Pulling The Plug On The Pedagogical Pacifier By Placing Programming Courses On-line

Thomas Walker, Jan Helge Bohn
Virginia Polytechnic Institute and State University

Introduction

For decades, engineering classrooms have changed little. The blackboard is still very prominent in engineering classrooms as is the overhead projector. These two appliances are used more than any other to present material to engineering students in the classroom. Certainly, some classrooms have other mediums present that are used occasionally; the TV/VCR combination, the computer monitor, the computer/LCD projector are examples. However, in most cases, these are used in such a way that they are simply color blackboards and color overhead projectors.

There has been, on the other hand, continuous improvement in engineering text books; most significantly in the quality of the graphics and in the use of color, but also in overall readability. Additionally, many texts come with software to assist the student and the instructor. The CDROM that comes with the classic Halliday and Resnick physics text is an example.[1] The reason for this continuous improvement is simple – there is tremendous competition among publishers and they are constantly pushing their authors to produce new testbook editions with multimedia content and/or computer-aided learning modules.

The most notable change is in student "equipment". Powerful, programmable calculators with graphical user interfaces and constant memory allow students to solve differential equations and LaPlace transforms easily. Many of these calculators now have infrared input/output capability for connection to a personal computer. Personal computers offer the students more than the main frames of thirty years ago because of inexpensive application software, Web browsers and EMAIL. CAD software has replaced the drafting kits of yesterday. Availability is a key factor – these tools are at the student's beck and call "24 by 7".

Of course, instructors have access to the same tools with the same availability. The common bookkeeping tasks have become easier as well as those involving word processing and presentation. Instructors communicate with their students and have Web pages for their courses. Some instructors use internet "chat rooms" and "list servers" to encourage student interaction. Using software such as Microsoft's Netmeeting, an instructor can take control of a student's computer remotely and simultaneously talk to them over the internet. Even a small color camera

can be used to transmit video between students and faculty. "Whiteboards" can be shared with numerous individuals participating in a virtual meeting.

Surprisingly, the new instructor and student equipment and improved textbooks have not produced much change in the actual content or pedagogy of engineering courses.[2] The engineering classroom has not changed and that is a reflection of the lack of substantive change in how engineering is taught or what is required of engineering students. For example, instructors are not lecturing less because the textbooks are vastly improved and come with multimedia software. Therefore, students do not use the software[3] nor do they take advantage of the improved textbook. While much of this might be due to the rise of aliteracy in our culture,[4] much of it is do to a lack of requirement. Course examinations and homework typically do not take advantage of the powerful student calculators. Therefore, most of the students do not even know how to use any features other than the "slide rule" features. In other words, the students are being "pedagogically pacified" and the typical instructor is at fault. This is not beneficial to engineering students. Instructors need to force a radical change into their courses and the students will do what is expected of them.

Programming Language Courses as Change Agents

Most undergraduate engineering curricula still require a course in a programming language. Modern programming languages provide an excellent opportunity to wean students off the pacifier because:

- They are comparatively easy to learn. Many high school students have taught themselves computer programming just for the fun of it.
- There are numerous "Teach Yourself…" and "XXXX For Dummies" books in addition to superb textbooks for programming languages.
- Many of the texts come with multimedia CDROM software and/or interactive tutorials.
- The modern programming environment is very user-friendly, consisting of a completely integrated development environment (IDE) with editor, compiler, linker, debugger, and on-line help facility. The tools run extremely well on the modern PC so students have total access to them.
- Programming requires active participation by the student. They have to type in the program, compile it, and run it. This involves simultaneous physical and mental activity, the best learning environment.[5]

  > "Research on learning has demonstrated that students understand best, remember ideas most effectively, and think most incisively when they feel personally responsible for getting meaning out of what they are learning instead of waiting for a teacher to shovel it into them."[6]

- Students receive immediate feedback from the programming environment when they test their programs. They do not need to wait for the instructor to grade their paper.

However, it's not just the student who needs to be weaned. The faculty needs to similarly be weaned into the 21st century and this change will require powerful incentives. With respect to programming language instruction, one of the incentives is that the entire process of distributing, collecting, grading, and checking for plagiarism can be automated. This represents an incredible savings of time for faculty and students.

On-line C++ Programming Courses at Virginia Tech

The Engineering Fundamentals Division in the College of Engineering at Virginia Tech began to teach C++ as a programming language in Fall, 1998, with a 2-credit introductory course for the Mechanical, Civil, and Industrial and Systems majors (EF 2314). This course was followed up in the Spring with a 3-credit course designed for Electrical and Computer Engineers (EF 1574). From the outset, these two courses were unique in that the homework was collected, graded, and checked for plagiarism by a computer. Homework grades were posted to each students individual Web page on the server. Over 200 students took EF 2314 in the first offering. Over 4000 submitted homework programs were graded. The computer highlighted approximately 90 cases of plagiarism involving 43 students resulting in a 100% conviction rate when reviewed by the Virginia Tech Honor Court. Two instructors were assigned to that course with no teaching assistance and both instructors also had other teaching and advising responsibilities. The first offering of EF 1574 served 325 students with three instructors, again with no teaching assistance. The computer "grader" evaluated over 8000 homework programs and identified 16 cases of plagiarism. In both of these course offerings the instructors provided a course Web site for major course documentation and instructions as well as the specific homework assignment instructions. Two list servers were provided for each course, one for student discussion and one for faculty announcements. The student discussion server was monitored by the instructors and they often contributed to it. Otherwise, the course offerings were traditional, with lectures. All instructors noticed that lecture attendance was sparse, most notably in EF 1574 but also in EF 2314. Students requested that the course presentation slides be posted to the course Web site. This was not done in EF 2314 but the slides were posted for EF 1574.

