

INFORMATION SYSTEMS EDUCATION JOURNAL

In this issue:

4. **A Paradigm for Student Learning Outcome Assessment in Information Systems Education: Continuous Improvement or Chasing Rainbows?**
Bruce Saulnier, Quinnipiac University
15. **Big Data in the Information Age: Exploring the Intellectual Foundation of Communication Theory**
Debra J. Borkovich, Robert Morris University
Philip D. Noah, Robert Morris University
27. **Entrepreneurial Health Informatics for Computer Science and Information Systems Students**
James Lawler, Pace University
Anthony Joseph, Pace University
Stuti Narula, Pace University
42. **Confronting the Issues of Programming In Information Systems Curricula: The Goal is Success**
Jeffrey Babb, West Texas A&M University
Herbert E. Longenecker, Jr., University of South Alabama
Jeanne Baugh, Robert Morris University
David Feinstein, University of South Alabama
73. **An Active Learning Activity for an IT Ethics Course**
David M. Woods, Miami University
Elizabeth V. Howard, Miami University Regionals
78. **Swipe In, Tap Out: Advancing Student Entrepreneurship in the CIS Sandbox**
Connor Charlebois, Bentley University
Nicholas Hentschel, Bentley University
Mark Frydenberg, Bentley University

The **Information Systems Education Journal** (ISEDJ) is a double-blind peer-reviewed academic journal published by **EDSIG**, the Education Special Interest Group of AITP, the Association of Information Technology Professionals (Chicago, Illinois). Publishing frequency is six times per year. The first year of publication is 2003.

ISEDJ is published online (<http://isedj.org>) in connection with ISECON, the Information Systems Education Conference, which is also double-blind peer reviewed. Our sister publication, the Proceedings of ISECON (<http://isecon.org>) features all papers, panels, workshops, and presentations from the conference.

The journal acceptance review process involves a minimum of three double-blind peer reviews, where both the reviewer is not aware of the identities of the authors and the authors are not aware of the identities of the reviewers. The initial reviews happen before the conference. At that point papers are divided into award papers (top 15%), other journal papers (top 30%), unsettled papers, and non-journal papers. The unsettled papers are subjected to a second round of blind peer review to establish whether they will be accepted to the journal or not. Those papers that are deemed of sufficient quality are accepted for publication in the ISEDJ journal. Currently the target acceptance rate for the journal is about 45%.

Information Systems Education Journal is pleased to be listed in the 1st Edition of Cabell's Directory of Publishing Opportunities in Educational Technology and Library Science, in both the electronic and printed editions. Questions should be addressed to the editor at editor@isedj.org or the publisher at publisher@isedj.org.

2014 AITP Education Special Interest Group (EDSIG) Board of Directors

Wendy Ceccucci
Quinnipiac University
President – 2013-2014

Scott Hunsinger
Appalachian State Univ
Vice President

Alan Peslak
Penn State University
President 2011-2012

Jeffrey Babb
West Texas A&M
Membership Director

Michael Smith
Georgia Institute of Technology
Secretary

George Nezek
Univ of North Carolina
Wilmington -Treasurer

Eric Bremier
Siena College
Director

Nita Brooks
Middle Tennessee State Univ
Director

Muhammed Miah
Southern Univ New Orleans
Director

Leslie J. Waguespack Jr
Bentley University
Director

Peter Wu
Robert Morris University
Director

S. E. Kruck
James Madison University
JISE Editor

Nita Adams
State of Illinois (retired)
FITE Liaison

Copyright © 2014 by the Education Special Interest Group (EDSIG) of the Association of Information Technology Professionals (AITP). Permission to make digital or hard copies of all or part of this journal for personal or classroom use is granted without fee provided that the copies are not made or distributed for profit or commercial use. All copies must bear this notice and full citation. Permission from the Editor is required to post to servers, redistribute to lists, or utilize in a for-profit or commercial use. Permission requests should be sent to Nita Brooks, Editor, editor@isedj.org.

INFORMATION SYSTEMS EDUCATION JOURNAL

Editors

Nita Brooks
Senior Editor
Middle Tennessee
State University

Thomas Janicki
Publisher
University of North Carolina
Wilmington

Donald Colton
Emeritus Editor
Brigham Young University
Hawaii

Jeffrey Babb
Associate Editor
West Texas A&M
University

Wendy Ceccucci
Associate Editor
Quinnipiac University

Melinda Korzaan
Associate Editor
Middle Tennessee
State University

George Nezek
Associate Editor
Univ of North Carolina Wilmington

Samuel Sambasivam
Associate Editor
Azusa Pacific University

Anthony Serapiglia
Teaching Cases Co-Editor
St. Vincent College

Lawrence Cameron
Teaching Cases Co-Editor
University of Montana

ISEDJ Editorial Board

Samuel Abraham
Siena Heights University

James Lawler
Pace University

Monica Parzinger
St Mary University Texas

Teko Jan Bekkering
Northeastern State university

Paul Leidig
Grand Valley State University

Bruce Saulnier
Quinnipiac University

Eric Breimer
Siena College

Michelle Louch
Robert Morris University

Li-Jen Shannon
Sam Houston State University

Ulku Yaylalicegi Clark
Univ North Carolina Wllmington

Pacha Malyadri

Michael Smith
Georgia Institute of Technology

Gerald DeHondt II

Cynthia Martincic
Saint Vincent College

Karthikeyan Umapathy
University of North Florida

Janet Helwig
Dominican University

Muhammed Miah
Southern Univ at New Orleans

Stuart Varden
Pace University

Scott Hunsinger
Appalachian State University

Marianne Murphy
North Carolina Central University

Bruce White
Quinnipiac University

Mark Jones
Lock Haven University

Anene Nnolim
Florida Institute of Technology

Peter Y. Wu
Robert Morris University.

A Paradigm for Student Learning Outcome Assessment in Information Systems Education: Continuous Improvement or Chasing Rainbows?

Bruce Saulnier
bruce.saulnier@quinnipiac.edu
Computer Information Systems
Quinnipiac University
Hamden, CT 06518, USA

Abstract

A paradigm is presented for student learning outcome assessment in information systems education. Successful deployment of the paradigm is illustrated using the author's home institution. The paradigm is consistent with both the scholarship of teaching and learning and the scholarship of assessment. It is concluded that the deployment of the paradigm allows us to address program constituent concerns regarding student learning in higher education while simultaneously being consistent with accreditation requirements at the program (ABET), school (AACSB) and institutional (NEASC) levels.

Keywords: Assessment, Accreditation, Program Constituents, Program Educational Objectives, Student Learning Outcomes.

1. INTRODUCTION

Over the course of the last decade there has been an increased emphasis for student learning outcome assessment at the national level. Although "No Child Left Behind" (United States Congress, 2002) and "Race to the Top" (United States Department of Education, 2009) have garnered the most press, this national movement for educational accountability is now directly impacting accreditation requirements of the regional higher education accrediting agencies (NEASC, 2011). Additionally, both school accreditation requirements (AACSB, 2003) and program accreditation requirements (ABET, 2012) have issued calls for the assessment of student learning.

Further impacting the need for higher education institutions to address assurance of learning is the public's demand for proof that graduates will have a reasonable opportunity for a successful

career at graduation given both the catastrophic student debt levels and the ever-changing economic landscape. All of these increasing demands for accountability are arising at a time of both a decreasing traditional college-age population and the emergence of Massive Open Online Courses (MOOC's).

In response to these increased demands for accountability institutions of higher education have placed an increased emphasis on assurance of learning by measuring student learning outcomes. In many cases desired student learning outcomes have been defined at the university, school, and program level. But are we in higher education simply chasing rainbows? Can student learning be measured, in any real sense of the word? The fallout from "No Child Left Behind" is legendary, and the "Race to the Top" is increasingly being met with skepticism. But if we in higher education fail to respond to the increasing demands for

accountability with measures and processes that are meaningful to us, and if we fail to convince the public that our results are meaningful, then it is likely that the measures and processes will be defined for us.

2. INFORMATION SYSTEMS ACCREDITATION REQUIREMENTS

For undergraduate programs in Information Systems, accreditation requirements exist at least at two levels: (1) regional accreditation agencies, for which requirements must be met; and (2) at the program level, through ABET, which is a program-level option. Additionally, for programs existing within Schools of Business, accreditation requirements exist through the Association to Advance College Schools of Business (AACSB), although this is also a voluntary option. All three levels of accreditation require attention to the assessment of student learning, though the individual requirements vary in terms of the language they employ.

While much prior work has been done in terms of information systems assessment, and this prior work has appeared in major Information Systems (IS) journals, only a few articles (Beard, Schweiger & Surendran (2008); Mills, Hauser, & Pratt (2008)) appear to link IS assessment to larger issues of school-wide assessment. An exhaustive literature such has failed to turn up a single article that links IS assessment to larger institutional assessment concerns associated with regional accreditation. Further, most of the current literature is micro in its scope advocating for either (1) a particular method associated with a particular course and/or learning outcome or (Carpenter, Snyder, Slauson, & Bridge (2011); Murray, Perez, & Guimaraes (2008); Wagner, Longenecker, Landry, Lusk, & Saulnier (2008)) or (2) the effectiveness of a particular method employed across the IS curriculum (Al-Mubaid, Abeysehera, Kim, Perkins-Hall, & Yue (2011); AAsheim, Gowan, & Reichgelt (2007); Saulnier, Landry, Longenecker, & Wagner (2008)). But none of the work to date has focused on the larger issue of providing a paradigm that addresses the link of assessment to IS ABET program accreditation while simultaneously addressing assessment requirements at both the school and institutional-levels.

At the institutional level, the New England Association of Schools and Colleges (NEASC), one of the six major regional accrediting

agencies, devotes standards 4.48 through 4.54 to the assessment of student learning, specifically requiring that each academic institution implements and provides support for systematic and broad-based assessment of how students are learning. NEASC further requires that each institution use a variety of quantitative and qualitative methods and both direct and indirect measures to understand the experiences and learning outcomes of its students, and that the institution use the results of these assessments for improvement.

At the school level, our information systems program is located in an AACSB-accredited school of business, and as such our school-wide accreditation must conform to AACSB Assurance of Learning standards. These standards are based on the premise that learning is the central activity of higher education, and that the definition of learning expectations and assurance that graduates achieve expectations are key features of any academic program. AACSB Standard 16 specifically requires that for each undergraduate degree program the school must define learning goals, and that for each academic program the school demonstrates that students meet the learning goals. Moreover, if assessment demonstrates that learning goals are not being met, that processes are in place and are being employed to eliminate the discrepancy.

At the program level, we choose to use ABET guidelines to maintain program-level accreditation, which requires that student performance be monitored to foster success in attaining student outcomes, thereby enabling graduates to attain program educational objectives. As such, we must define our program level constituencies consistent with the ABET definition of constituency, define our Program Educational Objective, define our student Learning Outcomes necessary for our students to obtain program educational objectives, and develop and execute a successful assessment program to insure that our program is meeting its educational objectives and modify it as necessary based on assessment results.

3. PRECURSOR/PRINCIPLES FOR EFFECTIVE STUDENT LEARNING

In response to the need to address how effective our students are learning, it is desirable for the faculty to engage in scholarly teaching; that is, whatever teaching and assessments they

employ should be consistent with what we know about how students learn. While the Scholarship of Teaching and Learning (SoTL) has been an object of higher education research for decades, the last decade has added significantly to our knowledge base.

Building on the earlier work of Chickering & Gamson (1987), Bransford (2000) has provided significant insight into the science of learning. Consistent with Bransford's findings, Fink (2003) asks us to move beyond the earlier taxonomy of Bloom (1956) to produce *significant* learning for our students by engaging in *backward course* design. Kuh (2008), writing on behalf of a nationally commissioned study group of the American Association of Colleges and Universities (AAC&U), extends the paradigm further by categorizing certain pedagogies as *High Impact Practices*; that is, a research-based group of instructional practices that have been shown to positively impact student learning.

We are fortunate at Quinnipiac University to be in an academic environment that has made an intentional commitment to become an exemplar of a *Learning Paradigm College* (Tagg, 2005). As such, all institutional resource allocation decisions are made based on the degree to which they have the potential to positively impact student learning (Barr & Tagg, 1995). Indeed, our program focuses on active "learning by doing" instructional practices, and among the ways we actively address student learning are our Information Technology for Good (IT4G) initiative, a commitment to service learning, project-based courses with real projects (usually service learning projects done for not-for-profits), and required internships.

4. PROBLEM STATEMENT

All of us in higher education in general, and in Information Systems in particular, seek to promote continuous improvement in our curriculum that results in improved student learning. But in order to objectively do so, we need a mechanism to "measure" our students' learning. Such has been the driver for the emergence of assessment requirements at the regional, school, and program levels.

While much has been done to promote and advance both continuous improvement and assessment of learning accreditation requirements, little appears to have been done to provide guidance and/or promote the use of

best practices to information systems programs attempting to meet these requirements. Rather than conduct separate assessments to meet separate accreditation requirements at the university, school, and program levels, it is highly desirable to employ an integrated procedure that meets all three sets of requirements in a single process. It is further desirable that the integrated process should promote the use of the AAC&U high impact practices consistent with the backwards course design espoused by Fink.

5. THE PARADIGM

The following 7-step process has been used to design and develop a program assessment and continuous improvement system consistent with the accreditation requirements of NEASC, AACSB, and ABET:

Step 1. Establish Program Constituencies

Although ABET does not specifically define what it means by a constituency of the program, ABET requires that the program must have published educational objectives that are consistent with the mission of the institution, the needs of the program's various constituencies, and the ABET criteria for accrediting computing programs. In other words, a program's educational objectives are based on the needs of the constituencies. It is therefore necessary for a program to have defined constituencies who are consulted to establish the program's educational objectives. While the ultimate responsibility for curriculum must necessarily lie with the program faculty (NEASC, 2011), the definition of program educational objectives is made with the input and concurrence of the program's constituencies.

Our CIS program has defined our program constituencies to be (1) the full-time program faculty, (2) the CIS Advisory Board, (3) Alumni of the program, and (4) employers of our graduates. The purpose of a constituency, then, is really to assist in the definition of program's educational objectives.

Step 2. Define Program Educational Objectives

Program Educational Objectives (PEO's) are by definition broad statements that describe what graduates are expected to attain within a few years after graduation (ABET, 2012). We have interpreted a "few years" to mean 3-5 year goals of the program. Specific PEO's were adopted by

a vote of the full-time program faculty after (1) presentation to and feedback from the program's advisory board, (2) interviews with internship supervisors and employers of our graduates, and (3) a formally conducted program alumni survey.

Specific PEO's adopted by our CIS program are within three-five years of graduation program graduates will have:

1. Helped an organization achieve its goals by applying knowledge and skills in the application of information systems;
2. Used information systems for decision making to help organizations achieve a strategic competitive advantage;
3. Served as liaisons between end-users and computing specialists by communicating effectively in both oral and written form;
4. Worked effectively in teams to manage both themselves and their colleagues; and
5. Demonstrated lifelong learning skills by attendance at continuing professional education courses/workshops, pursuit and/or attainment of professional certification, and/or higher-level academic education.

Step 3. Define Student Learning Outcomes

ABET accreditation requirements (ABET, 2012) specify that the program must have documented student learning outcomes (LO's) that prepare graduates to attain the program educational objectives, and that there must be a documented and effective process for the periodic review and revision of these student outcomes. The requirements specifically (p. 3) specify that the program must enable students to attain, by the time of graduation:

- (a) An ability to apply knowledge of computing and mathematics appropriate to the discipline;
- (b) An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution;
- (c) An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs;
- (d) An ability to function on teams to accomplish a common goal;
- (e) An understanding of professional, ethical, legal, and security and social issues and responsibilities;

- (f) An ability to communicate with a range of audiences;
- (g) An ability to analyze the local and global impact of computing on individuals, organizations, and society;
- (h) Recognition of the need for and an ability to engage in continuing professional development; and
- (i) An ability to use current techniques, skills, and tools necessary for computing practice.
- (j) An understanding of the processes that support the delivery and management of information systems within a specific application environment.

These specific outcomes (a)-(i) are recommended/viewed by ABET as the minimal set of required LO's. While individual programs are free to adopt additional LO's to support their PEO's and the specific needs of their constituencies, they are not required to do so.

Though there is no mandate that the ABET LO's by discussed with the program constituencies, we did so to obtain their input as to the appropriateness and completeness of the recommended ABET list. After this discussion, the faculty of the department unanimously adopted the ABET list as our approved student learning outcomes.

Step 4. Map LO's onto PEO's

To determine whether the list of LO's contribute to student attainment of all of the PEO's a mapping of the LO's onto the PEO's is constructed. This insures that the LO's are sufficient to attain the desired PEO's. The specific mapping of our LO's onto our PEO's is shown in Exhibit 1:

Mapping of LO's Onto PEO'S
(Learning Outcome Contribution to PEO's)

LOs/PEOs	PEO 1	PEO 2	PEO 3	PEO 4	PEO 5
LO-a	X				
LO-b	X	X			
LO-c	X	X			
LO-d			X	X	
LO-e		X			X
LO-f		X	X		X
LO-g		X			
LO-h					X
LO-i	X	X			
LO-j	X	X			

Exhibit 1 ... Mapping LO's Onto PEO's

Step 5. Assign LO's to Specific Required Courses

Once we are convinced that our LO's are sufficient to attain our desired PEO's, the next step is to assign responsibility for delivering learning outcomes to particular courses. Given that our students will not necessarily take elective major courses, it is necessary that all learning outcomes be addresses in required major courses. While some learning outcomes may also be covered in courses outside the major, we cannot necessarily control what is being taught in those courses.

Required courses specific to our program in Computer Information Systems (CIS) are as follows:

- CIS 125 Systems Analysis & Design
- CIS 225 Object-Oriented (OO) SAD
- CIS 244 OO Programming
- CIS 301 Enterprise Systems
- CIS 330 Networking & Data Com.
- CIS 351 Database Design & Prog.
- CIS 440 IT Project Management
- CIS 484 IT Internship

Each of these required courses is specifically assigned responsibility for one or more of the LO's within the framework of Bloom's taxonomy of educational objectives (Bloom, 1956). The specific mapping of our LO's onto our required courses is shown in Exhibit 2.

LO's/ Course	CIS 125	CIS 225	CIS 245	CIS 301	CIS 330	CIS 351	CIS 440	CIS 484
LO-a			X		X			
LO-b	X	X					X	
LO-c			X			X		
LO-d						X	X	
LO-e				X			X	
LO-f	X			X				X
LO-g	X							
LO-h								X
LO-i		X					X	
LO-j	X						X	

**Exhibit 2 Course Responsibility Matrix
(Mapping LO's onto Required CIS Courses)**

CIS 484 is a required internship which is supervised by company personnel. While we are comfortable that success in this internship experience is a highly professional capstone experience, we cannot necessarily document the experience across a range of LO's for every

student. Hence, we have mapped but two LO's onto the required internship.

Our CIS Alumni Survey, our CIS Advisory Board Input, and the Career Leader Assessment survey/test administered in *SB 112 Career Development* are the means via which we receive input and document our effectiveness in achieving outcome h – recognition of the need for continuing professional development. This outcome is further reinforced by the internship supervisor input regarding students' attitudes and experiences in CIS 484.

We administer the Information Systems Analyst (ISA) exam in CIS 440, though the information covered on the ISA exam is not necessarily covered in CIS 440; (please see particulars in CIS Assessment Plan – Appendix A).

The LO's that have been assigned to specific courses become the basis for both course design consistent with the principles of backward course design (Fink, 2003), and the adoption of high impact practices (Kuh, 2007) to address the assigned LO's specific to each required course. While courses may have individual learning objectives beyond those that appear in the Course Responsibility Matrix, the matrix becomes the driver for a minimal set of learning outcomes for each required course.

Step 6. Adopt an Assessment Plan to Monitor Attainment of LO's

Prior to constructing individual course syllabi and developing course assignments to specifically address assigned LO's, an assessment plan should be developed such that both the course syllabus and course assignments are constructed consistent with departmentally approved assessment criteria. Such a plan should minimally indicate specifically, for each LO, the following information:

- Where and with what frequency the outcome would be assessed;
- The specific assessment methods for each instantiation;
- The *a priori* target criteria for student performance to be deemed satisfactory;
- Specific assessment results that will be/ have been obtained from the execution of the plan;
- Document which specific actions have been or will be taken as a result of the evaluation of assessment results.

The adopted plan can/should be a combination of both direct and indirect assessment measures/methods. The field of higher education assessment has been an area of scholarly inquiry for decades, and we should avail ourselves of the scholarly body of knowledge about assessment to effectively construct such a plan. From the seminal works of Astin (1991) and Angelo & Cross (1993) to the more recent contributions of Suskie & Banta (2009) and Sambell, McDowell & Montgomery (2013), much is known about effective assessment techniques and strategies that are consistent with and complimentary to what we know about the most effective teaching methods and how people learn. While standardized testing has a place in higher education, the use of such tests can and should be primarily formative/diagnostic, not the common summative evaluative culture of the academy.

The CIS Department adopted assessment plan is included as information in Appendix A. The plan was constructed to be consistent with both NEASC reporting requirements via consultation with our campus-wide assessment coordinator, and with AACSB reporting requirements via consultation of with the associate dean of our school of business. While readers may employ the format of the plan for guidance, it is highly advisable that each department's faculty construct their own plan in consultation with the appropriate individuals on their respective campuses.

Step 7. Implement the Assessment Plan

The assessment plan was implemented during the 2012 calendar year with data collected from both the spring 2012 and fall 2012 semesters. Syllabi were constructed and course-embedded assessment measures were adopted consistent with the learning outcome responsibilities associated with each required course.

In addition to course-embedded assessments, the following activities were undertaken consistent with the requirements of the assessment plan:

- An alumni survey was conducted to ascertain alumni opinions of both the PEO's and LO's, the effectiveness of our program, and other factors needed to measure our LO's consistent with our adopted Assessment Plan;
- Individual interviews were conducted with both the members of the CIS Advisory Board and CIS Internship

Supervisors consistent with the requirements of our Assessment Plan;

- Available data was collected from prior semesters CIS required courses to provide a baseline and effectively summarize results relative to our adopted plan; and
- Data was collected from all of the School of Business courses that contribute to our assessment plan.

The faculty of the department met during January 2013 to discuss the results of the data gathered to date and plan for changes that were implemented during the spring 2013 semester. Further assessment results were obtained from spring 2013 courses and this data has been discussed by the faculty during their end of the year meeting to plan for appropriate changes for the fall 2013 semester.

6. RESULTS

The adopted paradigm has been employed during the 2012 ABET reaccreditation process. Although the results of the reaccreditation process are not known at the time of this paper submission, the process has been well received by the members of the ABET site visitation team.

The paradigm itself is consistent with the measures required for both NEASC institutional accreditation and AACSB school of business reaccreditation, and results to date are consistent with both NEASC and AACSB requirements.

A very real benefit of the deployment of this paradigm has been the almost universal adoption of backwards course design principles by the department faculty. That is, course syllabi are constructed to support student learning outcomes associated with the course, in-class activities are designed to support the syllabi, assignments are constructed consistent with desired learning outcomes, and assessment measures are adopted which specifically address course learning outcomes. Consequently, student performance is rising as we become much more intentional in our teaching and learning focus.

7. CONCLUSIONS

The presented paradigm has been developed and deployed and has yielded results that are

consistent with ABET program reaccreditation requirements, AACSB school-wide assessment requirements, and university regional accreditation requirements.

A major benefit of intentional involvement in the accreditation and assessment processes is that it drives department faculty to consider questions and issues that they ought to be considering on a regular basis, but frequently get overlooked during the rapid rhythms of the normal semester activities. In particular, construction of the assessment matrix forces an intentional consideration of student learning as the primary driver of course design.

The major limitation of the paradigm is not in the design of the paradigm, but rather in its implementation. The outcome measures developed to date are predominantly indirect measures of student learning because successful exam performance does not necessarily mean that students can effectively perform the tasks in a professional work environment. While the use of service-based projects provides for better measures of student performance, those measures are very difficult to quantify.

So we return to the title of this article. Are we just chasing rainbows? We think not! The deployment of this paradigm has provided the CIS department with reliable high quality data to provide to stakeholders concerning the learning of our CIS students while simultaneously addressing the needs of our program constituencies, and it has done so in a manner that supports both school-wide and university-wide assessment and accreditation requirements. Further, maintenance of accreditation at the program level provides the department with data to support ongoing preferential treatment in terms of budget allocation to support faculty professional development activities and student learning outcomes.

One final point – the adoption of this paradigm has directed the faculty toward a much more intentional focus on the learning of our students. In the final analysis, isn't that what our courses should really be about?

8. REFERENCES

AACSB International (2003). *2003 Business Accreditation Standards: Assurance of Learning*. Retrieved May 10, 2013 from

<http://www.aacsb.edu/accreditation/business/standards/aol/> .

Aasheim, J., Gowan, J.A., and Reichgelt, H. (2007). Establishing an Assessment Process for a Computing Program. *Information Systems Education Journal*, 5 (1). <http://isedj.org/5/1/>. ISSN: 1545-679X

ABET (2012). Criteria for Accrediting Computing Programs 2012-2013. Retrieved May 10, 2013 from <http://www.abet.org/DisplayTemplates/Docs/Handbook.aspx?id=3142> .

Al-Mubaid, H., Abeysekera, K., Kim, D., Perkins-Hall, S., Yue, K. (2011). A Model for Long Term Assessment of Computing and Information Systems Programs. *Information Systems Education Journal*, 9(3) pp 59-67. <http://isedj.org/2011-9/> ISSN: 1545-679X

Angelo, T. and Cross, K.P. (1993). *Classroom Assessment Techniques: A Handbook for College Teachers*. San Francisco: Jossey-Bass.

Astin, A. (1991). *Assessment for Excellence: The Process and Practice of Assessment and Evaluation in Higher Education*. New York: American Council on Education.

Barr, R. and Tagg, J. (1995). *From Teaching to Learning: A New Paradigm for Undergraduate Education*. Change: 27:6.

Beard, D., Schweiger, D., Surendran, K. (2008). Integrating Soft Skills Assessment through University, College, and Programmatic Efforts at an AACSB Accredited Institution. *Journal of Information Systems Education*, 9(2) pp. 229-240.

Bloom, B. S. (1956). *Taxonomy of Educational Objectives, Handbook I: The Cognitive Domain*. New York: David McKay Co Inc.

Bransford, J., Brown, A. & Cocking, R. (2000). *How People Learn: Brain, Mind, Experience, and School*. Washington, DC: National Academy Press.

