PERCEPTIONS OF THE IMPORTANCE OF OBJECT PROGRAMMING LANGUAGE CONTENT BY INDUSTRY PROFESSIONALS: A PRELIMINARY STUDY

Joseph P. Laverty, Robert Morris University, laverty@rmu.edu
David F. Wood, Robert Morris University, wood@rmu.edu
Frederick G. Kohun, Robert Morris University, kohun@rmu.edu
Gary J. DeLorenzo, California University of PA, delorenzo@calu.edu

ABSTRACT

The purpose of this paper is to compare the insight gained between preparing IS (information systems) graduates with relevant skill sets to their ability to function in an entry-level technical position in industry. Based on the needs of educational programs and professional practitioners, model curriculums generally position graduates with relevant technical programming skills. A survey was prepared and disseminated to compare perceptions of object-oriented programming language content knowledge from classroom theory to practical application in the field across three (3) groups of stakeholders (students, faculty, and professional programmers). As an initial effort to analyze the data results of the survey, this paper analyzes the findings for the professional programmer group while identifying overall conclusions.

Keywords: ABET, Accreditation, Object-Oriented Programming, Programming and Programming Language

INTRODUCTION

IS Model Curriculum 97, IS 2002, and IS 2005 have recognized computer programming languages and implementation as a significant component of Information Systems curriculums. Fundamental programming language structures, procedural and object-oriented concepts, design, implementation object-orientation and data structures are included in the IS Body of Knowledge [6]. While no specific programming language or platform is specified, IS curriculum models do recommend a comparison of programming languages.

The importance of preparing IS graduates to function in an entry-level position with a technical orientation based on the needs of educational program and professional IS practitioners has been long recognized. Model curriculums should reflect input from both industry and universities. "It responds to industry requests for both increased emphasis in technical orientation and improved skill in individual and group interactions" [6]. However, the process of implementing IS curriculum model has been a battle between 'Hard' and 'Soft' approaches in the IS curriculum [15] and challenges to adding new course content or tracks to existing foundation course requirements [3].

In addition, research has shown the challenges that exist in building information system programming courses that meets both the needs of industry professionals while also meeting the course content requirements for ABET CAC accreditation, as noted by McCauley et al [21].

To continue research in this area in comparing the hard and soft skills for a programmer, a survey was prepared to study and compare perceptions of object-oriented programming language content across three separate, distinct groups of stakeholders: 1) students who have taken object-oriented programming language courses, 2) the faculty who teach object-oriented programming language courses, and 3) professional programmers who hire entry-level programmers. The intent was to gain quantitative data and analyze the results concerning the knowledge base and experience of participants in "writing code". As a basis to capture the essence of programming knowledge, the object-oriented program concept survey questions were not programming language specific. For the purpose of this study, the data discussed was drawn from a sample taken from industry professionals.

BACKGROUND AND METHODOLOGY

The initial efforts in drafting the survey was to compare student's technical skills to practical application in the field focused on ABET-CAC curricular issues related to programming language content. The programming language content of the ABET-CAC model [1] was reviewed and used as the basis for building the survey questions. As the survey content was scrutinized, content from object-oriented programming language text books [4, 5, 7, 9, 16, 18, 19, 20, 22, 23, 26, 27, 32] and professional language certification exams [10, 11, 24, 28, 33] were also reviewed to improve the survey instrument. The intent was to prepare a hierarchy of language-independent content categories and to establish an object-oriented programming language content domain. Whenever possible, Computer Science (CS) and IS course descriptions were used to align content boundaries within the realm of normal CS/IS program courses. A subsequent review of faculty and industry programmers

recommended the following six sub-domains for the survey: 1) Fundamental Object-Oriented Programming Concepts, 2) Programming Language Operators, Control and Algorithmic Structures, 3) Program Development Concepts, 4) Client Application Technologies, 5) Data Structures, and 6) Enterprise and Integration Technologies

There has been considerable research into computer language courses in the IS curriculum. The research has been conducted in areas such as language of choice, features and functionality, and trends of computer languages, as noted by Raadt et al [30], Ali et al [2] and Parker et al [29]. Each author shares concern over the general theme of comparing the perceived industry pressure for graduates to ascertain certain skill sets versus academic training gained in the classroom through generic programming skills. Also, features and trends for programming language content continue to evolve with the integration of eCommerce technologies into the IS curriculum such as client-side applications (HTML, JavaScript, Applets) [17] and server-side applications (Websphere) [31].

