AC 2011-1263: INFORMATION TECHNOLOGY COURSES CHANGING CONSTANTLY: A CASE STUDY MODEL

Richard G. Helps, Brigham Young University

Information Technology Courses Changing Constantly:  
A Case Study Model

Abstract
Information Technology (IT) courses change frequently. These curricular changes follow the frequent changes in the technical discipline. Implementing course changes is made difficult by several exacerbating factors. Firstly, current technology is often used as a medium for instruction as well as a learning topic, so changes affect many parts of the course. Secondly, IT is relatively young as a separate discipline at universities, and thus lacks a deep pool of educational resources. Finally higher education reward systems often favor research and direct teaching for promotion and tenure, and do not usually reward course re-design and updating, even though the redesign effort is significant. As a result updating curricula frequently places a significant load on already-busy faculty.

A research case study with was undertaken to analyze and describe the process of changing courses. Qualitative methods were used and a model was developed to explain the processes observed in this case study. The model proved to be a useful vehicle for describing the change process in terms of instructional design domains. It also identifies and discusses relationships between design decisions, and the underlying structures, functions and activities in course design. (Related to Structures, Behaviors, Functions (SBF) analysis). The model helped to explain both positive and negative outcomes in IT course design. Finally, although the study was too limited to propose a generalized model it provided some insights into the design process and a theoretical foundation for future research.

The research was based on a collective case study, using thematic qualitative analysis of interviews from seven faculty members at two different institutions, representing three IT programs. The research methodology and validity is discussed in the report.

Introduction
Information technology is a rapidly changing field encompassing many disciplines related to computing. In the past decade a university-level academic discipline, also somewhat ambiguously called Information Technology (IT), has emerged. The academic discipline of IT faces a number of challenges, including a shortage of educational academic resources; a recently emerged program definition; a very rapidly changing, complex technical discipline; instructional designers with little training in instructional design; and an academic system that fails to reward ongoing and time consuming efforts to keep instructors and curricula up to date.

Historically IT is a relatively new discipline at the university level; four year IT degree programs at American universities emerged from related Computer Science, Information Systems, and Electronics and Computer Engineering or Technology programs, starting about 1995. Standards for a four-year IT curriculum, consisting of lists of technical topics, were accepted by the ACM and IEEE Computer Society professional organizations in 2008\(^1\). Formal standards for accrediting four-year IT programs were negotiated with ABET over several years and finally approved for evaluating programs in 2007\(^2\).
The vast majority of this foundational work was focused on the technical domain content of the curriculum, with some discussions on strategies for teaching. Much less work has been done in studying the instructional architecture and how curricula in this field evolve as changes occur in the field. The net effect of this pattern of development is that instructors at various institutions are pragmatically developing and modifying classes at a rapid rate, either using textbooks from related but different disciplines or generating their own materials, while they continue to investigate new technologies for their own professional interest and incorporate them into their teaching.

Methodologically, both because it is a professional degree program and because it follows the tradition of other technology degree programs, IT instruction emphasizes and champions authentic instruction using current technology\(^3,4\). Many technology programs use phrases like “hands-on” or “experiential” to characterize their educational approach\(^5-7\). This instructional approach differentiates technology programs from science programs and even from some engineering programs. For IT students to graduate as competent professionals in their discipline, it is necessary for them to be able to understand and apply current technological developments. Since the computer technology used in this experiential education changes constantly and rapidly\(^8-13\), as opposed to the relative stability of the underlying concepts and theories, the student learning experiences need to be constantly updated.

Courses are usually designed by IT instructors, whose educational background emphasizes technology rather than instructional design. Their having little or no formal training in instructional design impacts their approach to design of instruction: they emphasize technical topics in their instructional design. It appears that the change process is a natural and on-going one, with little or no conscious consideration of the impact of on-going change on the structure of the designed educational experiences. Liddtke\(^14\) comments,

> The process of curriculum design in the computing sciences has changed little in the last three decades, while the content of computing curricula has changed continuously because of the technology upon which it is based and the explosive growth of knowledge in the field. (p. 954)

As further evidence of the relative emphasis on technical advances in preference to theoretical educational design considerations in this domain, Lister and Box\(^15\) completed an analysis of papers published in the proceedings of the Computer Science Education Special Interest Group (SIGCSE) 2008 annual conference. They concluded, “The epistemology of the SIGCSE community is primarily objectivist, with a focus on content, rather than a constructivist, student-centered focus on learning.” A similar informal analysis of papers published in the Special Interest Group for Information Technology Education of the ACM (ACM-SIGITE) conference proceedings supports this conclusion, with the majority of the published papers focusing on new technology developments that could be introduced into the classroom, with little or no consideration of the educational curriculum architecture required or affected by these changes\(^16\).

