

2006-01-03-1
MAY 31, 2006

GRANDON GILL

THE MYSTERY OF THE SELF-PACED COURSE (A)

“What the heck is going on?”

Dr. Grandon Gill, Associate Professor of Information Systems & Decision Sciences (IS&DS) at the University of South Florida, asked himself the question as he finished looking at the Blackboard grade sheet for his introductory programming course, ISM3232. With 9 days left in the school’s 10-week Summer C semester, only 2 out of his 34 registered students had accumulated enough points to pass the course with a grade of “C”. None had achieved a “B” or better. He was used to seeing procrastination—but this was worse than anything he’d ever seen before. Short of a miracle, it looked as if he was going to have to fail the majority of the class unless a drastic reduction in course requirements were made. He had the uncomfortable feeling his students were playing “chicken” with him. The question was: should he turn the wheel (and in what direction)?

Actually, the problem he faced had two aspects. In the short term, there seemed to be three obviously available approaches:

1. Do nothing, and let students be held accountable for their own actions
2. Relax the course requirements
3. Provide a more flexible policy for “incomplete” grades

None of these seemed particularly palatable. Do (1), and the student course evaluations would be a blood bath; more importantly, failing an introductory required course might also lead students to abandon the whole MIS major. Do (2) and he’d be caving in on a firm policy that his summer courses would cover precisely the same amount of material as the normal 16-week fall and spring semester courses, in spite of their compressed time frame. Do (3) and he’d be creating a whole pile of extra work—all of which he’d have to do by himself, since TA support ended in 10 days. Also, giving “I” grades for non-emergency situations seemed like an abuse of the grading system. Given how unappealing all the obvious alternatives seemed, he also wondered if he could come up with some better choice.

This case was prepared for the purpose of class discussion, and not to illustrate the effective or ineffective handling of an administrative or classroom situation and is copyrighted by the Informing Science Institute. Permission to make digital or paper copy of part or all of these works for personal or classroom use is granted without fee provided that the copies are not made or distributed for profit or commercial advantage AND that copies 1) bear this notice in full and 2) give the full citation on the first page. It is permissible to abstract these works so long as credit is given. To copy in all other cases or to republish or to post on a server or to redistribute to lists requires specific permission and payment of a fee. Contact Publisher@InformingScience.org to request redistribution permission.

The short term problem was simple, however, compared with the longer term. In the 10 days before the end of the term, substantive course redesign was out of the question. But Gill was assigned to teach 3 sections of the same course in the fall. Was the course design that he rolled out over the summer fundamentally flawed? If it was, what changes was he going to have to make? And where would he find the time to make the changes?

The MIS Major

Two disciplines tended to focus on teaching computer-related concepts: computer science (CS) and MIS. As a general rule, CS curricula offered 2-3 times more computer-related courses than MIS curricula—which also had to cover core business concepts. Furthermore, CS itself could be broken down into two distinct camps, computer science as engineering—emphasizing performance and hardware—and computer science as a liberal art. In the latter camp, computer applications were often aligned with disciplines such as cognitive science and information science, examining the use of computers in more abstract ways. For example, the technical and social implications of potential computer uses, such as artificial intelligence, may be considered at some length. According to the National Center for Educational Statistics, in 2002 roughly 60% of all computer-related degrees fell into the two CS categories.

The remaining 40% of all computer-related degrees were offered through MIS (or business information systems) programs. These programs emphasized using information technology (IT) to solve business problems and the managerial challenges resulting from such uses. As a result, MIS majors were much more likely to view the computer as a tool for getting things done rather than as an abstract tool for scientific or social inquiry. MIS programs were also likely to emphasize the desirability of combining knowledge of the task domain (i.e., business) and the tool (i.e., IT) if real world problems are to be solved.

In addition to differing from its CS cousins, MIS majors differed from most other majors in a College of Business in that students had to demonstrate proficiency in a number of technology-related areas (e.g., programming, databases, data communications) as well as acquiring managerial skills. Students in the major tended to be career focused, and instructors in IS-related courses frequently emphasized the excellent long-term job prospects for graduates with IT skills. Indeed, as shown in Exhibit 1, in 2002 the top seven occupations for job-growth whose normal prerequisite was a bachelor's degree were all IS-related.

At USF, the MIS major was completed during the student's Junior and Senior years, and consisted of: 1) the business core (39 hours of courses in business functions applicable to all business majors), 2) a set of required MIS courses (18 hours), and 3) MIS-approved electives (6 hours). To ensure students were adequately prepared for more advanced courses, the major had a four layer course sequence (see Exhibit 2), with IS-specific courses outside the business core beginning in the second layer (the entry courses, ISM3232 and ISM3113). It was also possible to take a 4-course MIS minor, consisting of ISM3232, ISM3113, ISM4212 and an approved elective.

Over the previous 5 years, enrollment in the USF MIS major had been extremely volatile. In the late 1990s through 2001, the major had exploded in size—more than doubling in enrollments and growing to well over 1000 students. By the year 2000, it had become the largest undergraduate major in the College of Business. During that period, the department offered at least 5 sections of entry courses, and these would typically fill within 24 hours of being opened. During registration week, dozens of students could be seen roaming the corridors searching for overrides into closed sections.

The situation changed abruptly during the period from 2001 to 2004. With the bursting of the “Internet bubble”, students no longer perceived the MIS major to be the path to instant riches. In consequence, the major contracted considerably—enrollments dropping by an estimated 30-40% percent. By Fall Semester 2004, three sections of an entry course, such as ISM3232, were more than sufficient to meet student demand. Although alarming, the department felt it was doing rather well by national standards. At various national meetings, many chairs had reported enrollment drops in excess of 50%—some even reporting

80% or more. Nonetheless, the situation was of concern to the department. Further declines in enrollment could force the department to cut back on its use of instructors and increase the percentage of undergraduate required courses assigned to tenure-track faculty.

Computer Programming

The heart of every computer, the *central processing unit* (or CPU), is controlled by a collection of on-off switches (that can be viewed conceptually as 0s and 1s) in memory known as machine language. Since humans are not very good at thinking entirely in terms of 0s and 1s, programming languages have been developed to aid in the creation of applications to run on the computer, referred to as *computer programs*. The process of creating a program typically involves three steps:

- *Writing the source code.* In this activity, the programmer actually creates the application using a programming language and some type of text editor. While many different programming languages exist, at the time of the case the most common choices for introductory courses were C++ (a superset of the C programming language) and Java (a pure object-oriented programming language).
- *Compiling the source code.* A process whereby source code written by the programmer is translated into executable code that is “understood” by the computer itself. As part of this process, the code is also *linked* to executable code written by other programmers—such as Microsoft developers—known as library code. During the compilation and linking process, errors in the source code are typically identified as a list of error messages.
- *Running and debugging the application.* After an application has been successfully compiled and linked, the programmer can run the application and test it. As part of the testing process, a tool known as a *debugger* is normally used. The principal task of a debugger is not to remove code errors (a.k.a. bugs) but to allow the programmer to examine the inner workings of the code as it is running on the machine. Through careful inspection using the various windows provided by the debugger—dozens of different window types are available—the skilled programmer can often detect anomalies that lead to correction of code defects (see Exhibit 3 for an example).

In order to write programs, the developer needs a collection of specialized tools (e.g., editor, compiler, linker, debugger). Frequently, such tools are combined into a single application, often referred to as an integrated development environment (IDE). Such tools, by their very nature, are among the most complex software applications on the planet—even experienced developers rarely master (or even know about) more than 25% of their available feature set. The most popular IDE was Microsoft’s Visual Studio .NET, used in about 50% of all commercial development (according to Microsoft’s estimates). Its vast feature set also made it one of the most difficult environments to learn. Compounding the difficulty, its design objectives—to be a development tool for software professionals—meant that its help system and error messages were almost *never* comprehensible to novices.

