Education at the Seams: Preliminary Evaluation of Teaching Integration as a Key to Education in Information Technology

Joseph J. Ekstrom, Barry Lunt, C. Richard Helps
Brigham Young University

Abstract
Information Technology (IT) is widely considered to be an integrative discipline. Many four-year IT programs accept programming, networking, web systems, databases and human-computer interaction as core topics in IT. Active discussion continues as to the best way to teach and sequence these topics. We have proposed and begun to implement a curriculum that reflects a change in orientation from focusing on the technologies to focusing on the interfaces between technologies. We believe that this approach is fundamental to IT as an academic discipline. Students receive a broad introduction to computer and communication technologies combined with an in-depth exploration of the interactions between key technologies. This enables IT students to stitch systems together with understandable, manageable and deployable seams. We report on early results of applying this curricular approach within BYU's four-year IT program.

Introduction
There is an emerging consensus among Information Technology programs that the core of an IT curriculum consists of Programming, Database, Web Technologies, Networking and Human Computer Interfacing [1]. The initial curriculum for Information Technology at BYU took the approach of including topics from Electronics Engineering Technology, Computer Science and Computer Engineering in a traditional topic-oriented approach. Other IT programs have been following a similar track [1]. During the last three years we have observed several problems in attempting to implement IT courses by tailoring courses from related disciplines to the requirements of an IT curriculum. We reported on these experiences in our networking course development at ASEE 2002[2]. We discussed similar issues for web systems and database courses at CITC III[3] and CIEC 2003[4]. At CITC IV[5] we proposed focusing on the interfaces between technologies as an overriding philosophy that should guide Information Technology curriculum design in relation to sister disciplines. This paper is a follow-up to the CITC IV work that reports on the first year of implementing the curriculum changes that were proposed. In addition, we discuss the issues that have arisen and additional changes that we are implementing to address them.

Initially we thought of IT as a "breadth" rather than "depth" coverage of topics from Computer Engineering, Computer Science, with some ideas from other disciplines and an applications orientation. However, we have come to understand that IT students require depth, but not depth on how to implement technology components. IT students require deep knowledge of the interfaces between technologies. This insight has significant

implications for IT curriculum. Students require a working knowledge of the technologies that are to be integrated, plus deep understanding of the interactions between the technologies involved. This deep understanding of the intricacies of integration should be a major focus of IT as a discipline. Initially, IT students need a broad introduction to computer and communication technologies followed by in-depth treatment of their interactions. This requirement for a broad introduction has necessitated reorganization of the original BYU curriculum to include overview courses, one at the freshman and one and the sophomore level. These courses provide context and motivation for the entire curriculum. One of their primary goals is to provide a hands-on introduction to IT that creates a map of the concepts central to IT and where in the curriculum the student can expect to learn the details. There is an additional benefit that students find no surprises when they reach the upper division core. They have already been introduced to the concepts and expectations have been set correctly.

In this paper we first summarize the current status of IT curriculum development and accreditation efforts; we then present a brief history of the BYU experience in creating an IT program. Next we present the conceptual framework that motivated the curriculum changes and show how this conceptual orientation helps us understand several of the problems we observed. We then discuss the experience of the last year as we implemented the curriculum changes. In conclusion we discuss the implications of the ideas presented along with outstanding issues. We freely admit that this work is preliminary and that much work remains to be done. Results can only be preliminary when a course has been taught only once. We present these ideas with the hope that it will help other emerging IT programs analyze the issues they are facing and also to elicit feedback from the academy.

Status of IT Curriculum and Accreditation

IT has emerged as a discipline separate from Computer Science, Electrical and Computer Engineering, and Information Science. April of 2002 saw the second Conference on Information Technology Curriculum (CITC-2), attended by representatives of 35 universities with 4-year programs in Information Technology (IT), as well as representatives of related professional societies (IEEE Computer Society; ACSE) and accreditation (the Technology Accreditation Commission and the Computer Accreditation Commission of ABET, the Accreditation Board for Engineering and Technology). The CITC-2 conference gave birth to SITE, the Society for Information Technology Education, which in 2003 was organized as a special-interest group (SIG) of ACM (Association for Computing Machinery), and is known as SIGITE. Membership in SIGITE comes from over 30 institutions of higher education and from professional and accreditation bodies. Membership in SIGITE is open to all interested parties; efforts are being made to invite all who would be interested.