Carpenter, D., Snyder, J. E., Slauson, G. .., Bridge, M. (2011). Additional Support for the Information Systems Analyst Exam as a Valid Program Assessment Tool. *Information*

- Systems Education Journal*, 9(4) pp. 73-79.
<http://isedj.org/2011-9/> ISSN: 1545-679X
- Chickering, A. W., & Gamson, Z. F. (1987). Seven Principles for Good Practice in Undergraduate Education. *AAHE Bulletin*, 3-7.
- Fink, L. D. (2003). *Creating Significant Learning Experiences: An Integrated Approach to Designing College Courses*. San Francisco: Jossey-Bass.
- Kuh, G. (2008). *High-Impact Educational Practices: What They Are, Who Has Access to Them, and Why They Matter*. Washington, DC: AAC&U.
- Mills, R., Hauser, K., Pratt, J. (2008). A Comprehensive Two-Level Framework for Information Systems Curriculum Design, Assessment, and Improvement. *Journal of Computer Information Systems*, 48(4) pp. 1-14.
- Murphy, M. C., Sharma, A., Rosso, M. (2012). Measuring Assurance of Learning Goals: Effectiveness of Computer Training and Assessment Tools. *Information Systems Education Journal*, 10(5) pp. 87-94.
<http://isedj.org/2012-10/> ISSN: 1545-679X.
- Murray, M, Perez, J., Guimaraes, M. (2008). A Model for Using a Capstone Experience as One Method of Assessment of an Information Systems Degree Program. *Journal of Information Systems Education*, 9(2) pp. 197-208.
- NEASC: Commission on Institutions of Higher Education (2011). *Standards for Accreditation*. Retrieved May 10, 2013 from http://cihe.neasc.org/standards_policies/standards/standards_html_version/.
- Sambell, K., McDougall, L., and Montgomery, C. (2013). *Assessment for Learning in Higher Education*. Oxon, England: Routledge.
- Saulnier, B., Landry, J., Longenecker, B., Wagner, T. (2008). From Teaching to Learning: Learner-Centered Teaching and Assessment in Information Systems Education. *Journal of Information Systems Education*, 9(2) pp. 169-174.
- Suskie, L. and Banta, T. (2009). *Assessing Student Learning: A Common Sense Approach*. San Francisco: Jossey-Bass.
- Tagg, J (2003). *The Learning Paradigm College*. San Francisco: Jossey-Bass.
- United States Congress (2002). *PL 107-110. The No Child Left Behind Act of 2001*. Retrieved May 8, 2013 from <http://www2.ed.gov/policy/elsec/leg/esea02/index.html>.
- United States Department of Education (2009). *Race to the Top*. Retrieved May 8, 2013 from <http://www2.ed.gov/programs/racetothetop/index.html>.
- Wagner, T., Longenecker, B., Landry, J., Lusk, S., Saulnier, B. (2008). A Methodology to Assist Faculty in Developing Successful Approaches for Achieving Learner Centered Information Systems Curriculum Outcomes: Team Based Methods. *Journal of Information Systems Education*, 9(2) pp. 181-186.

Editor's Note:

This paper was selected for inclusion in the journal as an ISECON 2013 Distinguished Paper. The acceptance rate is typically 7% for this category of paper based on blind reviews from six or more peers including three or more former best papers authors who did not submit a paper in 2013.

Appendix A - CIS ASSESSMENT PLAN

Student Learning Outcomes	Where Assessed/ How Often Assessed	Assessment Methods	A-priori Target Criteria	Assessment Results	Recommendations for Improvement / Documentation
<i>a. An ability to apply knowledge of computing and mathematics appropriate to the discipline;*</i>	CIS 245 / yearly	Programming Assignments	80% of students will score 75% + on 80% of programming assignments.	. 80% of students scored >= 75% or above on all programming assignments 95% of students scored above 75% on all course networking assignments Average score on EC 271 final exam was 83% CIS ETS Quant Avg. = 44%	Meets criteria. Continue current assignments and assessments. <i>Meets criteria. Continue current assignments and assessments.</i> Meets criteria. Continue with current focus. Below average. Area of concern.
	CIS 330 / yearly	Networking Assignments	90% of students will score 75% + on 75% of networking assignments		
	EC 271 / Each semester	EC 271 common statistics final exam	CIS Students will have an average score of 75% + in common exam in EC 271.		
	SB 450 / Each Semester	ETS exam – Quant Bus Anal / yearly	CIS students will have an average score of 50% + on the ETS Statistics questions		
<i>b. An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution;</i>	CIS 125 / yearly	“Requirements” Assignment	125 - 90% of students will score satisfactory or higher 225 - 80% + on 80% of SAD assignments in 225 CIS majors will have average score 60% + in Systems Analysis section.	125 - 96% scored satisfactory or higher 225 – 88% scored above 80% QU average was 63% in 2010 ISAE section	Meets criteria. Continue current assignments and assessments. 225 – above target, but need additional work on activity modeling ISA - Meets criteria
	CIS 225 / yearly	Course assignments			
	CIS 440 / yearly	ISA Exam (Systems Analysis) / Yearly			
<i>c. An ability to design, implement, and evaluate a computer-based</i>	CIS 245 / yearly	Programming assignments	80% of students will score 75% + on 80% of programming	80% of students scored >= 75% on all	Meets criteria. Continue current assignments and assessments.

<i>system, process, component, or program to meet desired needs;*</i>	CIS 351 / yearly	Database assignments	assignments. 90% of students will score 75% + on database projects	programming assignments 90% of students scored at 85.4%	Meets criteria, continue with current assignments and assessments.
d. An ability to function effectively on teams to accomplish a common goal;*	CIS 351 / yearly CIS 440 / yearly	In-course team project In course team project	90% of students score 75% or above on team rubric 440 – All teams score 75% or above	90% of students scored at 85.4% 440 – 86% scored at 86%; 14% scored below 50%	351 - Meets criteria, continue with current assignments 440 – Meets Criteria
e. An understanding of professional, ethical, legal, security, and social issues and responsibilities;*	CIS 301 / yearly CIS 440 / yearly SB 450 / yearly	In-course assignment: Write Ethics Report ETS Exam – Legal & Social / yearly	301 – 85% > 90% 440 – 85% > 90% ETS Exam Avg. > 50%	301 – 88% scored 90%+ 440 – 86% > 90% CIS ETS Avg. Score = 59%	301 - Meets criteria 440 – below criteria, 3 students basically no show skew results ETS – Meets Criteria
f. An ability to communicate effectively with a range of audiences;*	CIS 125 / yearly CIS 301 / yearly CIS 484 / yearly Internship Supervisor Survey	In course presentation and analysis In course assignments and presentations Company Analysis Paper Internship feedback / yearly	90% of students will score satisfactory or higher 90% adequate 90% of CIS 484 Internship form respondents must grade QU CIS students “adequate/above average” in communication.	92% scored superior; 100% scored satisfactory or higher 301 – 95% Adequate or better 484 – 100% of papers Internships - 100% grade outstanding or above average	125 - Meets criteria 301 – Meets criteria 484 – paper meets criteria Survey – meets criteria
g. An ability to analyze the local and global impact of computing on individuals, organizations, and society;*	CIS 125 / yearly SB 450	In-Course assignment assessed with common rubric ETS Exam – International / yearly	80% score satisfactory or higher on assignment related to local/global impact CIS Avg. > 50%	96% scored satisfactory or higher; 76% at superior CIS Avg. = 62%	125 – Meets Criteria ETS – Meets Criteria
h. Recognition of the need for and ability to engage in	SB 112 Alumni Survey / every 3	Career Leader Assessment Alumni Survey;	Avg. CIS Score on Creative, Critical, and Strategic	AVG CIS Score > 75% for all 3 subsections	112 – Meets Criteria for all three subsections

<i>continuing professional development;</i>	years Advisory Board Input / yearly	Advisory Board Input	Thinking > 75% 75% of alumni and advisory board engaged in prof development activities yearly	Alumni - 81% engaged; 77% yearly; 100% of advisory board engaged yearly; 55% grad degrees	Meets criteria for both alumni and advisory board.
<i>i. An ability to use current techniques, skills, and tools necessary for computing practice.</i>	CIS 225 / yearly; CIS 440 / yearly	In-Course assignments ISA Exam (Programming, Networking, Database) / Yearly	90% score at or above 85% QU CIS majors will score 50% or better in ISA Exam programming, 50% in data management and 50% + in networking.	90% scored >87% QU CIS majors scored 48% in programming, 50% in database, and 50% in networking in 2010 ISA exam.	225 – meets criteria marginal in database and networking; need to strengthen tech ISA – Below standard in programming portion of curriculum
<i>j. An understanding of processes that support the delivery and management of information systems within a specific application environment.*</i>	CIS 125 / yearly CIS 440 / yearly SB 450	Analysis of processes for all SAD phases ISA Assessment Exam (Information Systems in Business) / Yearly ETS Exam / Yearly	125 - 80% score satisfactory or higher 440 - QU CIS Majors will score 60% or better in ISAE Info Systems in Business 450 - CIS avg. score >50% across all business areas	125 - 100% satisfactory or higher; 87% superior 440 - QU CIS students scored 65% in 2012 ISAE exam on this section. 450 - Avg. score across all business areas = 53%	125 – Meets Criteria ISA – Meets Criteria ETS – Meets Criteria
LO's with an * are consistent with Quinnipiac University Essential Learning Outcomes (ELO's)					

Big Data in the Information Age: Exploring the Intellectual Foundation of Communication Theory

Debra J. Borkovich, D.Sc.
Robert Morris University
borkovich@rmu.edu

Philip D. Noah, D.Sc.
Robert Morris University
philipdnoah@gmail.com

Abstract

Big Data are structured, semi-structured, unstructured, and raw data that are revolutionizing how we think about and use information in the 21st century. Big Data represents a paradigm shift from our prior use of traditional data assets over the past 30+ years, such as numeric and textual data, to generating and accessing petabytes and beyond of images and social media. Traditional databases stored only structured data consisting of letters and numbers, but in the era of Big Data a need arose to incorporate unstructured data as part of overall information management. The shift to Big Data started with the Internet boom of the mid to late 1990s and the "rich data" that could be collected through semi-structured and unstructured petabytes of behavioral, image, textual, and social media data. Social media, such as Facebook, Twitter, Wikis, Blogs, YouTube, etc., has also changed our view of data. Big Data is a relatively nascent field of study that has spawned the development of new hardware, software, and database architectures to handle the large volume of structured and unstructured data. However, the foundation for the exploration and analysis of Big Data is as old as the Information Age itself and is rooted in the field of communications. In this paper, the authors show how communication theory has developed in parallel with the development of the Information Age and how the application of convergent theories resulted in Big Data producing Business Intelligence.

Keywords: Big Data, Information Age, Communication Theory, Business Intelligence, ADIK, Web 2.0

1. INTRODUCTION

The history of Big Data is the history of the development of the computer from the first simple data processing machines to the large-scale data centers of the modern Information Age organization. It is also the history of communication theory. Communication theory is the foundation for analyzing and understanding data. The use of communication theory allows data to be combined, processed, and organized into information; then analyzed and transformed

into knowledge culminating into the core of Business Intelligence (BI). Without the use of communication theory, data collected and stored by companies would be of little value. It is therefore vital for students, researchers, and practitioners in the fields of Information Systems (IS) and Information Technology (IT) to have a working knowledge of communication theory.

This paper explores how communication theory plays an important role in understanding Big Data and its influence upon BI. We start by

defining what are Big Data, BI, and the Information Age organization. We explain the differences between data, information, and knowledge, and distinguish how they contrast. With this basic understanding of data and information, the discussion moves into the basics of communication theory. A brief history of the Information Age follows with a smattering of historical chronology and a brief literature review of selected subject matter experts and their respective theories. This leads to an examination of why and how communication theory applies to the clear and cogent transmission of information, and its logical relationship and transference to Big Data. Finally, we conclude with a brief look ahead into the social dimensions and future of Big Data.

What is Big Data?

Big Data is structured, unstructured, and raw data stored in multiple disparate formats that ultimately resulted in a paradigm shift in how we think of data. Traditional databases store only structured data consisting of letters and numbers (transactional data), but in the era of Big Data a need arose to store semi-structured and unstructured data. The shift to Big Data started with the Internet boom of the mid-1990s and the "rich data" that could be collected through e-commerce transactions (Noah & Seman, 2012). The unstructured data from e-commerce sites includes data on what items the user viewed, such as click-through data and sub-transactions. Social media also changed our view of what data are. Companies such as Facebook capture and store photos, video clips, sound bits, "likes," and instant messages. To store and process Big Data new highly scalable, available and low latency database systems are needed (Noah & Seman, 2012). Big Data systems often have databases in the petabyte range; a petabyte is equal to 1,000 terabytes or 10^{15} bytes. To turn Big Data into actionable information new ways of exploring and analyzing data were needed. This complex analytical process manifested itself into the role of Business Intelligence.

Business Intelligence

Business Intelligence (BI) is the process of generating actionable information from raw data. BI is not a technology; it is a combination of hardware, architectures, tools, methods, and databases (Turban, Sharda, Delen, & King, 2011). Modern BI allows decision makers to analyze real-time or near real-time data to make better decisions and to act on those decisions.

The purpose of BI is to transform companies into Information Age data driven organizations. The term "business intelligence" has been used since the late 1950s but modern BI systems only became possible with the advent of the data warehouse in the 1990s. The predecessor to the BI system was the Management Information Systems (MIS) of the late 1960s and 1970s. MIS were reporting systems that provided simple static, often times printed, two-dimensional reports (Turban, et al., 2011). The MIS lacked analytics and any changes to the reports that required a specialized programmer to modify the computer code. With the advances in technology the MIS systems of 1960s and 1970s gave way to the Executive Information Systems (EIS) of the 1980s. The advent of the personal microcomputer in the 1980s allowed the transformation of the MIS system from static reports to a more dynamic system. Users were able to generate ad-hoc reports and view reports that allowed the user to drill down from a high level overview or summary to detailed lower levels. The EIS also provided advanced features such as forecasting, predictions, and trending analysis (Turban, et al., 2011). By the mid-1990s, the EIS had become more powerful and were being used by mid-level management, as well as executives. The term "Business Intelligence" started to appear in technology journals and as common parlance within the popular media. The modern BI systems were developed to take advantage of "Big Data" created by the Internet, e-commerce, organizational textual documentation, social media, and traditional transactional data through the large-scale data warehouse. By 2005, BI systems were incorporating advanced analytics such as data mining, semantic analysis, and artificial intelligence (Turban, et al., 2011). BI is integrated into most large-scale software systems such as SAP, PeopleSoft, and IBM Cognos. What started out as a simple reporting tool for executives turned into a powerful system that allowed employees at all levels to be "data driven." BI evolved into creative and innovative analytics for problem-solving and decision-support.

Information Age organizations are data driven entities that use analytics to transform information into insight and action (LaValle, Lesser, Shockley, Hopkins, & Kruschwitz, 2011). LaValle and co-authors identified three stages that organizations go through to become data driven Information Age organizations. The first stage is the aspirational; the organization uses

tools more like those of the EIS than BI. The aspirational organization is focused on using technology to increase automation processes and cutting costs, as they do not have the resources in place to use analytics (LaValle, et al., 2011). The second stage is the experienced phase in which organizations start to build on the efficiencies gained in the aspirational stage (LaValle, et al., 2011). Organizations in the experienced stage further optimize their information systems and technology by developing BI to collect more data, analyze it, and act upon it (LaValle, et al., 2011). The final stage is the transformed stage. The transformed organization derives a competitive advantage from the use of BI and analytics (LaValle, et al., 2011). The transformed organization is less focused on cost cutting and optimization and more focused on the strategic use of analytics (LaValle, et al., 2011) by implementing analytics across a wide range of functions and through all levels of the organization. The Information Age organization captures, analyzes, and uses data to drive revenue growth and profits. Data are used not just to guide future strategies but also to guide day-to-day operations, providing decision-support and solving problems.

2. THE AGE OF INFORMATION: THE SCIENCE, TECHNOLOGY, AND BUSINESS

The Information Age, otherwise known to many as the Information Era, the Computer Age, or the Digital Revolution is a phenomenon that did not occur overnight. It is a concept directly linked to the computing, electronics, engineering, science, and communications reformation from the analog system to the digital system, characterizing the current age by the ability of individuals to transfer information freely and to have instant access to information previously unavailable or difficult to locate. Information Technology (IT), the overall concept of digital technologies used to process, store, and transport information, has provided vast benefits to many, such as the ability to control, access, share and manage data, information, and knowledge transfer; all powerful and progressive concepts.

It is equally important to remember that the emergence of a new and important paradigm does not always represent the same significance to all and indeed true to history, opposition opinions have surfaced as well. For example, some have asserted that the overwhelming bombardment of information via easy access to

multiple media has manifested itself as information silos, human isolation, and slaves to technology, particularly by those inexplicably and singularly driven to it. And the have-nots, those with neither funds nor access or those not interested in technological progress, internalized an even greater fear that they will be left even farther behind and perhaps, jobless. Standage (1998, p. 212) suggested that concerns and fears "are the direct consequences of human nature, rather than technology." These opposition views all have merit and deserve to be explored, researched, and examined for the important adversarial positions they represent to the future of the Information Age, therefore as space allows several will be included in the discussion. However, the emergence of the Information Age, through selected technological inventions, theories, theorists, and historical situations culminating in the business, social, and cultural benefits, now known as "Big Data," will be examined here.

Most agree that the period of Information is generally said to have begun in the latter half of the 20th century and continues through the present; although the precise date varies, because of the difficulty in pinpointing the specific dates for the global and public use of personal computers, Internet, e-mail, cell phones, personal communications devices, and social networking sites. Therefore, many dates for the important contributing factors to the spread of information within the common man's routine implementation are approximate and widely recognized as such. It is generally accepted that the Information Age was embraced and accepted by the masses since the late 1980s and into the 21st century, although the intellectual concepts and original inventions were developed somewhat earlier in the 20th century.

Another important matter that arose from the emergence of the Information Age was the overarching need to define the construct of information. Just what is information? And how does it apply to this context? Raw data is not information. Data is not information until it is collected, saved, stored, organized, transmitted, received, and understood. Data certainly must have meaning, hopefully the same meaning to both the sender and the receiver. Shannon's (1948) contribution to our understanding of this process was his development of the *Transmission Model of Communication* (Figure 1) that revolutionized our understanding of

transmissions by radically introducing the construct of digitizing information. Shannon opened the door to early Big Data development through his theory of a digital communication infrastructure and network that was cleaner, faster, more efficient, and generally an error-free mode than the analog process of an ordinary arithmetic computer or calculator.

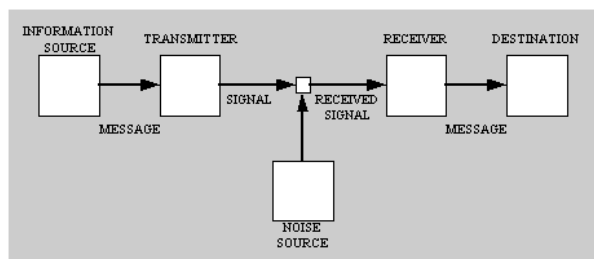
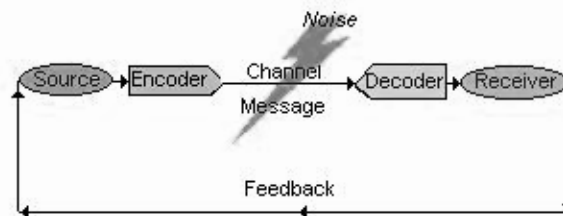


Figure 1. Transmission Model of Message Communication (Shannon, 1948)

Concisely paraphrased, the communication process works when a message is coded by the sender in a language that has meaning to the receiver, and the receiver can confirm the message as understood by returning feedback to the sender. This information transmittal process appears very simple to accomplish, but it is not as easy as it sounds as there are many codes, signs, symbols, languages, intuitions, sensory, and cultural perceptions incorporated into the message. All have applicable subjective meanings making it a very tricky undertaking to ensure that the message sent makes it through a certain amount of noise, interference, or distortion from outside sources; and equally challenging to ensure that when it is received it is truly decoded and comprehended as originally intended. Although Shannon was specifically focused on the technological and mathematically quantitative problem of developing a relatively error-free and accurate transmission model; his co-author and research partner, Warren Weaver, was particularly interested in conveying meaning and applying understanding to the message. Figure 2 represents and addresses Weaver's interest in the semantic and the effectiveness problems of the message, by adding a feedback loop to the communication model (Shannon & Weaver, 1949).

Not merely content to explain the process of communication, Shannon (1948) provided us with the term 'bit' to quantify the tiniest amount of explicit data possible in the form of a single binary unit. When collected, organized, and evaluated, 'bits' may even be determined as a

set of facts. Overall, information is commonly accepted as the result of processing, manipulating, and organizing data in a way that adds to the knowledge and learning of the person receiving it. And from data through information to knowledge, Liebowitz (2006, p. 78) cleverly reminded us that the discipline of Knowledge Management (KM) is continually challenged with "apply[ing] more rigor to the field so that it becomes more of a science and less of an art."



Shannon-Weaver Model

Figure 2. Shannon's Communication Model Appended with Weaver's Feedback Loop (Shannon & Weaver, 1949)

Furthermore, it is important to note that not all information is communicated through digitally recorded media or is computer generated. Other types of information, such as visual, aural, vocal, written, recorded, sensory, physical, tactile, signs, symbols, semiotics, language, instruction, knowledge, meanings, perceptions, feelings, intuition, cultural, and so many other things that when coded, transmitted, received, interpreted, and understood have specific and subjective meaning to human beings. In addition, all people in some way routinely convey, receive, share, and store countless types of information, regardless of whether or not we are computer scientists or professional librarians. In its most basic form, information is just a message received and understood because it has application. Knowledge occurs when the information becomes meaningful, is understood, and learned. However, as previously stated, this phenomenon did not happen overnight; a series of critical and sometimes disruptive events had to occur to generate this important historical transition.

3. HISTORY OF THE INFORMATION AGE

Many factors in the United States signaled the emergence of a new era. For example, most agree that during the mid-1950s through the 1960s, the number of Americans holding white-

collar jobs, professional, office staff, administrative, and sales, slowly started to exceed the number of people holding blue-collar jobs, those that were manufacturing-oriented and manual labor-driven (Toffler, 1984). In his book, "White Collar: The American Middle Class," Mills (1951, p. 182) referred to this middle class change commencing during the mid-1950s as a shift to marketing, sales, and service asserting that, "In a society of employees dominated by the marketing mentality, it is inevitable that a . . . great shift from manual skills to the art of servicing people, personal or even intimate traits of employees are drawn into the sphere of exchange and become commodities in the labor market." Mills argued that this change in the American labor force signaled that the Industrial Age was coming to an end and a new era was beginning to emerge as the Western world shifted into an information service driven economy.

During World War II and continuing thereafter, the U.S. Federal Government recognized the need for higher technologies to enhance the protection of its borders and citizens and to stabilize and further its dominance as a world power. This need for power and control incentivized Government agencies to collaborate with universities and non-profit think tanks whom together conceptualized and developed large main frame computers to process and organize data previously processed manually, enabling electronic document output and transfer and the eventual transmissions of local emails between networks and systems. Technology abounded from 1946 when the first general purpose electronic digital computer was developed (U.S. patent filed in 1947), known as the Electronic Numerical Integrator and Computer (ENIAC) funded by the U.S. Army and developed in collaboration with the University of Pennsylvania, originally designated as a post-WW II top secret project (Gertner, 2012). Built upon the research of Bell Labs in the late 1940s, the mass production of silicon transistors, a low cost key element of most modern electronics, appeared in the 1954 when Texas Instruments commercialized and mass produced the first transistor radios (Gertner, 2012). These inventions, among others, were instrumental in jump-starting the decline of blue-collar labor driven employment and were key in transitioning the workforce to white-collar service positions. Toffler (1984) chronicled these newsworthy events as "farms, factories, and floppies" in his book, "The Third Wave."

Concurrently, corporations continued their own research and development toward independent commercialization beginning with electronic typewriters, calculators, processors, and by the end of the 20th century eventually culminating in the miniaturization of computer hardware to a manageable, personal, and transportable size. The creation of the originally narrow-focused local Internet provided a way to connect computers together as part of a limited Government or a corporate network, and software developers were encouraged and incentivized to create new systems, productivity enhanced products, and personal applications. Berners-Lee's invention of the World Wide Web in 1989 permitted the Internet experience to be a global network available to everyone at little or no cost and at a remarkable speed eventually becoming the generally accepted platform for moving information. Table 1 (see Appendix) illustrates an approximate timeline of selected emerging inventions and significant activities highlighting the fundamental progress of the Information Age.

Not only were jobs and careers slowly changing, but large factories and manufacturing facilities were starting to implement tools and fixed machines to do the work traditionally performed by humans. In the 1960s, the U.S. Government implemented widespread use of computers, but it was not until the 1970s that they were introduced to corporate employees and in the 1980s on a more widespread and substantial offering to the general public. The earliest form of the Internet appeared in 1969 and limited networked email soon followed in 1971. But it was the first appearance of the personal computer in the late 1970s and the introduction of the World Wide Web in 1989 that propelled the Internet into full-scale operation by 1992. Dumb terminals of the 1980s were replaced by desktop computers in the 1990s and laptops during the onset of the millennium. Cell phones arrived in 1984, became cost effective, prevalent, and widely accepted in the 1990s subsequently emerging as Smartphones in the 2000s. Other personal assistance and communications devices followed shortly thereafter. The digital Information Age had begun in earnest and thanks to the arrival of the global Internet, Pandora's Box was forever opened and there was no turning back.

Theorists, educators, and consultants quickly emerged because these ever-evolving digital,

Internet, and cyberspace phenomena had to be critically explained to the rest of us. Vast sources of information technology and communication tools became a significant part of the economy and these new innovations rapidly overwhelmed our psyche. For example, computers, microcomputers, computerized machinery, broadband, fiber optics, communications satellites, Internet, software, biometrics, robots, and other personal communications devices altered not only Governments, business, and industry forever, but also changed personal lives, careers, families, cultures, and societies, as well. Fears of the unknown needed to be quelled, training needed to be provided, and overall explanations for the technology, scientific, business, social, economic, environmental, and cultural changes were warranted. Although this paper highlights only a few of the important Information Age theorists, the list in Table 2 (see Appendix) illustrates a brief timeline of some of their significant accomplishments that contributed to this new era of information, thereafter, illuminating our understanding of this inevitable and unavoidable adventure into the future. The dates listed are directly associated with the theorists' publications and are not intended to reflect any other significant events other than those discussed in this paper.

Emergence of Big Data

By 2004, Web 2.0 made its appearance in the form of social media, blogs, wikis, images, photos, rich site summaries (RSS), aggregate feeds, and other social-cultural behavioral transmissions that quickly augmented and often surpassed email as the primary form of electronic communication. Providing the public with an interactive form of consumer participation, such as commercial online ordering, banking, shopping, bill-paying, stock trading, game-playing, in addition to alerting a select group of followers or the public in general of an individual's every move, became addictive and self-perpetuating (Borkovich, 2011). And that was just the software. Innovative, creative, and attractive hardware had to keep up, as well. A computer that was once the size of an entire floor became room-sized; then personalized and desk-sized; then portable and lap-sized; then a tablet and palm-sized; and finally wrist-sized and positioned within the corner of an eye-glass frame; soon to become even smaller and more compact. Modern computers can perform just about anything a user or consumer wants and needs. Big Data had arrived, and we embraced

it; but after the giddy glow of excitement had abated, the construct of information overload quickly became a sobering reality.

Information Overload

Information overload, a term coined by Gross (1964, p. 856) and popularized by Toffler (1970), refers to the difficulty a person can have understanding an issue and making decisions that can be caused by the presence of too much information. It is a bombardment to the senses, causing confusion, distraction, disorientation, and lack of responsiveness long before assessing the validity of the content or recognizing the risk of misinformation. The anxiety and frustration caused by information overload can lead to a blind acceptance of all data, news, tweets, blogs, definitions, headlines, photos, social commentary, scores, trades, prices, RSS alerts, etc.; or an overwhelming outright rejection of all incoming data; or perhaps a paralytic response akin to a collaborative shutdown within the perceived safety of an information silo (Borkovich, 2012). This new paradigm shift (Kuhn, 1960) to Big Data continues to contribute to the effect of information overload and will require greater scrutiny and study as individuals learn to cope with the overwhelming daily bombardment of data. Furthermore, in order to prevent or at least manage this inevitable overload, organizations must continually seek advanced technologies to process and cope with voluminous amounts of data to support strategic Business Intelligence, competitive analysis, and decision-making.

4. IT IS THE BUSINESS: INFORMATION DOES MATTER

Carr (2003) in his well-known article, *IT Doesn't Matter*, argued that information technology is no longer relevant. The days of getting ahead by having the latest server or the newest network are gone, Information Technology (IT) is a commodity and Information Systems (IS) are part of the cost of doing business (Carr, 2003). Carr's argument is valid if the definition of IT focuses only on technology, servers, switches, and other systems that make up the traditional IT infrastructure. If we view IT in terms of the Information Age organization, we find that not only does IT matter but it represents the life blood infrastructure of the business. The storage of data, the retrieval of information, and the creation of knowledge are key business processes for any Information Age organization.