In the Spring of 1999 the authors received a grant from Virginia Tech's Center for Innovation in Learning. The vision was to develop EF 2314 and EF 1574 as 100% online courses available in a true distance-learning environment. With this goal in mind, EF 2314 was offered twice in the Summer of 1999 (once in a concentrated "Maymester" and once in a typical six-week summer session). Supplementary reading modules were posted to the course Web site for each assignment. There were no lectures but optional classes were provided to answer student questions. EF 1574 was offered in a standard six-week summer session and the format was the same as the previous semester. Throughout the summer, the authors continually improved the on-line components of the two courses and developed strategies for the Fall, 1999, semester course offerings.

Both courses were offered again in the Fall. EF 2314 was to be a fully on-line course with no class meetings. There were 220 students enrolled in the course with three instructors. EF 1574 was also offered with 168 students and one instructor. Classes met for EF 1574 but for student questions only, no lectures. Lecture slides from the previous semester were posted to the Web.

EF 1574 class attendance was extremely sparse – less than twenty students total attendance and always the same students. Of these students, the same seven or eight asked questions. The same sequence was followed in EF 2314. However, the students in EF 2314 began complaining about the lack of traditional instruction early in the course and the decision was made to hold two question and answer sessions each week to provide student-instructor contact. Additionally, ten hours per week of undergraduate student assistance was provided to the EF 2314 students. The combined courses produced computer grading of more than 10,000 homework submissions, fifteen graded computer programming assignments per student.

Course Web Site Design

The EF 1574 course homepage is shown in figure 1, (EF 2314's page is similar).