When it comes to rewards for educational design faculty tenure and promotion decisions in computing disciplines are commonly based on successful research and peer-reviewed publication; upgrading curricula is seldom well recognized and rewarded for faculty in higher
education academic environments. Thus, in a rapidly changing technical discipline, updating their own knowledge as well as updating their constantly changing curricula places significant stresses on busy faculty members with little reward other than meeting the needs of their students.

The combination of these competing forces—a developing discipline with an emphasis on current technology, a constantly and rapidly changing set of current technologies, and a lack of reward for curricular re-design strongly suggests that the impact of instructional design for instructors needs to be minimized.

While it is apparent that curricula are evolving rapidly, the nature of the changes is not as clear. A clear understanding of what and how changes occur is a necessary prerequisite to developing better paradigms for course design for IT. The problem thus posed is to attempt to explore on an abstract level what is happening as IT curricula evolve. What happens to the instructional design as curricula are updated? What are instructors and designers doing and how are they doing it? This deeper understanding of what is happening will create a foundation upon which a methodology for evolving curricula in rapidly changing environments may be developed. This study is an exploration of the change process. The result is a theoretically based description of the change domains and structure.

**Research methodology**

The research was done as a collective case study, where the ‘case’ was a selection of course changes that reflected the process of change in IT. Seven faculty members at two institutions were interviewed and the interviews were transcribed. A number of change events were identified from the interviews and the change process was analyzed. Word frequency analysis was attempted with poor success but a follow-on analysis led to recurrent themes being identified in the interviews. Domains of design emerged from the analysis and the change process was analyzed in terms of domains, or layers, of design, which were then further classified in terms of the design decisions they represented and in terms of structures, activities and functions, which is related to Structure-Behaviors-Functions (SBF) analysis. This is a design architecture representation of the results of the case study.

This research did not interview students nor directly evaluate student learning, however the effect of course changes in the efficacy of student learning was considered as a factor in the interviews.

Within exploratory research both quantitative and qualitative methods can be used. In choosing between a quantitative and a qualitative approach, Stake suggests that the choice is driven by intention:

> A distinction between what knowledge to shoot for fundamentally separates quantitative and qualitative inquiry. Perhaps surprisingly, the distinction is not directly related to the difference between quantitative and qualitative data, but a difference in searching for causes versus searching for happenings. Quantitative researchers have pressed for explanation and control; qualitative researchers have pressed for understanding the complex interrelationships among all that exists. (p. 35)
Since explanations, rather than control, are sought a qualitative research approach was employed. In keeping with qualitative research norms the results include a number of direct quotes of the participants.

Rigorous research is important. The traditional scientific paradigm of research quality requires rigor in carefully and objectively isolating effects and linking them to causes. Guba and Lincoln\textsuperscript{18} suggest the conventional scientific criteria for rigor are: internal validity, external validity, reliability, and objectivity. In a constructivist (qualitative) inquiry Guba and Lincoln identify “parallel or foundational criteria” for ensuring the trustworthiness of the inquiry. Their four parallel criteria are: credibility, transferability, dependability and confirmability (pp. 233-243). Each of these criteria for trustworthiness were addressed in this research project.

Inasmuch as objective research strives for generalizable results, qualitative research strives for \textit{transferable} results, where the results do not claim to accurately represent a whole class of findings but rather to represent the findings clearly enough for the reader to gain insights into their own domains of application. Typically in qualitative research this includes direct quotations from the interviewees.

Seven faculty members were interviewed using the interview protocol. These faculty members were all professors of various ranks (from assistant professor to full professor), at two different institutions and in three different IT departments. The seven participants were interviewed face to face and the interviews were audio-recorded. The interviews were all fully transcribed. The professors’ names were removed and replaced with alias names to preserve anonymity. Throughout this report professors will be referred to by their alias names. The seven alias names are Adam, David, Lisa, Tom, Geoffrey, Susan and Jack.