In recent years, Internet applications and IS enterprise development have to include beyond web services and other service-oriented architectures (SOA) which present new challenges throughout the systems development life cycle as noted by Kohun et al [13] and Lawler et al [14]. To address the challenges to integrate web services into the IS curriculum, Lawler et al studied the factors that contributed to the factors that contributed to the effectiveness of SOA as a strategy in industry. Based on the findings of this study, the authors saw opportunities for a new program of study for SOA in the IS curriculum from a management perspective than a technical perspective. Using a more technical approach to integrate web services to the curriculum, Kachru and Gehring [12] contrasted the use of using J2EE and .NET Platforms for Teaching Web Services. One of our intentions was to further quantify the professional programmers perceptions on the "readiness" of graduating IS students to contribute quickly in the workforce as an entry-level programmer.

The survey was prepared using the Vovici web based survey tool. The survey design included a front end demographic section, sections on institutional information, on accreditation, and on programming language experience that allowed the respondent to answer questions related to programming concepts germane to specific object-oriented programming languages (i.e., Visual Basic/VB.NET, C/C++, and/or Java).

The survey instrument was designed to present the program content categories to all three target populations—students, faculty, and industry professionals, but vary the response format to each target population. The purpose of this survey and study was to use an industry professional sub-sample to test the research design methodology and survey instrument before being disseminated to all target populations.

The researchers utilized a convenience snowball sampling technique to gather as much random sample data as possible in an expedient manner as possible. Each researcher sent email to industry professional contacts that included and introduction to the research project, a link to the survey on the Vovici website, and a request for the contact to forward the email to other industry professional colleagues with a personal request to participate in the survey so as to "snowball" the sample and response to the survey. The survey response data was captured and stored in the Vovici site for later analysis.

Participants were asked to rate the perceived importance of each topic for entry-level programmers. Four categories were listed, "No importance was valued as 1, "Awareness" as 2, "Functional" as 3 and "Mastery" as 4. "Awareness" was defined as to recognize the topic, and know what it is, but not any more. "Functional" importance was defined to mean that a new programmer should be able to implement the concept in an application or activity. "Mastery" was defined as being an expert in the nuances and issues involved in the application of the concept.

Weighted by the values defined above, an average importance was created for each topic, and the results presented in descending order of importance. Our intention for future research was to compare the importance thus defined with the emphases of current information systems courses.

The survey results were analyzed and subsequent modifications in the methodology and instrument will be used for the next phase of the project—inclusion of student and faculty populations along with an expanded industrial professional population.

FINDINGS, ANALYSIS, AND DISCUSSION

Survey Participants Academic Backgrounds

The participants consisted of 15 computer professionals. 40% of the participants had a Computer Science background, and 33% were Information systems. See Table 1.

Table 1 Academic Background		
Academic Background	**N**	**Pct**
Computer Science	6	40%
Information Systems/Sciences	5	33%
Internet/Network Technology	1	7%
Computer Security/Forensics	1	7%
Business	1	7%
Mathematics	1	7%

Programming Language and Professional Experience

The average age of respondents was approximately 39 years. Table 2 summarizes professional programming language experience and academic preparation by specific programming language. The data indicated that while the respondents had academic C and C++ academic or technical preparation, no respondent reported professional use of C or C++ beyond two years experience. The data indicates increased academic preparation and professional use of VB.NET or JAVA. The data indicates a lack of C# training, but yet increased professional use.

Table 2 Academic and Professional Programming Language Experience				
	>= 2 classes	**< 2 classes**	**>=2 years professional program experience.**	**<2 years professional program experience**
VB.NET or Visual Basic	20%	20%	33%	20%
Java	27%	20%	27%	40%
C/ C++	14%	33%	0%	60%
C#	7%	0	7%	27%

The roles played by these professionals emphasized both systems analysis and application development. Their specific professional roles are highlighted in Table 3. The most prevalent role most played in their organization had to do with systems analysis and project management. At least 60% of the respondents had professional programming language experience. Since these individuals were chosen from experienced professionals, these roles are not surprising.