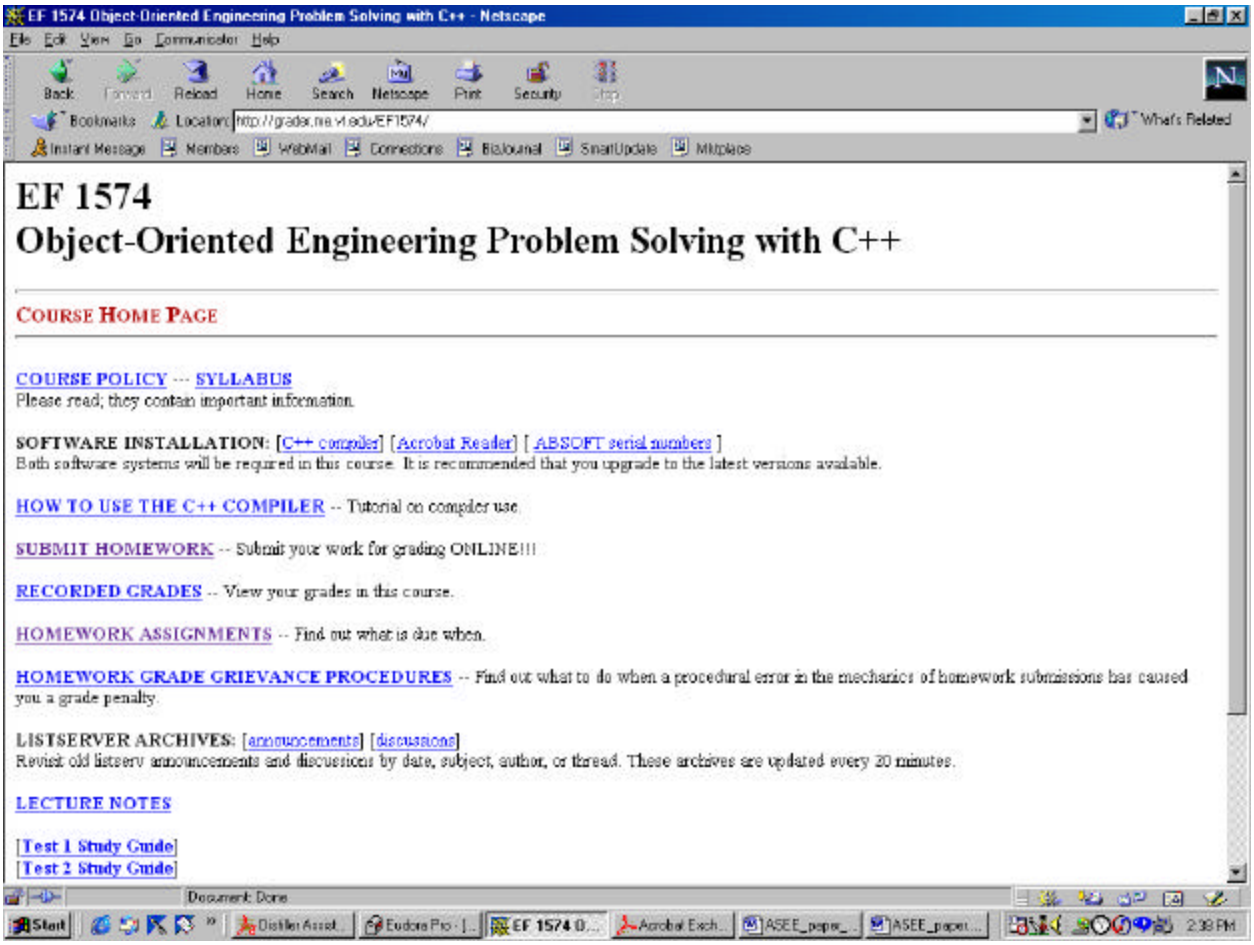

Figure 1

The course Web sites were developed to be as simple as possible for two reasons:

- to keep the bandwidth requirement as small as possible so that students with only modem access can still use the Web site efficiently
- to allow instructors to update and edit the pages easily

Experience has shown that the latter is probably more important. Instructors do not have time to keep up with the latest extensions to HTML, therefore pages were deliberately designed to be editable using the built in browser editor in Netscape or Internet Explorer.

The homepage is available publicly, without password. All other links require a student to be registered in the course to gain access. The site map is also kept as simple as possible for the same reasons noted above.

The "HOMEWORK ASSIGNMENTS" link takes the student to figure 2.

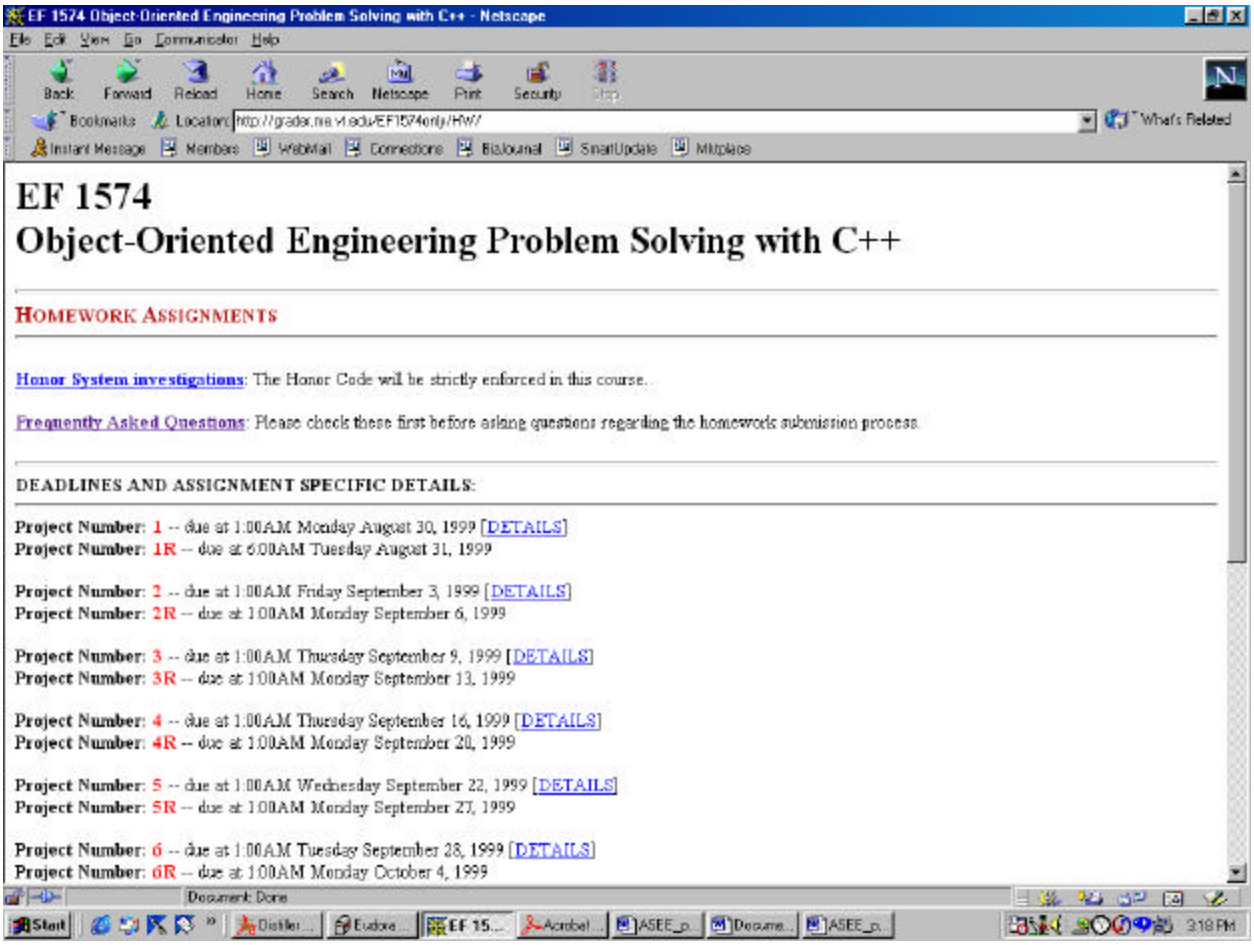

Figure 2

The "DETAILS" link takes the student to a PDF document explaining what is required in the programming assignment. The details are the same for the regular assignment and the "R" assignment. The difference between these will be explained later.