The Information Age organization is highly interconnected with customers, suppliers, and employees. IT and IS are an integral part of the organization in which people, processes, and technology-driven data come together to form knowledge. In the Information Age organization “the interaction between systems and humans become more collaborative” and “the [information] system also acts as an extension of the human ability to store and process knowledge” (Davenport, 2000, p. 169). This idea was further developed in Debons’ (2008) Augmented Data, Information, and Knowledge (ADIK) system in which IS augment human intellect. IT is the business for the Information Age organization, but we must first define what are data, information, and knowledge as well as examine how they relate to each other. We will explore the social dimensions of IT using the social network theory, and then examine some of the problems the Information Age organization faces with the storage and retrieval of data.

Data, Information, and Knowledge: What are the differences?

The distinctions between data, information, and knowledge are important to understand for our study of the Information Age organization. It is only when we see how these three terms are related, but different from each other, that the value of IT becomes apparent.

Debons (2008, p. 4) described data as “the collection of numbers, measurements, and simple signals that surround us every day.” Examples of data include a person’s name, a social security number, a house number, or a street sign. For the Information Age organization data are the raw materials of the Information System (IS). In the IS data are the characters stored in database files or records.

Information is created when the data are given a structure and organization (Debons, 2008). The number 42 is data, but when we learn that the number represents an age, and that age is related to the name John Smith, who lives in Anytown, USA, information is created. Information is stored in records in a database file. A record for John Smith may look like the depiction in Table 3.

John	Smith	42	Anytown	USA
------	-------	----	---------	-----

Table 3. Sample Record in a Database File

Knowledge is formed when patterns are found within the information (Debons, 2008). When the data are analyzed, it is disclosed that the population of Anytown is 6,540 people, Smith is the most common last name and half the population of Anytown is past the age of 42. Meaning has been applied to the information created from the data; and knowledge resulted from the learning process. More information is also known about John Smith: he has a common last name; and he is younger than 3,270 other people in Anytown. This new information was gleaned only by extracting, sorting, collating, and analyzing the raw data stored within the IS.

Augmented Data, Information, and Knowledge (ADIK) systems bring together data and information to create knowledge. An ADIK system “include(s) people, technology, and the functions/procedures that bring these together to achieve a goal” (Debons, 2008, p. 70). In the ADIK system knowledge is created when data are organized to form information and patterns are recognized in the information (Debons, 2008). The augmentation occurs when the system is used to allow someone to solve a complex problem more rapidly than one otherwise could (Debons, 2008). The ability to solve complex problems rapidly is the core of what Information Age organizations do. Figure 3 depicts Debons’ visual conceptualization of a feedback loop to determine the sender’s and receiver’s understanding and meaning attributed to transmitted information.

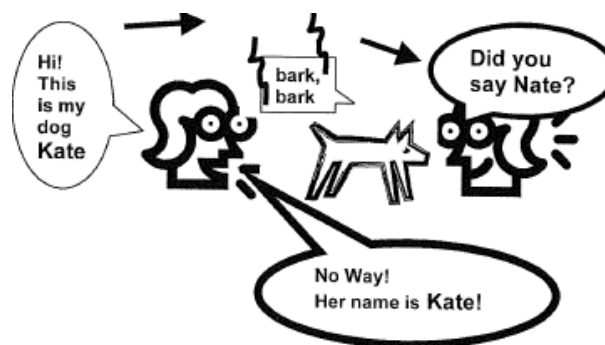


Figure 3. Debons’ (2008, p. 21) Visual Conception of Feedback Loop

Why IT Matters: The Information Age Organization.

The Information Age organization is knowledge-centric and relies on IT not just to process transactions and store data but also to create and transfer knowledge. Developing an

information and knowledge-centric focus is a necessity for all organizations, not just the typical information-centric actuarial-type industries, such as health care, banking, and insurance.

Davenport (2000) showed how the business environment changed for the manufacturing sector. Advanced enterprise systems allow manufacturers to be agile and respond to market changes. Davenport (2000, p. 164) explained that "a truly robust sense and response capability also requires process integration throughout the supply chain." To obtain a high level of integration with supply chain partners advanced Knowledge Management (KM) and Information Systems (IS) are needed. The Information Age organization also needs to know who its competitors are, who potential partners are, and when the two are the same. The amount of data collected and stored needs to be properly managed to turn it into actionable information and knowledge. The Information Age organization requires advanced technology systems to manage this wealth of data. Davenport (2000, p. 173) explained: "For the first time in the history of information systems, it will be possible to connect hard data and soft information and knowledge." There is also a need to "develop an ability to identify what information is really worth its users' time" (Davenport, 2000, p. 164). Without IT and IS it would not be possible to mine the raw data to create information, much less to analyze it and see all the patterns relevant to the problem that needs to be solved.

Information Technology (IT) and IS are at the heart of every Information Age organization. Pemsel and Widen (2010) show how KM is important to the construction industry by arguing that to stay competitive, project-based organizations such as construction need to retain and make good use of knowledge. In the construction industry like most project-based industries, knowledge is tacit; it is stored in the experience of the employees. Pemsel and Widen found that "firms want to turn tacit knowledge into explicit knowledge, and then they want to transfer it to the firm" (Pemsel & Widen, 2010, p. 127). In a project-based organization, it is often the tacit knowledge of the project manager that determines if a project will be profitable or not. The project manager's experience combined with data from other projects is needed to bid properly on a project. By storing this knowledge, the company can build up a

knowledge base or knowledge system that will grow over time and allow the company to make better decisions. When organizations do not invest in or maintain IS, they run the risk of losing valuable knowledge. And when an employee leaves a company that does not have a KM system, it is equally critical that the experience of that employee will be lost. The information stored needs to be retrievable and used in a systematic manner, as merely collecting the data and not acting on it adds no value to the organization.

Social Dimensions of Big Data

As we have seen, Information Technology (IT) and Information Systems (IS) are an interaction and interdependency of people and processes to turn data into information, information into knowledge, and knowledge into results. For the Information Age organization, "Knowledge creation and learning can be regarded as a social and dynamic process; it is not solely the transfer of information and data" (Pemsel & Widen, 2010, p. 123). Knowledge is also a social-cultural phenomenon that is transmitted through social networks and not just by the computer networks that transmit data.

Sasidharan (2006) examined the relationship between the level of social connectivity and success in implementing large-scale IS. The study used a social network analysis to measure how social connectivity and large-scale IS implementation are related. A social network model "views individuals as being embedded within an organization, enmeshed in a network of concrete interpersonal relations and structures" (Sasidharan, 2006, p. 33). The social connections measured in the study explained how the users were connected to each other, which parties they approached for advice and assistance, and to whom they provided help. The research found that organizations with a high level of social connectivity were more successful at implementing large-scale IS. Sasidharan (2006, p. 105) concluded, "It is not knowledge gathered at training but knowledge transferred in the organization workplace that influences implementation success." The findings in Sasidharan's research were consistent with Pemsel and Widen, who stated that knowledge is more than the transmission of information; and with Debons who showed that IT and IS systems only augment human knowledge and do not replace it. Communicating by moving data, developing information, and creating knowledge produces results.

5. LOOKING AHEAD: THE FUTURE OF BIG DATA IN THE INFORMATION AGE

Virtually any type of data required can be delivered instantaneously, and any business, organization, or academic institution that wants to stay competitive has no choice but to continually embrace new forms of technology. Ironically, the improvements made to any technology face the danger of a paradigm shift and the possibility that a new invention will render the old obsolete. The nascent emergence of new disciplines, such as cybersecurity, robotics, biometrics, and others are empirical proof of this continual digital advancement, fostered by the emergence of Big Data, coupled with the burgeoning requirement for advanced Business Intelligence, and supported by analytics.

6. CONCLUSION

Big Data and Business Intelligence (BI) are best understood when examined through the lens of communication theory. Information is not static, it is the result of a process in which data are transmitted, received, and understood. This is the model developed by Shannon in 1948 that inspired the early Management Information Systems (MIS). Shannon's model unveiled a unit of information called a 'bit' that transmitted from the sender to the receiver by assembling several bits to form information within a system. The early MIS system simply arranged data into static reports that were easier for people to view than the large tables of unsorted data. The modern BI system is an application of Debons' Augmented Data, Information, and Knowledge (ADIK) model. ADIK is a process in which data are transformed into actionable information and knowledge is created and learned. The augmentation is the use of analytics and artificial intelligence to allow people to have a better understanding of complex problems, enabling the human interface to arrive more efficiently and confidently at better data driven solutions (Debons, 2008). The ability to turn data into actionable knowledge relies on the interaction of people, processes, technology, and systems. Knowledge is a socio-cultural construct transmitted through social networks as well as computer networks. People receive information from BI systems, apply meaning and take action based upon that information; and through this learning process create knowledge through understanding and by communicating it to

others. For without the implementation of communication theory in concert with the analytical ability and creativity of the human element, data, *big or otherwise*, will remain literally and explicitly locked within its digital, virtual, or physical confines; or dwell tacitly memorialized within a mortal brain silo of a figurative data warehouse, *forever*.

7. REFERENCES

- Borkovich, D. J. (2011). The social science of data warehousing: Its ever-evolving corporate culture. *Issues in Information Systems, 12*(1), 23-35.
- Borkovich, D. J. (2012). When corporations collide: Information overload. *Issues in Information Systems, 13*(2), 269-284.
- Carr, N. G. (2003, May). IT doesn't matter. *Harvard Business Review Onpoint, 35*66, 41-49.
- Debons, A. (2008). *Information Science 101*. Lanham, MD: The Scarecrow Press.
- Davenport, T. H. (2000). The Future of Enterprise System-Enabled Organizations. *Information Systems Frontiers, 2*, 163-180.
- DiNucci, D. (1999). Fragmented future, *Print, 53*(4), 32-34.
- Gertner, J. (2012). *The idea factory: Bell Labs and the great age of American innovation*. New York, NY: The Penguin Press.
- Gross, B. M. (1964). *The managing of organizations: The administrative struggle*. New York, NY: The Free Press.
- Kimball, R. (2011). The evolving role of the enterprise data warehouse in the era of big data analytics. [White Paper] pp. 1-28. Retrieved from: http://www.informatica.com/downloads/1597_EDW_Big_Data_Analytics_Kimball.pdf
- Kuhn, T. (1996). *The structure of scientific revolutions* (3rd Ed.). Chicago: The University of Chicago Press. (Original work published 1970)
- LaValle, S., Lesser, E., Shockley, R., Hopkins, M. S., & Kruschwitz, N. (2011). Big Data, analytics and the path from insights to

- value. *MIT Sloan Management Review*, 52(2), 21-32.
- Liebowitz, J. (2006). *What They Didn't Tell You about Knowledge Management*. Lanham, MD: Scarecrow Press, Inc.
- Mills, C. W. (1951). *White collar: The American middle classes*. New York: Oxford University Press.
- Noah, P., & Seman, S. (2012). Distributed Multi-Level Query Cache: The Impact on Data Warehousing. *Issues in Information Systems*, 13(2), 51-57.
- Pemsel, S., & Widen, K. (2010). Creating Knowledge of End Users' Requirements: The Interface between Firm and Project. *Project Management Journal*, 41, 122-130.
- Sasidharan, S. (2006). Social knowledgetransfer and social network influences on enterprise resource planning systems implementation. *ProQuest Information and Learning Company*. (UMI No. 3259177).
- Shannon, C. E. (1948). A mathematical theory of communication. *The Bell System Technical Journal*, 27(July/October), 379-423; 623-656.
- Shannon, C., & Weaver, W. (1949). *The mathematical theory of communication*. Chicago: University of Illinois Press.
- Standage, T. (1998). *The Victorian internet: The remarkable story of the telegraph and the nineteenth century's on-line pioneers*. New York, NY: Walker Publishing Company, Inc.
- Toffler, A. (1970). *Future shock*. New York, NY: Random House.
- Toffler, A. (1981). *The third wave*. New York: NY: Bantam Books.
- Turban, E., Sharda, R., Delen, D., & King, D. (2011). *Business intelligence a managerial approach*. Saddle River, NJ: Prentice Hall.
- Wang, X. (2008). Matching records in multiple databases using a hybridization of several technologies. *ProQuest Information and Learning Company*. (UMI No 3308336).

Appendices

Table 1. Information Age – Historical Time Line of Relevant Innovations		
Type	Description	Date Invented/Widespread Use
ENIAC	First general purpose electronic digital computer	1946 ¹
Transistor	Invention/Commercialized by Texas Instruments	1947 / 1950s ²
Designation of the 'Bit'	Binary Code for Ones and Zeros	1948 ³
More "White Collar" than "Blue Collar" workers in U.S.	"Industrial Age" coming to an end; Moving into "Information Age"	1956 ³
Computer Systems	Government Use	1960s ⁴
Earliest form of Internet	Limited Government Use	1969 ⁴
Email	Limited Government Use	1971 ⁴
Personal Computer	Limited Business Use	Late 1970s ³
Personal Computers	Public Use	1970s / 1980s ³
Timesharing	Outsourcing Services	1970s / 1980s ⁵
Mobile Phone	Motorola	1983 ⁵
World Wide Web	Invention	1989 ⁴
Computers in Homes	Widespread Public Use	1980s / 1990s ⁵
Laptop	Widespread Public Use	1990s ⁵
World Wide Web	Globalization	1996 ⁵
Cellular Phones	Globalization	1990s / 2000s ⁵
Webcams/Digital TV	Widespread Public Use	1990s / 2000s ⁵
Broadband Mainstreamed	Widespread Public Use	2000s ⁵
Wireless Networking	Widespread Public Use	Early 2000s ⁵
GPS Mainstreamed	Widespread Public Use	Mid 2000s ⁵
Satellite Radio	Widespread Public Use	2003 ⁵
Digital Radio	Widespread Public Use	2004 ⁵
HDTV	Widespread Public Use	Mid 2000s ⁵
Smart Phones	Widespread Public Use	Mid 2000s ⁵
Internet Population	Over 1 Billion	2005 ³
Cell Phones	Over 3 Billion	2008 ³
Semi- & Autonomous Robots; Biometrics	Gov't., Academia, & Health Care Use	2001 - Current ⁶
Web 2.0	Interactive, User-Created Web Content	Approx. 2004 - Current ⁷
Big Data	Structured, Semi- & Unstructured Data	Approx. 2005 - Current ⁸

1. US Army Research Lab (ARL): <http://ftp.arl.mil/~mike/comphist/eniac-story.html>
2. Texas Instruments: www.ti.com/corp/docs/company/history/timeline/semicon/1950/docs/54commercial.htm
3. Computer History: http://www.computerhistory.org/internet_history/internet_history_80s.shtml
4. DARPA & Internet Timeline: http://www.darpa.mil/Docs/Internet_Development_200807180909255.pdf
5. Inventors - 20th Century of Technology Decade by Decade: http://inventors.about.com/od/timelines/a/twentieth_5.htm
6. NASA - ISS - DARPA: http://www.nasa.gov/mission_pages/station/main/robonaut.html
7. Web 2.0 term coined by DiNucci (1999, p. 32); popularized in 2004: User-generated content of social media in a virtual community.
8. Big Data appeared shortly after the emergence of Web 2.0 (Kimball, 2011)

Table 2. Information Age – Historical Time Line of Relevant Theories

Theorist	Publication	Description	Publication Date
Shannon, C. E. & Weaver, W.	"The Mathematical Theory of Information"	Transmission Model of Communication	1948; 1963; 1998
Kuhn, Thomas	"Structure of Scientific Revolutions (3 rd Ed.)"	History of Normal Science; & Paradigm Shift	1960; 1970; 1996
Simon, Herbert	"Theories of Bounded Rationality" & "Human Problem Solving" (with Allen Newell)	Human Cognition; Bounded Rationality; Satisficing; Design Trees (Parallel & Binary); Fragmentation	1972
Toffler, Alvin	"The Third Wave"	Iron; Industrial; & Information Ages	1984
Beniger, James	"The Control Revolution: Technological and Economic Origins of the Information Society"	Crisis Leads to Controversy; Loss of control brings forth technology innovation and change	1986
Utterback, James	"Mastering the Dynamics of Innovation"	Dominant Model succeeds but is not always the best choice	1994
Negroponte, Nicholas	"Being Digital"	Changes from analog to digital (atoms to bits)	1995
Christensen, Clayton	"The Innovator's Dilemma: the Revolutionary Book That Will Change The Way You Do Business"	Rationales for Great Companies' Failures	1997; 2000; 2003
Carr, Nicholas	"Does IT Really Matter"; "Does IT Matter? Information Technology and the Corrosion of Competitive Advantage"; & "The End of Corporate Computing"	Strategic IT; Tech is Not the Advantage; Process is the Advantage	2003; 2004; 2005
Von Hippel, Eric	"Democratizing Innovation"	Share Information; Open Source Code; Must Disclose for Long Term Solutions & Success	2005
Debons, Anthony	"Information Science 101"	ADIK System; EATPUT: Event, Acquisition, Transmission, Processor, Utilization, Transference	2008

Entrepreneurial Health Informatics for Computer Science and Information Systems Students

James Lawler
lawlerj@aol.com

Anthony Joseph
Ajoseph2@pace.edu

Stuti Narula
sn42066n@pace.edu

Seidenberg School of Computer Science and Information Systems
Pace University
New York, New York 10038 USA

Abstract

Corporate entrepreneurship is a critical area of curricula for computer science and information systems students. Few institutions of computer science and information systems have entrepreneurship in the curricula however. This paper presents entrepreneurial health informatics as a course in a concentration of Technology Entrepreneurship at a leading institution of technology. In the concentration, and in the course, students are learning to be business entrepreneurs in interdisciplinary fields, such as health. This paper can be beneficial to educators in schools of computer science and information systems desiring to enrich offerings to be contemporary with the demands of industry.

Keywords: computer science, computing curricula, entrepreneurship, interdisciplinary, health, health informatics, information systems

1. BACKGROUND

Corporate entrepreneurship is an approach for applying creativity and innovation as a means to entrepreneurial opportunity (Ireland, Kuratko, & Morris, 2006) in a process or product in industry. Entrepreneurship is a discipline of initiative, innovation, opportunity recognition, and pursuit of reward and risk (Phillips, & Garman, 2006). The criticality of enabling entrepreneurship for creativity and innovation is cited in the business literature (Byers, Dorf, & Nelson, 2011). The enablement of entrepreneurship from creativity and innovation fosters entrepreneurial opportunity in industry.

The health industry can benefit from entrepreneurship coming to life from the current Affordable Care Act (ACA) of 2010 - Obama Care - legislation.

The health industry is considered costly (Ratten, 2012), impersonal, and inefficient in methods of operation (Cutler, 2010). The industry is \$2.6 trillion or 17.9% of the gross domestic product (GNP) of the country (Norbeck, 2012). Entrepreneurial innovation in an electronic health medical or record system (EMR) can help in cost cutting in a hospital. The industry is additionally considered inefficient in patient-oriented service (De Regge, Gemmel, Degadt,

Verhaeghe, Sijnave, & Duyck, 2012) and performance of staff. Innovation in an intrapreneurial predictive personalized-prescription process and provider system can improve patient service. The industry is also considered in need of improved consumer-oriented systems (Ratten, 2012). Innovation in medical mobile systems (Milian, & MacMillan, 2012 and Briley, 2013) can improve non-patient and patient self-service (Howard, 2008). The health industry is clearly fertile for increased entrepreneurial opportunity (The Economist, 2013) and from increasingly required organizational response (Deluca, & Enmark, 2002) – a field that can be infused by schools of computer science and information systems.

Graduates of schools of computer science and information systems having entrepreneurial skills can be innovators in the health industry. As millennial students, they can be, for example, innovators in medical monitoring applications (apps) on smart-phones (Horowitz, 2012b) and Twitter; in integration of applications, patient record systems and reimbursement systems; in mining of patient record systems (Mathews, 2013) and of patterns and relationships in the systems (Srinivas, Rani, & Govrdhan, 2010); and in process re-engineering and research of health information exchanges (McNickle, 2013) and of service systems and tools (Ebling, & Kannry, 2012). They can be innovators in modeling intervention programs for issues, such as obesity and diabetes, or policies, such as HIV (Greengard, 2013). They can be further innovators in new DNA genetic profiling tools (Howard, 2008) and smart clothing tools (Velshi, 2013). They could explore favorable funding from investors in health industry incubators – a \$1.1 billion field in 2012, a 70% increase from \$626 million in 2011 (The Economist, 2012b). To be exploiters of health industry opportunity, computer science and information systems students have to learn not merely technology but also the business of the health industry and the economic potential of technology in an interdisciplinary Technology Entrepreneurship concentration. Few schools of computer science and information systems have a Technology Entrepreneurship concentration integrating a health informatics course opportunity for students, an opportunity addressed in this paper.

2. INTRODUCTION

The Seidenberg School of Computer Science and Information Systems of Pace University began a concentration in Technology Entrepreneurship in its Bachelor of Arts in Computer Science (Lawler, & Joseph, 2010). The concentration is designed for students to learn the fundamentals to be business opportunists, not mere scientists or technologists. The emphasis of the concentration is in the development of cutting edge ideas for a process, product or service, infused by entrepreneurial innovation if not invention of systems and tools, in a fictitious firm, or if feasible in an actual firm. The emphasis of the concentration currently is in a course in Entrepreneurial Health Informatics in the disciplinary domain of the health industry, as conceptually depicted in Figure 1 of the Appendix, inasmuch as Obama Care is encouraging entrepreneurship in the industry (The Economist, 2012a). Encouraging for schools, Obama Care is expanding innovation in popular mobile tools (Everett, 2013) interesting to students.

The generic learning objectives of Entrepreneurial Health Informatics are defined below:

- Define a business competitive or cost effectiveness idea for a process, product or service for the health industry that can be infused by innovation or invention of new systems or tools;
- Design and develop a process, product or service, or prototype, for the health industry that can furnish opportunity for productivity if not profitability from integration of new solution systems or tools;
- Design and develop a business and financial plan for the process, product or service for the health industry – hospital or physician practices, provider systems, or mobile non-patient or patient self-service tools - infused by the potential of productivity if not profitability of the new systems or tools;
- Design and develop a customized plan for marketing the process, product or service for the industry, infused by the new solution systems or tools, to marketplace providers and society; and

- Integrate contemporary innovation in the marketplace, such as cloud methodology, data mining and data warehousing technology, which might improve the solution systems or tools of the new venture.

The course is consistent in design with other interdisciplinary domains in the concentration of Technology Entrepreneurship in the school, as depicted in Figure 2, and is 4 credits. The outcomes are in adaptability, analysis, business, collaboration and communication - aspects of an entrepreneurial mind in creative critical thinking, problem-solving and risk-taking - in an expanding field (Brill, 2013). The outcomes are for students to be not mere technologists but business opportunists of technology. These outcomes conform to demands of consumers (Everett, 2013) and of the health industry for persistent and self-motivated students skilled in solution technology (Malugani, 2012). Entrepreneurial Health Informatics, in the concentration curriculum of Technology Entrepreneurship, can be beneficial to Bachelor of Arts in Computer Science students in the Seidenberg School, in the crafting of an experience that can improve hiring prospects (Khan, 2013) in health if not other industries.

3. FOCUS OF PAPER

The course in Entrepreneurial Health Informatics, in the concentration of Technology Entrepreneurship in the Bachelor of Arts in Computer Science program in the Seidenberg School of Computer Science and Information Systems of Pace University, is the focus of the paper. The domain of the health industry, as an entrepreneurial field of interdisciplinary study, is current to the expectations of government, industry and society for computer science and information systems students. The entrepreneurship in the health industry is not as formed as in other industries (Phillips, & Garman, 2006), nor as fulfilled in information technology (Kellermann, & Jones, 2013), furnishing innovation opportunity for students. The paper is not focused on the other courses in the concentration of Technology Entrepreneurship, as depicted in Figure 2, of which the course of Entrepreneurship and Technology was covered in an earlier study (Lawler, & Joseph, 2010). This paper will be beneficial to instructors in schools of computer science and information systems desiring to

improve offerings to be current with industrial and societal trends.

4. CONCENTRATION METHODOLOGY - TECHNOLOGY ENTREPRENEURSHIP

The concentration in Technology Entrepreneurship began officially in 2011, from basic courses in the Bachelor of Arts in Computer Science ebbing into courses in entrepreneurship, in a desired sequence:

- Entrepreneurship and Technology, a concept course currently integrating computer science and entrepreneurship on a project for business competitive decision-making;
- Customer Relationship Management (CRM) and Entrepreneurship, a concept course currently integrating data mining and data warehousing on a project for decision-making in strategy;
- Entrepreneurship and Financial Computing, a domain course currently integrating algorithmic computing, entrepreneurship, finance, financial analysis and information systems on a project for financial decision-making and strategy;
- Cloud Sourcing, a domain-enabling course currently integrating cloud methodologies and platforms on a project for financial decision-making and strategy;
- Modeling of Financial Processes, Products and Services through Technology, a domain-enabling course currently integrating cloud platforms, finance and information systems on a project for financial decision-making, on reengineering of systems through prototyped or real software technologies;
- Entrepreneurial Health Informatics, a domain course of the paper;
- Energy Efficiency Entrepreneurship, a domain course in the future, integrating entrepreneurship on a project for decision-making in the energy industry;
- Entrepreneurship and Governmental Security, a domain course in the future,

integrating governmental and industry policy on a project for crisis decision-making in national security strategy; and

- Special Topics in 21st Century Technologies and Ventures, an optional survey course further integrating leading edge marketplace technologies impacting new ventures.

The concentration of the courses of Technology Entrepreneurship covers a 2011-2015 period in the school.

The methodology of Technology Entrepreneurship, with Entrepreneurial Health Informatics, is depicted in Figure 2.

Those majoring in computer science or information systems may be fast tracked through the concentration of Technology Entrepreneurship, so that they may finish the courses, especially Entrepreneurial Health Informatics or Entrepreneurship and Financial Computing, sooner than other undergraduate students in the Seidenberg School; and so that they may be more marketable to domain internship prospects at industry ventures. Otherwise, the concept courses of Entrepreneurship and Technology and Customer Relationship Management (CRM) and Entrepreneurship are the prerequisites, as depicted in Figure 2.

5. COURSE MODEL – ENTREPRENEURIAL HEALTH INFORMATICS

The course in Entrepreneurial Health Informatics, in Technology Entrepreneurship, began in 2012, engaging students in diverse or dueling entrepreneurship innovation in a process, product or service - mobile monitoring applications to process re-engineering of systems in the health industry. They are focused on identifying opportunities from problems learned from industry research and on iterating prototyped simulations or solutions for minimal ventures, and may be iterating solutions even in failure, an integral part of problem-solving and risk-taking. The projects for formulating individualized new solutions and subsequent ventures are initiated in small (3-5) student teams, which are mentored by alumni executives from the health industry in New York City. The executives may have furnished the problems to the teams. The students are also mentored by the instructor. The student teams

may shadow a number of the executives in health industry start-up ventures, which may be pursuing solutions similar to the teams. These teams may moreover, in a mission of municipal service, share solutions of systems or tools with non-profit organizations in the industry. The instructor interacts with the student teams through the Blackboard Academic e-Education Suite – Discussion Boards and through the classroom – 4 hours in the classroom labs a week in a semester of 14 weeks.

At the end of the 14th week, the teams present the projects of systems and tools and the proposals of ventures to the instructor, investors invited by the mentors and regional representatives of the health industry; and they reflect on the results of the semester.

The specific learning of Entrepreneurial Health Informatics is depicted in the syllabus in Table 1 of the Appendix.

The text is Gartee, J (2011), Health Information Technology Management; with Christensen, C. (2011), The Innovator's DNA: Mastering the Five Skills of Disruptive Innovation, Hagen, J. (2010), Concepts in Health Care Entrepreneurship, and Timmons, J., and Spinelli (2009), New Venture Creation: Entrepreneurship for the 21st Century, as supporting text, in the current 2012-2013 semesters.