Under the best of circumstances, teaching introductory programming is a challenging task. To get a program to compile and run properly requires a high level of logical thinking and considerable attention to detail. Even in computer science programs, attrition rates of 30-40% are relatively common for the first programming course. In many ways, however, the challenge is even greater in MIS programs. The sources of additional difficulty include:

- *Greater diversity of course objectives.* Because MIS majors have far fewer courses to work with than CS majors, it is critical that certain concepts (e.g., computer organization, data

representation, programming practice) be introduced in the first course—to feed subsequent courses and ensure overall student literacy.

- *Greater diversity of background and experience:* Huge variations in programming experience among incoming students can be present. For example, as shown in Exhibit 4, 48% of incoming students to ISM3232 were found to have no previous programming courses, while 27% had taken more than one such course.
- *Greater diversity of aspirations:* Many MIS majors specifically choose the major (in preference to CS) for its reduced emphasis on programming. Students could actually be quite vocal in their desire not to be programmers. Having stated this, it is equally true that a substantial fraction were interested in becoming programmers, as is evident from the Exhibit 4 survey results.

Despite the difficulties inherent in teaching programming to MIS majors, doing so nonetheless remained an important part of most undergraduate MIS majors. Teaching programming languages was also perceived to be a central component of MIS programs in the future (Source: Gorgone, J.T., “Draft Information Systems Accreditation Criteria for 2006”. *Inroads: The SIGCSE Bulletin*. Vol. 36, No. 2, June 2004). The importance of programming in MIS programs resulted largely from the role it played in developing a fundamental conceptual skill. While not every MIS-related job involved programming, nearly all required the individual be able to “think like a programmer”. This applied whether an individual was creating a dynamic web page, managing user login scripts on a network, designing applications or managing a group of programmers. Thus, even if an individual didn’t plan to become a professional programmer, elementary programming skill was a valuable asset.

The Instructor

Grandon Gill had a varied background that strongly impacted his teaching styles and pedagogy. In 1975, he completed the requirements for his undergraduate degree from Harvard College in Applied Mathematics and Economics at the age of 19. He then joined the U.S. Navy, where he served as a nuclear trained submarine officer. That training introduced him to a novel pedagogy, based on the submarine qualification process. After completing his 5 year tour of duty, he enrolled in Harvard Business School’s MBA program. Here, he was introduced to the “case method” pedagogy—indeed, the two year program at HBS consisted entirely of case discussions.

Upon leaving the school, Gill pursued a number of entrepreneurial ambitions and worked for an agribusiness consulting firm that had been founded by one of his professors. Within one year of joining the 30 person firm, he had been promoted three times—to Senior Vice President of Technical Services. Despite his success in the consulting arena, Gill found the process of selling his high-priced services to wealthy Fortune 500 companies to be rather unsatisfying. What he really wanted to do was to teach. He also found that he had a growing interest in computer modeling, but lacked more than rudimentary programming skills (although, he notes candidly, he never emphasized that fact when billing clients for the models he developed for them). As a consequence, he decided to change directions in his career, and entered the HBS doctoral program, in information systems.

Upon completing his doctorate, Gill joined Florida Atlantic University (FAU), in Boca Raton. While there, he was heavily involved in the reorganization of the FAU undergraduate MIS major. His particular area of interest was the programming track, where he developed a 3-course sequence that took students with little or no exposure to programming and built the skills necessary to make them solid entry-level programmers. His enthusiasm for this process was so great that he routinely requested additional (uncompensated) course assignments so he could continue actively teaching all three courses.

Another aspect of Gill's teaching at FAU was his “leading edge” use of technology in his classes. In 1994, well before the Internet had become a household word, he had—in a spare bedroom and at his own expense—set up a 4-line bulletin board system (BBS) giving students access to local email, online discussion groups, chat and file downloads. During fall 1994, almost 2500 phone calls were made to that BBS. Buoyed by the success of the technology in supporting his students, he pressed the FAU College of Business (COB) to adopt a similar Internet-based system. To drive home the point, in fall 1995 he developed a survey course on the Internet for COB faculty. Meeting from 9 AM to 12 PM for eight consecutive Fridays, he had over 30 faculty members in regular attendance. By the year 2000, Gill's attention had become focused on creating multimedia materials to support his courses. Using these materials, he found he could dramatically reduce the time required to learn the complex Visual Studio IDE that he used in his classes.

While at FAU, Gill regularly developed fairly substantial Windows-based software applications, a process he had started while earning his doctorate. Most of these were developed in conjunction with the University, such as a geographic information system (GIS) built to help McDonalds locate restaurants in Latin America, initiated by a research center located within the COB. He also developed numerous programs to support his classes. His reasons for doing so were twofold. First, the applications were tailored to help the students learn concepts that had proven elusive in the past. Second, constantly creating new applications kept his programming skills fresh—critical in the ever changing environment of software development.

In assessing his own interest in programming, Gill took a rather bemused perspective. While he took great pleasure in the skills he had acquired, he was forced to concede that his programming career had not gotten off to a very promising start. The sense of confusion he had experienced in his first exposure to programming still evoked discomfort when he thought about it. As he wrote in the class notes for his 25th college reunion in 2001:

Looking back on my undergraduate experience, I think that the event that probably had the greatest impact on me, career-wise, was the C I received in Applied Math 110 (Introduction to Programming)... [given my obstinate nature] it therefore makes complete sense that I have organized my life, over the past 25 years, so that my occupation now consists, almost entirely, of writing computer programs and teaching computer programming. On the plus side, I guess I should be thankful that I do not live in an alternative universe. Had Wilfred Cantwell Smith seen fit to flunk me Hum 11 (History of Religion), I'd probably be a priest by now. And I doubt my loving wife and two fine sons would appreciate that one bit.

Among the teaching accolades garnered by Gill while at FAU were two "best professor" awards from the Executive MBA program, the university-wide award for excellence in undergraduate teaching in 1997 and two \$5000/year teacher incentive pay (TIP) raises. A summary biographical sketch, prepared by Gill for an NSF grant application, is provided as Exhibit 5.

ISM3232: Business Application Development

In fall 2001, at the height of MIS enrollments across the country, the IS&DS department at USF hired Gill. A major part of his assignment was to act as a coordinator for the five Tampa sections of ISM3232, the department's introductory course in computer programming. The course was described in the USF catalog as follows:

Presentation of business application development using a modern programming language (C). Topics include data structures, indexing, file processing, and user interfaces. Good program design techniques are emphasized. Business applications are developed.

Fall 2001: Gill's First Semester

At the time Gill joined the faculty, the course was in transition. Several years earlier, in response to student complaints about the difficulty of the course, the department had added a required prerequisite course, ISM3230, over half of which was devoted to introducing C programming. Unfortunately, the prerequisite course (which had been offered mass-lecture style) did not seem to solve the problem. Instead it had suffered from chronically low evaluations; furthermore, student skills upon entering ISM3232 seemed unchanged. To make matters worse, the course extended the time required to complete the MIS major. Based on these outcomes, the department chose to eliminate the prerequisite and have students return to taking ISM3232 as their introduction to programming. Gill's first class was roughly a 50-50 split between students who had taken the prerequisite as a result of having entered the major in spring 2001 or before, and 50% who had not taken the prerequisite.

The course design Gill used in his first semester at USF was modeled after the highly successful version of ISM3232 he had taught at FAU (under the Florida state system's uniform course numbering system, the course numbers were the same at both schools). He lectured for three hours per week, had two exams—a midterm and a final, both open book tests consisting of essay/short answer questions—and seven assignments (which could be done individually or in self-selected groups) that were worth 50% of the student's grade. Based on his experience at FAU, he used an expanded grading curve—fixed at the start of the semester and given to students—that was defined as follows:

- A: 80% and above
- B: 60-79%
- C: 40-59%
- D: 20-39%

In place of lab sessions, Gill prepared about 10 hours of multimedia content that walked students through the use of the tools and example programming problems. The CD could be purchased at the bookstore for a handling fee of \$3.00. Students were also able to download a 400+ page packet (in .pdf form) containing all the lecture notes for the semester—with hard copy also available for optional purchase at ProCopy.