If a student has prepared an assignment for submission to the grader, they simply select the "SUBMIT HOMEWORK" link from the home page which takes them to figure 3.

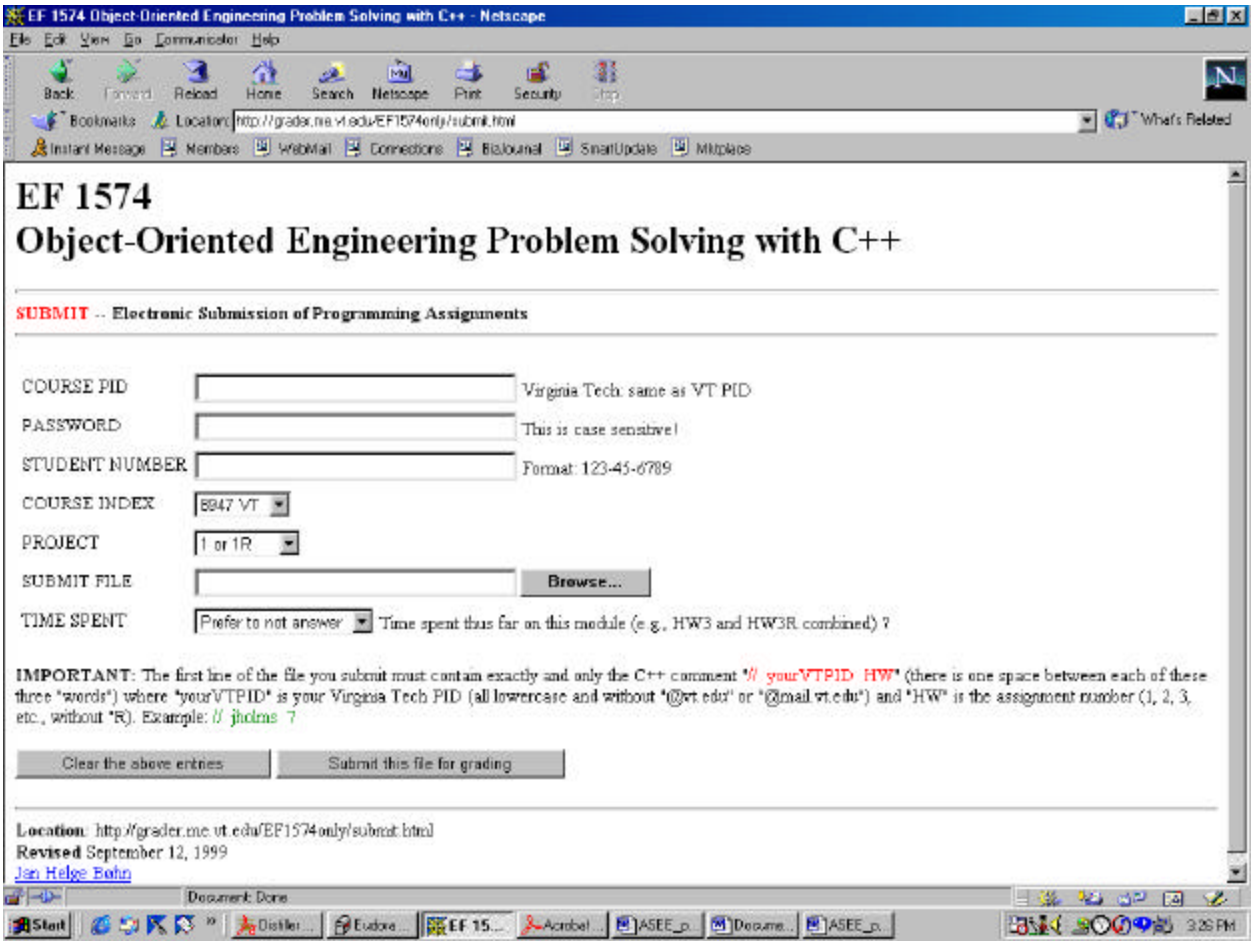

Figure 3

This Web page has evolved the most throughout the different course offerings. Every possible precaution has been taken to eliminate the possibility of a student's grade being affected by a simple mistake during homework submission. It should be noted that the homework can be submitted from any computer, anywhere, as long as it has Web access and a reasonably current edition of Netscape or Internet Explorer.