6. IMPLICATIONS

The course in Entrepreneurial Health Informatics is enabling a firmer foundation for the concentration of Technology Entrepreneurship of the Seidenberg School. The course is enabling entrepreneurial and interdisciplinary learning in an industry in which the highest investment in health technology is in this country (Norbeck, 2012) - an industry formulating as proactive in innovation investment in the technology (Horowitz, 2012a). The course is also enhanced by inclusion in Technology Entrepreneurship. Entrepreneurial Health Informatics is furnishing a product that is reacting to the requirements of the industry (Wagner, 2012). Other schools are frequently furnishing students that are not reacting to the skills of the industry. The implication of the paper is that Entrepreneurial Health Informatics is a marketable product to students desiring an interdisciplinary Technology Entrepreneurship program.

Entrepreneurial Health Informatics is enabling an exciting initiative of students to be innovators in

health industry technology. Evaluating opportunities and problems, and furnishing prototypes and solutions (Hyman, 2013), is ideal learning – not passive but proactive problem-solving. Partnering with a health organization and a technology organization on patient problems is an ideal project (Analytics, 2013). In the period from 2011, the bulk of the 36 students in Technology Entrepreneurship (12 in Entrepreneurial Health Informatics) are notably positive on the marketplace potential of even modest projects and on the overall progress in the semesters (Joseph, & Lawler, 2013). They are especially notably positive on mentor relationships (Joseph, & Lawler, 2013) with small start-up ventures (Cortese, 2013), through Entrepreneurs’ Organizations and Startup America Partnership (Max, 2013), and on the potential of recruitment with these ventures, which in the literature is noted as a product of Technology Entrepreneurship programs (Austen, 2013). The implication of the paper is that Entrepreneurial Health Informatics is an appealing proposition to students eager to pursue interdisciplinary potentials of technology.

Lastly, Entrepreneurial Health Informatics is facilitating growth into the Bachelor of Arts in Computer Science program of the school. The future of the health industry (McKenna, 2013) is evident to intelligent undergraduate students in the country intending to leverage the opportunities (Tribune Media Services, 2012) and the promises (Horowitz, 2013) of health technology. Schools of computer science and information systems may integrate the cloud methodologies and platforms of Bachelor of Arts in Computer Science programs (Lawler, & Joseph, 2011) into entrepreneurship and innovation projects, as on the projects of Entrepreneurial Health Informatics of the Seidenberg School. As a result, they may inexpensively invest in more entrepreneurship and innovation projects with cloud technology (Sobel, 2012, and Pratt, 2013). The implication of this paper is that Entrepreneurial Health Informatics, as an example of Technology Entrepreneurship, is a definite proposition to instructors of any university intending to be in tandem with industrial trends.

7. LIMITATIONS AND OPPORTUNITIES

Evaluation of Entrepreneurial Health Informatics as a course of the concentration of Technology Entrepreneurship of the Bachelor of Arts in Computer Science program may not be finished

until late 2013. Evaluation of perceptions and performances, including products, and of recruitments, of the fall 2012 – fall 2013 students may not be fulfilled until a survey then. The limited number of Technology Entrepreneurship students is also a limitation of the paper. The features of Entrepreneurial Health Informatics may nevertheless be helpful immediately to instructors interested in new offerings for STEM students. Students may be pleased with programs increasing marketable skills, so that they may be more than pure technologists.

8. CONCLUSION

The paper presents a course in Entrepreneurial Health Informatics in the concentration of Technology Entrepreneurship at Pace University. The course is one of entrepreneurship innovation projects, from which students of the Seidenberg School of Computer Science and Information Systems of the university are positioned to be business opportunists, not pure technologists. The course provides the creative thinking, problem-solving and risk-taking of business professionals required by not only the health industry but by other industries. The concentration in Technology Entrepreneurship, and the course in Entrepreneurial Health Informatics, further provides interdisciplinary skills to undergraduate students that may not have nor otherwise would have had such skills. Overall, this paper provides schools of computer science and information systems with a proposition that is timely to industrial and societal trends.

9. ACKNOWLEDGEMENTS

The authors of this paper acknowledge funding from the National Science Foundation (NSF) in 2010 – 2013 for the Technology Entrepreneurship program in the Seidenberg School of Computer Science and Information Systems of Pace University in New York City.

10. REFERENCES

- Austen, I (2013). Once BlackBerry Focused, A Campus Widens Its View. *The New York Times*, Business Day, February 4, B1, B4.
- Briley, J. (2013). The Power of Mobile. *National Geographic*, March, 10.

- Brill, S. (2013). Why Medical Bills Are Killing Us. *Time*, Special Report, March 4, 20.
- Byers, T.H., Dorf, R.C., & Nelson, A.J. (2011). *Technology Ventures: From Idea to Enterprise*. McGraw-Hill Publishing, New York, New York.
- Christensen, C. (2011). *The Innovator's DNA: Mastering the Five Skills of Disruptive Innovation*. Harvard Business Review Publishing, Cambridge, Massachusetts.
- Cortese, A. (2013). For Small Business, an Unclear Path to Crowdfunding. *Times Digest*, January 6, 4.
- Cutler, D.M. (2010). Where Are the Health Care Entrepreneurs? The Failure of Organizational Innovation in Health Care. *National Bureau of Economic Research*, 16030, 1.
- De Regge, M., Gemmel, P., Degadt, P., Verhaeghe, R., Sijnave, B., & Duyck, P. (2012). Health Care Redesign: Managing a Changing Health Environment. *Proceedings of the American Health Information Management Association (AHIMA) / EHMA Annual Conference 2012*, 1.
- Deluca, J.M., & Enmark, R. (2002). *The CEO's Guide to Health Care Information Systems*. John Wiley & Sons, Inc., San Francisco, California, 16.
- Ebling, M., & Kannry, J. (2012). Healthcare. *IEEE Pervasive Computing*, 1536(1268), 14.
- Everett, J. (2013). Tap Here for Better Health: Health Apps Are All the Rage. Which Ones Are Worthy of a Download? *Money*, January / February, 51-52.
- Gartee, J. (2011). *Health Information Technology Management*. Pearson. Boston, Massachusetts.
- Greengard, S. (2013). A New Model for Healthcare. *Communications of the ACM*, 56(1), 17-19.
- Hagen, J. (2010). *Concepts in Health Care Entrepreneurship*. Remedy Books, Portland, Oregon.
- Horowitz, B.T. (2012a). AT&T Unveils Cloud Video Patient-Monitoring Service. *eWeek*, December 5, 1-2.
- Horowitz, B.T. (2012b). Innovative Health Care Apps Debut for Personal and Professional Use. *eWeek*, December 21, 1-10.
- Horowitz, B.T. (2013). Health Care Information Technology (IT): Best Practices for Coordinating Care with Digital Tools. *eWeek*, January 30, 1-10.
- Howard, P. (2008). Health Care's New Entrepreneurs. *City Journal*, 18(3), 1,7.
- Hyman, P. (2013). David Patterson's "Big Data" Project Takes Aim at a Cancer Cure *Communications of the ACM*, Member News, 56(1), 19.
- Ireland, R.D., Kuratko, D.F., & Morris, M.H. (2006). A Health Audit for Corporate Entrepreneurship: Innovation at All Levels. *The Journal of Business Strategy*, 1, 1-10.
- Joseph, A., & Lawler, J. (2013). Technology Entrepreneurship in Computer Science – Preliminary Findings and Impacts. *2013 Principal Investigators Conference of the American Association for the Advancement of Science and the National Science Foundation Abstract and Poster*, Washington, D.C., January.
- Kellermann, A.L., & Jones, S.S. (2013). What It Will Take to Achieve the As-Yet-Unfulfilled Promises of Health Information Technology. *Health Affairs*, (32)1, 63-68.
- Khan, S. (2013). What Colleges Could Be Like: Imaging an Optimized Education Model. *Communications of the ACM*, 56(1), 41.
- Lawler, J.P., & Joseph, A. (2010). A Financial Technology Entrepreneurship Program for Computer Science Students. *Proceedings of the Information Systems Educators Conference (ISECON)*, 27(1315), Nashville, Tennessee, 1-7.
- Lawler, J., & Joseph, A. (2011). Cloud Computing as a Core Discipline in a Technology Entrepreneurship Program. *Proceedings of the Information Systems Educators Conference (ISECON)*, 28(1607), Wilmington, North Carolina, 1-13.

- Malugani, M. (2012). Tips for Aspiring Healthcare Entrepreneurs. *All Health Care*, December 20, 1-2.
- Mathews, A.W. (2013). Researchers Mine Data from Clinic, Big Insurer. *The Wall Street Journal*, January 15, B3.
- Milian, M., & MacMillan, D. (2012). Seeing Steve Jobs Everywhere. *Bloomberg Businessweek*, December 17-23, 38,41.
- Max, S. (2013). Helping Start-Ups with Local Support and National Networks. *The New York Times*, February 8, 5.
- McKenna, M. (2012). The New Age of Medical Monitoring: Mobile Phones and Tiny Sensors Are Making It Easier to Quickly Flag Health Trends. *Scientific American*, March, 33.
- McNickle, M. (2013). Five Ways to Improve Healthcare Information Exchanges. *Information Week Health Care*, February 19, 1-4.
- Norbeck, T.B. (2012). Drivers of Health Care Costs: A Physicians Foundation White Paper. *The Physicians Foundation*, November 1, 2,27.
- Phillips, F.S. & Garman, A.N. (2006). Barriers to Entrepreneurship in Healthcare Organizations. *Journal of Health and Health Services Administration*, Spring, 473.
- Pratt, M.K. (2013). Making Time for Innovation: A Healthcare Chief Information Officer (CIO) Moves Aggressively to Cloud Computing. *CIO*, March 1, 22.
- Ratten, V. (2012). A Theoretical Framework of Entrepreneurship and Innovation in Healthcare Organizations
International Journal of Social Entrepreneurship and Innovation, 1(3), 223.
- Richardson, G., & Butler, C. (2006). Readings in Information Technology Project Management. Thomson Course Technology, Boston, Massachusetts.
- Sobel, A.E.K. (2012). The Move Toward Electronic Health Records. *IEEE Computer*, 0018(9162), 23.
- Srinivas, K., Rani, B.K., & Govrdhan, A. (2010). Applications of Data Mining Techniques in Healthcare and Prediction of Heart Attacks. *International Journal on Computer Science and Engineering*, (2)2, 250-255.
- Timmons, J., & Spinelli, S. (2009). New Venture Creation: Entrepreneurship for the 21st Century. McGraw-Hill, New York, New York.
- Urban, J.B., Osgood, N.D., & Mabry, P.L. (2011). Developmental Systems Science: Exploring the Application of Systems Science Methods to Developmental Science Questions. *Research in Human Development*, 8(1), 1-25.
- Velshi, A. (2013). The Internet Is Warming Up: And What Is Next Will Change Your Health Care, Your Shopping, and with Luck Your Commute. *Money*, March, 30.
- Wagner, T. (2012). Creating Innovators: The Making of Young People Who Will Change the World. Scribner, New York, New York, 154.
- _____ (2012a). Fighting Fit: Obamacare Is Inspiring a Horde of Hopeful Entrepreneurs. *The Economist*, December 1, 74.
- _____ (2012b). The Dream of the Medical Tricorder: The Hand-Held Diagnostic Devices Seen on "Star Trek" Are Inspiring a Host of Medical Add-Ons for Smartphones. *The Economist*, Technology Quarterly, December 1, 12-14.
- _____ (2012). New Health Care Technology Will Result in New Opportunities. *Tribune Media Services*, August 9, 1-3.
- _____ (2013). IBM's Watson Transforming Patient Care. *Analytics*, February 18, 2-4.
- _____ (2013). The Great Innovation Debate: Fears That Innovation Is Slowing Are Exaggerated, but Governments Need to Help It Along. *The Economist*, January 12, 11.

Editor's Note:

This paper was selected for inclusion in the journal as an ISECON 2013 Meritorious Paper. The acceptance rate is typically 15% for this category of paper based on blind reviews from six or more peers including three or more former best papers authors who did not submit a paper in 2013.

APPENDIX

Figure 1: Course of Entrepreneurial Health Informatics in a Concentration of Technology Entrepreneurship in a Bachelor of Arts Program in Computer Science: 2012-2013

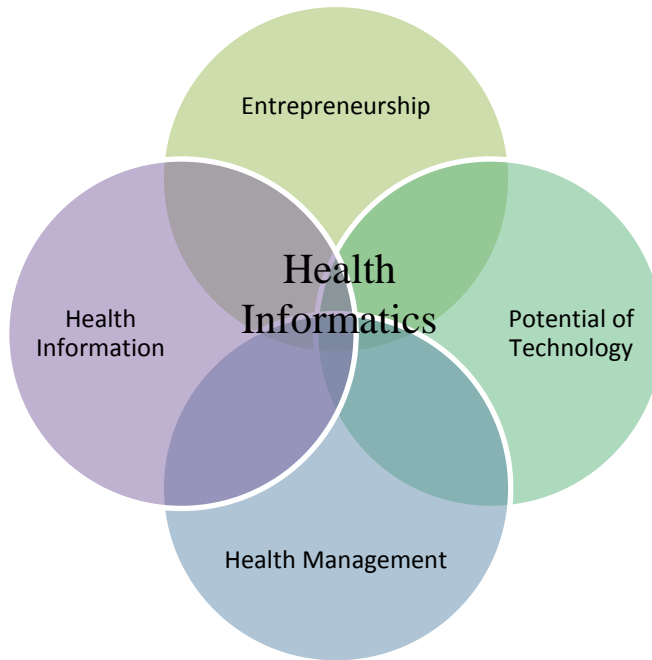
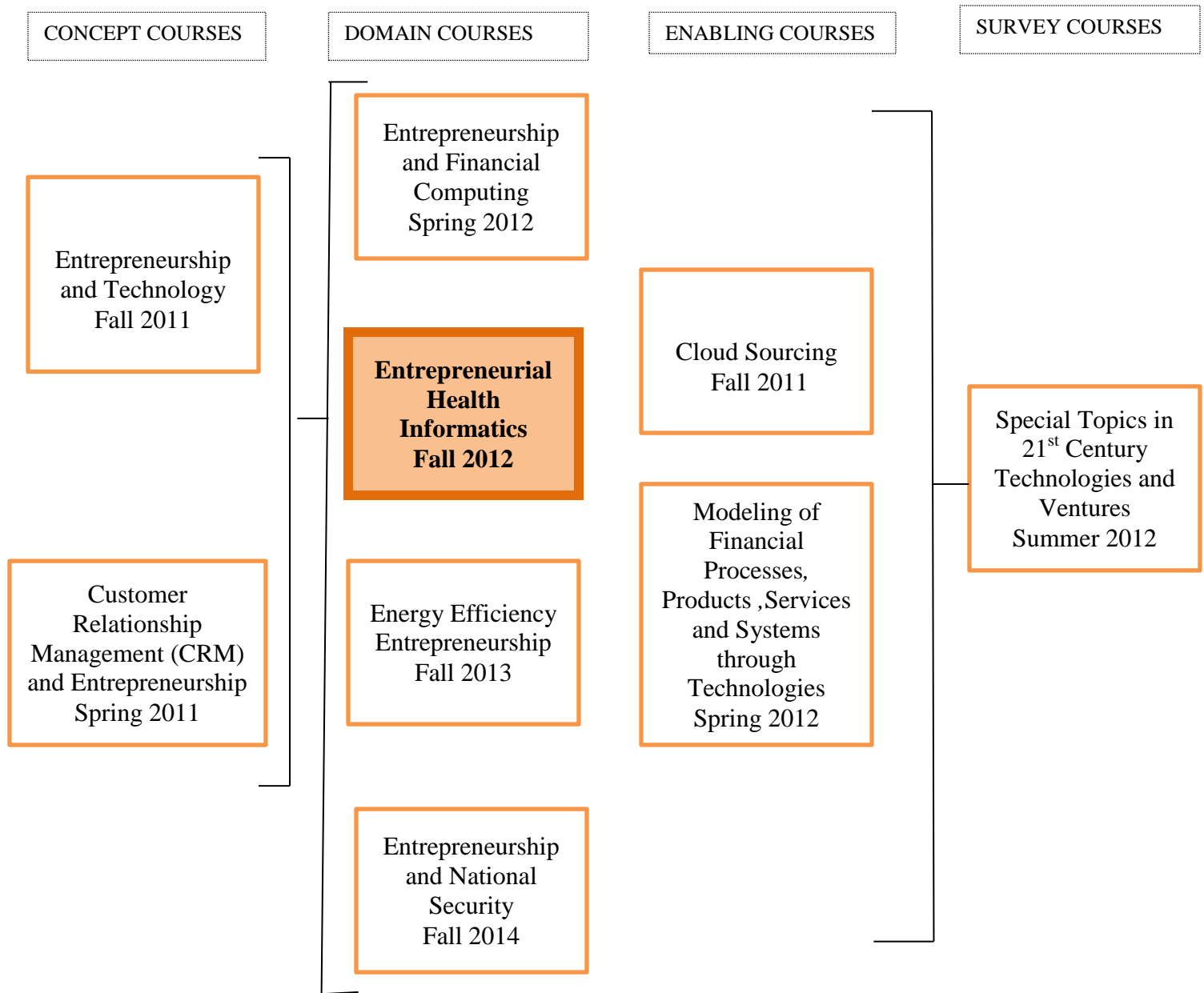


Figure 2: Concentration of Technology Entrepreneurship, with the Course of Entrepreneurial Health Informatics, in a Bachelor of Arts Program in Computer Science: 2011-2015

**Bachelor of Arts in Computer Science
Concentration in Technology Entrepreneurship**



Note: Periods represent inaugural semesters in the Seidenberg School of Computer Science and Information Systems.

Table 1: Course of Entrepreneurial Health Informatics, within the Concentration of Technology Entrepreneurship, in the Bachelor of Arts Program in Computer Science: 2012 – 2013

Semester Week	Topics	Optional Treks
1	Business Entrepreneurial Mind Business Entrepreneurial Process Business Entrepreneurial Strategy Ideas vs. Opportunities Rewards vs. Risks <i>Formation of Project Teams</i> (Random Selection by Students)	
2	Contemporary Health Entrepreneurship in Health Industry Evolution of Health Record Systems Functions of Health Systems Clinical Data Repository Systems Integrated Device Systems Managed Care Systems Medical Management Systems Systems for Telemedicine Health Insurance Portability and Accountability Act (HIPAA) Legislation Obama Health Legislation Organizational Policies and Procedures Privacy, Risk Management and Security <i>Mini-Presentations on Health Industry (by Teams) to Instructor</i>	
3	Entrepreneurship Innovation Ideas vs. Opportunities Innovation Opportunities through Systems Medical Mobile Monitoring Applications (Apps) Mining of Patient Record Systems Process Re-engineering of Service Systems Re-engineering of Self-Service Systems and Tools Research of Systems and Tools Level of Maturity of Industry Technology Review of Standards and Terminologies	Executive Mentor Firm(s)

<p>4</p>	<p><i>Integration of Mentors on Project Teams</i></p> <p>Entrepreneurship Modeling for Health Policy Opportunities</p> <p>Computer Modeling and Decision-Making</p> <p>Economic Efficiency Models Mathematical Models Statistical Models</p> <p>Entrepreneurship Models for Health Policy Problem-Solving</p> <p><i>Preliminary Presentation of Innovation Opportunities (by Teams) to Instructor and Mentors</i></p>	<p>Hospital Organization</p> <p>School of Health Professions of Pace University (Pleasantville, New York)</p>
<p>5</p>	<p>Entrepreneurship Program Management Methodology</p> <p>Analysis and Design Development Integration and Testing Deployment Implementation</p> <p>Systems Science Methodology</p> <p>(*)</p> <p>Prototyping or Simulation vs. Solution Systems</p> <p>Rules for Student Team Playing</p> <p><i>Preliminary Presentations of Innovation Opportunities through Technologies (by Teams) to Instructor</i></p>	
<p>6</p>	<p>Entrepreneurship Scenario – Interdisciplinary Health Project</p> <p>Process, Product or Service Scenario</p> <p>Business Plan</p> <p>Critical Success Factors Marketplace Forces and Opportunities New Project Rationale Scenario of Story Outcomes of Story</p> <p>Financial Plan</p> <p>Funding Plan</p>	<p>Equity Investor Firm</p> <p>Start-Up Venture</p>

	<p>Crowdfunding Equity Investors Private vs. Public Investors Investor Motivations in New Ventures</p> <p>(**)</p> <p><i>Preliminary Presentations of Process, Product or Service Scenarios (by Teams) to Instructor and Mentors</i></p> <p>7 Entrepreneurship Scenario – Interdisciplinary Health Project</p> <p>Process, Product or Service Strategy</p> <p>Objective Definition of New Project Differentiation in Edge of New Project Industry Perspective on New Project Project Scope Strategy</p> <p>(***)</p> <p><i>Preliminary Presentations of Process, Product or Service Strategies (by Teams) to Instructor and Mentors</i></p> <p>8 Entrepreneurship Scenario – Interdisciplinary Health Project</p> <p>Process, Product or Service Prototype or Solution System Technology</p> <p>Project Specifications Prototyping of Stages of System</p> <p>Rapid Application Development (RAD) and Iteration Steps</p> <p>Scenario Technology</p> <p>(***)</p> <p>9 Entrepreneurship Scenario – Interdisciplinary Health Project</p> <p>Process, Product or Service Prototype or Solution System Technology</p> <p>Project Specifications Prototyping of Stages of System</p> <p>Rapid Application Development (RAD) and Iteration Steps</p>	<p>Start-Up Venture</p> <p>Start-Up Venture</p> <p>Start-Up Venture</p>
--	--	---

<p>10</p>	<p>Scenario Technology (***) Entrepreneurship Scenario – Interdisciplinary Health Project Process, Product or Service Prototype or Solution System Technology Project Specifications Prototyping of Stages of System Rapid Application Development (RAD) and Iteration Steps Scenario Technology (***)</p>	<p>– Start-Up Venture</p>
<p>11</p>	<p>Scenario Technology (***) Entrepreneurship Scenario – Interdisciplinary Health Project Process, Product or Service Prototype or Solution System Technology Project Specifications Prototyping of Stages of System Rapid Application Development (RAD) and Iteration Steps Scenario Technology (***)</p>	<p>– Start-Up Venture</p>
<p>12</p>	<p>Scenario Technology (***) Entrepreneurship Scenario – Interdisciplinary Health Project Process, Product or Service Prototype or Solution System Technology Project Specifications Prototyping of Stages of System Rapid Application Development (RAD) and Iteration Steps Scenario Technology (***)</p>	<p>Start-Up Venture</p>
<p>13</p>	<p>Scenario Technology (***) Entrepreneurship with Cloud Technology Infrastructure-as-a-Service (IaaS) Platform-as-a-Service (PaaS)</p>	<p>Equity Investor Firm Non-Profit Organization</p>

14	<p style="text-align: center;">Software-as-a-Service (SaaS)</p> <p>Platforms of Cloud Service Providers (CSP) for Scenario Technologies and Ventures</p> <p><i>Preliminary Presentations of Project (by Teams) to Instructor</i></p> <p>Field Opportunities</p> <p>Bioelectronics, Biotechnology and Genomics Pharmaceuticals Telemedicine</p> <p>Trends in Entrepreneurship Technologies</p> <p><i>Final Presentations of Projects of Systems and Tools and Proposals of Ventures (by Teams) to Instructor and Investors, Mentors and Health Industry Regional Representatives</i></p> <p><i>Reflections on Results of Semester (by Students and Teams)</i></p>	<p style="text-align: center;">Start-Up Venture</p> <p style="text-align: center;">Internship Opportunity – Start-Up Venture</p>
-----------	---	--

(*) Urban, Osgood, & Mabry, 2011

(**) Byers, Dorf, & Nelson, 2011

(***) Richardson, & Butler, 2006

Confronting the Issues of Programming In Information Systems Curricula: The Goal is Success

Jeffrey Babb
jbabb@wtamu.edu
Computer Information and Decision Management
West Texas A&M University
Canyon, TX 79106 USA

Herbert E. Longenecker, Jr.
longeneckerb@gmail.com
School of Computing
University of South Alabama
Mobile, AL 36688 USA

Jeanne Baugh
baugh@rmu.edu
Computer and Information Systems Department
Robert Morris University
Pittsburgh, PA

David Feinstein
dfeinstein@usouthal.edu
School of Computing
University of South Alabama
Mobile, AL 36688 USA

Abstract

Computer programming has been part of Information Systems (IS) curricula since the first model curriculum. It is with programming that computers are instructed how to implement our ideas into reality. Yet, over the last decade numbers of computing undergraduates have significantly declined in North American academic programs. In addition, high failure rates persist in beginning and even advanced programming courses representing losses of students to the anticipated production of future professionals. Perhaps the main reason the current model curriculum in undergraduate information systems education has removed programming is to enable a higher degree of success with higher rates of program completion. Ironically, in the face of this decision, national skills expectations demand programming abilities from graduates of computing programs. Further, most all IS programs business schools require programming, and all ABET-accredited IS programs have multiple courses in programming. While there are challenges in a programming sequence, there is evidence that multiple approaches can be taken to improve the outcomes and perception of success. There is the perception that the problems with this sequence will be improved significantly.

Keywords: programming, class performance, outcome improvement, curriculum, skills achieved

1. INTRODUCTION

Information Systems model curricula - IS'90 (Longenecker & Feinstein, 1991), IS'95 (Couger, et al 1995), IS'97 (Couger, et al 1997), and IS2002 (Gorgone, et al, 2002) - have stated that a principle focus of these curricula was to produce graduates who are competent and confident in developing and deploying Information Systems. These exit-level goals have pervasively guided and shaped the development of these model curricula. The skills necessary to achieve these goals were identified by a survey of faculty and practitioners (Landry et al, 2001), and were reaffirmed by the work of Colvin (2008) based on surveying graduates 3-5 years out of school.

Interestingly, Longenecker, Feinstein, & Babb (2013) have demonstrated that the skills expected of IS practitioners have not changed over this time, despite current trends in industry. In fact, the Department of Labor expectations for IS related Science, Technology, Engineering, and Math (STEM) jobs mirrors the expectations of the earlier curricula (Longenecker, Feinstein & Clark, 2013). Clearly, the marketplace expects and demands that our graduates possess technical skills such as programming and database.

Furthermore, the bulk of business schools with IS programs believe in the necessity for programming and database skills (Apigian & Gambill, 2012); approximately 99% of these schools offer at least one programming course and all offer database. Likewise, the 47 ABET-accredited IS programs offer multiple programming courses as well as database (Feinstein, Longenecker, & Shrestha 2013).

In the IS 2010 model curriculum (Topi et al, 2010), programming was omitted from the list of requirements for an IS degree and relegated as being optional. Given the response of business schools and ABET accredited programs, it can be difficult to understand the reasoning being the omission of programming in the IS 2010 model curriculum. Since the "Dotcom" bust of the early 2000's, the number of students showing up for IS degree programs has decreased significantly; many programs have disappeared. Given that programming is difficult and there is a high degree of failure of students in these classes, it is not surprising that IS 2010

designers withheld programming as a requirement.

Ultimately, as programming is an important endeavor for the discipline, and as there are difficulties in teaching/learning the skill, then effort must be spent in mediation of the difficulties. Some of these methods will be addressed within this paper. We hold that a few things are fairly certain: the need for computing professionals will remain high; that computing is a diverse field where room exists for information systems as a discipline; and, that ABET's Computing Accreditation Commission (CAC) is correct in requiring that all computing disciplines share a core concern in learning about programming. As the spectrum ranges from concerns about the machine up to individual and organizational needs (Shackleford, 2006), programming remains a "lingua franca" as a means for all computing professionals to understand how data and information continue to transform our world.