Table 3 Professional Experience	
Significant Roles Played	**Pct of Respondents**
System Analyst	80%
Project Manager	67%
Standalone Application Developer	60%
Enterprise Application Developer	60%
Requirements Modeler	60%
Design Modeler	53%
Application Tester/Quality Assurance	47%
Database/Warehouse Specialist	33%
Business Intelligence Specialist	27%
Training	27%
Mobile Application Developer	7%
Network Specialist	7%

Perceptions regarding fundamental Object-Oriented Programming Concepts

The participants were asked the importance of understanding and applying each of the following programming language content categories for an entry-level programmer. Table 4 presents the frequency and a weighted average of their responses. The weighted average is ranked them in decreasing order of importance:

Table 4 Fundamental Object-Oriented Programming Concepts					
	No Importance	**Awareness**	**Functional**	**Mastery**	**Weighted Average**
Objects, classes, primitives and enumeration types, methods and functions	0	0	7	5	2.93
Set and Get Functions/Methods	0	1	6	5	2.86
Inheritance: concrete	0	2	7	3	2.64

Table 4
Fundamental Object-Oriented Programming Concepts

	No Importance	Awareness	Functional	Mastery	Weighted Average
classes, subclasses, abstract classes and interfaces					
Scoping: Information hiding and exposing objects, e.g., public, private; Persistence: static, dynamic, final, etc.	0	2	7	3	2.64
Coupling and Cohesion between Classes	1	6	5	2	2.57
Encapsulation: Overriding or overloading methods	0	2	9	1	2.50
Constructors (new) and Deconstructors	0	4	7	1	2.36
Program-to-Interface Principle	1	7	3	2	2.29
Extend versus implement	0	6	5	1	2.21
Polymorphism	2	3	7	0	2.07
External Classes	1	7	4	0	1.93

As can be seen from Table 4 most of the fundamental object-oriented concepts were highly desired at the functional or mastery level. Of lesser importance was an understanding of more abstract principles and concepts. Most of the fundamental object concepts are typically covered in an introductory object-oriented programming course.

Perceptions regarding Programming Language Operators, Control and Algorithmic Structures

Table 5 presents the frequency and a weighted average of responses concerning programming language operators, control and algorithmic structures. The weighted average is ranked them in decreasing order of importance:

Table 5
Programming Language Operators, Control and Algorithmic Structures

	No Importance	Awareness	Functional	Mastery	Weighted Average
Method/Function parameters and returns	0	0	5	8	2.61
Selection structures: If…Else, Switch (Break and Continue), Exception Handling (Try, Catch and Finally), etc.	0	0	5	8	2.61
Repetition Structures: While, Do-While, For and For-each Loops, etc.	0	0	6	7	2.56
Array Processing (single and multiple dimensional)	0	5	5	5	2.50
Assignment, arithmetic, relational and logical operators; operator precedence	0	2	4	7	2.44
Character, String, Date objects, methods and operators	0	1	8	4	2.33
Standard File I/O and streams	0	0	12	1	2.22
Network-based object programming	1	7	3	2	1.78
Object serialization	1	8	3	1	1.67

Function usage and the basic control structures were emphasized. File I/O, Network based object programming, and object serialization were of lesser importance. Only the basic control structures were expected to be known at the mastery level. Most of the significant content categories are typically covered in an introductory object-oriented programming course. No conclusion can be made at this point concerning the coverage of lesser important content categories.