Project Grading

The course server handles the grading of all projects including accepting project submissions, grading, and posting results. The process involves having the students submit only source code, compiling that code, executing it, and comparing the project output with the output obtained by the solution already existing on the server. Each project is tested with input data published with the details of the assignment (published data set) and also with an unpublished data set. The results of both tests are posted to the student's individual Web page as shown in figure 4. The page is cumulative, showing the grader output from all submissions. The results from the published data set are displayed at the top of the Web page in the format shown. The output from the student's homework submission must match character by character with the grader's expected output to receive full credit.

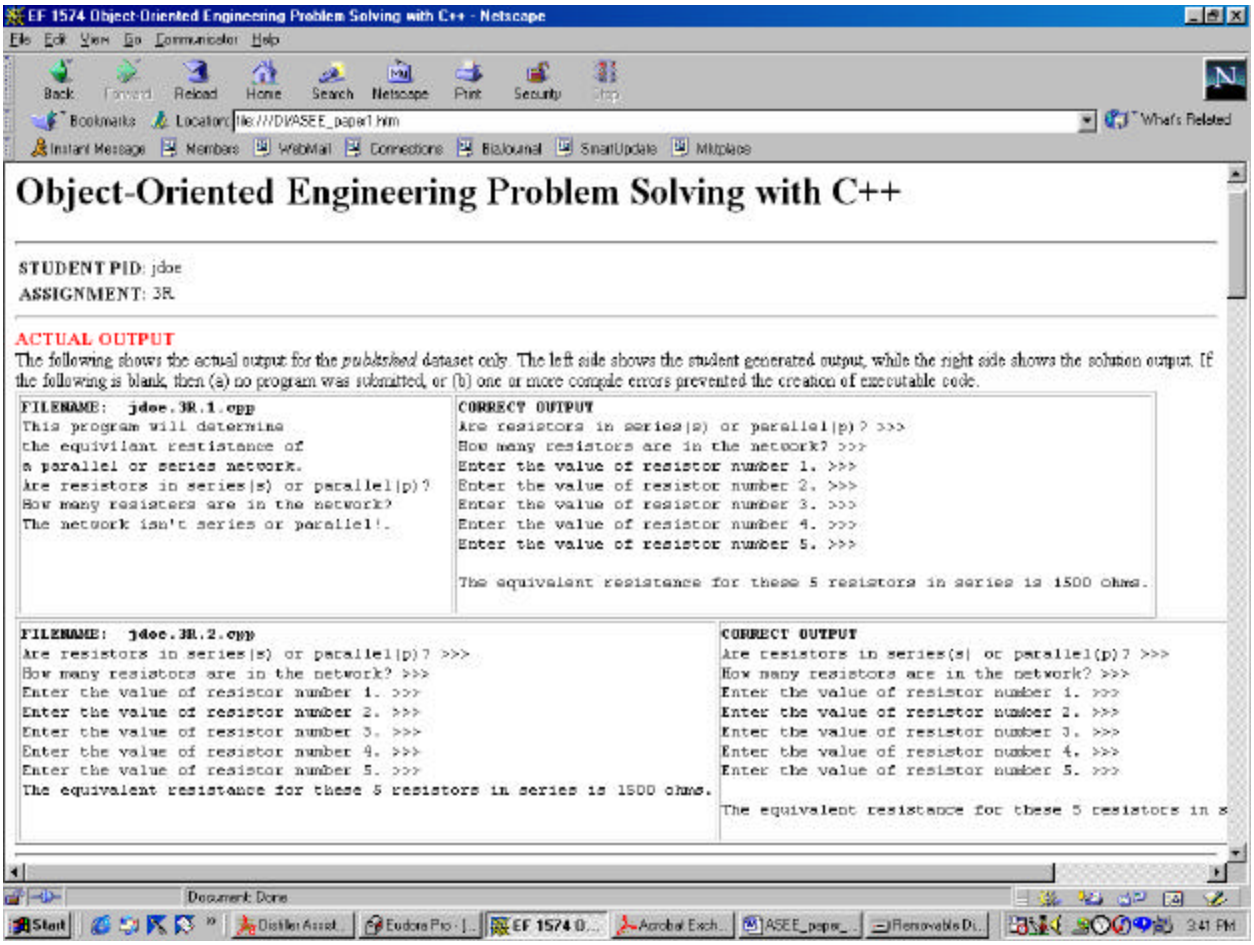

Figure 4

To keep from revealing the unpublished data set, the results of the test with this data are published at the bottom of the Web page as shown in figure 5. The color-coding in this page is important. The student's incorrect output is indicated in red and preceded by ">" while the expected grader output is printed in green and preceded by "<".