As we argue for renewed effort for IS educators to remain grounded in the fundamentals of computing by holding fast in our commitment to instruction in computer programming, we make our case as follows. First, we present evidence that programming has appropriately remained at the core of the IS curriculum as a requisite skill for our graduates. We next delineate and explicate what we understand as goals for the programming sequence in an IS curriculum. As this is a paper concerned with achieving success in teaching IS students how to program, we next discuss the various means by which students, educators, and programs fail in the delivery of programming instruction. We address these failures with a discussion of several cases and techniques which have garnered success. We then discuss the issue of achieving success in programming instruction and conclude with thoughts moving forward. Our ultimate aim is that information systems remains among the relevant computing disciplines which will deliver on the need for computing professionals.

2. CURRICULUM AND PROGRAMMING

Information systems curricula have existed for about fifty years (Longenecker, Feinstein & Clark, 2013). Programming and database have always been integral to these programs. With the exception of IS 2010 (Topi, et al, 2010), programming and database have expanded in

their specification. A complete specification of the skills of these curricula including IS'90 (Longenecker & Feinstein, 1991), IS'95 (Couger et al, 1995; Gorgone et al, 1994), IS'97 (Couger et al, 1997; Davis et al, 1997a; Davis et al, 1997b), and IS 2002 (Gorgone et al, 2002a; Gorgone et al, 2002) have been synthesized and is presented in terms of skills specified and expected in IS curricula (see Tables 1 – 5). Recent evidence leaves no reason to suspect that the guidance in these past curriculum models has changed.

The skills presented in model curricula prior to IS 2010 are consistent with the department of labor specifications (see DOL1, 2010; DOL2, 2010; DOL3, 2010; and DOL4, 2010). The applicability of these specifications is reviewed by (Longenecker, Feinstein, & Babb, 2013), and is compatible with the skills lists presented as Tables 1 – 6. Furthermore, Computerworld (Pratt, 2013) stated that 60% of new hires will be hired as programmers; our personal observations corroborate this.

Apigian and Gambill (2010) studied the academic catalog of 240 schools of business and found that 99.17% taught at least one course in programming. A study of ABET accredited programs shows that these programs required at least several courses in programming. Aasheim et al. (2012) also found that there is an expectation of industry that students must know programming.

During the past decade there has been a decrease in the number of students in IS programs. In addition, in some programs as many as 70% of students fail to complete coursework in programming sequences. So the challenge of suggesting that IS programs require multiple courses in programming combined with the difficulty in successfully helping students to survive and thrive is an understandably hard sell. Yet, many jobs and careers which rely on programming are available. There is evidence that legislators are awakening to this reality: "Computer programmers are in great demand by American businesses, across the tech sector, banking, entertainment, you name it. These are some of the highest-paying jobs, but there are not enough graduates to fill these opportunities" (Marco Rubio, Senator, Florida, <http://www.code.org>).

The recommendation for IS curricula (Longenecker, Feinstein, & Babb, 2013)

suggests that three courses in programming with database as a prerequisite to the third course will be necessary for a successful two-course capstone sequence (Reinicke & Janicki, 2010). This capstone sequence will produce desired skills needed by the IS industry.

3. GOALS OF PROGRAMMING SEQUENCE

The goals of the programming sequence are clearest if the goals of the major, degree, and discipline are clear as well. For our purposes, the goal of a program in information systems is to develop professionals who are able to design, develop, implement, manage, maintain, and strategically and tactically use information systems. While we do not purport that each graduate will engage in all of these activities during their professional career, we do assert that such a foundation is optimal. Thus a foundation in analysis and design, data management, and application development are each essential.

Data Management

An information system's ability to consume, produce, and transform data and information is quite fundamental to its existence. Students must be versed in the means by which the computing devices – which constitute the information system – handle data. While this concern typically reduces to the study of relational database management systems (RDBMS), an understanding of data and information must extend into the realm of abstractions (such as the degree to which we synchronize systems analysis and design with Entity-Relationship Diagrams) and into the detailed realm of computing architecture (understanding how computers and operating systems work). Thus, while understanding operating systems, computer hardware, and the implementation details of software are all equally-important data management concerns, RDBMSs remain a central concern for data management as RDBMSs are engineered to handle the myriad concerns of data and information.

Among reasons that we include "database" in the curriculum is that we recognize that understanding the techniques and knowledge associated with RDBMSs remain critical to information systems. Also, instilling within students an awareness of the tight relationship between data and logic is also required.

The topics in Table 7 are critical to an understanding of how an RDBMS works. Many of the features and aspects of an RDBMS underscore the requirement that data management is governed by both business logic and internal integrity logic (hence the presence of "stored procedures" as an extension to many RDBMS product).

Programming

Few would argue that information systems students should study both systems analysis and data management. In fact, perhaps some of the most "management"-leaning information systems programs generally retain these topics. What we hope to demonstrate is that each are not only tied to programming, but both are dependent on programming.

As this paper is about success with teaching information systems students programming – a success we hold as essential and non-optional – we offer a detailed accounting of the knowledge, skills, and competencies essential to facilitating student success. However, we bound this success by framing the material against an overarching goal: that students are prepared to achieve success in a comprehensive, immersive, and applied capstone course which focuses on the design, development, testing, implementation, and management of an information system.

Toward this end, we envision and present Table 8, the sequence of programming topics which lead to success in the capstone course. Table 8 is a list that is reasonably representative of the major milestone concerns for the programming topic for information systems students. It is a list that surely must be broken into multiple course receptacles, which may not be possible for all programs. However, we propose that the topics in the list are requisite for students to be able to successfully enter into a capstone experience as described by Reinicke & Janicki (2010).

Towards Success

While our proffered list of programming knowledge and topics is meant to serve as a viable list, it is debatable if it serves as a minimal (or even optimal) path in support of the capstone course. And, while the capstone is not the sole aim of this paper, the goals and intent of the capstone justifies the extent to which list is useful. Thus, for students and programs alike, we must distinguish between overarching

goals towards ultimate mastery and goals that are achievable.

Students must be made aware of the overarching goals that lead to mastery as such a target is a healthy for our discipline. However, during the short time that students are in our care, we must also remain cognizant of goals that are achievable. It is clear that we can't achieve mastery, in most cases, during the short time that students are in the major. After university and college requirements have been met, many programs face a serious deficit in the number of credit hours that can be used towards the degree, let alone programming.

Programs in information systems will necessarily have to scale their programming sequence to remain consistent with their capstone sequence. Our full list of programming competencies would likely require three courses in programming, and some programs, as currently designed, may not be able to accommodate this.

Getting the Balance Right

Action required for getting over the "hump" that many students experience while learning programming means successfully engaging at least the first 15 steps in Table 8. Regardless of the depth which a program in information systems can handle with respect to available credit hours, it is likely that those first 15 items should/would be covered. It is with these first steps that we should focus on the aspects of mastery that develop automaticity (or muscle memory). We use the term "mastery" in the sense that it is ultimately an inclination towards an endeavor which requires immersion, engagement, and tenacity. Or, simply stated: hard work. Our students will, step-wise and incrementally, engage in the persistent and iterative pursuit of programming (topics 1 - 15) in a manner that benefits from the transformative benefits of mastery (gaining in levels of competence and confidence).

Coaching

Steps 1 through 15 in Table 8 (and programming in general) require the application of effort that is well suited to a coaching methodology. Coach John Wooden, the winner of 10 championship NCAA games utilized a concept of a "pyramid of success" (2005). His attitude of success could easily become a goal for programming education: *"What was under my control was how I prepared myself and our team. I judged my success, my 'winning', on*

that. It just made more sense. I felt if we prepared fully we would do just fine. If we won, great; frosting on the cake. But at no time did I consider winning to be the cake... It's true everywhere in life. Hard work is the difference. Very hard work." (Wooden and Jamison, 1997)

4. KNOWN FAILURE MECHANISMS

The literature on pedagogy, computing pedagogy, and IS pedagogy provides myriad failure mechanisms known to stymie student success. We discuss several categories of these failures here. There are failures related to the mode and delivery from the faculty: lecturing; lack of hands-on experience; lack of student follow-up (in the case of absences in particular). Some failure mechanisms are related to the situation of programming within the curriculum or with the structuring of the course. In particular, a programming course may lack the correct pre-requisites. In other cases, there is a lack of continuity between the courses in the sequence. Another known failure mechanism is the lack of proper teamwork structures that encourage team and peer-driven learning. A last category of failure is that of leadership, motivation, and correction. While students may lack maturity or may encounter issues related to their ability to perceive, receive, and manage failure, we can and must do more than chalk these issues up to being out of our control. The reader may consult Table 9 to see an elaborated set of student failure issues.

Dealing with Lack of Student Maturity

While there is little doubt that students exhibit a paucity of maturity in many regards - failure to come to class; failure to have an attitude of success; immature reaction to our correction; returning a haughty response; lowering standards for everyone - it remains our obligation to sustain leadership and motivation to do our best to correct these behaviors. We are all aware that some students will wait until the last minute to try to complete a programming assignment. They need to be encouraged to begin the design of the solution as soon as the work is assigned. This type of behavior is seen in many college courses, not just a programming course, and is reflective of bad habits. This also shows a lack of maturity on the part of the student. As we are instructing for eventual automaticity in exercise of knowledge and skill in our students, we can also aim for automaticity of students' response such that maturity will become a default response.

Among the possible responses to student immaturity is to address students' transition into college (and the requisite maturing required for success) are "freshman seminar" or "first year experience" programs. Typical goals, learning outcomes, and requirements of such programs will include modules on academic success and responsible behaviors that lead to maturity. As we discuss failures, it is important to note that the imperatives underscored by these programs also extend into the context of teaching programming (and perhaps all of our courses).

Faculty Responsibility for Great Patience

It takes a faculty member with a great deal of patience to teach programming. Some faculty members feel that the student either has the ability to program or they do not. But the truth is that almost everyone can learn to program at some level. However, the faculty member must put in the time to help the students. Sometimes it requires a one-on-one session to review the programming code line-by-line with the student; showing him what he was doing wrong in his code. Coding mistakes made by a student are a great learning tool, however, an instructor must explain those mistakes and help the student to understand why the corrections are needed. It provides an insufficient learning experience to simply mark an assignment wrong without feedback. However, it may be more important to follow-up immediately on an absence from class, particularly one that occurs during the early phase of a course. This is so as the path to mastery requires constant engagement.

5. SUCCESSFUL APPROACHES

There are many different approaches to teaching a programming course. According to Wang (2010), programming requires thinking with abstract concepts, which is difficult for novices. Second, programming includes many different tasks, such as problem solving, algorithm and data structure design, programming language comprehension, testing, and debugging (Wang, 2010).

Table 10 presents a number of approaches that have been utilized to facilitate successful results to learning programming.

Speed of the Course

Of course there are always a few students who will not be able to understand the topics in the course. This can be said of any course in a

college curriculum. The marginal student may need additional actions taken to facilitate their success, such as extra help sessions, personal guidance during office hours, and tutoring sessions offered by the educational institution.

Actually it has been the experience of these authors that some students often want to create additional functionality in their programming assignments beyond what is required. Having some flexibility in the course topics allows the instructor to facilitate both the struggling student and the more advanced one.

Everyone Can Do IT

We all need to adopt a new attitude: everyone can do it. The web site <http://code.org> supports a non-profit organization dedicated to promoting computer science (specifically computer coding) as a requirement for all students. Their vision states that "every student in every school has the opportunity to learn how to code." Additionally, the organization supports the view that "computer science and computer programming should be part of the core curriculum in education, alongside other science, technology, engineering, and mathematics (STEM) courses, such as biology, physics, chemistry and algebra".

Leaders in all phases of industry and education advocate for all students learning to code. At code.org, Bill Gates states that "Learning to write programs stretches your mind, and helps you think better, creates a way of thinking about things that I think is helpful in all domains."

6. DISCUSSION

Programming is a necessary skill. According the United States Bureau of Labor Statics (BLS) website (www.bls.gov), students with a computing degree can expect to earn a starting salary around \$70,000 per year. The BLS lists Computer Science as the highest-paying college degree and forecasts computer jobs as growing at two times the national average (code.org). However, as can be seen in Figure 4, students are not going into the field in sufficient numbers to match the projected need.

Also, according to BLS, Systems Analysts are one of the fastest growing professions (22% growth per year) and have added 120,400 jobs since 2010 with a salary of \$77,000 per year. Analysts require programming skills in a business context. Programmer analysts still are

an entry point to the analyst profession at a salary of \$71,380 per year.

Computer and Information Scientists (see BLS) hold a doctorate degree and invent and design new technologies, and find new uses for existing technologies in business, medicine and other areas, and earn \$100,660 per year.

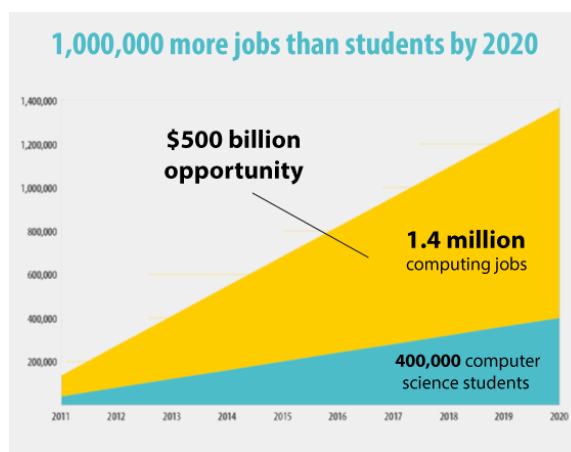


Figure 4 Job projections according to code.org

A programmer's job is to first solve problems. This entails a developing a detailed analysis of the problem, design of the solution (the computer instructions) and finally the test of the ultimate solution. Learning a programming language helps the IS student to understand how the computer actually works and processes instructions, not just how to use a computer. The student learns how to think logically and how to "tell" the computer what to do. As such, there are many different job opportunities for a student with an IS degree, many of which are not in coding. But with some experience in programming, the IS student will better understand things such as:

- what a file is and how it is accessed
- how an algorithm is used to solve a real life problem and how it is coded in the computer world
- being able to understand some basic principles such as variable assignment and conditional branching

Problem-Solving Training

The skill set required for employment in the "real world" is constantly evolving. (Gallivan, Truex, & Kyasny, 2004) Employers want their employees to program as well as communicate and document their work. Programming skills

along with communication and ethical, interpersonal and personal skills go hand in hand for successful employment (Aasheim, 2012). In fact, employers are increasingly demanding these skills of their entry level employees (Gruba & Al-Mahmood, 2004). In this light, the imperative for success in our instruction in programming matters as programming provides, at the very-least an applied and useful means of developing a mental acuity for problem solving that is relevant for our times.

7. CONCLUSION

Many disagree about the importance of programming in an undergraduate IS degree. (Topi, Valacich, Wright, Kaiser, Nunamaker, Sipior, & de Vreede, 2010).

It is the opinion of others (Longenecker, Feinstein, & Babb, 2013), and these authors, that a programming course can be extremely beneficial to the IS student even if the student has no intention of writing code upon graduation. Todd Park U.S. Chief Technology Officer said: "... technology and computers are very much at the core of our economy going forward. To be prepared for the demands of the 21st century—and to take advantage of its opportunities—it is essential that more of our students today learn basic computer programming skills, no matter what field of work they want to pursue." (<http://www.code.org>)

Writing a computer program is an exercise in logic. Since the basics of all computer work centers around these logical foundations, programming is fundamental to everything in computing. The computer can do nothing without a set of instructions (a program) to tell it what to do. A program can be anything from operating system instructions to instructions for a specific application, such as a game or a business process.

So, the issues at hand are straight forward and critically important. On one hand, the impact of programming can be seen daily in almost every aspect of our lives in personal technologic advances, in commerce and banking, and in health care and more; IS development requires great programmers and more of them (Pratt, 2013). On the other hand, teaching students programming remains very difficult. The incorporation of critical thinking, programming logic, technology and syntax must be blended in the process. Teachers of the discipline must not

only be excellent with the skill, they must be attentive to the needs of young and beginning students. As educators begin to understand the coaching skills of John Wooden as applied with the technology of programming, the success of Wooden may translate into our own success. Of course we must define carefully and monitor progress, and make daily adjustments to the plan.

Also highly important is that students need to feel that the instructor is accessible. A study done by Yang and Cornelius (2004) also supports the concept that the students must receive feedback on a timely basis. This is but one part of coaching.

While we call for very high demands, and illustrate very high stakes, at some point faculty will want to question the structure of our professional lives as academics – do we have the incentive and reward for the effort this will require? It is our position that we are dealing with an existential imperative. Either we provide value to the marketplace, or it will forget us. IS educators have a hard enough time to have to describe and explain that we exist, why we exist, and that we play a role and complement among the computing professions. To reverse on fundamental computing skills is to only increase our burden.

Faculty can, and must, be heroes and coaches. There is nothing more rewarding than hearing the student say that he never imagined that he could write a program such as the one he was turning in at the end of a semester. The educational institution must recognize not only how important programming is, but also how time-consuming it is for the instructor who teaches such a course. Successfully teaching programming does require more effort on the instructor's part (Escalante, 2008). A great deal of student follow-up needed for the beginners. Because faculty may spend a lot of extra time helping the first time programmer, it is essential that the institution provide support in areas such as class size, tutoring and course release for extra office hours or help sessions. If the support is not there, the success of any such course cannot be assured (Gopalakrishnan, 2006).

As IS educators, we must also accept that the reward for the extra effort in the instruction of programming will not likely come in the form of financial compensation; the rewards will likely be cerebral and personal in nature. The feeling

of walking out of the classroom knowing that the students "got it" is immeasurable.

The student who has never programmed before feels an enormous sense of accomplishment when the code all comes together and produces correct results. Students are very proud of their work. The authors of this paper have taught programming for many years and from their experience it is clear that programming students show success in both the areas of increased technical skills and personal growth. In addition to the students' positive experiences, watching the transformation of the students from not understanding a simple output statement to writing a programming project encompassing many methods/functions is extremely rewarding to the instructor.

Once a student in a lab asked how the instructor could stand to go from computer to computer helping students with the same programming tasks. The answer to that question is very simple, it is what teachers do. We have little doubt that going these "extra miles" is normative behavior present in many institutions and also present in many who read this article. However, we illustrate the problem ensuring success in the teaching/learning of programming for our students as being profoundly important for the discipline. Maintaining a viable discipline is a concern we should all share. Lastly, the authors of this paper hold that a love and thirst for technology – learning it, teaching it, using it, studying – should be in our blood. "Programming is exciting, stimulating, fun and develops new ways of thinking". (<http://www.code.org>) We, as instructors, are charged with helping to successfully prepare our students for the digital future.

8. REFERENCES

- Aasheim, C., Shropshire, J., Li, L., Kadlec, C. (2012). Knowledge and Skill Requirements for Entry-Level IT Workers: A Longitudinal Study, *Journal of Information Systems Education*, Summer 2012, Vol. 23 Issue 2, p193-204
- Apigian, C.H. and Gambill, S.E. (2010). Are We Teaching the IS2009 Model Curriculum? *Journal of Information Systems Education*, Vol. 21(4), p411-420.
- BLS (Bureau of Labor Statistics) (2013) retrieved July 21, 2013 at <http://www.bls.gov/ooh/computer-and-information-technology/>
- Barki, H. and Hartwick, j. (1989). Rethinking the Concept of User Involvement, *MIS Quarterly*, 13(1), 53-63.
- Baugh, J. M. (2011). Make it Relevant and They Just May Learn it, *ISEDJ* 9(7), December 2011.
- Baugh, J.M., Kohun, F. (2005) Factors that Influence the successful completion of a Doctoral Degree, *IACIS Conference Presentation*, Atlanta GA
- Baugh, J. M., Kovacs, P. (2012). Large Programming Projects for the beginning programmer, *Issues in Information Systems*, 13(1), 85-93.
- Baugh J. M., Davis G., Kovacs, P., Scarpino, J., Wood, D. (2009). Employers And Educators Want Information Systems Graduates To Be Able To Communicate, *Issues in Information Systems*, 10(1), 198-207.
- Choobineh, J., & Lo, A. W. (2004). CABSYYDD: Case-Based System for Database Design. *Journal of Management Information Systems*, 21(3), 281-314
- Courte, J. and Bishop-Clark, C. (2005). Bringing Industry and Educators Together, *Proceedings of the 6th Conference on Information Technology Education*, October 20-22, pp. 175-178.
- Colvin, R. (2008). Information Systems Skills and Career Success, Masters Thesis, University of South Alabama, School of Computer and Information Sciences.
- Cyrillo, M. (2011). Lean UX: Rethink Development, *Information Week*, Nov. 14, 2011.
- Couger, J. D., Davis, G. B., Dologite, D. G., Feinstein, D. L., Gorgone, J. T., Jenkins, M., Kasper, G. M. Little, J. C., Longenecker, H. E. Jr., and Valachic, J. S. (1995). IS'95: Guideline for Undergraduate IS Curriculum, *MIS Quarterly* 19(3), 341-360.
- Couger, J. D., Davis, G.B., Feinstein, D.L., Gorgone, J.T. and Longenecker, H.E. (1997). IS'.97: Model Curriculum and Guidelines for

- Undergraduate Degree Programs in Information Systems, Data Base, 26(1), 1-94.
- Davis, G., J. T. Gorgone, J. D. Couger, D. L. Feinstein, and H. E. Longenecker. (1997). IS'97: Model Curriculum and Guidelines for Undergraduate Degree Programs in Information Systems. ACM SIGMIS Database, 28(1).
- Davis, G.B., Couger, J. D., Feinstein, D.L., Gorgone, J.T. and Longenecker, H.E. "IS '97 Model Curriculum and Guidelines for Undergraduate Degree Programs in Information Systems," ACM, New York, NY and AITP (formerly DPMA), Park Ridge, IL, 1997.
- DOL1 (2010). Summary Report for: 15-1121.00 - Computer Systems Analysts, retrieved from June 1, 2013.
- DOL2 (2010). Summary Report for: 15-1141.00 - Database Administrators, retrieved from <http://www.onetonline.org/link/summary/15-1141.00> June 1, 2013
- DOL3 (2010). Summary Report for: 15-1132.00 - Software Developers, Applications, retrieved from <http://www.onetonline.org/link/summary/15-1132.00> June 1, 2013.
- DOL4 (2010). Summary Report for: 15-1134.00 - Web Developers, retrieved from <http://www.onetonline.org/link/summary/15-1134.00> June 1, 2013.
- Deperlioglu, O., Sarpkaya, Y., & Ergun, E. (2011). Development of a Relational Database for Learning Management Systems. Turkish Online Journal Of Educational Technology - TOJET, 10(4), 107-120.
- Escalente, J. (2008). Jaime Escalente On Being a Teacher, retrieved July 21, 2013, at <http://www.youtube.com/watch?v=FFMz8JRg8Y8>
- Feinstein, D.L., Longenecker, H.E., and Shrestha, D. (2013). A Study of Information Systems Programs Accredited by ABET In Relation to IS 2010, Proceedings of ISECON, San Antonia.
- Foltz , C., Bryan, O'Hara, Margaret T., Wise, Harold, (2004). Standardizing the MIS course: benefits and pitfalls, Campus-Wide Information Systems, 21(4), 163 - 169.
- Gallivan, M., Truex, D., and Kyasny, L., (2004). Changing patterns in IT skill sets 1988-2003: a content analysis of classified advertising, ACM SIGMIS Database, 35(36).
- Gopalakrishnan, A. (2006). Supporting Technology Integration in Adult Education: Critical Issues and Models, Adult Basic Education: An Interdisciplinary Journal for Adult Literacy Educational Planning, 16(1) 39-56.
- Gorgone, John T., J. Daniel Couger, Gordon B. Davis, David L. Feinstein, George Kasper, and Herbert E. Longenecker 1994. "Information Systems '95," DataBase, 25, (4), 5-8.
- Gorgone, J.T., Davis, G.B. Valacich, J., Topi, H., Feinstein, D.L. and Longenecker. H.E. (2003). IS 2002 Model Curriculum and Guidelines for Undergraduate Degree Programs in Information Systems. Data Base 34(1).
- Gruba Paul, Al-Mahmood Reem, (2004). "Strategies for communication skills development", ACE '04: Proceedings of the sixth conference on Australasian computing education - Volume 30
- Kim, Youngbeom, Hsu, J., and Stern, M. (2006). An Update on the IS/TT Skills Gap," Journal of Information Systems Education, 17(4), 395-402.
- Hunton, J.E. and Beeler J.D. (1997). Effects of User Participation in Systems Development: A Longitudinal Field Experiment, MIS Quarterly, 21(4) 359-388.
- Hutchings, P. and A. Wutzdorff, 1988, "Experimental learning across the curriculum: Assumptions and principals." New Directions for teaching and Learning, 35, 5-19
- Kazemian, F., and-T. Howles, 2008. Teaching challenges: Testing and debugging skills for novice programmers. *Software Quality Professional*, 11(1), 5-12.

- Larson, S., Harrington, M.C.R. (2012). A Study of ABET Accredited Information Systems Programs in the USA, *2012 Proceedings of the Information Systems Educators Conference, New Orleans, Louisiana, USA*, p1,18.
- Longenecker, H.E., Feinstein, D.L., Babb, J.S., and Waguespack, L.J. (2013). Is There a Need For a Computer Information Systems Model Curriculum?, *Proceedings of ISECON, San Antonio*.
- Longenecker, H.E., Feinstein, D. L. and Clark, J. (2013). Information Systems Curricula: A Fifty Year Journey, *Information Systems Education Journal (ISEDJ) 11(6) December 2013*.
- Markus, M. L. (1983). Power, politics, and MIS implementation. *Communications of the ACM, 26(6), 430-444*.
- Mathews, J. (2010) Jaime Escalanti dies, inspired 1988 film 'Stand and Deliver', *The Washington Post, March 31, 2010*.
- McCauley, R., S. Fitzgerald, G. Lewandowski, L. Murphy, B. Simon, L. Thomas, and C. Zander. 2008. Debugging: A review of the literature from an educational perspective. *Computer Science Education 18(2), 67-92*.
- Pratt, M.K. (2013). 10 hot IT skills for 2013, *Computerworld*, http://www.computerworld.com/s/article/print/9231486/10_hot_IT_skill. Last Accessed: 07/15/2013.
- Reinicke, B. A. and Janicki, T. N (2010). Increasing Active Learning and End-Client Interactions in the Systems Analysis and Design and Capstone Course, *ISEDJ 8(40) June 30, 2010*.
- Robbert, M. A., Wang, M. Guimaraes, M., Myers, M. (2000). The Database Course: What Must Be Taught, *SIGCSE Bulletin Proceedings of 31st SIGCSE Technical Symposium on Computer Science Education, March*, pp. 403-404
- Schwalbe, K (2010). Information Technology Project Management, Course Technology, Cengage, United States.
- Seyed-Abbassi, B., King, R., & Wiseman, E. (2007). The Development of a Teaching Strategy for Implementing a Real-World Business Project into Database Courses. *Journal Of Information Systems Education, 18(3), 337-343*.
- Shannon, L., Bennett, J.F., and Schneider, S. (2009). Cycle of Poverty in Educational Technology, *ISEDJ 7(71)* retrieved at <http://isedj.org/7/71/>
- Shannon, L., Schneider, S. and Bennett, J.F.. (2010). Critical Think Measurement in ICT, *ISEDJ. 8(1)*, retrieved at <http://isedj.org/8/1/>
- Shannon, L. and Benette (2012). A Case Study: Applying Critical Thinking Skills to Computer Science and Technology, *ISEDJ, 10(4)*, retrieved <http://isedj.org/2010-12/issn:1545-696X>.
- Subramanian, N. and Whitson, G. (2010). Implementing a CNSS 4012 Certification in an Information Systems Curriculum, *Proceedings of the 14th Colloquium for Information Systems Security Education, Baltimore Inner Harbor, Baltimore, Maryland June 7 - 9, 2010*
- Topi, H., Valacich, J., Wright, R.T., Kaiser, K.M., Nunamaker, J.F., Sipior, J.C., and Vreede, G.J. (2010). IS 2010 Curriculum Guidelines for Undergraduate Degree Programs in Information Systems, Association for Computing Machinery (ACM), Association for Information Systems (AIS)", retrieved July 14, 2012: <http://www.acm.org/education/curricula/IS%202010%20ACM%20final.pdf>
- Wang, X. O. P. H. I. E. (2010). Teaching programming skills through learner-centered technical reviews for novice programmers. *Software Quality Professional, 13(1), 22-28*
- White, G. (2012). Visual Basic Programming Impact on Cognitive Style of College Students, *ISEDJ 10(4)* retrieved at <http://isedj.org/10/4>
- Wooden, J. R. and Jamison, J. (1997). Wooden, McGraw Hill, New York.
- Wooden, J.R. and Carty, J. (2005). Coach Wooden's Pyramid of Success, Building

Blocks of Life for a Better Life, Regal Books,
Ventura, California.