The main efforts of SITE have been to define the academic discipline of IT, which has involved work on IT curriculum and IT accreditation. Efforts to this point have resulted in draft documents defining IT curriculum and outlining the requirements for IT accreditation. In the most recent CITC (October 2003), both documents were discussed. The accreditation document was approved with minor changes.[6] The curriculum

document was approved and a writing committee was formed to write the formal document defining IT curriculum. The model for this document is the CS volume of the CC2001 document. [7]

Most SIGITE members have indicated close agreement with both documents; only relatively minor changes have recently been made. Additionally, most SIGITE members have indicated that their academic institution fully intends to seek accreditation when it becomes available. In summary, the strong national movement to define the 4-year academic discipline of Information Technology, has made significant progress. SIGITE members have published several papers, given presentations at national conferences, and participated on national panels to publicize the efforts. [8] Additionally, a current report on the curriculum and accreditation progress is being presented in a paper at this conference. [1]

Creating a Curriculum: an Evolving Consensus

The Information Technology program at BYU began officially in fall 2001 with a faculty consisting of:

1. Two Electronics Engineering Technology professors that were instrumental in the evolution of the existing EET program at BYU into an IT program,
2. One Electrical Engineering, Ph.D. newly arrived from the aerospace industry.
3. One computer scientist instructor who had done part time teaching and had been part of the department for 1 year with several years in system development in health care.
4. One computer science Ph.D. with recent executive management responsibilities in network hardware and service provider businesses.
5. One former department chair of the technical education program for secondary schools.

This is very diverse group of people, each of whom joined the department because they thought that the existing computer programs at BYU did not prepare students for the practical aspects of system delivery to customers. We are evenly divided between long-term academics and recent retreads from industry. However, the academics have also each had significant industrial experience which provided the motivation for them to accept positions in the new IT program.

As we worked on the course descriptions, each of us had pet topics that could be justified as valid in Information Technology, but that didn't fit in the time available and/or weren't "core" to the discipline. Our Ph.D. computer scientist loves to teach compiler theory and implementation, but it didn't really appear to fit at all. The EET professors wanted to keep a lot more of the electronics topics than one 4-year degree allowed. In conjunction with our Industrial Advisory Board (IAB) we settled on the topics that we thought belonged in an IT program. We repeated the exercise with the participants in CITC I and received similar results to those we obtained with our IAB and faculty[1]. We agreed on the topics, but not on relative importance of some of them. We also found many issues with sequencing of the topics that caused redundancy and/or a lack of prerequisite knowledge that became problematic as we taught the courses we had designed.

As we taught the original curriculum envisioned with the junior core treated in typical "stovepipe" fashion, we encountered the problem of students with insufficient background to understand the basic functioning of a web server and its relationship to the browser. This was the first major flag that appeared when the web systems class was taught for the first time. It became obvious why web systems are typically a senior level, optional topic in CS programs. This class has also been called "The unteachable class", for reasons that will be explained in following paragraphs. [9]

One needs a basic understanding of the breadth of computing technology to understand web implementation architectures. As we discussed this problem as a faculty, we also realized that the class dealing with human computer interfacing would benefit from students having experience with web design as a prerequisite. The HCI course was originally taught in the sophomore year and the web systems course was taught in the junior year. We initially decided that we would teach introductory concepts of operating systems, database, and networking in the web systems course, and that it would be a prerequisite for the database course. As we discussed the implications of these changes and worked on curriculum issues with other institutions in SIGITE curriculum committees, we came to realize that there was something fundamental going on. IT courses, by their nature, exist on the boundaries between technologies; hence the focus of this paper on education at the seams or boundaries. With this in mind, we decided to create a web systems course at the sophomore level and move the HCI class to the junior core. This also has the advantage that by using this course as gateway to the junior core, instructors can assume a certain level of proficiency in the fundamentals of IT platform technologies as they prepare curriculum for these core courses that cover the "pillars" of IT curriculum in depth.