One unusual feature of Gill's original ISM3232 course design was the use of oral exams for the final two assignments (worth 50% of the total assignment grade, 25% of the course grade). The exams, administered by Gill himself, involved asking students detailed questions about the completed assignments that they had handed in. If a student could answer the questions, he or she was given full credit for the assignment. If not, it was as if the assignment had never been handed in. Students were allowed to retake an oral exam as many times as necessary in order to pass—but the questions on retakes tended to get successively harder with each retake. Gill had modeled the oral exam approach on the submarine qualification process that he had participated in during the late-1970s. He had first instituted it at FAU in response to some student complaints that group members were not pulling their weight yet were getting full assignment credit. Usually, it took about 30 seconds to tell if a student understood the code handed in by his or her group, but exams typically ranged in length from about 3 minutes (for very strong students) to 20-30 minutes (for weaker students, or students who had difficulty expressing themselves).

Gill's confidence in the oral examination process was so high that he established a policy of allowing students to substitute their assignment grades for their final and midterm grades provided they both:

- a) received a B or better on the last two assignments, AND
- b) passed both oral exams

At FAU, all but the weakest students had taken advantage of this policy—understandable since assignment grades tended to be in the 80s and 90s, whereas the median score on any of Gill's written exams rarely exceeded 60%. Gill encouraged students to take the assignment route, since he preferred their time be spent problem solving on the computer rather than studying his notes in preparation for an exam.

Gill's first USF classes were among the quietest he had ever encountered. Students virtually never initiated questions and seemed reluctant to respond to the numerous questions Gill posed to them during lectures. Nonetheless, their median score on the midterm was in the high 50s, slightly above what he was used to seeing at FAU. Student course evaluation forms were handed out very early, in mid-November. The reason for this was that the last three class meetings covered topics not related to assignments—so designed to give students time to complete their last two projects—and Gill knew they would be sparsely attended. Starting right before Thanksgiving, and continuing through the end of the semester in mid-December, Gill administered well over a hundred oral exams on the last two assignments. The general atmosphere of these exams seemed to be quite upbeat, with relatively few retakes being required. It seemed to Gill that the students had finally gotten their act together—although he naturally wished that they had started earlier.

When student evaluations came back in January, Gill was in a state of shock. His overall instructor rating average of 2.63 (on a 1 to 5 scale, with 5 being best) was so far below anything that he had ever received in the past that he initially thought he was reading the scale backwards. Student written comments were also humbling (see Exhibit 6 for some examples)—although some very positive ones were mixed with the negative.

Spring 2002

Part of the reason Gill was so unprepared for the reaction he had experienced in the evaluations was the rapport that he felt he had developed with the students during the oral exam process. Although students were nearly universal in their complaints about the excessive course workload, he really didn't believe it was the source of the problem. Indeed, he had never heard of a programming course without an "unfair" workload. So, instead of modifying the class based on student recommendations, in Spring 2002 Gill took the risk of being guided by his instincts and made the following changes:

- He doubled the size of the first major assignment (Assignment 3) and made credit contingent on an oral exam.
- He instituted Blackboard discussion groups for each individual assignment
- He changed the timing of the evaluations so that they would be given out on the last possible class day

These changes led to a totally different outcome at the end of the semester, with instructor evaluations climbing to 4.47. During this period, he also held extensive discussions with his department chair relating to the future direction of the class.

Fall 2002-Spring 2004: Evolution to Blended Design

Although the results of spring 2002 were gratifying (at least with respect to Gill's self-esteem), they still left many problems unsolved. Chief among these was that of consistency across sections and faculty assignment issues. Consistency was problematic because different instructors were using different textbooks and—at satellite campuses—different development tools. Such variations were an issue because subsequent courses in the major expected students with reasonably similar preparation. Faculty assignments were also becoming a concern because accreditation criteria were becoming increasingly strict with respect to individuals teaching undergraduate courses. As a result, the department was forced to cease using a number of highly qualified (but under-credentialed) adjuncts and instructors, creating a shortage of faculty available to teach technical courses.

Together, the department chair and Gill developed a plan to transform ISM3232 into a blended course. Under the new design, initiated in fall 2002, Gill took overall responsibility for all ISM3232 sections (whether or not he was listed as the instructor of record). Each week, he lectured to one section in the TV studio (typically on a Friday or early on Monday) and the tapes were replayed at the remaining sections, in the presence of a TA or an instructor who was there to answer questions. TAs were always undergraduates who had recently taken the course. Gill preferred these to graduate students for a number of reasons:

- Having had an opportunity to observe them in the class, he had more confidence in their suitability for TA work.
- They were familiar with the dynamics of the class and its unusual organization
- Being peers of the students in the class, they were less intimidating than graduate students
- They tended to view the job as an educational opportunity (indeed, a number of students had offered to act as TAs for free)
- They were 30-40% cheaper than graduate students

As the course continued to evolve, a many changes were instituted. In fall 2002, he introduced a draft textbook—available in .pdf form (for free downloading) and at ProCopy (for the cost of copying). Gill felt this was needed since existing texts didn't really match the course design. In spring 2003, Gill created online versions of all the lectures—posted on Blackboard and available for student download. By fall 2003, Gill felt that grades were getting too high (over 50% A's in spring 2003) so he therefore added an additional module, introducing object-oriented programming, representing about 30% more content. His motivation for doing so was not to set himself up as a proponent for "rigor", or to act as a one-man tsunami against the rising tide of grade inflation. Given how hard his students seemed to be working, he could have slept easy giving 70-80% of them A's. Rather, it was his intent to ensure that the class was sufficiently challenging for that 10-20% of the students who might actually choose to become programmers in their career. Towards this end, he changed assignment weights such that the requirements for a "C" remained largely unchanged, while the requirements for a "B" and an "A" (in particular) rose considerably.

Another change instituted in fall 2002 was offering TA "lab" sessions during half of all class time. Rather than consisting of formal lectures, these sessions focused on helping students understand and complete assignments. Gill gave the TAs almost complete discretion with respect to the content of these sessions, which students viewed as very helpful. Students were also given the option of going to TAs for their first oral exam on any assignment (all subsequent retake exams had to be done with Gill). Overall, TA evaluations (gathered in an optional, extra credit, end-of-semester survey) tended to be high (usually ranging from 4.2-4.7 on a 1 to 5 scale).

Another change made in fall 2003 was the elimination of midterm and final exams. The principal reason for this change was that students—knowing assignments could substitute for exams—had simply stopped showing up for the exams. During that semester, it also became clear that “live” and taped lectures were bordering on irrelevant. Not only were students not showing up to lecture sessions, at TA sessions students often wanted the “web” version to be played in the classroom rather than the taped version. Among the reasons given were: a) the better visual clarity of the online lectures (created with Camtasia, an animated screen capture application allowing the instructor to add audio narration), b) the fact that the online lectures tended to be slightly shorter (lacking any pretense of social interaction) and c) the fact that these lectures were available to students for download.

By spring 2004, median attendance at the live lecture section had dropped to 2 students (out of 35) and TAs reported that taped lecture replays were faring little better. At that time, the decision was made to eliminate traditional lectures altogether and to remaster the web-based lectures to improve their quality and add additional features, such as indexing. The new online lectures were completed by the end of April 2004.

Throughout the entire period of transition from traditional to blended (i.e., mixture of distance learning and face-to-face communications), course evaluations hovered in the 4.0 range—reasonable for a programming course and, in Gill’s view, pretty good given that the GPA distribution for the course was below the departmental average and students were reporting working twice as long per week on ISM3232 as on other MIS courses. The one exception to these “adequate” evaluation scores was in summer 2003, when the 10-week schedule—without any reduction in course requirements—seemed to take a toll and ratings dropped to 3.2.