Yang, Y, and Cornelius, (2004) "Students'
Perception towards the Quality of Online

Education: A Qualitative Approach",
Association for Educational Communications
and Technology, 27th, Chicago, Il. October
19-23 17 pp

Appendix

Low level data structures	bits, bytes, number representation, money representation, character representation, rounding operations, overflow
Algorithmic Design, Data, Object and File Structures	analysis, design, development, debugging, testing, simple data structures (arrays, records, strings, linked structures, stacks, queues, hash functions). Functions, parameters, control structures, event driven concepts, OO design, encapsulation, classes, inheritance, polymorphism, sorting, searching
Problem Solving-identify problems, systems concepts, creativity	devise questions to help identify problems, apply systems concepts to definition and solutions of problems, formulate creative solutions to simple and complex problems, Fishbone-root cause, SWOT, Simon Model, Triz, ASIT; embracing developing technology; methodologies (waterfall, object, spiral etc.), dataflow, structured
Programming-principles, objects, algorithms, modules, testing	principles, concepts, control structures (sequence, selection, iteration); modularity, objects and ADTs, data structures, algorithmic design, verification and validation, cohesion, coupling, language selection, user interface design, desk checking, debugging, testing, error correction, documentation, installation, integration, operation; writing code in a modern programming language (e.g.,VB.net, Java, C#); interpreted and compiled computer languages; design tools; secure coding principles and practices
Application Development-requirements, specs, developing, HCI considerations	principles, concepts, standards; requirements, specifications, HCI planning, device optimization (e.g. touch screen, voice), development and testing, utilization of IDEs, SDKs, and tool kits; configuration management, installation, module integration; conversion, operation
Web page Development-HTML, page editors, tools	FrontPage, HTML, page building/edit tools, frames; http, Dreamweaver, Photoshop; Sharepoint, Joomla, Drupal, IDEs, SDKs, Snagit, Jing
Web programming-thin client, asp, aspx, ODBC, CGI, E-commerce, web services, scripting	Visual Studio; thin client programming: page design; HTML, *.asp/aspx coding; session variables / page security; ODBC; CGI programming; integration of multi-media; e-commerce models; tools: Java Script, Perl, Visual Studio, Java, Web services, XML server / client side coding, web services, hypertext, n-tier architectures; integration of mobile technology

Table 1. Programming skills expected in model curricula (except IS2010)

Modeling and design, construction, schema tools, DB systems	Data modeling, SQL, construction, tools -top down, bottom up designs; schema development tools; desk-top/enterprise conversions; systems: Access, SQL Server/Oracle/Sybase, data warehousing & mining; scripts, GUI tools; retrieve, manipulate and store data; tables, relationships and views
Triggers, Stored Procedures, Audit Controls: Design / Development	triggers, audit controls, stored procedures, trigger concepts, design, development, testing; audit control concepts/standards, audit control Implementation; SWL, concepts, procedures embedded programming (e.g. C#)
Administration: security, safety, backup, repairs, Replicating	monitoring, safety -security, administration, replication, monitoring, repair, upgrades, backups, mirroring, security, privacy, legal standards, HIPAA; data administration, policies
Metadata: architectures, systems, and administration	definition, principles, practices, role of metadata in database design, repository, dictionaries, creation, ETL, administration, usage, tools
Data Quality: dimensions, assessment, improvement	Data Accuracy, Believability, Relevancy, Resolution, Completeness, Consistency, Timeliness; Data definition quality characteristics, Data model / requirements quality characteristics; Data clean-up of legacy data, Mapping, transforming, cleansing legacy data; Data defect prevention, Data quality employee motivation, Information quality maturity assessment, gap analysis
Database Security	SQL injection attacks and counter measures; encryption; limiting exposure in internet applications; risk management: attacks and countermeasures; Server Security management
Data sources and advanced types	Accessing external data sources; use of search engines; purchasing data; image data; knowledge representations
Database Server	Database server requirements, connecting from application to database, simple rules (no path to internet; local connection only), mounting and updating a database, use of script to enable application security; multi-user connections; replication and backup

Table 2. Database and Data Management

Personal Skills- encouraging, listening, being Organized, principles of motivation	Having integrity, honesty; responsible attitude of personal responsibility; encouraging, listening, negotiating, being persuasive, being organized; Personality types and relationships (DISC, MBTI, COLOR)
Professionalism-self directed, leadership, time management, certification, conferences	being self-directed and proactive, personal goal setting, leadership, time management, being sensitive to organizational culture and policies; personal development (conferences, read literature, use self-development programs)
Professionalism- committing to and completing work	Persistence, committing to and rigorously completing assignments, can-do
Cognition	concepts of learning; sequential levels of learning (recognition, differentiation, use / translation, apply); relationship of learning and emotion
Mathematical Fundamentals	Mathematics (algebra, trigonometry, variables, operations, expressions, logic, probability, limits, statistics)
HCI Principles: underpinnings	Cognitive Process, education learning levels, interface design, concepts of usefulness, the 8 golden rules
Critical Thinking	fact recognition, argument strength, analysis (break into components), synthesis(assembling the components); abstraction; qualitative research principles
Individual behavior	learning styles (visual, auditory, kinesthetic), motor skills, linguistic mechanisms, auditory mechanisms
Communication-oral, written, multimedia, empathetic listening	oral, written, and multimedia techniques; communicating in a variety of settings; empathetic listening, principle centered leadership, alignment technical memos, system documentation, technical requirements; necessity for involvement; development of resistance
Develop Consultant Characteristics	build relationship, identify need, present alternatives, provide assistance as needed, make recommendations, be supportive
Ethics- theory/concepts, setting an ethical example	ethical theory and concepts, codes of ethics--AITP/ACM; setting an ethical example; ethical policies, intellectual property, hacking, identity theft
Learning to learn	journals, learning maps, habits of reading, listening to tape/cd, attending professional seminars, teaching others, meta-thinking, life long learning; human learning: recognition, differentiation, use, application, analysis, synthesis and evaluation
Teams-team building, vision / mission development, synergy building and problem solving; leadership	team building, vision and mission development, planning, synergistic consensus team leadership, leadership development, negotiation, conflict resolution
Collaboration support by IT	IT Solutions for Individuals and Groups, Problem solving mechanisms in support of meetings, consensus development

Impact of IT on Society	IT impact on individuals, on groups, on enterprises, on societies; knowledge work and support by IT; computer industry and society, work force requirements
IT Career Paths	Programmers, Application Developers, Information Analyst, Systems Analysis, Data Management, CIO, CTO

Table 3. Personal, Interpersonal and Organizational Skills

Learning Business Process and Environment	learning business process and environment, exchanges, competitive position, e-business, global concepts, business models, Creating value, Value chain, improving value creation; financial markets, determining value of securities; organizational models
Accounting, Distribution, supply chain management, Finance, HR, Marketing, Production, payroll, inventory processing	accounting (language of money, representations of accounts, reports), distribution (purchasing, supply chain management, distribution systems), finance, human resources (laws, compensation, recruiting, retention, training), marketing (the market, customers and customer satisfaction, market strategies, cycle time and product life cycle; environment scanning), production, international business
Business Problems and Appropriate Technical solutions, end-user solutions	business problems and appropriate technical solutions; quantitative analysis and statistical solutions; decision formulation and decision making; business intelligence systems; business use of spreadsheets, desk-top databases, presentation software, word processing and publishing
Modes of Business	B to B, B to C, C to C, B to G, C to G; organizational span (individual, work group, department, enterprise, inter-organization)
Regulations	Federal and State Regulations; compliance, audits, standards of operation (e.g. FAR); agencies and regulatory bodies
IT Standards	ITIL, CORBA
Business Law	legal system, courts, dispute resolution processes (mediation, arbitration, conciliation, negotiation, trial); types of organizations, contracts, warranties, and product liability; policy and management of intellectual property
Disaster Recovery	identify essential system functions to support business functions for restoration and recovery after a catastrophic failure; define requirements for critical system performance and continuity of business function; backup, replication, fail-over processes in support of system performance subsequent to a disaster

Enterprise Information Systems and Business Intelligence	Alignment of business processes with large system structures; configuration of large systems; implementation and training; integration with business intelligence capabilities and optimization of business procedure.
IT Support for Business Functions	Business systems (budget, personnel, capital, equipment, planning, training, control); Specific systems (production, financial, accounting, marketing, supply chain, securities, taxation, regulation compliance)
Operational Analysis	scheduling, allocation, queuing, constraint theory, inventory management models, financial models, forecasting, real time analysis; linear programming, simulation
Managing the IS Function	Development, deployment, and project control; managing emerging technology; data administration; CIO functions; security management; disaster planning and business continuity planning
Information Center Service	PC Software training and support; application and report generators, IS Development, Development and operations staff; corporate application management, data safety and protection, disaster recovery

Table 4. The Context of Information Systems

Strategic Utilization of Information Technology	use of IT to support business process, integration of customer requirements; team development of systems, reengineering concepts and application, methodologies, interfaces, systems engineering, CRM and ERP concepts; Agile, Object, Lean UX and other methodologies; identification of security issues, incorporation of security concepts into designs ensuring security principles; development of IS policy
IT Planning	value of IT, integration of IT in reengineering, IT policy, end user advocacy and optimization, IT advocacy and alignment outsourcing / off-shoring (risks, benefits, opportunities), training; capture security controls and requirements, ensure integration of security objectives, assurance of people and information protection; ensure security in interface considerations
IT and Organizational Systems	types of systems relationship of business process and IT, user developed systems, use of packaged software, decision systems, social systems; information assurance and security designs; IT support of end-user computing, group process and computing, and enterprise solutions

<p>Information Systems Analysis and Design</p>	<p>investigate, information analysis, group techniques / meetings design, systems engineering, Information architectures, enterprise IS development with strategic process; consideration of alternatives; application and security planning; conversion and testing, HIPAA, FERPA, ISACA, GAAP; requirements analysis. cost analysis, cost/benefit, satisfaction of user need / involvement, development time, adequacy of information assurance controls; consideration / adoption of emerging technology (e.g. mobile computing), consideration of optimal life-cycle methodologies and tools; physical design (database, interface design, reports design, programming, testing, system testing)</p>
<p>Decision Making</p>	<p>personal decision making, Simon's model, structured, unstructured decisions, decision tools, expert systems, advanced problem solving (Triz, Asit); business intelligence, advanced reporting technologies.</p>
<p>Systems Concepts, Use of IT, Customer Service</p>	<p>develop client relationships, understand and meet need, involving the client at all phases of the life-cycle; review of customer functional requirements; consideration of improved business process; assurance of customer needs into requirements analysis</p>
<p>Systems Theory and Quality Concepts</p>	<p>system components, relationships, flows, concepts and application of events and measurement, customer expectations, quality concepts; boundaries, open systems, closed systems, controlled systems; effectiveness, measuring system performance, efficiency</p>
<p>CMMI and Quality Models</p>	<p>quality culture, goals; developing written standards, templates; process metrics development process improvement through assessment, lessons learned</p>
<p>Systems Engineering Techniques</p>	<p>scope development, requirements determination, system design, detailed design and specifications, Enterprise Architecture, System architecture, information architecture, make or buy, RFP/Bid Process verification and requirements tracing, validation planning and test case development, unit testing, integration, system testing, system certification, system acceptance, installation and operation of the system, post-implementation audit; ensuring security designs, secure configuration management; agency evaluation and validation of requirements; ensuring customer training and incorporation of installation teams</p>
<p>End-User Systems</p>	<p>individual software: word processing, spreadsheets, database, presentation, outlining, email clients, statistical packages; work-group software; enterprise software: functional support systems (e.g. PI), enterprise configuration</p>
<p>Enterprise Information Systems in Support of</p>	<p>Systems that support multiple enterprise functions (e.g. SAP); Electronic Medical Record Systems for physician-groups, and for hospitals; Cloud solutions for individual and organizational support; TPS, DPS, MIS, EIS, Expert System</p>

Business Functions	
Emerging Technology	Bleeding edge technologies; testing and adoption of new technologies; cost benefit of new technologies
Systems Roles in Organizations	operations, tactical, strategic
Organizational Models	Hierarchical, Flow Models, Matrix
Metrics and Improvement	Development metrics, quality metrics, metrics in support of 6-Sigma or CMMI, customer satisfaction; Learning Cycles (Understand the problem, plan, act, measure/reflect and learn and repeat the cycle), Lessons Learned (what was supposed to happen, what happened, what was learned, what should be done, communicate the observations)
Hardware selection, acquisition, and installation for project	Determination of capacity for process, storage devices, and communication systems; consideration of alternative hardware; bid preparation, bid evaluation, and final system selection; hardware installation and testing; system deployment and initial operation.
Facilities Management	Physical facility construction, access control, fire protection, prevention of flooding; power management (public utilities, generators--fuel storage, testing, battery management--lightening protection), air conditioning, fire prevention systems, physical security, protection from weather
Maintenance Programming	Fault detection and isolation, code correction, code testing, module testing, program testing; code, module, system documentation
Decision Structure	structured, unstructured decisions, decisions under uncertainty, heuristics, expert systems
Decision Tools	application results, idea generation, Delphi, nominal group, risk analysis, cost benefit analysis
Structured development	process flows, data flows, data stores, process logic, database design, program specifications and design
Object Oriented Development	UML; class diagrams, swim lane, use case, sequence diagram, design patterns
Screen Design	menus, input forms, output forms and reports, linkage of screen modules, navigation

Frameworks and Libraries	object libraries, source libraries, language extensions
Reports Development	simple lists, control break--group by--reports, error reports, exception reports, graphics reports, audit reports
Develop Audit Control Reports	Document new accounts with public information: names, addresses, organizations, items, events
Develop cash audits	deposits, batches, accounting variable controls, accounting distributions
Audit analysis of separation of function	establish roles of staff, validate transactions, validate personal functioning
Audit risk and disaster recovery strategies	determine risks, verify adequacy of mitigations; audit failure processes, replication, and failover mechanisms; audit backup strategy and physical results

Table 5. Organizational Systems Development

Strategic Utilization of Information Technology	use of IT to support business process, integration of customer requirements; team development of systems, reengineering concepts and application, methodologies, interfaces, systems engineering, CRM and ERP concepts; Agile, Object, Lean UX and other methodologies; identification of security issues, incorporation of security concepts into designs ensuring security principles; development of IS policy
IT Planning	value of IT, integration of IT in reengineering, IT policy, end user advocacy and optimization, IT advocacy and alignment outsourcing / off-shoring (risks, benefits, opportunities), training; capture security controls and requirements, ensure integration of security objectives, assurance of people and information protection; ensure security in interface considerations
IT and Organizational Systems	types of systems relationship of business process and IT, user developed systems, use of packaged software, decision systems, social systems; information assurance and security designs; IT support of end-user computing, group process and computing, and enterprise solutions
Information Systems Analysis and Design	investigate, information analysis, group techniques / meetings design, systems engineering, Information architectures, enterprise IS development with strategic process; consideration of alternatives; application and security planning; conversion and testing, HIPAA, FERPA, ISACA, GAAP; requirements analysis. cost analysis, cost/benefit, satisfaction of user need / involvement, development time, adequacy of information assurance controls; consideration / adoption of emerging technology (e.g. mobile computing), consideration of optimal life-cycle methodologies and tools; physical design (database, interface design, reports design, programming, testing, system testing)
Decision Making	personal decision making, Simon's model, structured, unstructured decisions, decision tools, expert systems, advanced problem solving (Triz, Asit); business intelligence, advanced reporting technologies.
Systems Concepts, Use of IT, Customer Service	develop client relationships, understand and meet need, involving the client at all phases of the life-cycle; review of customer functional requirements; consideration of improved business process; assurance of customer needs into requirements analysis
Systems Theory and Quality Concepts	system components, relationships, flows, concepts and application of events and measurement, customer expectations, quality concepts; boundaries, open systems, closed systems, controlled systems; effectiveness, measuring system performance, efficiency
CMMI and Quality Models	quality culture, goals; developing written standards, templates; process metrics development process improvement through assessment, lessons learned
Systems Engineering Techniques	scope development, requirements determination, system design, detailed design and specifications, Enterprise Architecture, System architecture, information architecture, make or buy, RFP/Bid Process verification and requirements tracing, validation planning and test case development, unit testing, integration, system testing, system certification, system acceptance, installation and operation of the system, post-implementation audit; ensuring security designs, secure configuration management; agency evaluation and validation of requirements; ensuring customer training and incorporation of installation teams
End-User Systems	individual software: word processing, spreadsheets, database, presentation, outlining, email clients, statistical packages; work-group software; enterprise software: functional support systems (e.g. PI), enterprise configuration
Enterprise Information Systems in Support of Business Functions	Systems that support multiple enterprise functions (e.g. SAP); Electronic Medical Record Systems for physician-groups, and for hospitals; Cloud solutions for individual and organizational support; TPS, DPS, MIS, EIS, Expert System

Emerging Technology	Bleeding edge technologies; testing and adoption of new technologies; cost benefit of new technologies
Systems Roles in Organizations	operations, tactical, strategic
Organizational Models	Hierarchical, Flow Models, Matrix
Metrics and Improvement	Development metrics, quality metrics, metrics in support of 6-Sigma or CMMI, customer satisfaction; Learning Cycles (Understand the problem, plan, act, measure/reflect and learn and repeat the cycle), Lessons Learned (what was supposed to happen, what happened, what was learned, what should be done, communicate the observations)
Hardware selection, acquisition, and installation for project	Determination of capacity for process, storage devices, and communication systems; consideration of alternative hardware; bid preparation, bid evaluation, and final system selection; hardware installation and testing; system deployment and initial operation.
Facilities Management	Physical facility construction, access control, fire protection, prevention of flooding; power management (public utilities, generators--fuel storage, testing, battery management--lightening protection), air conditioning, fire prevention systems, physical security, protection from weather
Maintenance Programming	Fault detection and isolation, code correction, code testing, module testing, program testing; code, module, system documentation
Decision Structure	structured, unstructured decisions, decisions under uncertainty, heuristics, expert systems
Decision Tools	application results, idea generation, Delphi, nominal group, risk analysis, cost benefit analysis
Structured development	process flows, data flows, data stores, process logic, database design, program specifications and design
Object Oriented Development	UML; class diagrams, swim lane, use case, sequence diagram, design patterns
Screen Design	menus, input forms, output forms and reports, linkage of screen modules, navigation
Frameworks and Libraries	object libraries, source libraries, language extensions
Reports Development	simple lists, control break--group by--reports, error reports, exception reports, graphics reports, audit reports
Develop Audit Control Reports	Document new accounts with public information: names, addresses, organizations, items, events
Develop cash audits	deposits, batches, accounting variable controls, accounting distributions
Audit analysis of separation of function	establish roles of staff, validate transactions, validate personal functioning
Audit risk and disaster recovery strategies	determine risks, verify adequacy of mitigations; audit failure processes, replication, and failover mechanisms; audit backup strategy and physical results

Table 6. Skills in Analysis and Design Course

Topic	Description
1. Data Definition	Data typing and relationship to information
2. Data Modeling	Implementing the requirements
3. Data Schema	Relate "real" objects to data representation
4. Entity-Relationship Diagrams	Specify the relationship between objects and how they may transform each other over time
5. Normalization	How to specify and structure schemas such that the observable relationships between entities can be maintained.
6. Referential Integrity	To ensure that we preserve the logical nature of relationships as dynamicity is introduced to the data store
7. Structured Query Language	The ability to issue instructions and questions to the RDBMS for results
8. Transactions	As the database lives and operates, data integrity is maintained by ensuring that transactions are <i>Atomic, Consistent, Isolated, and Durable</i> .
9. Concurrency Control	The RDBMS must handle multiple transactions and retain the ACID properties of each transaction.
10. Implementing the Data Model	Using DDL to write scripts to build the physical model
11. Ensuring Data Quality	Controlled attributes, data types

Table 7. Data Management Topics in Course

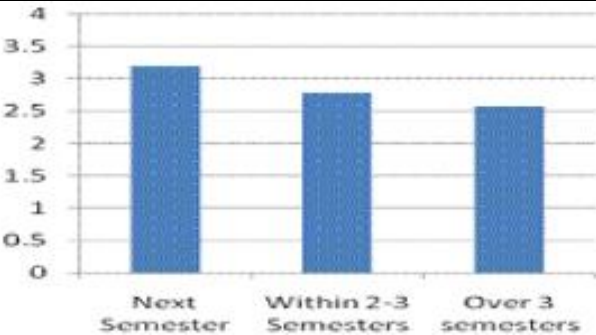
Topic	Description
1. Problem setting and problem solving	This is perhaps the primary competency that programming affords ANY student who undertakes the study of programming. Problem setting is an understanding of the problem domain and an articulation of the problem. Problem solving is the root of algorithmic thinking and the ability to harness the mental acuity that programming affords
2. Data (and Information)	Students must be aware of how data and information can be classified, quantified, notated, and accessed for transaction. Moreover, data transformed in act of problem-solving is often transformed as information.
3. Hardware and Software	Our students must be aware, even if at a cursory level – of the machines to which they communication and collude for computing outcomes.
4. Input and Output	If we accept that computer programs are reducible to Input -> Processing -> Output, then the basics of input and output, as facilitated by whichever programming language is being used, is covered
5. Logic	Students will solve problems, and transact and transform data, using tools that are rooted in logic. Logic is fundamental to problem solving as it prescribed valid reasoning
6. Algorithms	While both logic and algorithms can taken to a deep mathematical space, this will not be necessary for information systems students. Rather, algorithms teach students to express a methodical and reliable expression of their problem-solving.
7. Graphical User Interfaces	As a component of success is retention, our previous literature suggests that captivating and holding attention is key. Therefore, the event-driven Graphical User Interfaces that principally govern students' extant utilization of computing should appear earlier than later. The topic can't be mastered at this stage, but the immediacy and relevance of a GUI can help towards the journey of success.
8. Control Structures / Conditional Logic	This is the essence of programming and a stage where coaching, repetition, and similar antecedents to mastery must be engaged. These essential building blocks must be engaged throughout the period of instruction (at all levels leading up and during the capstone).
9. Debugging	Errors of syntax and semantics will plague both the beginner and professional. Tools, techniques, and strategies for reviewing and correcting these errors are fundamental.

10. Data Structures (Static and Dynamic)	At the very least, both arrays (static) and linear/list (dynamic) data structures are requisite for dealing with data complexities of any "real world" project.
11. Modularity I (Functions/Subs/Methods)	As problems become more complex, students will need techniques and approaches for modularizing and compartmentalizing problems
12. Scope, Promotion, Casting	Now that problems and programs become more complex, handing data across and within modules becomes a concern.
13. Modularity II (OOP)	Object-oriented programming offers a paradigm such that programs can more closely fit the characteristics of the problem space. Makes programming "fit" more closely with data management and systems analysis and design.
14. Error Handling	Complex programs are executed in an equally complex computing environment such that anticipated failures can be addressed with error handling.
15. Data Persistence	The complex computing environment requires that data is persisted beyond runtime. This is where interaction with local and network stores becomes important.
16. Data Connectivity	Programs frequently "converse" with other programs over data communication networks. Students must be aware of how to programmatically utilize appropriate communication protocols (TCP, UDP, FTP, SSH, etc.)
17. Modularity III (Components, Libraries, etc.)	Once a complex set of programs are collaborating in unision, we are truly developing information systems. As this work is performed overtime, the need to both develop and utilize (internal and external) code and binary libraries becomes evident.
18. Data Structures II	Increase complexity also means that data may need to be stored and manipulated in more sophisticated data structures. Even if these data structures are appropriated by way of a vendor-supplied library of these structures (such as C++'s STL or Java/.NET's Collections API) , students will need to understand and use common data access and manipulation algorithms.
19. Generics/Templating	OOP and even data structures and algorithms benefit from generic and template designs.
20. Patterns and Principles	As students prepare to work with a systems development project, the phenomenon of patterns and principles are worth consideration. Decades of practice has wrought various meta, design, and architectural (MVC, etc.), "patterns" that codify "best practice.
21. Multi-threading and Parallel Computing	Programs will operate in a multi-tasking and multi-processing environment. Approaches to facilitating this in programming languages are unique and must be studied

	(lest the students are fooled into thinking this is "magic").
22. GUI Variations (Web/Mobile)	While the web and mobile paradigms can be argued as proposing unique challenges which warrant consideration beyond "GUI variations," we use this term for convenience. Furthermore, library and tool support for developers on these platforms make workflows for making GUI-oriented programs very similar (Visual Studio, XCode, .NET and ASP.NET, Appcelerator Titanium, PhoneGap, Ximian, etc.)
23. Major Web-Oriented Application Development Framework (ASP.NET)	Ultimately, to write comprehensive and contemporary information systems/software solutions, students will need to be versed in a technology stack that provides full-service features for systems development – design, development, libraries, frameworks, testing, implementation, modification, and update. A representative tool for this is Visual Studio and the .NET Framework. For instance, in the web space, ASP.NET and ASP.NET MVC can integrate with Visual Studio, .NET, IIS, and Microsoft SQL Server for a fully-integrated development experience.