After the assignment submission deadline, a plagiarism program is executed which compares each student's submission with every other student's submission. If the project is the same as one assigned in previous semesters, it cross checks with those student submissions as well. The results are presented in a spreadsheet format showing how similar the files are. Final visual verification is used prior to filing charges of Honor Code violations. This system has been very effective at curtailing cheating. With a 100 percent conviction rate of those charged, the program's reputation for identifying cheaters is firmly established and tends to be fostering increased individual work. The authors feel that this is absolutely necessary when student homework counts for a large part of the course grade (30%), especially in light of recent reports on the level of cheating.[7]

Student Concerns

In the initial offerings of both courses (lecture with on-line submission), the students' only concern was with automatic homework grading. Since the grader operated under UNIX and the students were using Windows, the grader compiler was not the same as the student compiler. Additionally, the ANSI/ISO C++ standard was not finalized until July 1998. These two facts combine to cause occasional slight variations between the output on the grader and the output

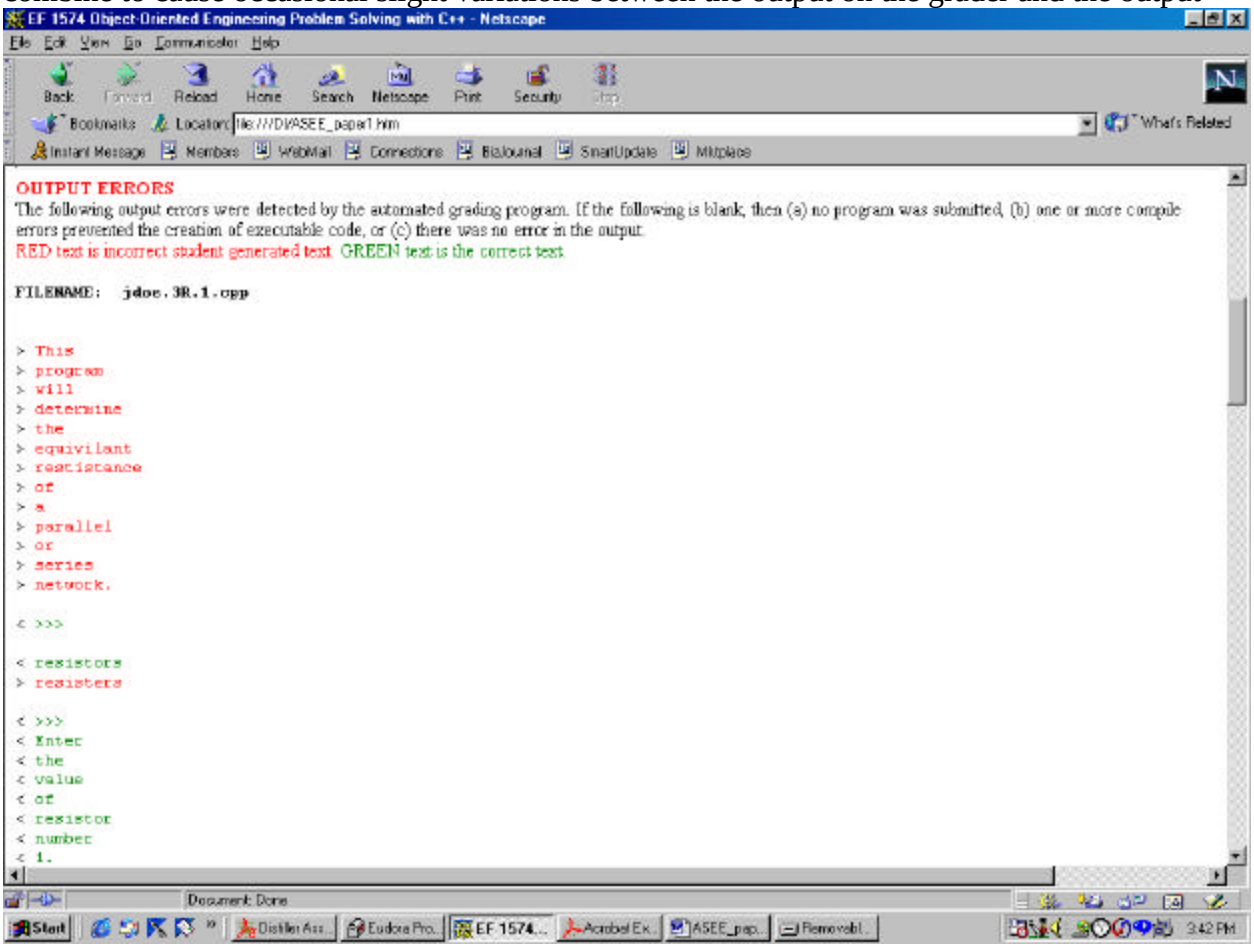

Figure 5

on the student computers for the same source code. To account for this, the grader allows multiple submissions (currently 4) of the homework before the initial homework deadline. A student can receive full credit on any of these submissions. Some students still do not like having to match the grader's output exactly, even though they are advised to "cut and paste" from the details of the assignment to prevent formatting or spelling errors. The real problems occurred when procrastinating students waited until the last minute to attempt a homework. The initial offerings of the courses only graded homework once daily. If a student made a spelling error on output prompting, the highest grade they could receive was a 70 percent. The grading program was changed in the Fall of 1999 to grade and post results every hour except between 1:00 A.M. and 6:00 A.M., when the server is down for maintenance. In an attempt to dissuade students from procrastinating, one of the submission opportunities is removed if it is not used 72

hours in advance of the published deadline. Another submission opportunity is removed if it is not used 48 hours before the deadline.