Perceptions regarding Program Development Concepts

Table 6 presents the frequency and a weighted average of responses concerning program development concepts. The weighted average is ranked them in decreasing order of importance:

Table 6
Program Development Concepts

	No Importance	Awareness	Functional	Mastery	Weighted Average
System Development Life Cycle (SDLC)	0	4	5	4	3.0
Requirements Analysis	0	3	7	3	3.0
Code Tracing and Debugging	0	1	8	3	2.9
Program Development and Deployment	0	4	4	4	2.8
Unit, Interface or System Testing	0	3	6	3	2.8
Program Editors and IDE	1	6	5	1	2.5
Application Security Principles, e.g., Type Safety, Buffer Overflows, etc.	1	6	3	2	2.3
User/Role-based or Code Access Security	0	7	4	1	2.3
UML	3	6	4	0	2.1
Application Encryption, Data integrity and Signatures	1	8	3	0	2.0
Platform/Container Security	0	10	2	0	2.0

Notice that the Systems Development Life Cycle was the most important concept reported. We found it interesting that UML and Security principles were some of the least valued. Entry-level people were expected to be able to trace code and debug it, but not have mastery level understanding of an Integrated Development Environment.

Perceptions regarding Client Application Technologies

Table 7 presents the frequency and a weighted average of responses concerning client application technologies:

Table 7
Client Application Technologies

	No Importance	Awareness	Functional	Mastery	Weighted Average
Event Handling	0	3	6	2	2.7
Local Client Applications, e.g., Applets, Active X, Standalone, etc.	0	4	6	1	2.5
GUI Forms, Components and Controls, e.g., VBForms, AWT, Swing, etc.	0	4	6	1	2.5
Text-based User Interfaces	2	5	3	1	2.1
Graphics	2	7	3	0	2.1
Multimedia, e.g., embedding graphics, audio, and video files	3	6	2	0	1.8

Event handling, client applications and their visual interface were most important, while graphics, multimedia and older text-based interfaces were valued least.

Perceptions regarding Data Structures

Table 8 presents the frequency and a weighted average of responses concerning the coding of data structures:

Table 8
Data Structures

	No Importance	Awareness	Functional	Mastery	Weighted Average
Lists and Sets	1	2	6	3	2.7
Collections	1	2	7	2	2.6
Searching and Sorting Algorithms	1	3	5	3	2.6
Recursion	1	2	8	1	2.5
Stacks and Queues	1	4	4	3	2.5
Hash Tables	2	2	6	2	2.5
Linked Lists	2	2	7	1	2.4
Binary Trees	2	4	5	1	2.2
Digraphs	3	4	4	1	2.1

It is interesting to note that the Data structures most valued were lists, sets, and collections. More complicated data structures were not considered important.

Perceptions regarding Enterprise and Integration Technologies

Table 8 presents the frequency and a weighted average of responses concerning Enterprise and Integration Technologies:

Table 9
Enterprise and Integration Technologies

	No Importance	Awareness	Functional	Mastery	Weighted Average
Database Connectivity Concepts, e.g., JDBC, ADO.NET, ODBC, JDBC, SQL, etc.	0	0	12	1	3.1
Dynamic Web Page Generation Concepts	0	4	8	1	2.8
Enterprise Layer Technology Models, e.g., client, presentation, business logic, and EIS tiers.	0	6	5	2	2.7
Server Application Technologies, e.g., JSP, Servlets, ASP.NET, beans, etc.	1	4	8	0	2.5
Web Services (XML, SOAP WSDL, and UDDI)	0	7	6	0	2.5
Remote Method Invocation	0	9	4	0	2.3
AJAX	2	6	5	0	2.2
Mobile Device Program Development	3	9	1	0	1.8

For these participants, Database Connectivity was by far the most important topic not only in this subdomain, but was higher than many fundamental object – oriented concepts and all other subdomain content categories. Students were also expected to be familiar with dynamic web page generation. We suspect the types of companies these professionals worked in,

mostly large, well-established firms, were the reason that mobile device programming was not considered important.

Languages Desired for New Programmers

Finally, we asked the participants how important each language was for entry level programmers to know. Participants were permitted to name multiple languages. This data is portrayed in Table 10.

Table 10
Language Importance

Language	Pct of Participants
Visual Basic or VB.NET	53%
Java	47%
ASP.NET	47%
C/C++	40%
JavaScript	40%
PHP	27%
Cobol	7%

It appears that Microsoft VB.Net with ASP.Net as well as Java were seen as the most important language for new programmers. We were mildly surprised that C and C++ were still considered important.

