Decision Sciences Journal of Innovative Education
Volume 3 Number 2
Fall 2005
Printed in the U.S.A.

TEACHING BRIEF

Assignment-Centric Design: Testing the Assignments, Not the Lectures*

Grandon Gill
Information System and Decision Sciences Department, College of Business Administration, University of South Florida, 4202 East Fowler Avenue, CIS1040, Tampa, FL 33620-7800, e-mail: ggill@coba.usf.edu

One of the principal practical challenges faced by programming instructors, particularly in introductory courses, is the diversity of student backgrounds (Roberts, 2000). The problem is particularly acute in Management Information Systems (MIS) courses, where the level of interest in learning to program is also highly variable because many MIS graduates pursue careers where programming is not a required job activity. These concerns are not hypothetical. In an Introduction to Programming in C++ course I taught as part of an undergraduate MIS major, 2003 survey data gathered from 116 students yielded the following profile:

- Fifty-two percent had previously taken at least one programming course (8% in middle school, 17% in high school, 24% in community college, 20% at the university level in non-business colleges, and 24% at the university level in a college of business). Twenty-seven percent had taken 2 or more previous courses. The remaining 48% had never taken a programming course.
- Thirty percent viewed programming as a somewhat attractive or very attractive career, while 27% rated it as very unattractive. Fourteen percent thought it pretty likely or very likely they would be programmers, 34% thought it very unlikely.

To compound the challenge of teaching the class, the course was normally taken in the first or second semester of the student's junior year, because the MIS concentration was part of a 2-year upper division business degree. Between 60 and 70% of all students entering that program had earned a 2-year associates degree at one of many surrounding community colleges, meaning even the same course numbers did not imply uniform preparation.

Presented with such diversity, it seemed inconceivable that a traditional programming course design (e.g., lectures, projects, exams) could adequately meet the needs of all participants—a supposition confirmed by numerous attempts. For this reason, a new design—the assignment-centric approach—was developed.

*Course materials and further information are available from the author (ggill@coba.usf.edu) and are also compiled in the instructor's manual for: Gill, T. G. 2005). *Introduction to Programming Using Visual C++.NET*. Hoboken, NJ: Wiley.

Figure 1: Course requirements (taken from Summer 2004 syllabus).

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

Using this approach, traditional midterms and examinations were eliminated, leaving only assignments in place. This new approach did not eliminate testing; it did, however, drastically change its focus, as shall now be described.

ASSIGNMENT-CENTRIC APPROACH

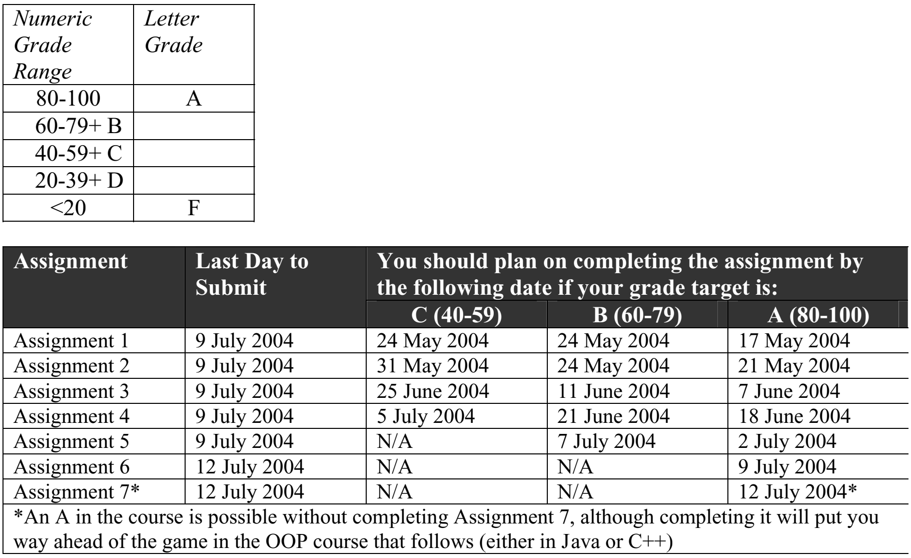
The principles of an assignment-centric course were simple, and were explicitly stated to all students on their first day of class:

- Completion of course assignments, in and of itself, meets all course requirements.
- The primary role of lectures and outside reading is to help students complete assignments.
- All testing is directed toward validating that students properly complete their assignments.
- The central focus of the course design process is the creation of appropriate assignments.