The authors also instituted a second set of submission requirements for all projects to ensure that the students mastered the topic covered by each homework assignment prior to continuing on in the course. These are the "R" projects shown in figure 2. If a student does not receive a grade of 80 percent or better before the initial homework deadline, they have three additional opportunities to pass the homework prior to a second deadline. These submissions are evaluated just as the original submission but do not change their assignment homework grade. The students must have an evaluation of 80 percent or better on the assignment prior to the expiration of the second deadline or a 5% penalty is imposed on their final course grade. Even though students have up to seven opportunities to avoid this penalty, some of them feel that this is "double jeopardy" and not fair.

When lectures were stopped, several students complained that this was unfair and that they could not learn by reading the book along with the on-line information. It is the authors' combined opinion that these students are examples of the aliteracy permeating student culture, aliteracy defined as "I can read but I won't."[8] This complaint has been addressed by holding "question and answer" sessions for the students but not lectures. Since many complaining students have not read enough to understand what questions to ask, these sessions have not stopped the complaining, however, it has eliminated the perception that the courses do not provide enough student/faculty interaction.

Assisting Students On-line

To encourage students to help each other, the instructors provide a monitored discussion list server for student use. The course policy allows students to provide assistance to each other as long as that assistance does not include actual C++ syntax to solve a homework problem. Most project assignments require some kind of problem solving. Students can solve the problem, flowchart, and/or pseudocode the problem together. However, the specific C++ code implementing the algorithm must be the student's own code. The instructors participate in the discussion, sending out hints and advice when appropriate and making sure that the subject matter of the emails does not violate the purpose for the list server. Students who use the list for reasons other than its intended purpose have their posting privileges revoked. Since the list is archived on the course Web page, this revocation does not keep the student from reading the assistance given to others.

The course policy requires students to make an initial submission to receive any kind of assistance. Once a student makes a project submission, the instructors have direct access to the student's source code and grade page so that they can provide assistance via email. Instructors will not assist a student who does not make an honest attempt at the project. This policy is important because significant numbers of students have to be taught that they need to work on projects themselves before seeking assistance. The policy does, however, foster some student complaints.

Finally, the students always know where they stand with respect to the rest of the class by looking at their grade page. The bottom portion of that page shows a course histogram as displayed in figure 6 with the student's position highlighted in red.

Instructor Pedagogical Concerns

The primary instructor concerns come from EF 2314. The student attitude in that course is substantially different from the attitude of the students in EF 1574. The latter group of students take for granted that they need to know how to program (this is the course for electrical and computer engineers). They adjust to the course quickly, provide appropriate assistance via the discussion list, and seldom complain. As a result, one instructor can handle the EF 1574 course with up to 200 students. Significant numbers of the students in EF 2314 do not feel they