CONCLUSIONS AND FUTURE RESEARCH

This paper was a preliminary study that will be expanded to include a larger industrial professional population as well as the inclusion of students and computer faculty. The professional target population response scale presented in this study focused on the perceived importance of various programming language content categories across six content subdomains for entry-level programmers. Future student responders will be asked to report their perceived awareness or knowledge of the same content categories using the scale of none, some, average, and above average. Future faculty responders will be asked to report significant coverage of the same content categories using the scale of no coverage, first programming language course, second programming language course and other course coverage.

In phase two of this project, we intend to: 1) compare the coverage of object-oriented programming language content categories in CS, IS and other programming language curriculums to the perceptions of professional programmers concerning entry-level program requirements, 2)) compare the perceived awareness or knowledge of object-oriented programming language content categories to the perceptions of professional programmers concerning entry-level program requirements, and 3) to determine if any possible differences may be associated with a background variable, e.g., academic major, experience in a programming language, or professional experience.

It was clear by the results of this preliminary study that Database Connectivity and Enterprise Application Development were more important to professional responders than some other traditional programming language concepts. Educators understand the importance of prerequisite skills for these topics. Enterprise application development, database connectivity, application security and testing, and demands of project management have presented new challenges to CS and IS curricula. While the nature and number of programming language courses may vary by curriculum, most CS and IS curricula do have one thing in common - programming language requirements. Whether a CS or IS graduate starts their career as a programmer, system analyst or project manager, most will be affected during their education by academic programming language courses.

It is important to periodically assess the level and sequence of programming language content in relationship to industry requirements. All CS and IS faculty and professionals are inherently biased to a specific programming language or prerequisite programming language content area. Programming language content assessment should be a constant process that is focused on the future demands of our graduates. This is goal of Phase 2 of this study and survey.

REFERENCES

1. ABET Computing Accreditation Commission (CAC) Computer Programs Forms and Criteria. Retrieved April 30, 2010 from http://www.abet.org/forms.shtml
2. Ali, A.I & Kohun, F. G. (2005) Suggested Topics for an IS Introductory Course in Java. Proceedings of the 2005 Informing Science Institute (June 2005).
3. Cameron, B. (2008). "Enterprise systems education: new directions & challenges for the future." Proceedings of the 2008 ACM SIGMIS CPR