Overall, Gill perceived the new course design to be quite successful. Completion rates of 70-80% were being achieved (quite good for programming courses). Furthermore, gender differences and prior experience—two factors reported to be highly problematic according to an extensive literature on teaching programming—seemed to have no impact on any of the ISM3232 student outcomes, including grades, self-reported hours worked and satisfaction. As a result, in Spring 2004, Gill proposed a \$500,000 curriculum materials development project to NSF, with the objective of further refining and disseminating the techniques that he had developed for the course. As part of the grant proposal, he attempted to map the course’s activities into Chickering and Gamson’s widely used “Seven Principles of Effective Undergraduate Education”. A table taken from that proposal is presented in Exhibit 7.

Summer 2004

As the summer of 2004 approached, Gill looked to complete his course redesign by eliminating some slight inconsistencies in organization. Prior to the summer, weekly lectures had been used to “synchronize” the course, and assignment due dates were based around the pace of lectures. In many ways, however, this arrangement didn’t make sense. For example, did it make sense to hold students content with a “C” to the same due dates as students targeting an “A”? Even more to the point, what should be done if the course’s policy of deducting 10% per week for late assignments caused someone who otherwise would have earned a “C” to fall into the “D” range? (Note: “C” or better was required if the course were to count for credit in the MIS major).

Summer 2004 Course Design

The obvious solution to the aforementioned inconsistency was to eliminate “late” deductions and firm due dates, replacing them with guidelines based on the desired grade. Gill took this approach in the 10-week summer 2004 semester, as shown in Exhibit 8. Replacing standard due dates for assignments in the syllabus was a grid of recommended due dates based on the desired grade. Late points were never deducted and the only clear requirement was that all assignments (excepting the last 2) had to be submitted at least a week prior to the end of the semester—in order to provide adequate time for the TAs to grade the assignments and administer the validation exams.

Because he did not want a repeat of the previous summer, Gill gave a great deal of thought regarding how students could be induced to step up their pace as required by the shorter semester. The solution that he arrived at involved two parts. First, since weekly lectures no longer needed to be given, Gill decided to require every student in the class (34 in all) to meet with him personally during the first week of class. To facilitate these meetings, he replaced the desk in his office with a small conference table. In a similar vein, he decided that each student in the class would be “assigned” a TA, and would be required to meet weekly with that TA throughout the semester. The idea here was to keep a close eye on student progress so that they did not fall behind.

Another design change introduced in the summer term involved the administration of oral exams. In the past, TAs had mentioned to Gill how uncomfortable it could be to “not pass” a student in an oral exam when that same student had been coming in for help from them in the previous days and weeks. To reduce pressure on the TAs, and to further strengthen the oral exam system, course policy was changed so that:

1. A student could not take an oral exam until his or her "assigned" TA signified that the student was ready
2. A student could not take the oral exam with his or her "assigned" TA, but had to seek out another TA instead.

A further revision of the oral exam policy allowed one retake (with a third TA) provided the assigned TA *and* the original examiner both signified the student to be ready. Gill hoped students would perceive this as an inducement—since being forced to take an exam with him was often viewed as a punishment. The new policy gave the students one more opportunity to settle the matter with their peers.

To track the new policies, Gill developed a “Student Validation Card” (see Exhibit 9 for the first page of the card), modeled after a submarine “qual” card. On the first day of class, each student was given a blank card (printed on heavy index stock) and was instructed to keep it throughout the semester. As student assignments were graded and validated, entries were made on the card signifying the student’s progress. The card provided a confirming source of information for grade results—the authoritative source, should grades posted on Blackboard be disputed. Students were cautioned that if they lost their validation card, they could be required to revalidate any or all assignments.

The final change to the course design involved the development of a course web site that paralleled Blackboard. Without weekly lectures to synchronize their progress, Gill reasoned that students might find it difficult match readings and online content with the assignments they related to. To address this, the site was organized into assignment pages, with material related to each assignment grouped together. Exhibit 10 shows an example of such a page.

The reader may view the archived course site at: <http://ism3232.coba.usf.edu/Archive/>

Summer 2004 Course Delivery

The summer 2004 semester of ISM3232 began smoothly. Within 10 days, Gill had met with nearly every student in the class for at least 10 minutes, 1-on-1. All had indicated a clear understanding of the course organization and all but a few had stated their desire to try for an “A”. TA assignments were made expeditiously and the only real setback was the time required to get students their own copies of Microsoft’s Visual Studio .NET IDE software. This was a problem every semester, however, so the assignments were designed to minimize use of the tool until at least three weeks into the class (even for “A” students).

By early June, three weeks into the course, Gill started to become a bit puzzled. Virtually no posts were being made to Blackboard relating to the assignments. This was quite unusual, since 100-200 postings per major assignment were fairly typical. He polled the TAs to discover that many students were not showing up for their weekly meetings. He then sent out a mass email to the students asking for an explanation. Responses given by the students were generally along the lines of “being forced to come in negates the benefits of having so much of the course online”. Since virtual contact was better than no contact at all, Gill sent out a mass email and an identical Blackboard announcement (Exhibit 11) permitting email to be used for weekly meetings.

By mid-June, Gill was on the road to meet with an NSF program director about his project proposal, then on to New Hampshire where did his research and writing during the summer. With just over three weeks left in the course, he was still not seeing the type of progress he had expected. While nearly all of the class had completed the first (non-validated) assignment, few had bothered to validate the second assignment and no one had completed the third. In other words, not a single student in the class was meeting the “C” pace (see the second table in Exhibit 9), much less the “A” or “B” pace. Another email went out to the TAs requesting insights, as Gill began to wonder if he was going to need to fly back to Tampa. Two replies he received were as follows:

TA 1. We are all wondering the same thing. Lately, a few more people have been coming in for help, but earlier in the semester there was no one. From what I have seen, people feel that they do not have to go to class, and there are no late deadlines, so it is a self paced course, which it is. A lot of them have the impression that this is a class where all of the course work can be completed in 2 weeks if you cram enough. We have tried to stress that they need to get started, but we have all gotten minimal status emails, and minimal weekly visits. Also, no one shows up to class. I get the impression that they feel this class is an easy class. Maybe some of the other TA's feel the same.

TA 2. I'm not sure what to do about the current status of most students in the class. I've been having the majority of my Monday night classes with only 2-3 students in them and never see anyone during office hours. I constantly think about this but can't come up with a solution since as far as I can see nobody is really trying. Even the people that do show up to class don't have any progress from the previous class.

As he thought about coming back early, Gill realized that it would be a futile gesture. Given the nature of “contract” he had made with the class in the syllabus, there was no way he could change the rules of the game until July 9th—the “no excuses” due date for 5 assignments. As an alternative he decided to try some damage control by adding an “I” option that students could take if they met the requirements for at least a “C”, as described in Exhibit 12.

7 July 2004

Two days before the 9 July deadline specified in the syllabus, only 2 out of Gill's 34 registered students had accumulated enough points to pass the course with a grade of "C". None had achieved a "B" or better. Although the TAs had informed him that a number of Assignment 3s and 4s had been submitted, their grades had yet to appear on Blackboard. And, of course, the oral exams had not been administered. Never before had he beheld such potential for a bloodbath.

The options available to him in the short run all seemed obvious—and flawed. He could let his students figure out how to get out of the mess they were in. Undoubtedly, he'd see many of them again in the fall, sadder but wiser. Unfortunately, there would be quite a few he wouldn't see—their blank, uncomprehending stares would be focused on some marketing professor (having dropped out of the MIS major). On the other hand, he could always reduce the course passing requirements. But did he really want to be sending students scoring a 20% in his class to the instructors teaching Java? "Give 'em a bit a challenge", he thought to himself, unable to keep from grinning. Of course, there was always the "incomplete" solution—postponing the problem rather than solving it. His students obviously thrived on procrastination. All he would need to do was to offer "I" grades on request (instead of requiring a "C") and he'd start an email blizzard that would bring the Net to its knees. Then he'd get to spend the next four months sorting things out. Furthermore, in his experience, relatively few mercy "I" grades ever became passes.