At the time these statements were made, students were also given all seven course assignments (Figure 1) and the final course curve with recommended due dates (Figure 2). There was no penalty for not handing an assignment in—only the loss of points that could have been awarded.

The spread of the grade curve should also be noted. It was specifically designed so that completion of the minimum number of assignments required to earn a C (the minimum acceptable grade for MIS majors) provided adequate background

Figure 2: Course curve and deadlines (from Summer 2004 syllabus).



Numeric Grade Range	*Letter Grade*
80-100	A
60-79+ B	
40-59+ C	
20-39+ D	
<20	F

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++)				

for subsequent courses in the major. Completing the requirements of an A grade, in contrast, implied a level of coverage far beyond what would normally be expected in an introductory programming course.

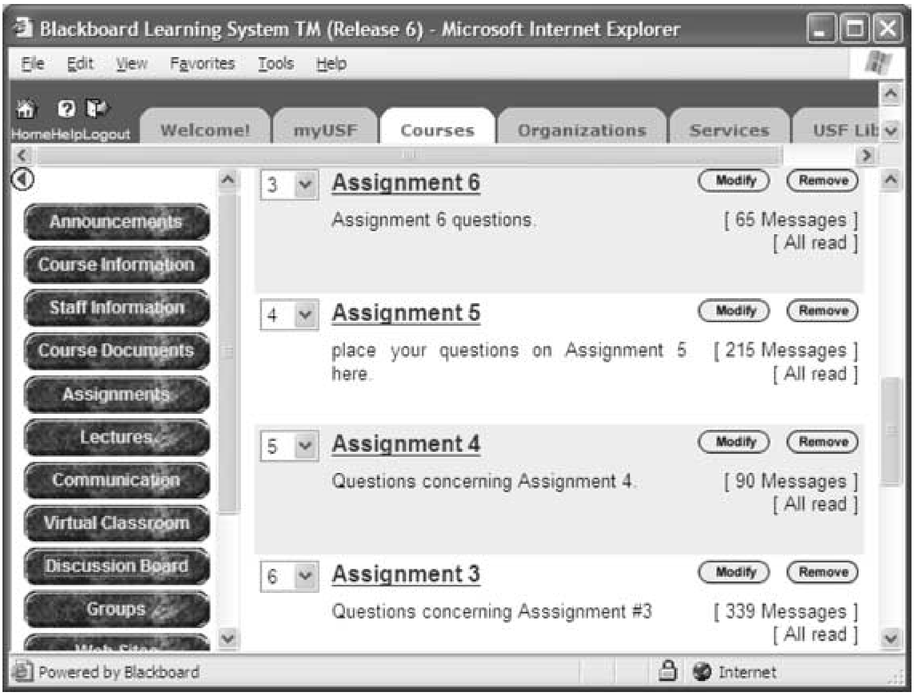
To support the assignment-centric approach, a number of changes to traditional course approaches were required. First, content needed to be provided in a manner flexible enough to support the wide range of paces at which students progressed. Multimedia versions of all lectures were therefore placed online, available for any-time/any-place delivery. Second, lab sessions needed to be oriented specifically toward assignment completion rather than teaching “general” principles. Otherwise, no students would show up. Third, technical support for assignment questions needed to be provided 7 days a week (because many students were employed and weekends were therefore “prime time” for completing assignments). This was accomplished primarily through the use of the school’s Blackboard course management system (Figure 3). Finally, and most critically, some mechanism for validating individual assignment performance was required.

VALIDATION EXAMS

The validation approach was one of the most unique aspects of the course design. Each time a student completed an assignment, he or she had to pass an exam specifically designed to ensure that the grade granted was a reasonable assessment of the student’s knowledge. Some characteristics common to all validations were as follows:

- Until the validation of an assignment was completed, no grade on the assignment was entered. It was as if the assignment had never been turned in.

Figure 3: Assignment discussion groups.



3 **Assignment 6**
Assignment 6 questions. [65 Messages] [All read]

4 **Assignment 5**
place your questions on Assignment 5 here. [215 Messages] [All read]

5 **Assignment 4**
Questions concerning Assignment 4. [90 Messages] [All read]

6 **Assignment 3**
Questions concerning Asssignment #3 [339 Messages] [All read]

- The intensity of the validation varied according to the performance on the assignment. Thus, a 95% score on an assignment meant a far tougher validation exam than a 60%.
- There was no set limit on the number of attempts a student could make in order to validate an assignment—other than the duration of the term.