This modified curriculum requires a pedagogical approach that is integrative rather than purely analytical. Early in our experience teaching IT courses, we observed [3] that teaching computer networking to IT students included the same concepts as teaching CS students, (in fact we use the same text [10]), but that the labs were very different. Similar differences occur throughout the IT curriculum; rather than a focus on the creation of a particular technology, the focus is on the external interface to the technology, how to tune implementations and how to access the functionality. We now realize that the explanation of how a technology works and the basic algorithms underlying it, naturally transitions into an implementation discussion in CS, while the natural transition in IT is to "standards", interoperability of implementations, and system integration issues, with practical applications being explored in labs and projects.

Education at the "Seams"

One of the major problems IT departments have faced is distinguishing themselves from Computer Engineering, Computer Science, and especially Information Systems programs. A major epiphany for us has been the realization that Information Technology can be distinguished by its focus on the places where technologies are integrated to create larger systems. The IT perspective of each of the five pillars can best be explained in terms of how they are used to create systems. Programming is generally used in IT to glue things together. The IT perspective of a network is the design, installation, and

maintenance of the whole network, and not any particular box. We teach databases and modeling, emphasizing use cases for modeling the human side of the system and SQL access from Java and PHP, since most databases are part of a web system in current IT situations. Web systems and human computer interaction are considered the soul of current IT programs and they are both the study of specific integration points in particular models. Indeed the major distinction between IS and IT may be the amount of time spent on analysis of particular interfaces. As determined by course hours taught, IS focuses on the interface of technology to the organization, while IT focuses on the interfaces between technologies [11].

Once we came to this understanding it became clear why several of the problems we were facing occurred. We were trying to use pedagogy created to develop skills for implementation of components rather than teaching interface use and integration. CS and CE curriculum and pedagogy focus on the creation of components, hence CS networking labs focus on the software's view of an interface from the inside of the box and the creation of a system that can implement the interface, while IT labs focus on connecting the interfaces of two separate boxes to create a network and what knobs to turn on the box to make the composite system function more efficiently.

Curriculum Changes

Our modifications of the curriculum to support education at the seams now include the following:

1. Development of a new *Introduction to Information Technology* cornerstone class that gives an introduction to IT in the context of the sister disciplines. This includes single lecture introductions to topics by IT instructors in the specialties of IT with simple lab experiences and seminar experiences including having CS, CE, and IS department representatives describe their disciplines to the students. The major goal of this course is to make sure students understand which of the computing disciplines offered best matches their interests. Originally this course was a 1 semester hour optional course. It has been converted into a 2 semester hour course required for the IT major.
2. Creating Fundamentals *of web-based Information Technology* class that incorporates the introductory concepts from operating systems, networking, database, human computer interaction, and software engineering, including elementary data modeling and use cases.
3. Removing the concepts taught in the fundamentals course from the corresponding classes in the junior core, leaving time for better intermediate coverage of the material.
4. Moving HCI from the sophomore to the junior year and the fundamentals course to the sophomore year.
5. Adding a web emphasis to the HCI course.
6. Emphasizing modeling as a tool for management of complexity throughout the system development process. A strong introduction is given within the sophomore class and usage is encouraged within the junior and senior level classes.
7. Including team projects in most of the courses, including the sophomore level course, and continuing the application-oriented labs in all IT courses.

8. Including the use of state of the art tools through educational copies and the use of the 30 day free trial of the software. (Vendors who allow free academic use of their products get a big advertising bonus here.)
9. Providing deeper coverage of enterprise systems deployment in the database class.
10. Redesign of our operating systems course to focus on OS applications rather than OS design.
11. Increased emphasis on technological evolution and integration with legacy systems to motivate an orientation of life-long learning.

In this paper we report on the experience of creating and teaching the Introduction to web-based Information Technology course. The key successes were as we expected; the major issues were involved with the transition in the curriculum. We will begin teaching the revised cornerstone course in fall 2004. Clearly, many of the changes require an on-going process to evolve all courses toward the new pedagogical orientation.