Table 8. Programming Topics in Courses

Lack of Student Maturity	There is little doubt that students exhibit a paucity of maturity in many regards - failure to come to class; failure to have an attitude of success; immature reaction to our correction; and returning a haughty response.
Lack of Preparedness from High School	Some of the difficulty in achieving success may be due to high school preparedness (Shannon et al 2012) in critical thinking (Shannon et al, 2009, 2010). In those studies, almost 90% of high school students could not apply critical thinking skills. Basically, students entering into college-level work are not capable of interpreting data, of problem solving, or of proposing solutions.
Failures of Delivery	<p>Lecturing to the student about computer programming is often not a successful method to teach such topics. Students should have hand-on sessions in a computer lab setting to "try" specific programming tasks along with the instructor. A lack of hands-on work with the instructor can lead to frustration on the student's part when he runs into various "strange" computer behavior, such as integer math operations.</p> <p>A student may come to the course unprepared in terms of a background in logical thinking and or mathematics. They could also have a fear of a programming course because they have a fear of math. A good instructor can create good exercises in both logic and math operations to help encourage the student lacking in these skills.</p>
Moving too fast	It is always difficult to decide what the pace of the course should be. There will be some students who will understand the programming concepts better than others. There will also be some who will never understand the concepts. This is the age old question: "what should the pace of the class be"? Perhaps at the beginning of the class meeting, the instructor could announce what concepts are being covered on that particular day. The students who report they have read the section in the book and are comfortable with the topic could be sent to a lab on campus to complete an exercise with that concept. Thus they can work at a faster pace than those who stay to listen to the entire lecture on the concept. One of the authors has successfully used this method with several students each semester. Another related issue is the lack of available tutoring for advanced students. Often there are tutors available for a beginning programming course, but not for an advanced one. Therefore, a great deal of pressure falls on upon the instructor to provide help outside of class to struggling students. Perhaps a cohort or group-based project could help with this issue.
Failures of Curricular Structure	There are a number of factors which might contribute to failure in the introductory programming sequence (Hoskey, et al, 2010). Worse yet, " <i>Numerous studies document high drop-out and failure rates for students in computer programming classes. Studies show that even when some students pass programming classes, they still do not know how to program.</i> " (Hoskey et al 2010):

	<table border="1" data-bbox="662 226 1269 596"> <tr><td>Time Lapse Since Programming 2</td><td>Yes</td></tr> <tr><td>Introductory Programming Language</td><td>Yes</td></tr> <tr><td>Track (Concentration)</td><td>Yes</td></tr> <tr><td>General GPA</td><td>Yes</td></tr> <tr><td>Logic Course</td><td>No</td></tr> <tr><td>Major</td><td>No</td></tr> <tr><td>Faculty</td><td>No</td></tr> <tr><td>Gender</td><td>No</td></tr> <tr><td>Number of Programming Courses Taken</td><td>No</td></tr> <tr><td>Math Courses Taken</td><td>No</td></tr> </table> <p data-bbox="513 600 1276 655">Table from Hoskey et al, 2010 showing variables explored and significance in failure from programming class.</p>	Time Lapse Since Programming 2	Yes	Introductory Programming Language	Yes	Track (Concentration)	Yes	General GPA	Yes	Logic Course	No	Major	No	Faculty	No	Gender	No	Number of Programming Courses Taken	No	Math Courses Taken	No
Time Lapse Since Programming 2	Yes																				
Introductory Programming Language	Yes																				
Track (Concentration)	Yes																				
General GPA	Yes																				
Logic Course	No																				
Major	No																				
Faculty	No																				
Gender	No																				
Number of Programming Courses Taken	No																				
Math Courses Taken	No																				
<p>Delay Between Courses</p>	 <p data-bbox="513 995 1328 1050">Figure from Hoskey et al, 2010 shows a clear decrease in student performance as time is allowed to pass after taking the initial class.</p>																				
<p>Failures of Leadership, Motivation, and Correction</p>	<p data-bbox="513 1050 1419 1192">As is the case with John Wooden’s advice on coaching, our student’s flaws are not entirely our fault or responsibility, but they are our problem. While this position may be anathematic to many, a coaching pedagogy would dictate that we engage all of our students with leadership, maturity, and correction.</p>																				
<p>Choice of Wrong Programming Language</p>	<p data-bbox="513 1226 1399 1369">White (2012) studied the impact on cognition while learning Visual Basic. He found that “... since left hemispheric cognitive style is required to be successful in Visual Basic and Visual Basic does not create such cognitive style, this research, as well as other research, supports the need for prerequisites for Visual Basic to ensure students’ success.”</p>																				
<p>More with Less</p>	<p data-bbox="513 1402 1419 1717">A common mistake, given the length and depth of topics we must teach in the programming sequences, is to attempt to cover the “full menu” of topics. In the interest of positive forward progress and the maintenance of confidence and focus, students can gain the essence of a topic without full grasp. For instance, for an information systems student, can the while-loop be taught as their only tool for repetition structures? For a considerable length of time, this tool will suffice. It may turn out then, that after using the while-loop and developing real craft and mastery with it, the introduction of the for-loop may be trivial and intuitive to the student with a greatly reduced (if not eliminated) period of instruction from the educator.</p>																				
<p>Scope Creep versus Establishing Confidence</p>	<p data-bbox="513 1751 1406 1860">We seek to avoid scope creep (or the “kitchen sink” approach) as our aim is success – we can’t teach them everything. Rather, we advocate for an approach which lets students be good now! We advocate for confidence over competence. We must be very careful to avoid over complicating</p>																				

	presentations lest students fall into a "whooped dog syndrome".
Trying to Put too Much in the Course	We are reminded of this maxim: it takes time to build humans. Do not go faster with the material than the class can handle. As new programming tools are introduced, students should be given assignments and/or lab work with those tools. They need time to incorporate the new programming tools into their own programming knowledge base. The makeup of the students in a programming course will differ each time the course is taught, therefore, what is covered may be slightly different from one semester to another. It will not be possible to cover everything in the text book. The instructor should develop a list of "must have" topics that are required teaching each semester and another list of "add on" topics that could be taught if the class is progressing at a faster pace.

Table 9. Possible Failure Mechanisms in Programming Courses

<p>Everyone Can Succeed</p>	<p>The character in the Movie "Stand and Deliver" portrays is a real-world Jaime Escelente high school Advanced Placement (AP) Calculus teacher. What is remarkable about his story is that ALL of his students PASS the difficult AP exam. What is even more remarkable was that all of the students were fraught with many life-problems. His devotion and coaching style netted 17 years of repeat performance. "He rejected the usual markers of academic excellence and insisted that regardless of a student's GPA, he would let her take the AP course if she promised to work hard." (Mathews, 2010)</p>
<p>Use Tutoring</p>	<p>When especially difficult topics are introduced, such as methods/functions and abstraction through class definition, extra class sessions could help. Tutoring on campus is also something that is extremely beneficial. Good student should be encouraged to sign up to be tutors. We should use these advanced students as experts. Interestingly, it is a wonderful way for the student who is doing the tutoring to increase their own coding skill set. Debugging someone else's code is difficult and helps both students on each side of the tutoring process.</p>
<p>Write Short Sample Programs</p>	<p>Writing very short sample programs to illustrate one specific programming concept can be very helpful to the student. Often the programming books are difficult for the student to read, in that a specific programming concept may be embedded within a complicated example. The student is just trying to see where to put the beginning ending braces for a loop and the book may have an example of a loop running across three pages. Therefore, the sample code can be a valuable resource to the students.</p>
<p>Project Based Learning</p>	<p>Researchers have investigated project-based learning in a wide variety of disciplines and settings. They have generally found project-orientation to be effective in increasing student motivation, improving student problem solving, improving higher-order thinking skills, addressing different learning styles, and providing students with an integrated learning situation (Hutchings & Wutzdorff, 1998).</p> <p>The successful completion of each programming assignment should also include the analysis and design of the problem, data requirements and logic needed to code and test the program. Students should be required to turn in their pseudo code and/or their design with each programming coding effort. Programming can be viewed as a process of building a plan, in the form of source code, to achieve a certain goal (McCauley, Fitzgerald, Lewandowski, Murphy, Simon, Thomas, and Zander, 2008). But Kazemian and Howles found that among the students they surveyed, only 5 percent of the students always developed a design prior to starting to program (Kazemian, Howles, 2008). Therefore, designing a solution before coding should be a requirement for the first time programmer.</p>
<p>Large Coding Projects</p>	<p>In one study it was found that the beginning programmer can create very large projects by focusing on the programming concepts as they are needed for the project (Baugh and Kovach, 2012). In the course studied, students were first given a number of small programs to write that highlighted the basic building blocks of programming; input, output, variables, math operations, selection (if) statements, loops, and other control structures. Then a programming project was initiated that required these concepts, as well as the use of new programming concepts, as each phase of the project was assigned. This method of teaching programming</p>

	<p>showed a great deal of success with 60% of the students receiving an A for the course.</p> <p>Semester coding projects allow the student to see the real value of a computer program and what may be required in the real world. The students also feel a great sense of pride and accomplishment in the creation of a programming project that is many, many lines and pages of code. Susan Wojcicki Senior Vice President, Google said "Learning to code makes kids feel empowered, creative, and confident." (http://www.code.org/quotes) Students who are empowered, creative, and confident are the qualities we expect to witness in those who have successfully completed a large programming project.</p>
Cohort learning	<p>Creating a cohort of students to go through the curriculum together is another approach that has shown success in other areas of IS education. In one study, IS Doctoral students who worked in a cohort for the three years of their study reported that the main reason they felt they were successful was because of the cohort approach (Baugh, Kohun, 2005). The students reported that they were in it "together" and would do whatever they could to insure that all "made it". With a cohort approach, multiple programming courses with continuity from one course to the next would be easily accomplished. Students would be cheerleaders for each other.</p>
Group/Team Projects	<p>Another possible method of teaching programming that may be successful is allowing students to work in groups or teams on a coding project. This approach is often used on lab assignments by these authors. Students working together and helping each other can definitely have great benefits. But for writing code, care must be taken to ensure that everyone is helping with the code and it is not just one person who is taking on most of the work. A way to ensure this is to test the students on all concepts that are required in the specific programming assignment. One instructor has all students explain their semester project code on a final to insure that they actually wrote the code (Baugh, 2009). The student will not be able to adequately explain the code if they did not write it or help to write it.</p>
Make it Relevant to the Student	<p>Making the course material relevant to the student has shown to produce both increased student interest and success (Baugh, 2011). If the course material can be made more interesting to the student, then he will be more inclined to learn it. A real world project allows the students to "learn better through a particular domain of their interest" and "see the practical value of what they learned." (Robbert, 2000)</p> <p>Should an Introductory programming course be taught differently than an doctoral level course? The first answer that one might give to this question is yes, of course. But although the course work is obviously different, the same approach for assignments can be used in almost any IS course. If the course material can be made more interesting to the student, then he will be more inclined to learn it. A real-world project allows the students to "learn better through a particular domain of their interest" and "see the practical value of what they learned" (Robbert, 2000).</p> <p>One instructor designed a way to teach beginning C++ where the students choose their own area of interest, and thus created their own data set. (Baugh, 2011) Students wrote a menu-driven program that was</p>

	<p>broken up into phases. At the completion of the project, each student's project performed the following tasks utilizing the data of the students choice:</p> <ul style="list-style-type: none"> • Read data from data files • Wrote data to data files • Implemented various class structures • Manipulated data in multi-dimensional arrays, including inserting, deleting and modifying • Coded various error checking functions • Coded various search functions • Coded reports • Wrote user's and programmer's guides for the project <p>At the completion of this course, students reported that they were very proud of the large coding project they had written. Again, most of them had no previous programming experience. A number of the students said that they spent a great deal of time on the project, but because of the individualized data, the extra time was something they did not mind. They reported that they felt that the project was more interesting to work on because the data was of interest to them. One student said "without a doubt this was one of the best classes I have ever taken." Even the beginning programmer can write a large project.</p>
<p>Bring in Former Students</p>	<p>Bringing in former students from recent grads to accomplished professionals gives students the opportunity to see how they might be transformed in a few short years. Such speakers might spend time talking about how they made this transition—yes, it took a lot of hard work, but the benefits have become so large and exciting such as the ability to support a family comfortably...!</p>

Table 10. Successful Approach in Teaching Programming

An Active Learning Activity for an IT Ethics Course

David M. Woods
woodsdm2@miamioh.edu
IT Services
Miami University
Oxford, OH 45056, USA

Elizabeth V. Howard
howardev@miamioh.edu
Computer & Information Technology Department
Miami University Regionals
Middletown, OH 45042, USA

Abstract

Courses in Information Technology Ethics are often designed as discussion-intensive courses where case studies are introduced and evaluated using ethical theories. Although many of the case studies directly apply to our students' online lives, the stories can sometimes seem too far removed from their own experiences. While we read the news headlines about data being intercepted via networks, students may not fully understand how easy that data is to intercept. Incorporating a hands-on experience using a network sniffing program allows students to actually experience just how easily someone can intercept their data.

Keywords: IT Ethics, Active Learning, Network Sniffing, Interactive Exercise, Experiential Learning, Wireshark

1. THE COURSE

The topic of ethics is an important part of both the IS and IT curriculum (IS 2010; Information Technology 2008), however, there is little literature discussing active learning activities for an IT Ethics course. This paper extends previous discussions (Howard, 2006; Howard, 2007) of an IT Ethics course to discuss how an active learning activity is used to engage students. The IT Ethics course is designed as a discussion and writing-intensive course. We use traditional ethical theories such as Act Utilitarianism, Rule Utilitarianism, and Kantianism to evaluate scenarios on topics ranging from privacy and intellectual property rights to whistleblowing and vulnerable groups. Students participate in discussions both outside

of class in online discussion forums using the university's learning management system (LMS) and in-class discussions. Part of the student's grade is based on their participation in the discussions. In addition to discussions, students also write a number of position papers centered around the covered topics and create a final group project on the IT ethical topic of their choice.

The literature suggests that one of the challenges of teaching a discussion-intensive course is the underprepared or unengaged student (Benbunan-Fich, 1998; Greening, Kay, & Kummerfeld, 2004; Sanders, 2005). To help address the challenge of students being prepared to actively participate in class discussions, we provide the students with

specific questions about the chapter of the text (Brinkman & Sanders, 2012) and use a subset of those questions for quizzes. Students are required to prepare two (2) full pages of notes for each quiz and the notes are worth 50% of the quiz score. Students then use those notes during the quizzes. The quizzes and quiz notes are intended to encourage the students to read the text and to be prepared for class discussion. This combination of quizzes and notes seems to be an effective method to encourage student preparedness. Students strongly agree that the quizzes and notes help to better prepare them for the in-class discussions (Howard, 2007). To further encourage students to participate in the class discussions, students must cite their classmates within their position papers (Howard, 2006; Sanders, 2005). Although students find citing their classmates in their papers to sometimes be a challenge, the papers are more interesting and substantive than position papers assigned in other courses. As an added benefit, citing classmates has essentially eliminated plagiarism.

2. WIRESHARK ACTIVITY

Throughout the semester, students engage in online and face-to-face discussions on many case studies and scenarios that are the same type of situations that they face in their daily technology use. For example, students read and analyze examples where network traffic has been intercepted but those scenarios sometimes seem too removed from their personal experience (Greenemeier, 2007; McMillan, 2009). In addition, many students take this course for the general education requirement and are not computing majors and they have not yet seen the type of information that is shared in an internet search. The literature suggests that an active learning opportunity would be more effective than merely lecturing about the information that is transmitted (Schweitzer & Brown, 2007; Gao & Hargis, 2010). As Kolb wrote, "Knowledge is continuously derived from and tested out in the experiences of the learner." (Kolb, 1984). To provide an experience where students could actively consider ethical aspects of technology, a hands-on activity involving network packet sniffing was added to the course. This activity involved having each student run the Wireshark network protocol analysis software (www.wireshark.org) to capture network traffic while the student executed a Google search. Students then reviewed the captured network

traffic to explore all of the information sent to Google as part of the search.

This activity provided many technology ethics discussion topics. The first topics came from the need to get permission for the activity from the university's network managers. They approved of the exercise as long as the Wireshark software was not permanently installed on the classroom computers and the exercise was limited to the wired network. Fortunately, the university's switched network design meant that students would not be able to see network traffic from any other computers, which eliminated another potential issue. To avoid the effort needed to install and uninstall Wireshark from the classroom computers, bootable Linux DVDs (<http://networksecuritytoolkit.org/nst/index.html>) were used. Many of the students were unfamiliar with the Linux operating system and it provided us with an opportunity to discuss various operating systems as well as discussions surrounding open source software

Once students had used Wireshark to capture the network data generated by their search, the next discussion topic was how easy it was to see the specific search term, and by extension any text entered in the browser. Students were surprised at how easy it was to see this text and also by all the additional information such as browser and operating system information that was included. Figure 1 (see appendix) shows a snapshot of the information captured by Wireshark. The search term "wireshark," the type of operating system, the browser used, and other data collected are shown in the rectangles. This provided an opportunity to discuss the ethical practices of the university's network managers in restricting the use of network analysis tools and physically securing access to network switches and other hardware.

The discussion was extended by asking students how they would feel if usernames and password for accessing e-mail, online banking, and other sites could be accessed with the same ease. Most students had never considered this, but in light of what they had seen with the Google search were very concerned. This provided an opportunity to discuss the difference between http and https. While use of HTTPS is an obvious feature for banks, for others, there is a potential trade off between the security of using HTTPS and the additional time required to load encrypted pages. Facebook explicitly discusses this tradeoff in their announcement of the ability

to use the HTTPS protocol for interacting with Facebook (Rice, 2011).

Another follow up discussion topic is the difference between wired and wireless networks, and the additional security risks posed by wireless networks. In the Wireshark exercise, students were only able to see their data due to the switched design of the wired campus network. Due to the broadcast/receive nature of wireless networks all data is broadcast to all users. Poor network security can lead to significant risk for companies and individuals (Greenemeier, 2007). An additional ethical consideration with wireless networks is whether it is ethical to access a wireless network without permission. In every class where this exercise has been used, there has been at least one student who has accessed a neighbor's wireless network without asking for permission. This provides an excellent situation for applying the ethical theories used throughout the course.

During the exercise, many students spend time exploring the Wireshark data and bring up other questions. Figure 2 (see appendix) shows two items that students will often ask about - why the search results aren't actually coming from www.google.com (in Figure 2 results come from "ve-in-f104.1e100.net") or about all of the other network traffic that is captured.

Discussing these questions gives students a feeling for the massive complexity of the internet and the need to consider the interactions between components when discussing ethical considerations. For example, including browser and operating system information seems unnecessary for a simple Google search, but once students see the complexity of network traffic, they begin to appreciate how the additional information could be useful to network administrators.

3. STUDENT REACTIONS

Students have a variety of reactions to the Wireshark exercise. Most students seem to enjoy the opportunity for a hands-on exercise (Howard, 2007) and they are completely engaged during the activity. In course evaluations and in written reflections, students often mention how valuable they found the Wireshark activity. Many students, especially those with less technical backgrounds, are very concerned when they see how easy it is to intercept network traffic and how Google search

strings are transmitted in plain text. They often have questions about security of passwords, financial data, and personal information which present an excellent opportunity for a discussion about HTTP and HTTPS protocols and other network security measures.

An interesting outcome of the Wireshark activity is that it prompts many students to think more about the security of their interactions on the web. Many have not previously thought about the security of the data they send across the internet. The discussions during the Wireshark activity provide them with the knowledge needed to think critically about their use of the internet. In our most recent class, one student asked a terrific question, "What are five (5) things that I can do to better protect myself online?" This led to a discussion where other students were able to contribute their own ideas. Ideas included looking for the use of the HTTPS protocol, making use of two-factor authentication where available, keeping anti-virus software updated, using strong passwords, avoiding re-use of passwords, and not doing online banking while connected to public wireless networks. At the end of the class session at least one student typically comments that they plan on reviewing the security practices of web sites they visit for banking.

4. CONCLUSIONS

Including an active learning activity on network sniffing using Wireshark and Linux provided many topics for discussion in our IT Ethics class. Topics such as intercepting network traffic, network security, and open source software suddenly became topics that directly affected the students' lives and not merely stories in an article or case study.

One suggestion that we propose is that instructors be ready to offer tips on safeguarding personal information (Stern, 2013). This would provide information that students could act upon as a result of the Wireshark activity and discussion.

5. REFERENCES

- Benbunan-Fich, R. (1998). Guidelines for using case scenarios to teach computer ethics. *SIGCAS Comput. Soc.* 28, 3 (Sep. 1998), 20-24.

- Brinkman, W. & Sanders, A. (2012). *Ethics in a Computing Culture*. Cengage Learning, Boston.
- Gao, J. & Hargis, J. (2010) Promoting Technology-assisted Active Learning in Computer Science Education. *The Journal of Effective Teaching*, 10(2), 81-93.
- Greenemeier, L. (2007). T.J. Maxx Data Theft Likely Due To Wireless 'Wardriving.' Information Week. Retrieved on June 15, 2013 from <http://www.informationweek.com/tj-maxx-data-theft-likely-due-to-wireles/199500385>
- Greening, T., Kay, J., and Kummerfeld, B. (2004). Integrating ethical content into computing curricula. In Proceedings of the Sixth Conference on Australian Computing Education - Volume 30 (Dunedin, New Zealand). R. Lister and A. Young, Eds. ACM International Conference Proceeding Series, vol. 57. Australian Computer Society, Darlinghurst, Australia, 91-99.
- Howard, E.V. (2006). "Facing the Challenges of Teaching IT Ethics." Proceedings of Special Interest Group in Information Technology Education (SIGITE) 2006, (Minneapolis, MN, October 2006), 95-98.
- Howard, E.V. (2007). "Students Respond to IT Ethics." Proceedings of Special Interest Group in Information Technology Education (SIGITE) 2007, (Destin, FL, October 2007), 219-224.
- Information Technology 2008: Curriculum Guidelines for Undergraduate Degree Programs in Information Technology. Retrieved on Sept. 3, 2013 from <http://www.acm.org/education/curricula/IT2008%20Curriculum.pdf>
- IS 2010: Curriculum Guidelines for Undergraduate Degree Programs in Information Systems. Retrieved on Sept. 3, 2013 from <http://www.acm.org/education/curricula/IS%202010%20ACM%20final.pdf>
- Kolb, D.A. (1984): *Experiential learning: experience as the source of learning and development* Englewood Cliffs, NJ: Prentice Hall, p. 27.
- McMillan, R. (2009). The NSA wiretapping story that nobody wanted. *Network World*. Retrieved on June 15, 2013 from <http://www.networkworld.com/news/2009/071709-the-nsa-wiretapping-story-that.html>.
- Rice, A. (2011). A continued commitment to security. Retrieved on June 15, 2013 from <http://blog.facebook.com/blog.php?post=486790652130>.
- Sanders, A. F. (2005). A discussion format for computer ethics. In Proceedings of the 36th SIGCSE Technical Symposium on Computer Science Education (St. Louis, Missouri, USA, February 23 - 27, 2005). SIGCSE '05. ACM Press, New York, NY, 352-355.
- Schweitzer, D. & Brown, W. (2007). Interactive visualization for the active learning classroom. In Proceedings of the 38th SIGCSE technical symposium on Computer science education (SIGCSE '07). ACM, New York, NY, USA, 208-212. DOI=10.1145/1227310.1227384
- Stern, J. (2013). 10 Tips to Protect Yourself Online. Retrieved on June 15, 2013 from <http://news.yahoo.com/safer-internet-day-10-tips-protect-yourself-232418764--abc-news-topstories.html>.

Appendices

```
Internet Protocol Version 4, Src: probe-plpl.earthlink.net (192.168.1.111), Dst: ve-in-
Transmission Control Protocol, Src Port: 35531 (35531), Dst Port: http (80), Seq: 5066,
Hypertext Transfer Protocol
  [truncated] GET /s?gs_rn=18&gs_ri=psy-ab&cp=9&gs_id=5c&xhr=t&q=wireshark&es_nrs=true&
  [[truncated] Expert Info (Chat/Sequence): GET /s?gs_rn=18&gs_ri=psy-ab&cp=9&gs_id=5c
    Request Method: GET
    Request URI [truncated]: /s?gs_rn=18&gs_ri=psy-ab&cp=9&gs_id=5c&xhr=t&q=wireshark&es
    Request Version: HTTP/1.1
  Host: www.google.com\r\n
  User-Agent: Mozilla/5.0 (X11; Linux i686; rv:20.0) Gecko/20100101 Firefox/20.0\r\n
  Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8\r\n
  Accept-Language: en-US,en;q=0.5\r\n
  Accept-Encoding: gzip, deflate\r\n
  Referer: http://www.google.com/\r\n
  Cookie: PREF=ID=4a019527b394cb57:TM=1372202705:LM=1372202705:S=qld1zlwNtfDC83k9; NID=
  Connection: keep-alive\r\n
  \r\n
  [Full request URI [truncated]: http://www.google.com/s?gs_rn=18&gs_ri=psy-ab&cp=9&gs_

00  c8 d7 19 76 26 db f0 4d a2 7c 89 0b 08 00 45 00  ...v&..M .|...E.
10  03 33 b8 cc 40 00 40 06 c3 b6 c0 a8 01 6f ad c2  .3..è.è. ....o..
20  4b 68 8a cb 00 50 4e 96 c2 94 d3 50 78 9a 80 18  Kh...PN. ...Px...
30  01 4b bc 8c 00 00 01 01 08 0a 00 01 1c 74 69 08  .K..... ....ti.
```

Figure 1. Data collected by Wireshark. The first highlighted box shows the search term used. The second box shows some of the browser and operating system data that is sent as part of the Google search.

136	3.445612	probe-plpl.earthlink.net	googlehosted.l.googleuser	TCP	47019 > http [ACK]
137	3.449432	probe-plpl.earthlink.net	ve-in-f104.1e100.net	HTTP	GET /gen_204?v=3&s=
138	3.479622	192.168.1.1	Broadcast	ARP	Who has 192.168.1.1
139	3.503912	ve-in-f104.1e100.net	probe-plpl.earthlink.net	HTTP	HTTP/1.1 204 No Con
140	3.503942	probe-plpl.earthlink.net	ve-in-f104.1e100.net	TCP	35531 > http [ACK]
141	3.565082	probe-plpl.earthlink.net	ve-in-f104.1e100.net	HTTP	GET /s?gs_rn=18&gs_
142	3.643099	ve-in-f104.1e100.net	probe-plpl.earthlink.net	HTTP	HTTP/1.1 200 OK (a
143	3.643146	probe-plpl.earthlink.net	ve-in-f104.1e100.net	TCP	35531 > http [ACK]
144	3.644766	ve-in-f104.1e100.net	probe-plpl.earthlink.net	HTTP	Continuation or non
145	3.644790	probe-plpl.earthlink.net	ve-in-f104.1e100.net	TCP	35531 > http [ACK]
146	4.160709	probe-plpl.earthlink.net	rns2.earthlink.net	DNS	Standard query 0x40
147	4.160727	probe-plpl.earthlink.net	rns2.earthlink.net	DNS	Standard query 0x2e

Figure 2. Data collected by Wireshark showing that Google.com search results are actually returned from another address (ve-in-f104.1e100.net in this example). This image also shows the additional network traffic taking place at the same time as the Google search.

Swipe In, Tap Out: Advancing Student Entrepreneurship in the CIS Sandbox

Conner Charlebois
charleb_conn@bentley.edu

Nicholas Hentschel
hentsch_nich@bentley.edu

Mark Frydenberg
mfrydenberg@bentley.edu

Bentley University
Computer Information Systems Department
Waltham MA 02452

Abstract

The Computer Information Systems Learning and Technology Sandbox (CIS Sandbox) opened as a collaborative learning lab during the fall 2011 semester at a New England business university. The facility employs 24 student workers, who, in addition to providing core tutoring services, are encouraged to explore new technologies and take on special projects to support or enhance the day-to-day operations of the CIS Sandbox. Doing so creates a culture of entrepreneurship and encourages innovation among the student workers and their peers. This paper follows up on previous results by describing a student-initiated development project to rewrite a card-swipe desktop application that tracks student usage, as an integrated suite of web and mobile apps. The paper presents the architecture of the new system along with perceptions from the student developers around their motivations to contribute to the CIS Sandbox technology infrastructure. Initial use of the prototype suggests it will improve productivity among tutors and provide faculty with easier access to tutoring data.

Keywords: web development, mobile development, computer lab, tutoring, innovation, entrepreneurship

1. INTRODUCTION

The Computer Information Systems Learning and Information Technology Sandbox (CIS Sandbox) has become a valuable campus destination for learning about and with new technology, through the in-person peer tutoring services and online resources it provides. Staffed

by 24 undergraduate and graduate student assistants, the facility has applied startup values to sustain its day-to-day operations and encourage innovation. In addition to providing tutoring to undergraduate and graduate students as a core service, student workers take on administrative responsibilities of managing, promoting, and planning activities for

the facility. On several occasions, they have also identified improvements to processes in place. Previous research describes the results of transforming a computer lab into a collaborative environment (Frydenberg, 2013a) that supports both learning and technology. This redesigned physical space reflects the BYOD (Bring Your Own Device) paradigm of today's connected world, in which most students use their laptops and mobile devices to access the university digital resources and the Internet, and perform computing tasks.

Combining tutoring and technology in the same facility has been effective. Usage reports indicate that the number of students who came to the CIS Sandbox for tutoring increased as the role of the computer lab shifted. The lab was no longer a place where students simply could access technology. The CIS Sandbox has become a campus destination for students to learn to use the technology available to them to achieve their goals while simultaneously evoking further interest in technology.