\textbf{General Results}

Some expected results emerged from the interviews and from the change events, such as the fact that all faculty interviewed change their curricula frequently or very frequently; one professor indicated that he had redesigned or significantly modified seventeen courses within the past five years. The fact that he offered this datum spontaneously indicates that he is very aware of changing courses. Another professor indicated that he had changed or redesigned every course he was responsible for—also in a relatively short period of time. It should be emphasized that none of these professors were full-time instructional designers, but were normal university IT faculty with teaching and research expectations.

Professors did support the concept that constantly updating their curricula placed an extra burden on them. As participant Geoffrey expressed it,

\begin{quote}
[Our institution] is moving towards doing more formal research and expecting more publication and grantsmanship (sic) from the faculty, and so it's been made very clear to people that the reward system is going to lean heavily towards publication and grantsmanship and curriculum development is not always related and not always the best path towards that.
\end{quote}

Another professor said her method of keeping current with new technology is only successful, “because I work 75 hours a week.” These insights emphasize the reality of the problem of
constantly updating the curricula. Aspects of and inter-relationships relating to the processes of change were discovered by finding commonality in the data for the participants.

**Common Themes**

Several common themes were found. The first theme, which was expected, is that all those interviewed acknowledged constant technical change as a major factor in IT and in their instructional design. More revealing were the differences in attitudes towards the on-going process of technically driven change. Initially it was assumed that professors and students would be enthusiastic about new technologies and would be anxious to explore them. The actual attitudes range from considering change as enjoyable and a privilege (“It’s a labor of love for me”) to considering it an on-going duty and even considering it to be an unwanted driving force. (“We’re running ahead of the snowball”)

A common theme that was not anticipated, although perhaps it should have been, is that faculty members frequently change their courses to suit their preferred teaching style. At least four of the seven professors interviewed at both institutions redesigned their courses to follow a form of the studio-teaching model. Others changed their courses suit other preferences, but once changed future redesign work tended to use the same teaching approach.

Another common theme in this study was how few external resources and teamwork professors used in their design work. Most of the professors considered themselves competent in the technologies they were adopting and did not feel the need of support from educational experts or from material creation experts. There were very few exceptions to this finding in the seven interviews and twenty-plus course change events. Both institutions used for this study boast significant faculty support organizations or learning centers. These learning centers are staffed with professionals trained in instructional design and in creation of teaching materials (graphics, videos, assessment modules and so on). They are freely available to faculty to help with course development. All the participants were aware of these departments but none made any significant use of them.

Teaching using current technology was another common theme. Several of the professors interviewed try to use the latest ephemeral technology to teach enduring principles. The technical content can affect the teaching approach, and changes in the technical content can impact the design of the course. Susan describes what happened when a new version of database software (v. 10) replaced the previous version (v 9.9).

> For example, I used to love in Oracle, in Nine Nine [Ed: version 9.9] you were able to pull out the query tree as a graphic so you could actually see the tree in one of them. It worked for me of course. The most fun thing I liked about that course is the fact that relational algebra is actually there in the database. It isn't just something we made you learn; “Omigosh! there it is.” So we take the SQL code and convert it to a relational algebra statement, then we convert it to a tree, and then you could dump the tree right out of Oracle for your SQL – there it is! And then they had all the little notations on the performance on the tree and you could look at that. And then in Ten [Ed: version 10] they dropped that, and it was like so, Grrr! And they went to a text, and then you had to learn to read the text and the indentation and which ones were not indented, and to see what's
happening. So now what we have to do is pull out the plan as text and write the tree from the plan, so, Um! you know...

Geoffrey also talked about this phenomenon. He describes looking for “sweetheart applications.” The example cited below refers to the design of computer games and how characters in the game world interact with and encapsulate ideas of the software structure (methods and inheritance).

We say, what is it that we're gonna teach; what is it that they're gonna make, that they're gonna feel good about? And we spend a lot of time trying to find these little sweetheart applications where you can find some application you can build, that is almost like a description of the concept. … A concrete way of expressing an abstraction that is easy for people to get their head around, and then eventually we can move to a more abstract notion of that.

Other examples were found. Each of these examples expresses the strategic approach of the instructors designing their courses so that the students will understand underlying concepts by interacting with current technology. This approach is very effective in teaching but conflicts with the reality that as the technology changes all the carefully crafted teaching experiences become obsolete and need to be re-designed.