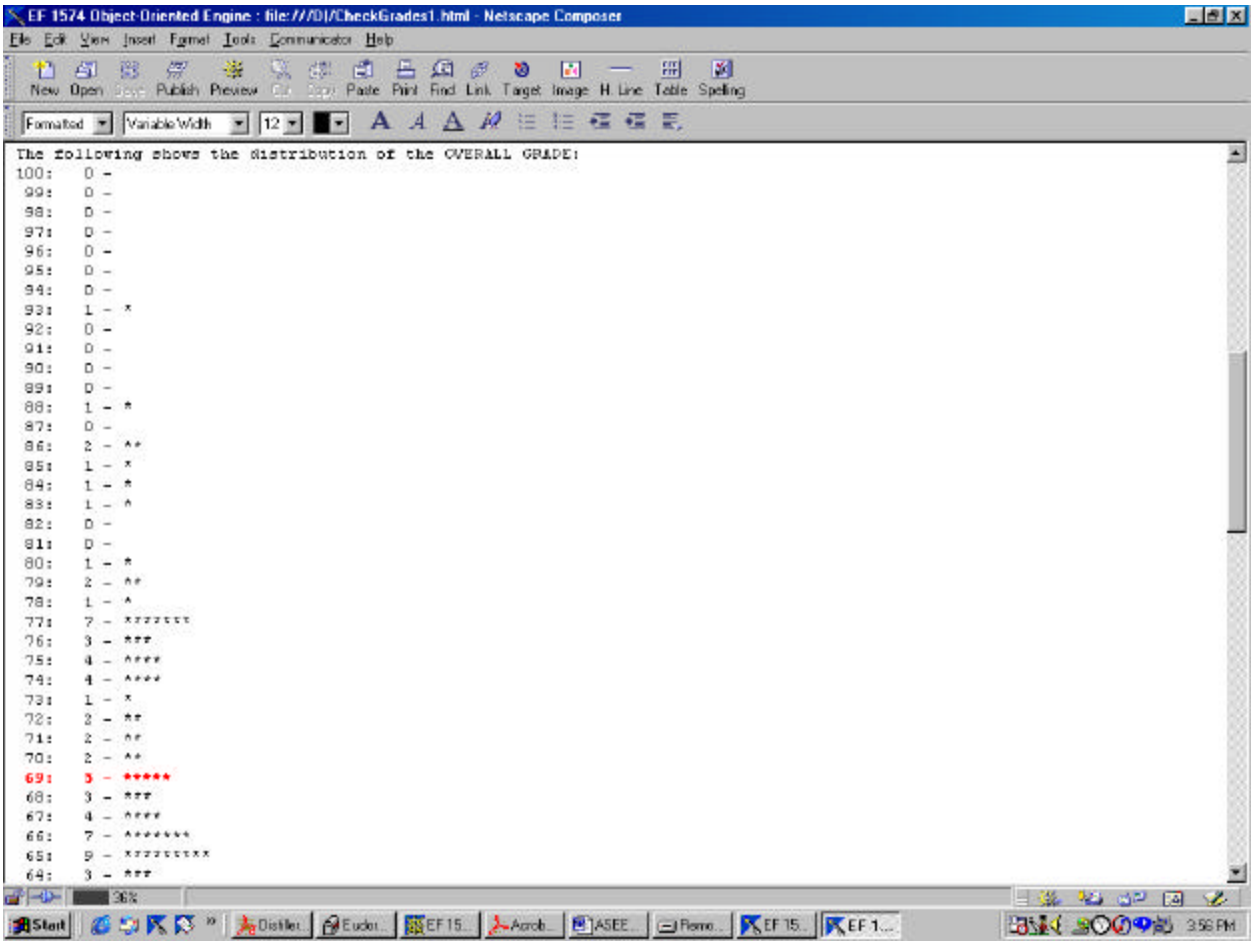

Figure 6

need to know how to program and they clearly dislike the course. Additionally, they have bombarded instructors with requests for one-on-one homework help sessions during instructor office hours. This has resulted in extensive faculty time requirements and is very inefficient. The instructors often voice the opinion that it would be better to use standard lectures than to continue the course in its present on-line format. The authors attempted to alleviate this problem by providing an accomplished undergraduate student to assist students with problems. A policy

was also instituted requiring students to attend the normally scheduled question and answer sessions and visit the undergraduate assistant prior to asking the course instructors for help. This policy has worked well, but again reinforces student feedback that there is not enough student/faculty interaction.

Conclusions

Computer programming courses can be offered in an on-line format with relative ease. Even more importantly, computer homework can be graded and checked for plagiarism with very little human intervention, allowing more homework projects to be assigned in a given semester.

A significant subset of students finds it difficult to do well in a programming course, on-line or otherwise. The exact requirement of the language syntax appears to provide more frustration to these students than it does to others. Offering the programming course in an on-line-only format appears to increase the level of frustration for this student subset. Face-to-face assistance must be provided for these students.

The majority of students transition well to the on-line format and the material covered. This adjustment is easier when the students perceive that they need to be knowledgeable in the programming language for follow-on courses.

Another subset of students thrives in the on-line environment. They appreciate the freedom of a lecture-free environment and enjoy being able to advance at their own pace. They participate well in the course chat rooms or listservers.

It is time that engineering instructors and students take advantage of the modern computing equipment, software, and engineering textbooks that are available. The contemporary engineering classroom should include student computer access and some in-class graded, computer-based work.

Finally, there is no question that the engineering profession requires “life-long learning”. We should be preparing our students to take advantage of the synchronous and asynchronous distance learning opportunities that will be available to them after graduation. The vast majority of these will require remote participation via computer link. On-line engineering courses (or components of courses) are good preparation.

Bibliography

[1] Halliday, D., Resnick, R., & Walker, J., CD-Physics, version 2.0, John Wiley & Sons, 1997

[2] Jones, J.B., “The Non-Use of Computers in Undergraduate Engineering Science Courses,” *Journal of Engineering Education*, vol. 87, no. 1, 1998, pp. 11-14

[3] Ibid.

[4] Healy, Jane M., *Endangered Minds – Why Children Don’t Think and What We Can Do About It*, Touchstone, 1990, pp. 23

[5] Ibid, pp. 297

[6] Ibid.,

[7] Kleiner, C., & Lord, M., “The Cheating Game”, U.S. News & World Report, Nov. 22, 1999, pp. 54-66

[8] Healy, pp. 23

JAN HELGE BOHN
Jan Helge Bohn is an associate professor in the Department of Mechanical Engineering at Virginia Tech. His research interests center about geometric modeling for product realization. He received his BS in computer science and MS and PhD in computer and systems engineering from Rensselaer Polytechnic Institute in 1988, 1989, and 1993, respectively.

THOMAS WALKER
Thomas Walker is an associate professor in the Division of Engineering Fundamentals at Virginia Tech. His research interests center about the proper use of computer technology in engineering education and distance learning. He received his BS in electrical engineering from Purdue University in 1973 and his MS in mechanical engineering from Naval Postgraduate School in 1978