The culture in the CIS Sandbox fosters a sense of ownership and innovation among student workers who are encouraged to explore new technologies and find ways to integrate them into the day-to-day operations of the facility (Frydenberg, 2013b). The CIS Sandbox relies on a variety of social and digital media tools (Frydenberg, 2013c) as part of its infrastructure.

This paper describes a web and mobile software development project proposed by two undergraduate CIS Majors to improve the process of managing student usage statistics and documenting tutoring cases in the CIS Sandbox. They began this project in their junior year as part of their work in the CIS Sandbox.

2. ENCOURAGING INNOVATION AND ENTREPRENEURSHIP IN THE CIS SANDBOX

The opportunity to learn about web and mobile application development tools enables students to develop marketable skills that the information technology industry seeks (Huang, Kvasny, Josh, Trauth, & Mahar, 2009). Providing students the opportunity to develop real-world software applications has been shown to be a valuable experience to Information Systems students (Abrahams & Singh, 2011; Wong, Pepe, Stahl, & Englander, 2013; Su, Jodis, & Zhang, 2007). Such projects enable students to "practice their content knowledge and workplace skills while

working on authentic, contextualized projects." (Dunlap, 2005). Developing entrepreneurial skills involves giving students opportunities to take initiative and ownership of projects they want to work on. (Bilen, Kisenwether, Rzasa, & Wise, 2005) (Faltin, 2001)

The CIS Sandbox is a collaborative learning facility that supports university courses through tutoring and student exploration of technology through the availability of computing equipment and environments, career presentations, and incentives for innovation. It strives to provide students with learning opportunities that "enabl[e] students to perceive the world, and especially ... technology, as a learning opportunity space. This technology can be used for critical inquiry that allows them to develop as humans and as professionals" (Senges, Seely Brown, & Rheingold, 2008, p. 126)

Inspired by an employment model at Google, (Mediratta & Bick, 2007) student workers who take on expanded roles are given one hour per week apart from their scheduled tutoring responsibilities to work on special projects.

Student workers in the CIS Sandbox have identified and taken on a variety of projects indicative of innovation and entrepreneurship. These include creating instructional video tutorials on using software applications taught in CIS courses, planning extracurricular workshops with industry leaders who speak about career opportunities and technology topics, designing and managing the CIS Sandbox WordPress, blogging on technology topics, promoting the facility using social media, and creating engaging demonstrations for fellow students.

It is worthwhile to note that opportunities for student entrepreneurship can extend to learning centers in other disciplines. Tutors cannot just be students who have mastered the required material; they must be knowledgeable, personable and ambitious. By giving tutors both the opportunity to follow their own passions and access to resources with which they can succeed, the CIS Sandbox has demonstrated a model from which other labs can learn: to have students achieve great things, let them.

3. RECOGNIZING AREAS FOR IMPROVEMENT

One area where student workers saw a need to improve was the way that the CIS Sandbox

gathers usage statistics about its users. When the CIS Sandbox opened in fall 2011, university support staff implemented, as a side project, a Visual Basic desktop application to record usage metrics. Students, upon entering the facility, swipe their ID cards using a magnetic card reader. The swipe application records the date and time of their arrival along with student's name and email address to an Access database. The application obtains the user's identity information by passing the ID number read from the ID card to a student directory database. The swipe application also displays a personalized welcome message to the student, confirming the scanner reads the student's card properly. Students view the acknowledgment, along with photos of tutors on duty, on a front-facing monitor as they enter (Figure 1).



Figure 1. Swiping in at the CIS Sandbox.

As they finish working with students, tutors complete an online tutor form in which they record student's name, class, instructor, and comments about the nature of the help provided. Because tutors easily could hand the tablet to students to enter their names and class information before completing the session summary themselves, most tutors found tablet devices to be conducive in completing this form.

Each week, a CIS Sandbox student worker reviews both the swipe usage data and the tutor form data, to prepare reports informing faculty about students in their classes who come for help and the type of questions they asked, along with overall usage statistics. These reports provide a link between what students are learning in the classroom and what they are learning in the CIS Sandbox. The swipe data shows periods when the facility is at its busiest; these patterns also inform decisions around staffing and scheduling.

Faculty use student usage statistics to design their lessons, see the questions students are having, and recognize students that are "going the extra mile" for their classes. Reports of students who seek assistance and their questions inform the instructors of what students are learning in the lab, and enable faculty to better cater to students' individual needs.

Improving the Swipe System

Obtaining usage data relied on accessing two disparate systems that did not share data electronically. The swipe system lived as a desktop app whose data was only accessible from one computer in the room. The tutor form application existed as a Google Form which required asking students for personal information already known to the swipe system in order to complete.

The reporting capabilities also needed a user-interface overhaul and more automation. As it stood, a CIS Sandbox student worker imported the raw data into Excel manually each week in order to generate charts and reports, and then send them individually to each professor in the department. The process usually took over an hour. This task seemed like it could be automated easily.

The original system was not user-friendly to setup and maintain, often requiring numerous resets during a given shift. Each time the front-end swipe application was started, the application window had to be moved to the front-facing monitor, where a clicking a button on the app would maximize its window on the screen in which it was displayed. Once the app was running, there was no easy way to update the information displayed. For example, in addition to capturing student swipe information, the Welcome screen also displayed pictures of tutors who were currently on duty. These tutors' pictures updated only when the next tutor on duty swiped in, and his or her photo appeared over the previous tutor's photo. Updating tutor pictures and information was no easy task. The process could be simplified by allowing tutors to swipe in as workers (rather than as students), and having the app determine who is currently working by retrieving this information from a database.

Furthermore, tutors were required to carry an additional employee ID card with a different ID number to distinguish between times when they were working and times when they were using

the lab as a student. The original system was unable to differentiate between students who were tutors and students who were not, even though the data allowed making this distinction.

Improving the Tutor Form

The original tutor form was implemented as a Google Form intended to be viewed from a desktop or laptop computer despite the fact that most tutors found it easiest to complete it using a tablet or their own mobile devices. Filling out the form requires the tutor to enter information that he, or the students he tutors might not know (such as student's name, the section number of the classes they are taking), some of which could have been previously recorded from their ID card upon swiping in (such as the student's name).

4. PROPOSING A UNIFIED WEB AND MOBILE SOLUTION

Tutors recognized deficiencies in the current system, but were unable to make any improvements to it because the source code was unavailable. Two student tutors took it upon themselves to create a new, unified web and mobile app that combined the best features of both the original swipe system and the tutor form, and added new capabilities.

The new app would employ a touch interface making it simple for students swiping in, responsive pages which will adapt to the laptops, tablets, or smartphones on which tutors view or complete the tutor form, and a web interface for faculty checking their students' progress. The new version, necessarily rewritten from the ground up because of the lack of access to the original source code, is all open source, and can be easily modified for future expansion and development of new features. The student-developers encourage their successors in the CIS Sandbox to tinker with the source code to see what cool new features they can implement.

This design approach allows the tutors and students to make better use of some of the new technology the CIS Sandbox has to offer, such as an HP all-in-one touch screen computer running Windows 8, and Google Nexus and Asus tablets. It also enables tutors to work effectively from their smart phones.

Student workers identified key areas and new features to improve upon prior functionality in

the new system. These included improved statistics and reporting, a better-integrated tutor form, and a wait queue to inform tutors on duty of students who swiped in and need help, in order to acknowledge them quickly and easily.

Statistics and Reporting

Because the new system records the times when students swipe in and out, as well as the time when the tutor submits the tutor form, deeper insights into how students are using the CIS Sandbox are available. It becomes possible to determine how much time tutors spend tutoring, and gather the most common questions they receive. For example, if students are spending a long time working on an Excel assignment with pivot tables, comments in the tutor forms will reflect these issues, and a faculty member could decide to spend more time going over the assignment in class. The new system also informs tutors how long a student has been waiting for help, and how much time tutors have been spending with individual students. These statistics are invaluable metrics to the CIS Sandbox in determining efficiency and effectiveness of the tutoring staff.

The new system also allows running a report of hours that all tutors worked during the previous pay period, and notifies individual student tutors via an automated, customized email message of the hours they swiped in during the previous pay period. This will reduce errors when students enter their hours worked on the university's time card reporting system in order to get paid.

Tutor Form

The new and improved tutor form requires only one input from the student being tutored: the time at which they take their IT class. Through working with students, tutors found that most students do not know the section numbers of their classes, but they do know when their classes are held. The form pre-populates the student's ID, name, and course tutored from data obtained when the student swipes in and specifies a course number for which they are seeking help. The only work left for the tutor is to enter a brief description of the help provided during the session. This is a marked improvement over the previous system, and allows tutors to spend more time tutoring, and less time performing administrative tasks.

Wait Queue

One of the most troublesome pieces of feedback received from visitors to the CIS Sandbox since

its opening is that the wait times for a tutor can be long, especially at peak times during the semester, or that students do not feel that tutors are seeking them out when they need help. There are usually between one and three tutors on duty at a given time, depending on the time of day and expected usage at that time. The original system had no way to track how long students were waiting, or inform the tutors which students were waiting for help.

To alleviate this, the new system makes a few key changes that shift the responsibility to seek out students needing assistance back to the tutors. Each time a student swipes in using the new system, it adds the student's name and the course for which the student would like help to a wait queue, along with the amount of time that the student has been waiting.

This queue is available to all of the tutors, accessible via a browser on any mobile device or laptop. Monitoring this screen, the tutors can acknowledge those students who need help and have been waiting the longest, allowing them to initiate the contact with students. This feature also demonstrates the power of the web as a development platform allowing the tutors to connect to a central location and view live-updating data simultaneously on their own devices.

5. DEVELOPMENT TECHNOLOGIES AND PROCESS

This application is built on the web, for the web. It makes heavy use of the traditional LAMP stack (Linux, Apache, MySQL, and PHP). Of those main components, Linux is the operating system of choice; the Apache web server handles the serving of the pages, MySQL powers database, and PHP is the server-side scripting language. The application also relies heavily on the Code-Igniter MVC framework for PHP. This framework keeps the application organized and coherent for future maintainers who may enhance it after the current developers graduate. On the front end, the jQuery JavaScript library assists in making numerous AJAX calls, allowing major sections of the app to be generated and updated without refreshing the page. Including the Twitter Bootstrap CSS allows for rapid app development with sleek and intuitive UI elements.

In keeping with the open nature of the CIS Sandbox, the application was developed to be entirely open source. One of the issues with the

original swipe system was that the code was inaccessible. Despite the need for changes, the tutors did not have the access to make them. Now, with all of the code managed with the git version control system and stored on Github¹, all of the tutors will have access to the code, enabling the system to grow and change with the CIS Sandbox.

Application development follows the standard tenets of the software development life cycle (SDLC). The student developers met to discuss the failings of the current swipe app, and ways to meet the needs of the tutors, students, and professors. One of the student developers describes their development process:

"We conducted interviews with current tutors, and talked extensively with the Sandbox director about the shortcomings of the old system and what features and functionality would return the most value to the users. This requirements gathering and legwork gave a solid foundation on which to build the new system. We designed the application based on the requirements generated and our respective technical backgrounds. Development was a series of sprints (features to develop within a given timeframe) with progress tracked in the web-based project management application Trello. At the end of each sprint, the development team would meet to debrief and select the next feature set to build. We had regularly scheduled meetings with the CIS Sandbox director to review our progress and the roadmap for upcoming development. We developed and tested using localized Apache servers while seeking hosting on a university server for production-level testing and deployment. We are also working on a full deployment plan and on how best to document the project for future maintainers."

6. ARCHITECTURE AND DATA FLOW

Appendix 1, Figure 1 shows the architecture and data flow of the Swipe In, Tap Out application.

Swipe In

A user swipes his or her ID card (1) using a magnetic card reader attached to a front-facing touch screen display running a web app displaying the Swipe In page shown in Appendix 1, Figure 2.

The app reads the user's ID from the card, and looks up the student's name from a student ID database (2) provided by the university's department of administrative computing.ⁱⁱ To ensure the security of student IDs, the database stores student ID numbers encrypted with the SHA-1 algorithm (Handschuh, Knudsen, & Robshaw, 2001) which the university's data security officer indicated would provide ample security. The app encrypts the student's ID obtained from the card using same algorithm, and matches that value with the one on file to retrieve the student's name. The student's name displays in a customized Welcome screen shown in Appendix 1, Figure 2.

The Welcome screen asks a user if he or she wants help. If the user taps Yes, the app displays a list of classes offered during the current semester, dynamically obtained by scraping the university registrar's website of course listings, as shown in Appendix 1, Figure 3. The user taps the class for which he or she is coming for help, and the app records this information into its database (3).

If a student is not here for help, the Welcome screen asks if he or she is an employee and signing in to work, in order to track the start or end of a tutor's shift.

Wait Queue Page

Tutors access a Wait Queue page (4) shown in Appendix 1, Figure 4 on their laptops or mobile devices to view the names of students waiting for help. Knowing who is waiting for help puts the onus on the tutors to reach out to students, rather than the other way around. After helping a student, the tutor taps or clicks the Tutor Form button near a student's name shown in Appendix 1, Figure 5. The tutor form loads, pre-populated with information already known. The only information required from the student is the time (or scheduling block) when the student takes the class in order to determine the student's instructor.ⁱⁱⁱ That the tutor form is also responsive so tutors can fill it out from their own mobile devices encourages the BYOD culture in the CIS Sandbox.

Tap Out

When students are ready to leave the CIS Sandbox after tutoring or completing their work, they now will tap out on an iPad mounted on the wall near the door as they exit (5). Appendix 1, Figure 6 shows the Tap Out page, which displays

the student's first name, first initial of the student's last name, and an "I'm gone" button. This allows keeping a real-time list of everyone in the room, while also capturing the exact amount of time that each student was in the room, how long it took them to get the help they needed, and how long they stayed in the CIS Sandbox afterward.

Administrative Pages

Future development calls for administrative pages, where instructors may access a web app to view usage statistics, or detailed information from tutor forms describing tutoring cases about their students.

A CIS Sandbox assistant may access an administrative web app to perform tasks such as generating reports, downloading data, and entering tutor information.

API

Future plans also call for the development of application programming interfaces (APIs) to enable external apps to incorporate data stored in the Swipe App database. For example, efforts are underway to create a WordPress plugin to display on the CIS Sandbox home page the names of tutors who are currently on duty.

7. STUDENT MOTIVATION

Student workers recognize the culture of entrepreneurship that pervades the CIS Sandbox. They are encouraged constantly to suggest new ideas and experiment with new technologies. Because students and tutors use the swipe system every time they work, it was easy to pinpoint exactly what needed to be changed, and how to improve upon the original system.

Said one of the tutors who took on the development of this system:

"We knew that the information systems of the swipe app and tutor form needed to be consolidated, and the entire system needed to be open, allowing easy modifications and for expansion down the road. So we paid close attention to succession planning throughout our planning and analysis process, and began the early stages of development. Once we had a proof of concept, we presented it to the CIS Sandbox Director, who commended our work so far, and encouraged us to continue on our own. He didn't try to steer us one way or another.

His style of 'management by enablement' really is what allowed us to do this. In addition to working on this project when the lab was quiet, he offered to pay us for part of the time we spent on this project outside of working hours, and offered to help us obtain computing resources we needed. It was as if we became the project managers, and he was working for us, offering to support us while not impeding our progress. In doing so, he evoked in us a great feeling of ownership and, ultimately, a sense of accomplishment in what we had built. That feeling of being able to do with this project whatever we wanted was a big factor in our continued efforts."

8. NEXT STEPS AND CONCLUSION

The initial prototype has implemented Steps 1 - 5 of the system's capabilities as described in Appendix 1, Figure 1. Development will continue to provide additional administrative capabilities.

Based on the promise this application has shown, and demonstrations to university administrators, the CIS Sandbox has been approached by several other learning labs on campus about using this system in the future. Although designed to meet the requirements of the CIS Sandbox, the application easily could be modified to suit the needs of facilities such as the Math Lab, the Academic Advising Center, and the Accounting and Economics Lab, each of which host students for tutoring, and could benefit from a student queuing and reporting system.

In the rapidly changing world of technology, there is always room for improvement. The students who created this system have identified potential areas for new development. For example, the app could benefit significantly from the development of a REST backend. Accessing the database through a REST API would enable further modularization of the application, easily separating the swipe system core from the administration backend. It would also allow for a simplification of the data model.

On the front-end, much of the interaction with the user is handled through AJAX requests routed through a controller to call methods in the application's models. This could be simplified significantly with a front-end JavaScript framework like Backbone.js or Angular.^{iv} These

frameworks would add more structure and coherence to the application, and allow it to operate from a single web address and appear much more like a standard desktop app.

The management style in the CIS Sandbox is effective in promoting innovation and exploration of new technologies. Creating the Swipe In, Tap Out app provided a practical development project and real world experience for two student workers who were empowered to recognize ways to improve efficiency in the CIS Sandbox, and build a system to accomplish their goals.

9. REFERENCES

- Abrahams, A. S., & Singh, T. (2011). A 'Rainmaker' Process for Developing Internet-based Retail Businesses. *Information Systems Education Journal* , 9 (2), 14-26.
- Bilen, S., Kisenwether, E., Rzasas, S., & Wise, J. (2005, April). Developing and Assessing Students' Entrepreneurial Skills and Mind-Set. *Journal of Engineering Education* , 233-243.
- Dunlap, J. C. (2005). Problem-Based Learning and Self-Efficacy: How a Capstone Course Prepares Students for a Profession. *Educational Technology Research and Development* , 53 (1), 65-85.
- Faltin, G. (2001). Creating a culture of innovative entrepreneurship. *Journal of International Business and Economy* , 2, 123.
- Frydenberg, M. (2013a). Creating a collaborative learning community in the CIS Sandbox. *Interactive Technology and Smart Education* , 10 (1), 49-62.
- Frydenberg, M. (2013b). Fostering Entrepreneurship in the CIS Sandbox. *Information Systems Education Journal* , 11 (3), 35-41.
- Frydenberg, M. (2013c). Aligning Open, Physical, and Virtual Spaces in the CIS Sandbox. In A. R. Tatnall, *IFIP Open & Social Technologies for Networked Learning* (pp. 121-130). New York: Springer.
- Handschuh, H., Knudsen, L., & Robshaw, M. (2001). Analysis of SHA-1 in Encryption

Mode. In *Topics in Cryptology: CTRSA 2001* (pp. 70-73). Springer Berlin Heidelberg.

Huang, H., Kvasny, L., Josh, K. D., Trauth, E. M., & Mahar, J. (2009). Synthesizing IT Job Skills Identified in Academic Studies, Practitioner Publications, and Job Ads. *SIGMIS CPR '09: Proceedings of the special interest group on management information system's 47th annual conference on Computer personnel research* (pp. 121-128). New York: Association for Computing Machinery.

Janz, K., & Owen, S. (2004). Organizationally Supporting Innovation in Technology Enhanced Instruction and Research. *Proceedings of SIGUCCS '04* (pp. 202-208). Baltimore: Association for Computing Machinery.

Lennon, R. G. (2012). Bring your own device (BYOD) with Cloud 4 education. *SPLASH '12 Proceedings of the 3rd annual conference on Systems, programming, and applications: software for humanity* (pp. 171-180). New York: Association for Computing Machinery.

Mediratta, B., & Bick, J. (2007, October 21). *The Google Way: Give Engineers Room*. Retrieved June 1, 2012, from The New York

Times:

<http://www.nytimes.com/2007/10/21/jobs/21pre.html>

Senges, M., Seely Brown, J., & Rheingold, H. (2008). Entrepreneurial learning in the networked age: How new learning environments foster entrepreneurship and innovation. *Paradigmes Journal of the Catalan Ministry of Innovation, Universities and Enterprise*, 1 (1), 125-140.

Su, H., Jodis, S., & Zhang, H. (2007). Providing an integrated software development environment for undergraduate software engineering courses. *Journal of Computing Sciences in Colleges*, 23 (2), 143-149.

Wong, W., Pepe, J., Stahl, J., & Englander, I. (2013). A Collaborative Capstone to Develop a Mobile Hospital Clinic Application Through a Student Team Competition. *Information Systems Education Journal*, 11 (4), 39-50.

Editor's Note:

This paper was selected for inclusion in the journal as an ISECON 2013 Meritorious Paper. The acceptance rate is typically 15% for this category of paper based on blind reviews from six or more peers including three or more former best papers authors who did not submit a paper in 2013.

Appendix I

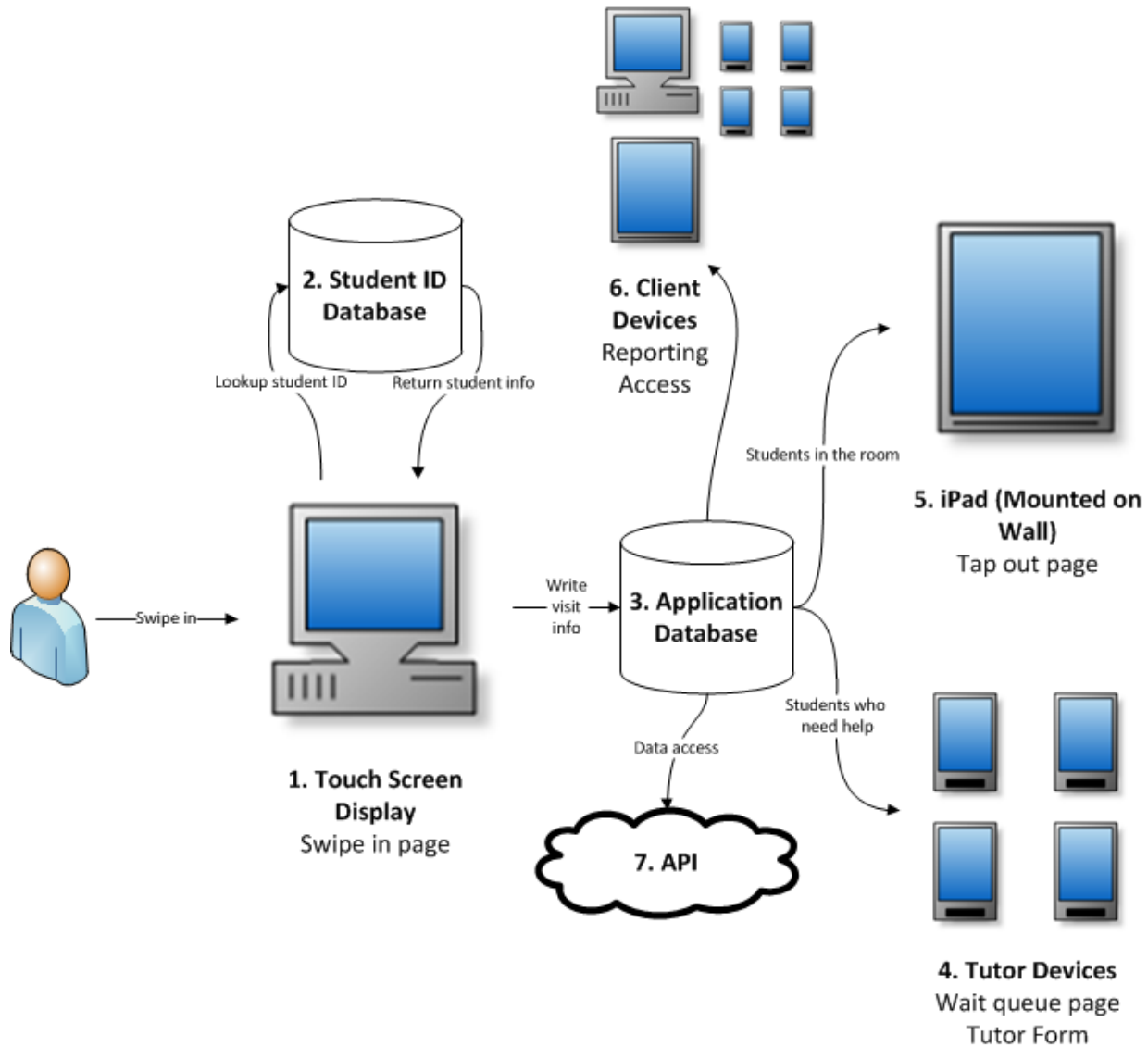


Figure 1. Architecture and data flow of the Swipe In / Tap Out Application.



Figure 2. The Welcome screen.

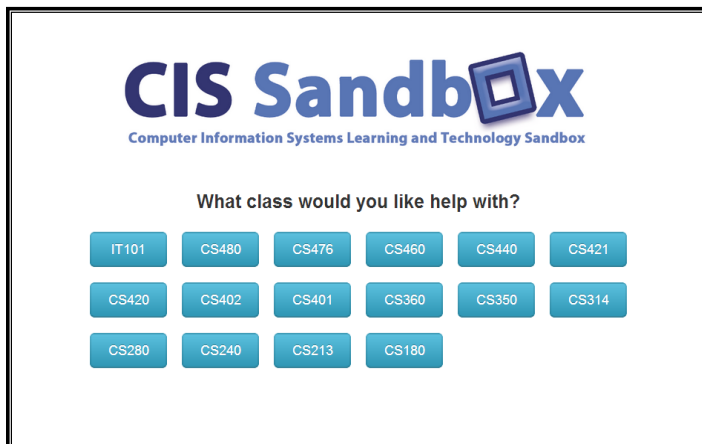


Figure 3. The *What would you like help with?* screen.

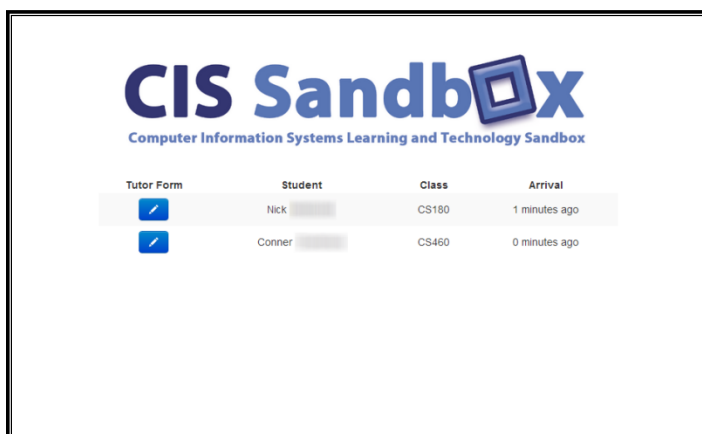


Figure 4. The *Wait Queue* Page.



The image shows a web browser window displaying the CIS Sandbox logo at the top, which includes the text "CIS Sandb" followed by a blue square icon with a white 'X' and the full name "Computer Information Systems Learning and Technology Sandbox". Below the logo is a form titled "The Tutor Form". The form contains several input fields: "Student" with the value "12345678" and a "Nick" dropdown menu; "Course" with the value "CS180"; "Section" with a dropdown menu showing "TR 5:00"; and "Tutor" with a dropdown menu showing "Conner". There is also a "Notes" section with a large empty text area and a "Submit" button at the bottom.

Figure 5. The Tutor Form, populated with already-available information.



Figure 6. The Tap Out screen as displayed on an iPad mounted by the door.

Endnotes

ⁱ The source code is maintained at <https://github.com/cisSandbox/sandbox-swipe-system>.

ⁱⁱ It would have been more elegant if the student developers could have access to an API from the administrative computing department's student database, but this was not available. The compromise was to have Administrative Computing automatically push a flat file containing updated encrypted student ID numbers to the server hosting this app on a monthly basis. When received, the app updates its Student ID database.

ⁱⁱⁱ A current limitation is that the app cannot handle the case if two different sections of the same course are scheduled to meet in the same block. This occurrence is rare at the University, so this disambiguation capability is not included in the prototype.

^{iv} Angular and Backbone.js are two examples of front-end JavaScript frameworks. Information about each is available at <http://angularjs.org/> and <http://backbonejs.org/> respectively.