All professors expressed a desire for their students to understand both the current technology of the discipline and the underlying principles and concepts. All the participants displayed a commitment towards these two somewhat contradictory driving forces. On the one hand, they aim to teach fundamental principles or concepts and use the technology to reinforce the teaching of those principles or concepts. They do not teach technology for the sake of technology. On the other hand, teaching using the current technology is important. The students expect it and it is necessary in order for them to become competent professionals in the IT discipline. Therefore instructors strive to teach with the current technology. This dichotomy can be seen in this pair of quotations from Susan. On the one hand she says, “It seems like every year there's a new version of JAVA that comes out, okay there's new capabilities in that version, so you have to look at it.” Later in the interview she states the following:

We've always felt that IT is an academic discipline. If you're just teaching, “Here's FrontPage, and let's do our web pages with FrontPage,” or Dreamweaver or whatever, what you're really doing is you're just teaching a software package. We've always tried to stay on the side of, these are the basic IT concepts, principles; this is what we can do. These other things are just our tools.

The Instructional Design Model
Creating or developing new course materials is a design activity. In addition to the recurring themes, the language that the participants used revealed an underlying structure or architecture of instructional design. Certain domains or layers of design could be identified in the transcribed interviews that represent an underlying architectural basis for understanding the process of instructional change. Schön\cite{19} in his analysis of the design process of architects designing a new school building identified a set of “Normative/Descriptive Design Domains,” which were derived from the conversations he observed (p. 59). In an analogous manner I found that professors used words or phrases to indicate that they were focused on a particular aspect or domain of the design process. For example technology product names or descriptions (“Oracle”,...
“ActionScript”, “LCD monitors”) related to new and changing technology. When professors used these words they were describing instructional design decisions about how to include new technology. Similarly time based words or phrases (“each summer” “start of each semester” “one at a time”) reflected design decisions about the pace of change of courses. By collecting similar types of language and seeking common factors in the language a number of design domains were identified. Table 1 shows the design domains that were found.

These design domains not only indicate the multi-faceted nature of the design architecture, they also show other characteristics. Examining each of the domains it appears that each domain represents a certain type of design decision. Parnas\(^2^0\), in his discussion of software modularity, indicated that a major purpose of modular design is to enclose design decisions in modules. The different types of decision for each domain were identified and shown in Table 2.

The domains can be further classified in terms of the type of decision or domain that is represented. The Structure-Behavior-Function model (SBF), discussed earlier, also provides further insights into the domains.

Designers and theorists have used Structures, Behaviors and Functions (SBF) for analyzing design in fields ranging from computing\(^2^1-2^6\), to educational design\(^2^7-2^9\). The design domains in this study can be classified in terms analogous to the SBF approach. This not only provides further insight into the design architecture but also reveals relationships between the different aspects of the design. Since the SBF approach has been explored and modified by multiple researchers there are multiple meanings of the terms ‘structure’, ‘behavior’ and ‘function’ as well as similarly named and closely related concepts (such as “form and function”) for describing aspects of design. I have chosen to use the terms Structure (S), Function (F) and Activity (A) and define them as follows.

Designers apply functional intentions (F) to abstract structures of instruction architecture (S) by executing a variety of instructional design activities (A). These structures, functions and activities are different abstract domains or layers of the design process; each domain captures some aspect of design decisions. The usage of these terms will be illustrated by an example.

Technical content changes rapidly and constantly but the professor has no control of that change process and cannot make design decisions affecting that process. Therefore the design decisions relating to technical change relate to how the design will be adapted relative to the fixed reality of change happening. This technical content change is therefore a structural element. By the same token the professor has a certain philosophical or intellectual preference for a particular teaching approach, such as “constructivism” or “coaching.” The professor thus indicates a preferred functionality in course design, but this does not define a process of instructional design. This, therefore, is described as a functional domain. To design and create the course the professor designs student learning interactions and selects representations of the material. These are described as activity domains.