The technique for validating each assignment varied according to the content. Two of the assignments were validated by online exams given in classroom labs, proctored by teaching assistants (TAs). These exams involved conceptual topics, and were generated using software developed by the instructor. That same software was also used to generate a practice exam test bank (that students could access online) and substantial portions of the assignments themselves. Using a common source to generate assignments, practice exams and actual exams served to guarantee that the exam performance would be a reliable measure of assignment understanding.

For each of the most heavily weighted assignments—the four major programming assignments—validation was accomplished using an individual oral exam with the instructor or a TA. A student would bring a copy of his or her completed and graded code to the exam. The student was then asked specific questions about the code, such as “What does this line do?” or “What would happen if we took this statement out?.” Considered “fair game” in an oral exam was any question that related directly to the code that the student has handed in. Questions testing more general understanding, such as a topic discussed in a previous week’s lecture, were out of bounds. Similarly, questions could only be asked on the parts of an assignment that were completed and graded. If a "C" was the student's ultimate goal, parts of early assignments might be omitted (to avoid difficult exam questions) and later assignments could be ignored.

OUTCOMES

The move to an entirely assignment-centric design was not instantaneous. Rather the technique evolved over a 3-year period. First, oral exams were conducted on the last two assignments—which could then be substituted for the final. In time, less than 10% of the class was taking the final, and the process was instituted for earlier assignments. Then attendance of the midterm began to shrink, as students opted for all-assignment options. Eventually, final and midterm exams were both dropped. This move precipitated the development of online validation exams for the two "concept" assignments. Throughout its evolution, course design was demand-driven; a more conventional path through the class remained available to students until there was an overwhelming consensus in favor of the pure assignment-centric approach.

There are a number of indicators of the success of the approach. During its evolution, the amount of "available" course content was increased by roughly 30% (consistent with the goal of challenging the top students). Also, an extensive questionnaire was developed and administered to every section starting in spring 2003. Participation was voluntary, but provided extra credit and 60–70% of students filled it in. A number of results are listed in Table 1. Three outcomes were particularly indicative of the assignment-centric approach accomplishing its goals: (i) students reported that they felt the validation exams were fair, despite the fact that when content was expanded the average course GPA awarded dropped to about .5 points below the departmental average—reflecting the increased difficulty of getting a B or an A, (ii) students did not want the course to rely more heavily on tests, and (iii) students reported spending far more time on the course than they spent on other department, college, or university courses—evidence that lower work load was definitely not motivating their preference for the assignment-centric design. While these self-reported time estimates are unlikely to be accurate, it should be noted that they have moved consistently with course changes, rising both as content was added and in summer terms—when the same body of material must be covered in 10 weeks versus the normal 16. The time estimates also correlated with an objective measure, Blackboard discussion board "hits" (.268, $p < .05$).

With respect to diversity of backgrounds, not a single significant difference was found between the groups with and without programming experience for 16 different measures of course satisfaction—as well as the other Table 1 measures (right column). In other words, the approach presented in this brief was able to accommodate diverse backgrounds and motivate unusually high time-on-task levels while being perceived as fair.

GENERALIZABILITY

In closing this brief, it is worth considering the domain of subject matter for which assignment-centric design would be most effective. Some speculation is

Table 1: Results of course surveys for 2003 (116 total responses).

			No to	Yes to	
	Scale	Mean	Previous	Previous	Sig.
Item	Code	(count)	Course	Course	Diff?
The online validation exam for Assignment 2 provided a fair assessment of my knowledge at the time	A	4.4 (33)	4.4 (16)	4.4 (16)	No
The oral exam on Assignment 3 provided a fair assessment of my knowledge at the time	A	4.4 (110)	4.3 (53)	4.4 (57)	No
The online validation exam for Assignment 4 provided a fair assessment of my knowledge at the time	A	4.2 (33)	4.2 (19)	4.3 (14)	No
The oral exam on Assignment 5 provided a fair assessment of my knowledge at the time	A	4.1 (92)	4.1 (44)	4.1 (48)	No
The oral exam on Assignment 6 (& 7) provided a fair assessment of my knowledge at the time	A	4.0 (62)	3.9 (30)	4.0 (32)	No
Time per week spent on the class	T	17.3 (110)	16.2 (55)	18.3 (55)	No
Time spent per week on other MIS classes	T	7.8 (105)	7.4 (49)	8.1 (56)	No
Time spent per week on other business classes	T	5.4 (106)	5.7 (55)	5.0 (51)	No
Time spend per week on other classes in the university, outside the College of Business	T	4.2 (89)	4.3 (46)	4.1 (43)	No
Satisfaction with number of assignments	S	3.6 (112)	3.6 (53)	3.5 (59)	No
Satisfaction with type of assignments	S	3.4 (112)	3.3 (53)	3.4 (59)	No
The course should put greater emphasis on tests and less on assignments	A	1.9 (112)	2.0 (54)	1.8 (58)	No
Overall rating of the instructor*	E	3.98 (88)	3.93 (44)	4.02 (44)	No