As he thought about it, though, it was not the short term problem that vexed him as much as the long term implications for his course. He'd designed ISM3232 to be a far more realistic depiction of real world programming than the traditional lecture-and-test course. The heart of his course—group work, use of existing code, demanding projects, self-guided learning and online support—were all common practice in commercial programming. Should MIS programs really graduate majors who fall to pieces when they encounter a "realistic" environment? How would students learn to cope if they were not held accountable for their actions or inactivity?

Finally, the half million dollar question: what could he do to redesign the course so this never happened again?

Exhibit 1: Outlook for occupations normally requiring a Bachelor's Degree

Job Title	Jobs Added (000s) 2002-2012	% Change 2002-2012	Typical degree requirement
Computer systems analysts	184	39	Bachelor's degree
Computer software engineers, applications	179	46	Bachelor's degree
Computer software engineers, systems software	128	45	Bachelor's degree
Network systems and data communications analysts	106	57	Bachelor's degree
Computer and information systems managers	103	36	Bachelor's
Network and computer systems administrators	94	37	Bachelor's degree
Database administrators	49	44	Bachelor's degree
Physician assistants	31	49	Bachelor's degree
Occupational therapists	29	35	Bachelor's degree
Environmental engineers	18	38	Bachelor's degree

Source: <http://stats.bls.gov/news.release/ooh.toc.htm>

Exhibit 2: MIS major course sequence

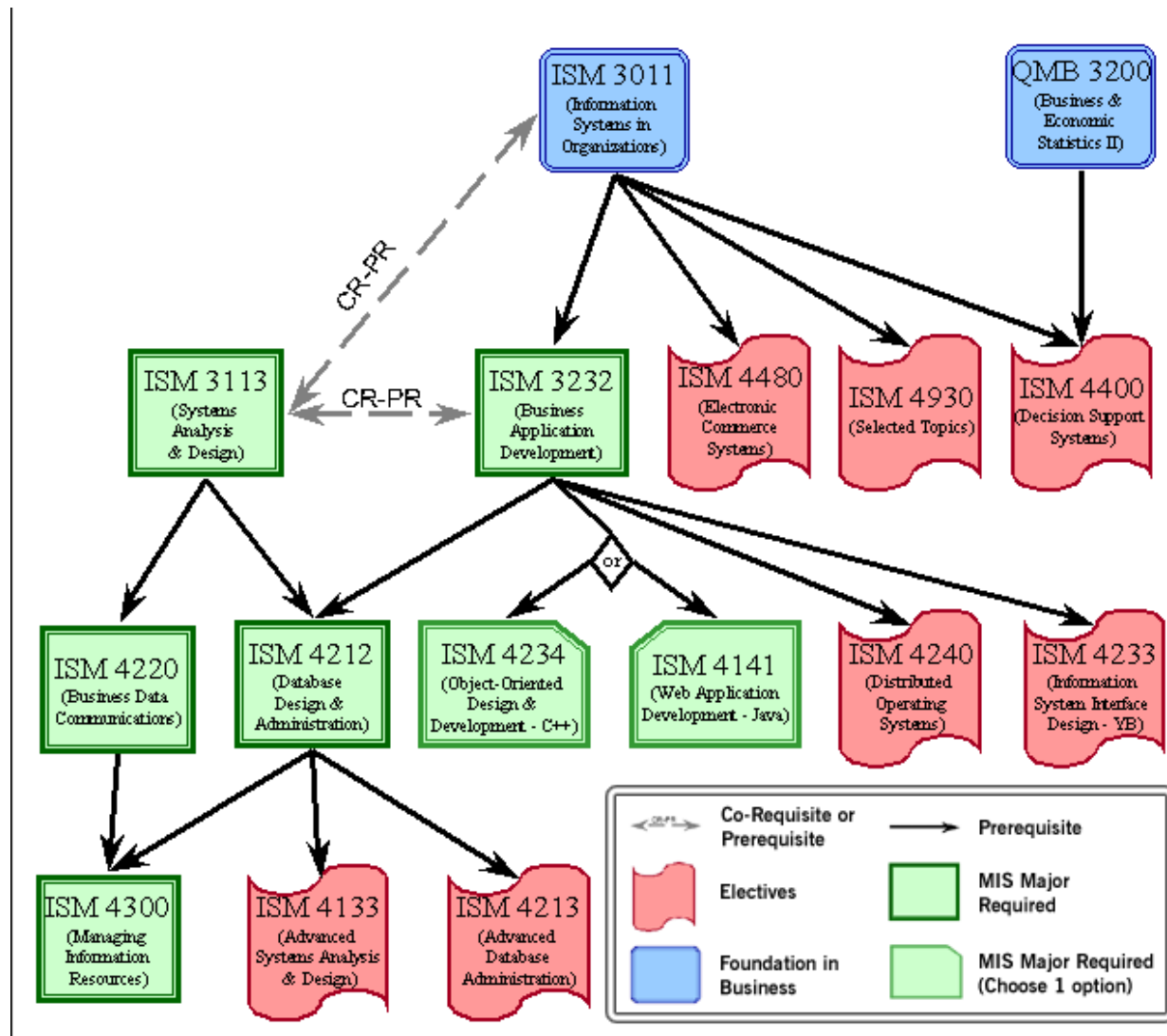


Exhibit 3: Visual Studio .NET IDE during debugging

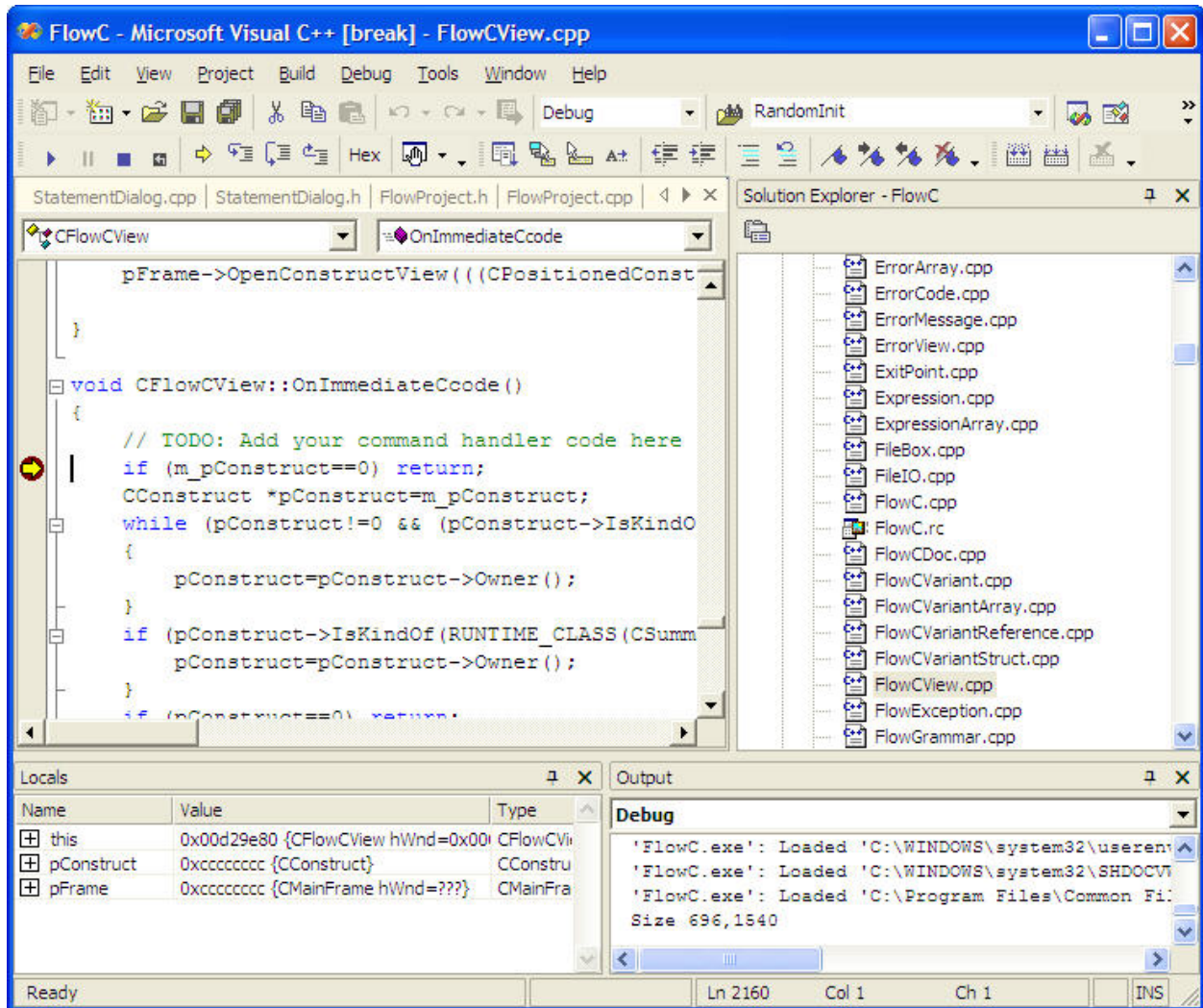


Exhibit 4: Background and aspiration diversity

Area	Item	Result
Experience	Percentage who have taken no programming courses before	48%
Experience	Percentage who have taken more than one programming course before	27%
Experience	Percentage who are working full time	34%
Aspirations	Percentage believing it pretty likely or very likely that they will be employed as programmers within the next 10 years	15%
Aspirations	Percentage believing it pretty unlikely or very unlikely that they will be employed as programmers within the next 10 years	45%

Source: ISM3232 end-of-semester surveys, 2003

Exhibit 5: Grandon Gill Biographical Sketch

Professional Preparation

Harvard College	Applied Mathematics & Economics	A.B. 1975
Harvard Business School	Management	M.B.A. 1982
Harvard Business School	Management Information Systems	D.B.A. 1991

Academic and Professional Appointments

2001 –	<i>University of South Florida, College of Business Administration</i> Associate Professor (tenured) in IS & DS department
1991 – 2001	<i>Florida Atlantic University, College of Business</i> Associate Professor (tenured in 1996) in ITOM department
1983 –1986	<i>Agribusiness Associates</i> Senior Vice President, Technical Services
1982 –1983	<i>SnCorp, Inc.</i> President
1975 – 1980	<i>US Navy, Submarine Force</i> Nuclear trained submarine officer

Publications (Closely Related to Proposed Project)

- Gill, T.G. 2005. Introduction to Programming Using Visual Studio .NET Hoboken, NJ: John Wiley & Son.
- Gill, T.G. 2004. "Teaching C++ Programming 'Submarine Style': A Case Study". *IEEE Transactions on Education*. Publication forthcoming.
- Gill, T.G. 2004. "Teaching Programming with FlowC" *Journal of Information Systems Education*. Vol 15, No. 1.
- Gill, T.G. 2003. "A New Approach to Teaching C Programming". *Proceeding of DSI 2003, Washington DC*.
- Gill, T. and Hu, Q. 1998. "The Evolving Undergraduate Information Systems Education: A Survey of the U. S. Institutions," *Journal of Education for Business*, Vol. 74, No. 5. 289-295.

Publications (Other Significant)

- Gill, T. 2001. "What's an MIS Paper Worth? An Exploratory Analysis," *Data Base*. Vol. 32, No. 2. 14-33.
- Hu, Q. and Gill, T. 2000. "IS Faculty Research Productivity: Influential Factors and Implications," *Information Resources Management Journal*, Spring. Vol. 13, No. 2. 12-22.
- Gill, T. 1996. "Expert Systems Usage: Task Change and Intrinsic Motivation", *MIS Quarterly*, Vol. 20, No. 3, September. 301-329.
- Gill, T. 1995. "Early Expert Systems: Where are they now?", *MIS Quarterly*, Vol. 19, No. 1, March, 1995.