Teaching the Unteachable Course

The title of this section is taken from [9]. That author focuses on the difficulty of teaching a class where so much peripheral knowledge is required to understand the topic. In reality the experience is significantly less daunting if you know ahead of time that you must provide all of the peripheral knowledge and focus on the introduction and simple application of the technologies.

This new sophomore course is titled *Introduction to Web-based Information Technology*. We chose this title to convey the idea that this course is what IT is really about. Since we were adding material and wanted to keep the coverage of web systems similar to the original junior-level course, the new course has 4 semester-hours of credit associated rather than 3. In our program this allows for 3 days of lecture per week plus 3 hours of scheduled lab in addition to open lab times with TA coverage as available.

The course begins with lectures on the importance of models in all communication. Syntax and semantics are defined and their importance to all communication is emphasized using examples where identical syntax represents totally unrelated semantics. For example, the instructor writes “red” on the board and asks what this three-letter syntax represents. The obvious answer is a color that means stop if attached to a traffic semaphore. However, “la red” means “the net” in Spanish, very different semantics for identical syntax. This example is especially appropriate for our students since so many are bi-lingual due to national origin and foreign mission service for BYU’s sponsoring institution. After the students are sensitized to the issues of ambiguity in using models for communication, we review computer architecture, operating systems, and networking using the traditional models associated with the topics. The goal is to provide enough background in these topics to describe how web systems use the platform technologies to deliver a web service. The initial labs cover operating system utilities and process management, using a ‘sniffer’ to dissect packets revealing the bottom 4 layers of the internet model with an emphasis on locating and understanding addresses at each layer.

We then introduce scripting languages and string substitution macro processors using Batch files, BASH scripts, M4 and PERL.

After this tour of platform technologies, we cover web architecture starting at the browser and working toward the back end. We look at browser architecture and how HTML, XHTML, and XML are interpreted to provide presentation. We then move into JavaScript and the DOM. The labs that are associated with these concepts include installation of Apache, creating a simple home page for the site, and modifying a JavaScript example to display buttons in one frame that cause a picture and sound file to be displayed in another frame.

The course then covers CGI from an architectural perspective with a lab where students modify a PERL script to display the buttons that cause a picture and sound to be activated in another frame. The students immediately see why server-side code has become the default way to do dynamic pages. We then cover template-based server technologies with a PHP lab where students modify a PHP script to display the same dynamic web interface. At this point in the lectures we also introduce Databases, SQL and 3-tier web system architecture. The PHP lab includes example code to access a MySQL database that contains the same information that was stored in text files for the previous labs.

The lectures then cover Java-based technologies with a JSP lab that duplicates the functionality of the previous labs. Each lab starts with example code that does all of the basic actions required to access the database and display information. The lab consists of modifying the example code to provide specific dynamic displays.

The last 3-4 weeks of the semester are devoted to a team project. The lectures during this part of the course cover system life-cycle models including the traditional waterfall and iterative development using the Rational Unified Process as the example. This year team projects included chat servers, web sites, and the construction of a small network using wireless and switched devices. The only requirement was that the students form a 2-3 person team, and propose a project related to anything we had covered and then get instructor approval.

As this brief description indicates, the course emphasizes the *architecture* of web systems. It becomes clear that different technologies can be used to implement each of the major components without impacting the rest of the system. The labs are specifically designed to help the students realize that standard interfaces make it possible to transparently substitute and integrate different implementations of the components. In fact, the simplest implementation of the lab requirements is to reuse PHP pages in the context of a system driven primarily through Java Server Pages.

The approach also demonstrates that each of the technologies has strengths and weaknesses and that a good technologist cannot become religious about any technology. The course is designed to demonstrate to the students that one must continually update skills in order to avoid obsolescence.

Results

This course was taught Fall 2003 with 76 students registered in two sections. The day section was at 8:00 AM three days/week and had 56 students. The evening section was taught two days per week and had 20 students. Though the course was challenging, it was not impossible, since 37% of the students were in the A range. However, 24% failed, withdrew, or simply stopped coming to class. Some of the dropouts can be attributed to the fact that this was an 8:00 AM class or a 4:00 -6:00 PM class. Even though the instructor had thought that the smaller class would have a positive impact on student performance, the grade distribution was indistinguishable between the sections. This may be due to the fact that the sections had mixed lab sections. During the semester the instructor felt that the smaller section understood the material better and was surprised by the similar grade distributions. This result is counterintuitive and merits further investigation.