Another example of a structural domain is the Environment domain, which includes institutional guidelines, course outcomes and similar factors decided at departmental or higher levels within the organization. Once again professors cannot (easily) decide to change these items when
<table>
<thead>
<tr>
<th>Domain</th>
<th>Language type</th>
<th>Example words or phrases</th>
</tr>
</thead>
<tbody>
<tr>
<td>Technical content (change)</td>
<td>Technology names or versions</td>
<td>“LCD and CRT Displays,” (A) “ActionScript 3” (J, G), “Oracle” (S), “[version] nine-nine” (S)</td>
</tr>
<tr>
<td>Topics</td>
<td></td>
<td>“Storage devices”, “Displays” (A), “object-oriented paradigm” (J)</td>
</tr>
<tr>
<td>Driving force for change</td>
<td>Change verbs or verb phrases</td>
<td>“It's always a changing target” (A), “Market forces and what's new” (A), “What's hot. We try to stay with the source, pretty much” (S) “industry standards” “market adoption” (D, L), “what's new” (A), “we switched to studio teaching” (D).</td>
</tr>
<tr>
<td>Instructional Philosophy</td>
<td>Theory language Names for instructional approaches</td>
<td>“Constructivist” (T, G), “Studio teaching,” (D, G) “Active learning” (G, J), “Coaching” (T)</td>
</tr>
<tr>
<td>Strategies</td>
<td>Concept, Problem, Interaction, Exercise, Instructional period, challenge, Unit, Lesson</td>
<td>“Basic IT concepts, principles” (S), “Fundamental [idea]” (A, D, G, J), “Assignment” (A, D, T, G, J, L, S), Lab (A, D, T, G, J, L, S), “Project” (A, T, G, J, L), “what is a better way to present this than I've done in the past?” (A), “this is one we will build together in class” (J)</td>
</tr>
<tr>
<td>Student interaction (Message/Control)</td>
<td>Explain, Discuss, negotiate, Email, meeting, advise, consult</td>
<td>“Call and Response” (L) “Lots of emails but a few phone calls. And emails are just conversations between you and the student” (A), “I can send emails to them and I can put them in groups” (S), “Clickers keep students involved” (A), “class is discussion of the things that they've read”</td>
</tr>
<tr>
<td>Presentation/representation</td>
<td>Perceivable elements</td>
<td>Images, videos, PowerPoints, present, teach, “I have an image and I talk about the image” (L)</td>
</tr>
<tr>
<td>Change management</td>
<td>Resource</td>
<td>“Collaborate”, “team”, “TA”, “resource center”</td>
</tr>
<tr>
<td></td>
<td>Management phrases</td>
<td>“The labs themselves didn’t change” (J)</td>
</tr>
<tr>
<td>Data Management</td>
<td>Filenames and names of data organizational systems</td>
<td>“MyCourses” (S) “Blackboard” (A, D), “I just file them in the folder for that class” (A), “I automatically do that [track change history] with file names” (J)</td>
</tr>
<tr>
<td>Environmental factors</td>
<td>Organizational and institutional.</td>
<td>“outcomes,” “Program,” “course,” “research,” “Lab staff” “diverse group [of faculty]” (G), “[new lab equipment] is a matter of budget always” (L)</td>
</tr>
<tr>
<td></td>
<td>Resources</td>
<td>“funding” (G), “It's just pure volunteer” (J), “we have a chair who's very supportive.” (J) “65 hours per week” (S)</td>
</tr>
<tr>
<td>Curriculum structure</td>
<td>Course names or numbers as part of a sequence</td>
<td>“programming sequence”; “prerequisites”; Course numbers (500-level, 350, 628, 4002-330, 4002-434 etc.)</td>
</tr>
<tr>
<td>Pace of change</td>
<td>Time-based words</td>
<td>“change one factor at a time” (J), “update at the start of each semester” (several), “do changes in the summer. (J, G)</td>
</tr>
</tbody>
</table>
Table 2. Decision oriented listing of domains, showing their structural, functional and activity aspects.