- Gill, T. 1995. "High-Tech Hidebound: Case Studies of Information Technologies that Inhibit Organizational Learning", *Accounting, Management and Information Technology*, Vol. 5, No. 1. 41-60.

Synergistic Activities

1. *Research in distance learning and the case method.* The subject of the MIS Quarterly submission currently listed as under review, this research involved studying how mixes of distance learning and in class delivery impacted student learning in a case method class. Was featured in an *Online Classroom* article (Kelly, R. 2002. "Adapting the Case method to Online Learning", *Online Classroom*. October. 4-5).
2. *Educational materials development.* Designed, authored and performed layout for the original *QuickStudy* guide series (Barcharts, Inc., 1994-1997) and *CyberCue* guide series (MacMillan Computer Publishing, Que E&T division, 1998-1999). Developed 34 of these 4-page and 6-page laminated guides, with unit sales of roughly 1 million in total. Also developed over a dozen published case studies (HBS, Prentice Hall and Journal of Information Technology) and authored the IS content for a general business textbook (Madura, J. 1998. *Introduction to Business*, Cincinnati: South-Western College Publishing). Currently completing *Object-Oriented Programming Using Visual C++.NET* with Wiley.
3. *Curriculum Development.* Awarded 3 grants (in the form of software licenses totaling in excess of \$100,000 at existing retail prices) for the development on online course materials as part of Microsoft's curriculum development grant program (1996, 1997). Materials for 5 different courses were placed online. Received \$6000 from USF Center for 21st Century Teaching Excellence to develop online version of ISM3232 C++ programming class.
4. *Programming.* Have developed a significant number of commercial software applications that have been used by individuals (e.g., the College Expert) and by major corporations such as CocaCola USA, McDonalds Latin American Division, United Fruit, and McKinsey & Co.
5. *Educational Policy.* Proposed a program designed to enhance opportunities for individuals reentering the workforce, particular focused on women (e.g., Gill, R & Gill, T. 1993. "A New Parental Bill of Rights". *Public Interest*. Spring. No. 111). Modeled after the GI Bill, the program would have provided educational benefits to any individual who chose to leave the workforce in order to raise small children. The proposal was widely discussed in the press (e.g., Will, G. 1997. "A GI Bill for Mothers". *Newsweek*. 22 December. p. 88).

Exhibit 6: Selected comments from ISM3232: the Good, the Bad and the Ugly

The following comments represent a selection the comments section of Gill's Fall 2001 ISM3232 course evaluations. All are all direct transcriptions with original grammar and spelling.

The good:

- I thought the course was wonderful. Professor Gill made information for the class accessible in many, many ways. The CD for the class is the greatest thing. I wish I had other classes like this one. My overall evaluation of Prof. Gill is perfect. I have not had a better teacher at USF.
- The class has been hard but worth the effort. Gill is really cool and easy talk with.
- I learned more stuff than I ever thought I would, I actually like C++ now. Great teacher. Knows his stuff.
- I think he is a good professor but our previous class didn't adequately prepare us.

The bad:

- Course materials were well developed, however the course was taught in poor manner. We did 4 really easy assignments then were given 3 impossible ones. Why couldn't we be given easy-harder-harder-hardest? We didn't learn how to program in C, we learned how to pass a hard class.
- 1) Assignments are not equally balanced. Assignments should start out easy to moderate in difficulty and then progress to more difficult. 2) Instructor stands in front of class and goes over 100 lines of code. This is not teaching nor facilitates learning.
- Class was extremely confusing. Needs more instructions on writing functions. Be more clear-don't just read the presentation.
- He needs to teach more about C and less from HTML and conversions. No psudo code. Was tough. And I learned from practice and not from this kind of teaching. He is unapproachable outside of class and goes off in tangents in lectures.
- He knows his information but it doesn't seem he conveys it good enough to be understood.