The grades were based 50% labs, 20% mid-term, 20% final, and 10% unannounced quizzes.

Day			Evening		Combined	
21	A	37.50%	7	35.00%	28.00	36.84%
8	B	14.29%	4	20.00%	12.00	15.79%
8	C	14.29%	2	10.00%	10.00	13.16%
7	D	12.50%	0	0.00%	7.00	9.21%
11	E/W/UW	19.64%	7	35.00%	18.00	23.68%
1	Audit	1.79%	0	0.00%	1.00	1.32%
56		100.00%	20	100.00%	76.00	100.00%

A survey was taken as a part of the final asking the following questions:

- What is the most important/useful thing you learned in the course?
- What is your favorite aspect of the course?
- What part of the course annoys you the most?
- What would you keep the same in the course?
- What would you change in the course?

The responses to the first question varied from "there is more than one way to do everything" to "The Rational Unified Process". No clear pattern emerged. The student's favorite part of the class was overwhelmingly the labs and final project even though lab write-ups were the most annoying part of the course! This was followed closely by know-it-all students that monopolize classes with questions over the head of the majority of the class. The consensus was that the course was excellent the way it is, but that more prompt return of labs needed to occur and that doing good lab documentation was time consuming. The promptness issue was a direct result of the unexpectedly large class. The evening section was added when the overflow list in registration passed 10 students for an original size of 60. However, we were unable to hire more TA's. For this lab

intensive class the ideal would be 1 TA for each 15 students. Our ratio was 1 to 38--more than twice the optimal, and the TA's spent time helping students during open labs to the detriment of prompt grading. No pattern emerged in the answers to the final question except that we should hire more TA's and get the labs graded more promptly. Since TA shortage is a common problem our experience suggests that further research is needed in more efficient advising and assessment in the lab component of the course.

Two students suggested that we divide the class into students with experience and those without; both of these were most annoyed by the know-it-alls and appeared to be members of the less experienced group of students.

Issues

Most of the issues seem to be related to the transition to the sophomore course. Some of the dropout rate and comments about know-it-all students can be attributed to the fact that many of the students in this course were juniors that were taking the sophomore class as a substitute for the defunct junior level class. However, several of the most aggressive question askers were sophomores that came to class better prepared. If we would have had one section of the old class format and one of the new, we could have obtained a better understanding of these issues. As it is, we need to be more sensitive to the needs of the less experienced students.

It would have been valuable to find out the reasons for the high number of UWs, Ws and Es in the course. If the students were finding out the IT was not the major for them, then the course was serving a useful purpose. If the course structure was part of the issue, feedback from these students would have been helpful.

We need to incorporate early feed-back mechanisms into the course that will allow us to identify students having problems and assist them in making a decision to drop the course or provide tutorial help that will allow them to keep up with the coursework.

Our experience teaching the "unteachable course" has demonstrated that "Education at the Seams" is a useful concept for structuring IT instruction. However, breadth-first approaches are not as common as topic oriented approaches in science and technology education. It was clear that some of the students felt uncomfortable with a course that covered a programming/scripting language a week focusing on similar semantics and leaving them to deal with syntax issues themselves. At least 3 students specifically mentioned this in the survey. One student mentioned that they were uncomfortable with the course's approach at the beginning, but liked it more and more as the semester progressed. A majority of the students expressed the opinion that the course should keep the same basic structure and content. The course clearly prepares students for the core IT curriculum better that any of our previous approaches. Several comments from junior and senior students taking this course mentioned that they thought the current curriculum restructuring was a significant improvement.

Conclusion
In the terminology of CC-CS 2001[7], IT is inherently a breadth-first discipline. We strongly believe, however, that it is not a breadth-only discipline. One needs to prepare students with an understanding of both sides of a technology boundary in order to begin examining the primary issues of importance to IT. This requires IT programs to provide broad introductions to technology early in the curriculum followed by more complete analysis of the seams in advanced courses.