Scale A:
1—Strongly disagree
2—Mildly disagree
3—Neutral
4—Mildly Agree
5—Strongly Agree

Scale T:
Estimated hours per week

Scale S:
1—Not at all satisfied (or very dissatisfied)
2—Somewhat dissatisfied
3—Neutral
4—Somewhat satisfied
5—Very satisfied

Scale E:
1 (Poor) to 5 (Excellent)

*Taken from university's course evaluation forms for non-summer sections. Typical results for technical ISM courses are approximately 4.0, but vary considerably.

Table 2: Characteristics of assignment-centric domains.

Characteristic	Explanation	Good Fit	Weak Fit
Depth of understanding is preferable to shallower understanding of more material.	The assignment-centric approach implies not all students will attempt all assignments, meaning not all assignments can be "critical."	Areas where skill acquisition is cumulative.	Survey courses.
Emphasis on understanding complex problems or skill acquisition rather than factual content.	Understanding can be readily assessed during validation exams—especially since being able to explain one part of a complex problem nearly always requires demonstrating understanding of related parts. Acquired skills, in turn, can usually be demonstrated fairly quickly. Factual recall, on the other hand, would require a much longer time to spot-check during validation, since facts are often learned independently. Furthermore, such recall-based exams break the tight linkage between assignment and validation that is part of the "contract" with the student.	Quantitative methods courses.	Course that traditionally rely heavily on memorization.
Projects can be defined whose creation or analysis requires most or all of the skills/knowledge the course is designed to impart.	In an assignment-centric approach, all use of books, materials and lectures is essentially demand-driven—meaning students will only attend to the content if they need it to complete the assignments.	Areas where objective analytical techniques are being taught and problems requiring these techniques can be readily defined.	Areas where problems tend to be highly subjective, turning validation exams into debates.
Situations where student backgrounds, abilities and/or interest levels tend to be highly variable.	The assignment-centric approach, by virtue of being self-paced, effectively provides separate paths for weak-background/low-interest students and those students likely to be highly motivated or familiar with the subject matter.	Areas likely to be a career option for some students but not for others (e.g., programming for MIS majors).	Areas where a course is a prerequisite for other required courses that expect uniform preparation.

involved here because I have only observed two situations where it was used: in teaching programming and in nuclear submarine training (Gill, 2005). Common threads from the two contexts are identified in Table 2. To summarize, the approach seems best suited for domains where: (1) students come in with a broad range of goals and experience, (2) the objective is to teach students specific techniques or skills, and (3) projects can be defined that require the use of most or all of the techniques be taught. The approach seems least applicable to domains with highly subjective content, where the course objective is to survey all facts in the domain or where subsequent courses presume familiarity with *all* course content. It should be noted, however, that the last of these sources of poor fit can, to some extent, be mitigated where the follow-on courses are also taught in assignment-centric fashion to compensate for non-uniform preparation (as is the case with an assignment-centric object-oriented course that I teach to students who have completed the introductory course). [Received: April 2004. Accepted: July 2004.]

REFERENCES

Gill, T. G. (2005). Teaching C++ submarine style. *IEEE Transactions on Education, 48*(1), 150–156.

Roberts, E. (2000). Strategies for encouraging individual achievement in Introductory computer science courses. *SIGCSE' 00*, March, 295–299.

T. Grandon Gill is an Associate Professor at the University of South Florida. He received his MBA and DBA from Harvard Business School. He has numerous published research and teaching case studies, as well as other education material. His research interests are currently focused on distance learning, organizational learning, and MIS education. His publication include articles in *MIS Quarterly, IRMJ, Data Base, Accounting, Management and Information Technologies,* and *Education and Information Technologies*.