The ugly:

- Up to this point I am still wondering why this monster became a professor. He is a self-righteous person. He needs to go back where he came from.
- I will never take another class with this instructor. I did not learn anything in here.
- Horrible professor.
- Horrible professor!!

Exhibit 7: Course design and the “7 Principles”

	Design Elements	Relevant Survey Items and Outcomes
1. Student-faculty contact	<ul style="list-style-type: none"> Instructor played a central role in responding to student postings Instructor role in 1-on-1 oral examinations validating major programming assignments TAs were used as instructors in the labs and during extensive office hours, leveraging the fact that they cost 10% as much as faculty TAs were specifically assigned blocks of lab time to help students with assignments 	<ul style="list-style-type: none"> 93% were either satisfied or very satisfied (66%) with using Blackboard, an activity they report spending 4 hours on each week. 79% found TAs to be helpful in their learning. Individual ratings of TAs were high, ranging from 3.6 to 4.7. Students reported spending an average of 4 hours per week interacting with either the TAs or the instructor.
2. Cooperation among students	<p>Encouraging students to work in groups though:</p> <ul style="list-style-type: none"> Use of a predetermined curve to reduce incentive to compete with fellow students Use of validating exams to ensure rigor, while permitting group work and inter-group cooperation Weighting assignments identically, whether completed individually or in groups 	<ul style="list-style-type: none"> 68% felt the grading system was much help or very much help to their learning. 75% felt that working with peers was much help or very much help in learning. 77% disagree or strongly disagree that they could have completed the assignments without help. Students reported spending an average of 6 hours per week working in groups. 71% were satisfied or very satisfied with group activities. Students indicate that they put in equal time at the computer as compared to their teammates: mean, median and mode for each assignment are all 3 on a five point scale.
3. Active learning	<ul style="list-style-type: none"> Focus on assignment completion rather than delivering facts and conventional examinations 	<ul style="list-style-type: none"> 75% reported that the assignments contributed to their understanding of the course material (12% disagreed). 72% disagreed or strongly disagreed (median and mode of 1 on a 5 point scale) that more emphasis should be placed on tests (15% agreed).
4. Prompt feedback	<ul style="list-style-type: none"> Assignments and oral exams were designed to give students rapid feedback on their progress, using undergraduate TAs to leverage instructor time. Blackboard coverage was used to provide quick feedback for student questions 	<ul style="list-style-type: none"> 78% felt level of feedback offered was moderate, much or very much help. Median response time to a Blackboard posting initiated by a student in 2002 was under 1 hour (measured using posting times). [An internal failure of Blackboard prevented 2003 data being gathered.]
5. Time on task	<ul style="list-style-type: none"> Use of an assignment-centric grading system, with a fixed (and pre-specified) curve to eliminate uncertainty in the linkage between effort and grade 	<ul style="list-style-type: none"> Students report spending 17 hours per week on the course, more than twice as much time reported spent on typical MIS courses and more than three times that for typical business courses.
6. High expectations	<ul style="list-style-type: none"> High quality course materials sets standard for class Incorporation of "state of the art" software tools into the curriculum Difficulty of later assignments far exceeds typical difficulty level in introductory course 	<ul style="list-style-type: none"> 76% were satisfied or very satisfied with course multimedia materials. 82% were satisfied or very satisfied with course handouts.
7. Respects talents and diverse ways of learning	<ul style="list-style-type: none"> Providing numerous options regarding how the course can be taken, ranging from a traditional classroom setting to pure distance learning Use of pedagogy different from "traditional" programming course intended to provide benefits even for those with prior experience Diverse group of teaching assistants has been recruited 	<p>Evidence of support for diverse learning styles includes the perceived effectiveness of techniques:</p> <ul style="list-style-type: none"> 35% found the textbook to be of little or no help to their learning while an equal number found it to be of much help or very much help. 22% found the textbook companion multimedia CD to be of little or no help to their learning while 56% found it to be of much help or very much help. 28% report finding Blackboard to be of little or no help to their learning while 53% found it to be of much help or very much help. 42% found discussion to be of little or no help to their learning while 33% found it to be of much help or very much help. 52% found lectures to be of little help or no help to their learning while 28% found them to be of much help or very much help. 63% report that course activities added to their skills in working effectively with others which suggests increased respect for the diverse talents of their peers.

For explanation of “7 Principles”, see Chickering, A. W. and Gamson, Z. F. (1987). Seven Principles for Good Practice in Undergraduate Education. *AAHE Bulletin*, 39 (7), 3-7.

Exhibit 8: ISM3232 Assignments & Due Dates for Summer 2004

Requirement	Description	Percent
<i>Exercise 1:</i> Compiler Exercises	Compiler installation and simple compiles (Hello, World! and simple multi-file project)	5%
<i>Exercise 2:</i> Numbering Systems	Conversions between decimal, hex and binary. Twos complement representation. Simple bitwise logical operations. <i>Credit for assignment will be dependent on the results of an online exam conducted in the lab.</i>	10%
<i>Exercise 3:</i> Logic and flow-charting	Creating flow charts for simple processes. Converting code to flow charts. Converting flow charts to code. <i>Credit for assignment will be dependent on the results of an oral exam.</i>	20%
<i>Exercise 4:</i> Debugging (4A) & Pointer Arithmetic (4B)	Taking a program with a variety of compiler, linker and runtime errors and finding/removing the bugs. Using a memory grid to locate items in memory. <i>Credit for assignment will be dependent on the results of an online exam conducted in the lab.</i>	15%
<i>Exercise 5:</i> Function exercises	Creating a series of functions that perform simple string tasks. <i>Credit for assignment will be dependent on the results of an oral exam.</i>	25%
<i>Exercise 6:</i> Structured CGI Application	Creating web-based application that takes input from a web form and returns it to a browser. <i>Credit for assignment will be dependent on the results of an oral exam.</i>	15%
<i>Exercise 7:</i> OOP CGI Application	Rewriting web-based CGI application using C++ classes. <i>Credit for assignment will be dependent on the results of an oral exam.</i>	10%

Assignment	Last Day to Submit	You should plan on completing the assignment by the following date if your grade target is:		
		C (40-59)	B (60-79)	A (80-100)
Assignment 1	9 July 2004	24 May 2004	24 May 2004	17 May 2004
Assignment 2	9 July 2004	31 May 2004	24 May 2004	21 May 2004
Assignment 3	9 July 2004	25 June 2004	11 June 2004	7 June 2004
Assignment 4	9 July 2004	5 July 2004	21 June 2004	18 June 2004
Assignment 5	9 July 2004	N/A	7 July 2004	2 July 2004
Assignment 6	12 July 2004	N/A	N/A	9 July 2004
Assignment 7*	12 July 2004	N/A	N/A	12 July 2004*

*An A in the course is possible without completing Assignment 7, although completing it will put you way ahead of the game in the OOP course that follows (either in Java or C++)

Source: Ism3232 syllabus, summer semester 2004.

Exhibit 9: First page of ISM3232 Student Validation Record

Name: _____ Blackboard ID: _____

Note: Each student must keep the original copy of this record and bring it with them to each validation exam. In the event the original is lost, you may be required to repeat validations.