<table>
<thead>
<tr>
<th>Domain</th>
<th>Type</th>
<th>Nature of design decision</th>
</tr>
</thead>
<tbody>
<tr>
<td>Technical Content</td>
<td>S</td>
<td>Professors decide to incorporate new technical content. They do not control the content, only decide to adopt it. This is identified through various specific technical references, such as names or versions of technologies.</td>
</tr>
<tr>
<td>Driving force for change</td>
<td>F</td>
<td>Various phrases or words indicate that participants' change decisions were influenced by internal or external forces. The driving force indicates the type of change the professor decides to make.</td>
</tr>
<tr>
<td>Instructional Philosophy</td>
<td>F</td>
<td>The professors chose an instructional approach, either theoretically-based or pragmatic.</td>
</tr>
<tr>
<td>Strategies</td>
<td>A</td>
<td>Creating instructional strategies, such as class presentations, assignments, discussions, tests etc. Strategies tend to follow the instructional design philosophy.</td>
</tr>
<tr>
<td>Student interaction</td>
<td>A</td>
<td>Student interaction elements of the design are chosen by the professor. Although they may be part of the strategy, they are often more fluid and rapidly changing. Professors may choose different interactions ad-hoc as the learning situation suggests it.</td>
</tr>
<tr>
<td>Present/represent</td>
<td>A</td>
<td>How the material is presented is a very personal decision. Sometimes it is chosen by the professor in advance, such as prepared demonstrations, PowerPoint slides and graphics carefully collected and included in teaching materials. At other times it may be much more dynamic as a professor spontaneously asks students to participate in a discussion or chooses to present a topic to them or challenges them to find materials to match the subject under discussion.</td>
</tr>
<tr>
<td>Change management</td>
<td>F</td>
<td>Change management is a decision relating to how the professors allocate their time to curriculum change, whether they do it as a regulated ongoing activity or in bursts of intense effort.</td>
</tr>
<tr>
<td>Data Management</td>
<td>F</td>
<td>Data relative to the course must be managed and handled. Most professors see this as an independent function and delegate it to a separate system, such as a learning management system.</td>
</tr>
<tr>
<td>Environmental factors</td>
<td>S</td>
<td>The environment that the professors design and teach within. The professors have little control over institutional policies, guidelines and expectations but they choose how their designs will interact with the environment.</td>
</tr>
<tr>
<td>Curriculum structure</td>
<td>S</td>
<td>Professors decide to how to include new content within an existing curricular structure in most cases. Changing the curriculum, in the sense of changing sequences of courses is frequently decided at the institutional level. From the point of view of the course designer the curriculum, with its pre-requisites represents a structural element to be designed against.</td>
</tr>
<tr>
<td>Pace of change</td>
<td>S</td>
<td>The pace of change is not under control of the professor but they choose how to adapt to it. The decision here is how they choose to synchronize their course changes with the ongoing technological changes and which aspects of their course they allow to be affected by any current change.</td>
</tr>
</tbody>
</table>
designing courses so the design decisions relate to how to adapt the course to these factors. These structural, functional and activity classifications for all of the domains are also included in Table 2.

A notable example of an environment domain is the ABET accreditation criteria. Many IT programs are accredited by ABET and the programs studied here are ABET accredited. The ABET requirements impact the change process in that the changes may affect the program’s objectives or outcomes, which are in turn reviewed by ABET as part of the accreditation process. The program outcomes are often defined at a departmental level and the instructor is not free to change them while changing course materials. This requirement constrains the change process. In at least one change event studied this constraint was found to have a negative impact on the students.

Finally, in classifying domains, the design layers architectural approach discussed by Gibbons and others \(^{30-32}\), also has a close match to some of these design layers. Design layer concepts such as data management, strategy and others emerged as design domains. The well-developed definitions of these design layers, such as the Strategy and Representation layers, provide additional understanding of the implications of these design domains. Table 1 shows the domains and the type of decisions represented in each design domain. It also shows the nature of each domain, whether it is a structural, functional or activity element and therefore how it will relate to other domains. Several of the domains correspond reasonably with known design layers, and in those cases the relevant design layer is shown in square brackets under the name.

**Conclusions**

The design domains and their sub classifications, which indicate aspects of the way the domain interacts with other domains, explain much of the process of design that professors follow in these rapidly evolving environments. Professors are constrained to constantly make changes to their courses due to changing technology. This change is a constant factor in information technology. They decide to incorporate the new technologies as they see them becoming essential to the course. What they intend to achieve functionally in terms of the course changes is driven by factors such as “what's new,” “what's hot,” or “market adoption.” The types of changes are also driven by other functional intentions, such as by their underlying instructional philosophy, either theoretical or pragmatic. To implement these functional elements they choose various design activities, for example teaching strategies such as assignments, class discussions or specific student interactions, and they choose different representations such as specific images, demonstrations or examples. Professors also follow functional intentions in choosing data management functionalities and change management functionalities. These functions and activities take place against a structural environment of institutional policy, curriculum structure and structural background of pace of change.

The differences between structures, functions and activities indicate the nature of design decisions that were by the designers. Design decisions relating to structural elements indicate that these are areas where the designer should not try to change the domain but must include the constraints of this domain in the design. Functional design decisions indicate that the designer should clearly establish preferred modes or directions for design. Design activities indicate where course content should be created or selected to implement the functional modes chosen.
Activity decisions reflect where course designers can spend time creating specific instructional artifacts. By recognizing different types of decisions designers can, in the spirit of modular design, create conceptually isolated modules with defined interfaces and reduce the need for complete course re-design when future changes are needed. Only one professor in the study did this consistently.