The key issues that have emerged are:

1. Many students are uncomfortable with a breadth-first approach. Expectations need to be set earlier in the curriculum that IT requires one to learn by getting a broad overview and then getting detail as needed. We are addressing this through the freshman cornerstone course.
2. Instructors need to be explicit about what level of detail and understanding they are trying to convey. Students can be very uncomfortable if they are intently watching for details that will not be provided.
3. Techniques need to be developed to teach abstraction through modeling and architecture as an intellectual tool for managing complexity. IT professionals are continually required to create an understandable and explainable model from a myriad of details and components.

"Education at the seams" is a useful concept for preparing students to stitch systems together with understandable, deployable and manageable interfaces. We believe that this skill is the essence of Information Technology as an academic discipline.

REFERENCES

[1] Lunt, Barry, et al., *Defining the IT Curriculum: The Results of the Past 2½ Years*, ASEE 2004, Session not yet assigned.

[2] Ekstrom, Joseph, Renshaw, Stephen, Curriculum and Issues in a First Course of Computer Networking for Four-year Information Technology Programs, ASEE 2002 Session 2793

[3] Ekstrom, Joseph, Renshaw, Stephen, *A Project-Based Introductory Curriculum in Networking, WEB and Database Systems for 4-year Information Technology Programs*, CITC 3 Rochester NY, September, 2002

[4] Ekstrom, Joseph, Renshaw, Stephen, *Database Curriculum Issues for Four-year IT Programs,* CIEC 2003, Tucson, AZ, January, 2003.[

5] Ekstrom, Joseph, Lunt, Barry, Education at the Seams: Preparing Students to Stitch Systems Together;Curriculum and Issues for 4-Year IT Programs, CITC IV Purdue University, West Lafayette, Indiana, October 2003.

[6] Society for Information Technology Education; IT Accreditation Guidelines, Unpublished Draft; September 2002.

[7] Computing Curricula 2001, Computer Science Volume, December 15, 2001 Copyright 2001 ACM/IEEE**.** http://www.computer.org/education/cc2001/final/index.htm

[8] The authors are aware of SITE member participation in national conferences of ASEE (American Society for Engineering Education), DSI (Decision Sciences Institute), CIEC (Conference on Industry and Education Collaboration), and SIGCSE (Special Interest Group of Computer Science Education). SITE members have also participated in related regional conferences.

[9] Treu, Kevin, *To Teach the Unteachable Class: An Experimental Course in Web-Based Application Design,* SIGCSE'02, Covington, Kentucky, USA,

[10] Peterson, Larry L. and Davie Bruce S. *Computer Networks: A Systems Approach*, 3rd Ed, San Francisco, CA, 2003

[11] Lunt, Barry, et al., *What Is the New Discipline of Information Technology? Where Does It Fit?*, CIEC 2003, Tucson, AZ, Jan 28-31, 2003.

JOSEPH J. EKSTROM
Joseph J. Ekstrom (Ph. D. Computer Science, BYU 1992) has been Associate Professor of Information Technology at BYU since 2001. During 30 years of industrial experience he held positions from developer through senior management. His research interests include network and systems management, distributed computing, system modeling and architecture, system development, and IT curriculum and instruction.

BARRY M. LUNT
Barry M. Lunt is an Associate Professor of Information Technology at Brigham Young University in Provo, UT. Dr. Lunt received a B.S. and an M.S. degree in EET from BYU, and a Ph.D. in Occupational and Adult Education from Utah State University in Logan, UT. He has spent seven years in industry as a design engineer. His present research emphases are the physical design of electronic circuits and systems, IT curriculum, and engineering education.

C. RICHARD G. HELPS
Richard Helps is the Program Chair of the Information Technology program at BYU. He is also a TAC-ABET program evaluator. He spent ten years in industry as a control systems design engineer. He completed BS and MS degrees at the U of the Witwatersrand, South Africa and a further graduate degree at the University of Utah in Electrical Engineering. His primary scholarly interests are in embedded and real-time computing with its instrumentation and control aspects and also in technology education.