Assigned TA

	TA Name	Weekly meeting		Date assigned
Initial		Day:	Time:	
Reassigned to		Day:	Time:	
Reassigned to		Day:	Time:	

Assignment 1 score: _____ (No validation required)

Assignment 2

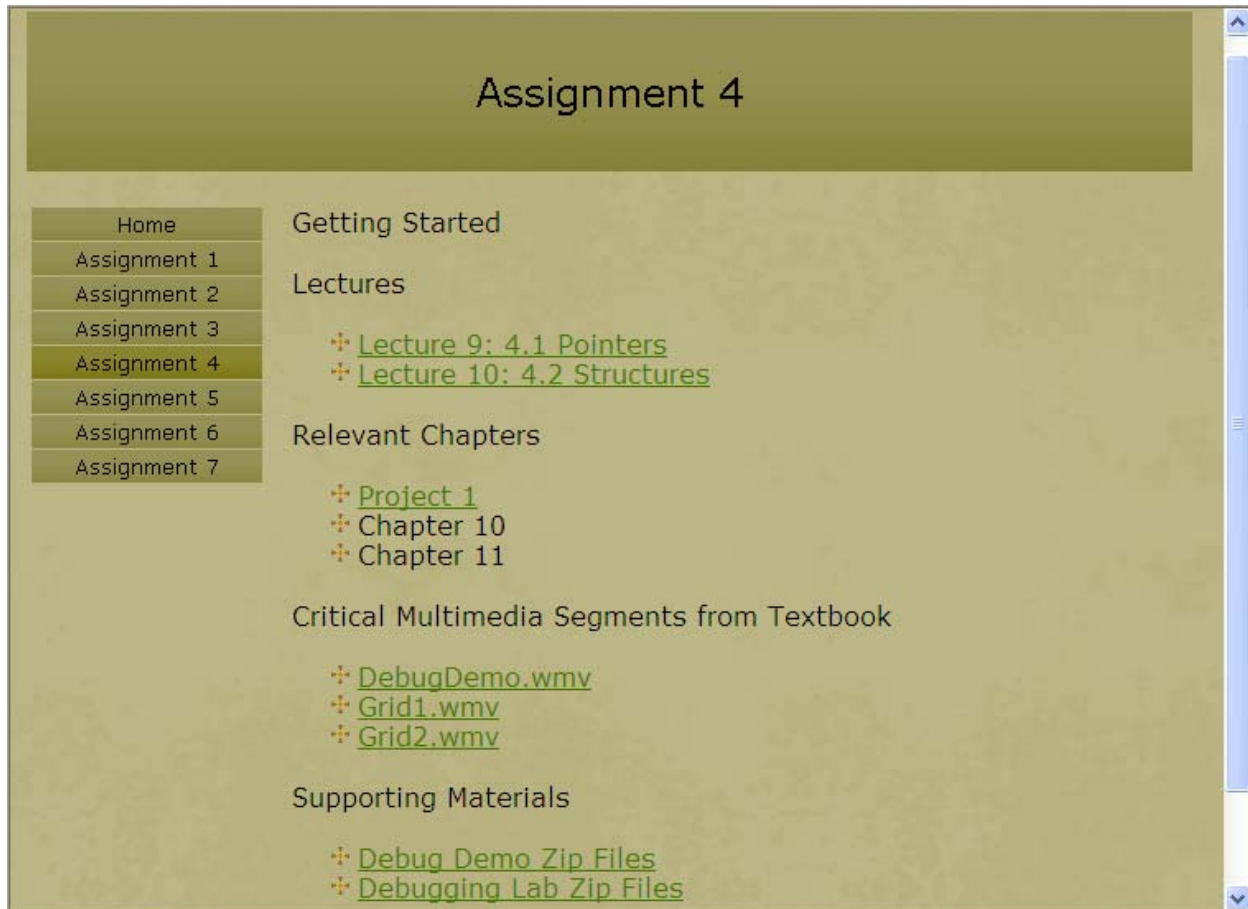
Raw Score	Validation Score	Verified by	Verification date:

Assignment 3

Raw Score	Ready to Validate (Assigned TA)	Date Ready:
Passed (Y or N):	Examined by (Signature of TA):	Date of Exam:
	If N, Ready to revalidate (Signature):	Date Ready:
Passed Retake (Y or N):	TA performing retake (Signature):	Date of Exam:
	If N, Ready to revalidate (Signature):	Date Ready:
Validated by Instructor	Instructor signature:	Date validated:

Assignment 4

Raw score on debugging part (out of 100)		Verified by	Verification date:
Grid Score	Raw Score	Validation Score	Verified by
			Verification date:

Exhibit 10: Assignment 4, ISM3232 Course Web Site, Summer 2004

Assignment 4

Home
Assignment 1
Assignment 2
Assignment 3
Assignment 4
Assignment 5
Assignment 6
Assignment 7

Getting Started

Lectures

- + [Lecture 9: 4.1 Pointers](#)
- + [Lecture 10: 4.2 Structures](#)

Relevant Chapters

- + [Project 1](#)
- + [Chapter 10](#)
- + [Chapter 11](#)

Critical Multimedia Segments from Textbook

- + [DebugDemo.wmv](#)
- + [Grid1.wmv](#)
- + [Grid2.wmv](#)

Supporting Materials

- + [Debug Demo Zip Files](#)
- + [Debugging Lab Zip Files](#)

Exhibit 11: Change to Email Policy

Mon, Jun 14, 2004 -- *Email alternative to weekly meetings*

A number of you have indicated problems attending weekly meetings. I, on the other hand, am concerned about the pace of many students in the class (given that most of you indicated to me that you were striving for an A). Here is my attempt to come up with a workable compromise.

If meeting weekly with your TA is too difficult, here's an alternative.

Each week, you can send an email to your assigned TA in the following format:

- 1) Activities I'd planned to accomplish last week
[N/A for the first message, will be a copy of item 3 from previous week's email]
- 2) Activities I actually accomplished last week
- 3) Activities I plan for the coming week

If the TA is concerned about your progress, he may request an in-person meeting with you. Should you fail to attend a requested meeting, I will be notified. Repeated failure (i.e., more than one) to attend either 1) requested meetings or 2) weekly meetings (without sending the TA e-mail) will lead to a - being added to your letter grade at the end of the semester.

Exhibit 12: New Incomplete Policy Announcement

Wed, Jun 30, 2004 -- Requirements for a temporary "I" grade

[This is a restatement a mass e-mail I just sent out]

Looking over the grade rolls, it appears that many of you may be somewhat behind in meeting your course objectives prior to the end of the semester (FYI, Grades must be turned in by July 16th). Since the requirements of the course are fixed, and will not change, the only adjustment I can make is to the timing. Here, then, is an option for a temporary "Incomplete" that you may choose to take. The requirements for getting an incomplete grade are as follows:

1) You *must* meet the requirements for a "C" in the course by the end of the course (July 15th, to give me a day to tabulate grades). As a practical matter, that means completing the oral exam on Assignment 3 and getting Assignment 4 validated. (This is necessary because Dr. Birkin will have to override anyone with an "I" to allow registration in subsequent courses for which Ism3232 is a prerequisite--so I have to be able to ensure him that every "I" student will get at least a "C" in Ism3232)

2) On or before July 15th, you must submit, to me, an email detailing your plan for completing additional course materials (ggill@coba.usf.edu). (This is necessary so I can distinguish "I" grades from "C" grades.)

If you should decide to take the "I", you will then have until Friday, August 27th 2004, to complete and validate additional material (e.g., Assignments 5 & 6) to increase your grade. No "late penalties" will be assessed, and I will post available times for validation exams on the Ism3232 web site (<http://ism3232.coba.usf.edu>). If you do not validate any additional assignments by that time, your "I" will become a "C". Otherwise, your grade will be changed to whatever your new score would earn.

You should be aware that no "I" grades will be given to students not meeting the requirements for a "C" in the class by the end of the semester. (The only possible exception to this would be circumstances that would normally justify an "I", such as verifiable health problems or bona fide family emergencies. These will be handled on a case by case basis).

I hope this will afford many of you the opportunity the get the "A" or "B" grades in the course that indicated you were striving for in your conversations with me at the beginning of the semester. Do not hesitate to contact me if you have any questions.

Regards,

Grandon

Biography



Grandon Gill is an Associate Professor in the Information Systems and Decision Sciences department at the University of South Florida. He holds a doctorate in Management Information Systems from Harvard Business School, where he also received his M.B.A. His principal research focus is in the area of IS education, and he has published many articles describing how technologies and innovative pedagogies can be combined to increase the effectiveness of teaching across a broad range of IS topics. Currently, he teaches programming, database and managerial courses to both undergraduate and graduate students.