Another significant factor is that of modularization. Modularization has been outstanding successful in software design. Realizing similar success in instructional design requires a clear understanding of the underlying design structure and how to separate modules. The domains model developed for this case study provides insights into this problem.

Baldwin and Clark\textsuperscript{33} make this interesting comment about the point at which modularization becomes economic.

Two interesting points arise as we move from simple to complex: (1) the point at which an artifact can no longer be made by a single person; and (2) the point at which an artifact can no longer be comprehended by a single person. Crossing into the first region requires a division of labor; crossing into the second requires a division of the knowledge and effort that goes into creating a design.

The complexity of instructional design revealed by this study strongly suggests that IT curriculum design has passed one, if not both, of these points. The fact that work within some of the identified domains indicates a need for specialized knowledge of some topics (such as instructional philosophy and data management – not typically within the expertise of the professor designing the course) also suggest that this type of design is probably in the region where multiple people are required or recommended for effective design.

An item of interest in the study is the effect that course changes have on student learning. No direct measures of student learning were done in this study but the intended effects of changes on learning were considered. In brief all the professors interviewed were aware of the issue and their approach to the problem can be summarized as, “good intentions.” They strove to make their course designs relevant and effective but did not measure the effects of course changes on student learning in any structured way.

**Problems revealed by this model**

Examples of modularization or lack of it, in design domains was seen in the interviews. Baldwin and Clark discuss the necessity of keeping domains separated or the benefits are lost\textsuperscript{34}. One situation that is described by the model is that of the change in database software described earlier. The Technical Change decision was to upgrade from version 9.9 to version 10. This change also affected the Strategy, Presentation and Student Interaction domains. The changes implemented by the instructor cut across multiple design domain. Thus the instructor found herself making changes that affected most of her course with a resultant large effort to effect the changes. An awareness of the multi-layer nature of changes can mitigate these efforts or at least help with understanding and planning for appropriate changes. In contrast to this, Adam describes how he continually inserts new technical content into his courses but keeps the course strategy undisturbed, with much less rework time.
One of the respondents described a situation where a two-course sequence was defined with fixed outcomes. The course outcomes were in terms of basic computer language constructs (e.g. variable, loop structure, recursion etc.). Over time the courses evolved from procedural languages to Object-Oriented (OO) languages. The outcomes remained unchanged. Some students took the first course (using procedural language) and then waited for some time to take the second (by then being taught in OO language). Clearly they were unprepared for the second course. However since the outcomes had not been changed the institutional documentation said they should have been prepared. This was described as a “worst-case scenario” In terms of the domains model a misunderstanding of the nature of the environmental domain (the fixed course outcomes) and a failure to consider it in the redesign led to a course design out of sync with its environment.

Other analyses of the data revealed other situations where the layers either came into conflict or designs intended to just update a technology artifact ended up impacting multiple domains and requiring huge efforts to achieve.

The concepts of modularity, domains and layers of design, shed further light on the problems of constantly evolving curricula. When reviewing the events described in the interviews it is apparent that although their language shows different domains of design, professors rarely attempted to separate these domains to allow for future changes. The approach to design was somewhat holistic. In some cases the professors attempted to keep the course outcomes stable, indicating an inherent awareness of the problem of multiple domains as aspects of design, but in some cases the problem persists and is merely hidden.

**Recommendations**

This study was a thematic case study and based on a limited number of interviews at two ABET-accredited institutions offering IT programs. It does not claim to be a generalized model of all IT course design. However, within the limits of those constraints the insights offered by the study have value for IT course designers.

Modular development is relevant in course design as it is in other fields of design. Modularity requires an understanding of the nature of the design decisions encapsulated in the each module. Different domains of design representing design decisions are apparent in the process of changing IT courses. Different types of design activity occur within those domains. An awareness of these factors and the problems that occur when crossing domain boundaries can be of great value to IT instructors in changing courses.

The research indicates that currently IT professors make little use of labor-sharing resources. Understanding the structure of the problem, committing to modularization with an awareness of the domains of instructional complexity and then delegating or sharing design and implementation responsibilities is likely to yield the benefits of modularization that have achieved such remarkable success in the field of computer hardware, software and systems design.
Bibliography