conference on Computer personnel doctoral consortium and research, pp 119-126.
4. Cashman, G., Cashman, T., Starks, J. and Mick, M. (2006). "Java Programming: Comprehensive Concepts and Techniques, 3rd Edition, Thompson/Course Technology.
5. Crume, J., Mukhar, K. and Weaver, J.. (2004) "Beginning J2EE 1.4: From Novice to Professional" . Apress L. P.
6. Davis, A., Gorgone, J., Cougar, J., Feinstein, D., Longenecker, H., (1997). IS'97 Model Curriculum and Guidelines for Undergraduate Degree Programs in Information Systems
7. Doyle, B. (2007). "C# Programming: From Problem Analysis to Program Design". 2nd Edition. Cengage Learning.
8. Ehie, I. (2002) "Developing a Management Information Systems Curriculum: Perspectives from MIS Practitioners," Journal of Education for Business, Jan/Feb., pp 151-158.
9. Evjen, B., McCarthy, T., Sheldon, B., Pinnock, J., and Lhotka, R. (2004) "Professional VB.NET" , Wiley, John & Sons, Incorporated.
10. Fundamentals of the Java Programming Language, Java SE 6 (SL-110-SE6). Retrieved on February 2, 2010 from: http://education.oracle.com/pls/web_prod-plq-dad/show_desc.redirect?dc=D61796GC10
11. Java Platform Overview for Managers. Retrieved on 2/12/2010 from: http://education.oracle.com/pls/web_prod-plq-dad/show_desc.redirect?dc=D63400GC10
12. Kacrhu, .S, Gehring, E. (2004). A comparison of J2EE and .NET as Platforms for Teaching Web Services. 34th ASEE/IEEE Frontiers in Education Conference (Session S3B) Retrieved on February 12, 2010 from: http://citseerx.ist.pse.edu/viewdoc/download?do1=10.1.1.86.8564
13. Kohun, F., Wood, D. and Laverty, J. (2007) "Systems Oriented Architecture, Unified Process Life Cycle, and IS Model Curriculum Compatibility: Meeting Industry Needs," Information Systems Education Journal, 7(15) pp. 1 -9.
14. Lawler, J., Benedict, V., Howell-Barber, H., and Joseph, A. (2009)." Critical Success Factors in the Planning of a Service-Oriented Architecture (SOA) Strategy for Educators and Managers ," Information Systems Education Journal, 7(94) pp. 1-9.
15. Lowery, G.; Turner, r. (2005) "Education Systems Education for the 21st Century: Aligning Curriculum Content and Delivery with the Professional Workplace." Idea Group, Inc. p 171.
16. Li, L. and L. Li. (2002) "Java: Data Structures and Programming". Springer-Verlag New York, LLC.
17. Mahmoud, Q., Wlodek, D. and Swayne, D. (2004) "Redesigning Introductory Computer Programming with HTML, JavaScript, and Java", Proceedings of the 35th SIGCSE technical symposium on Computer science education, pp. 120-124
18. Malik, D. (2006). Java Programming: Program Design including Data Structures". Thomson/ Course Technology.
19. Malik, D. (2007). "C++ Programming; From Problem Analysis to Program Design". 3rd Edition . Thomson/Course technology.
20. Matsik, B., West, J., Greenwood, J., Kauffman, J. , and Xie. D. (2003). "Beginning ASP.NET Databases Using VB.NET". Wiley, John & Sons, Incorporated
21. McCauley, R. and Manaris, B.. (2002) Comprehensive Report on the 2001 Survey of Departments Offering CAC -Accredited Degree Programs. Technical report CoC/CS TR#2002-9-1, Department of Computer Science, College of Charleston.
22. McDonald, M., and Szpista, M. (2007). "Pro ASP.NET 3.5 in C# 2008". 2nd Edition. Apress, L. P.
23. McMillan, M., (2006). "Data Structures & Algorithms Using VB.NET". Cambridge U.P.
24. Microsoft Learning Course 2565A: Developing Microsoft.NET Applications for Windows (Visual Basic .NET). Retrieved on February 2, 2010 from: http://www.microsoft.com/learning/en/us/course.aspx?ID=2565A&locale=en-us
25. Mitchell, T. and Strauss, J. (2001) "Practitioner and Academic Recommendations for Internet Marketing and E-Commerce Curricula," Journal of Marketing Education, 23(2), pp. 91-102.
26. Murach, M. (2006). "Murach's C# 2005". Mike Murach & Associates, Inc.
27. Nagel, C., Watson, K., Gylnn, J., Skinner, M. and Evien, B. (2008). " Professional C# 2008". Wiley, Johns and Sons, Inc.
28. Object-Oriented Analysis and Design Using UML (OO-226). Retrieved on February 2, 2010 from: http://education.oracle.com/pls/web_prod-plq-dad/show_desc.redirect?dc=D61808GC10
29. Parker, K. R., Chao, J. T., Ottway, T. A. and Chang, (2006) J. A Formal Language Selection Process for Introductory Programming Courses. Proceedings of the 2006 Informing Science Institute (June 2006).
30. Raadt, d.M., Watson, R. and Toleman (2002) Language Trends in Introductory Programming

Courses. Proceedings of the 2002 Informing Science Institute (June 2002).
31. Sandvig. J. (2007) "Selection of Server-Side Technologies for an E-Business Curriculum," Journal of Information Systems Education, 18(2), pp 215-226.
32. Schildt, H. and Herbert, S. (2006). "The Complete Reference to Java" , 7th Edition. McGraw-Hill
33. Sun Certified Programmer for the Java Platform, Standard Edition 6. Retrieved on February 2, 2010 from: http://education.oracle.com/pls/web_prod-plq-dad/db_pages.getpage?page_id=41&p_exam_id=1Z0_851