OMEN: An Online Grader for Engineering Programming Courses

T.D.L. Walker, J.K. Goodman
Engineering Fundamentals Division[1]
Virginia Polytechnic Institute and State University

Abstract

OMEN (Online Materials Education Network) is a system developed within the Engineering Fundamentals (EF) Division at Virginia Tech for the distribution and grading of programming projects in courses where programming assignments are made in C++ or Fortran 90. It can be used to distribute and grade programming projects in any language that supports command line compilation, linking, and execution from within a LINUX environment. The system was developed using "free" software operating systems, languages, database systems, and chat/collaboration facilities wherever possible. The system has been used by the Division for the past six semesters to provide grading for over 2000 students, each submitting up to 14 different programming projects with multiple submission opportunities and "instantaneous" grading. Continuous improvements have been made to the system throughout that time.

This paper will describe the OMEN system in detail, its strengths and weaknesses, how and why it was developed, and how it is used. The paper will also discuss lessons learned during development, how they have impacted continued improvement and teaching, and goals for future versions.

Background

The EF Division at Virginia Tech advises and instructs approximately 1250 new freshmen engineering students each year. Since 1984, these students have been required to have their own IBM compatible personal computers and, for several years, all dorm rooms have had high-speed Internet connectivity. The VT College of Engineering has no large "computer labs" for computer instruction.

The instruction component of the EF tasks is accomplished primarily using a required two course sequence (2-credit hours each). Prior to the fall of 1998 approximately one-third of the first course was programming language instruction (most recently, Fortran 90). During a complete review of the first year curriculum, the decision was made to drop the programming instruction material from the first course and to teach whatever computer programming language(s) the other engineering departments desired in separate courses. This resulted in a 1-credit hour Fortran 90 course (EF 2324) and a 2-credit hour C++ course (EF 2314). The Electrical and Computer Engineering Department decided to offer its own 3-credit C++ course (ECE 1574). Some departments dropped the programming language component from their curriculum completely.

The Problem

The first course offering of EF 2314 was scheduled for fall 1999. At that time, there was only one available C++ programming instructor in the division (Professor Walker) and a projected C++ course load of 200 – 300 students per semester. Obviously, teaching resources were a major issue. However, with the lack of large computer labs in the college, material delivery was just as critical. To state the obvious, computer programming is learned by programming on computers and, at Virginia Tech, the computers were in the students' dorm rooms.

> PROBLEM STATEMENT: How do you deliver effective programming language instruction to a large number of undergraduate engineering students with a very small number of instructors, no GTAs, and no computer labs?

A Possible Solution

The mechanical engineering department was one of the departments requiring C++ instruction and Dr. Jan-Helge Bohn[2], a systems engineer, is a member of that faculty. He stepped forward and volunteered to help the EF Division develop a WWW-based course delivery and homework grading system and to act as a visiting instructor until the system had been developed and proven. It was hoped that this would accomplish two things:

- Shift the student's focus from lecture to hands-on programming (remember – no labs)
- Reduce the number of faculty required since homework grading would be done by computer

Knowing that the computer science department had similar instructor resource issues in their first programming course (C), Dr. Bohn contacted the CS department and received permission to use their "Curator" automatic homework grading system software[3] as a prototype for the engineering course homework collection component of the system. He then developed a WWW-based delivery component for course documents, notes, and assignments with security based on the university password system. The entire system was hosted on an existing ME Sun computer with its proprietary operating system and compiler. The initial version of the course website was fully operational in fall '99 with the first offering of the course.

The Other Agenda

Dr. Bohn and Professor Walker were seriously interested in offering the course completely online, in a distance-learning model. This would make it the first computer-based distance-learning undergraduate course in the college. With that as one of the objectives, a small course development grant was provided by the Center for Excellence in Undergraduate Teaching (CEUT) with matching funds from the college. Therefore the course was developed with the presupposition that it would eventually be offered as an online course.

Course and System Development

The course was hosted on a Sun computer, with its operating system and compiler. There were eleven programming projects automatically graded by the computer once daily. Developmental issues were addressed as follows:

- **system security**
    - Students submitted only source code that would be compiled and executed on the grader in an isolated subdirectory where the compiled program had no other access to the system
- **computer gradable projects**
    - Project output requirements were given for a published data set and students were not allowed to deviate at all from that output. Projects were also graded with an unpublished data set to prevent students from using simple text output to fool the grader.
- **grader versus student compiler differences**
    - Each student received four opportunities to submit the project for full credit prior to an initial published deadline. The computer would disallow submissions for grade after that deadline. With each submission the student was provided with a results page indicating errors, resulting grade, and showing the code the grader compiled for them
- **ensure student mastery of course subtopics**
    - Students who were not successful (grade of 80% or better) before the initial project deadline were given three additional submission opportunities before a second deadline which were only graded in a PASS/FAIL mode. Final project averages were then multiplied by a factor equal to the number of successful projects submitted (all five submissions counted) divided by the number of projects assigned.
- **project grading scale**
    - Code compiles with no warnings, executes and gives exact output match – 100
    - Code compiles with warnings, executes and gives exact output match – 80
    - Code compiles with no warnings, executes but output does not match – 60
    - Code compiles with warnings, executes but output does not match – 40
    - Code does not compile – 20
    - Code is not submitted successfully to grader at all – 0
    - Student submits frivolous code in effort to get 40 or 60 – negative 10 (requires human intervention)

- **cheating**
    - The grader compared every student's homework submission with every other student's homework submission and gave indication of possible cheating. Students were warned in the course documents and in class that this was being done and that violations would be submitted to the Virginia Tech Honor Court for adjudication.
- **style grading**
    - Effective grading for style requires human intervention. Instructors must view the student's submitted code and points are deducted from the computer generated project grade if the style requirements are not followed. The grader includes a facility for uploading a style grade file for each project so that the students' grades can be adjusted and displayed accordingly.

By the spring of 2000 the following issues had arisen and were resolved as shown:

- **need for authorized communications channel for students outside of class**
    - A monitored discussion listserv was provided and its use encouraged by the instructors. The instructors also participated.
- **claim that course was inhibiting any collaborative learning experience**
    - Students were allowed to register as "partners" for some specific assignments.
- **claim that computer grading was too arbitrary**
    - Students were allowed to submit appeal letters to instructor detailing their specific situation for a specific project.
- **claim that multiple choice format test had too many choices**
    - A graduate student was hired to develop an online, interactive practice test.
- **students wanted the homework graded more often**
    - Homework grading was increased to every 20 minutes
- **need to discourage procrastination**
    - The grader was programmed to delete one homework submission opportunity 72 hours and another 48 hours prior to the initial deadline so that a student choosing to wait until the last day to submit would be limited to two submissions for grade

During the summer of 2000, course and web site development continued and more EF instructors became qualified to teach the course. The course was also taught by Dr. Bohn during the summer session. In the fall semester Dr. Bohn taught his section of the course in a totally online format with the exception that the students could attend class once a week for the sole purpose of asking questions. There were no lectures. The other two instructors met the class as regularly scheduled and used the standard lecture format. However, their students were told that they could treat the course as online if they desired, choosing when and when not to attend class. With those exceptions, all students were treated as if they were in the same class with joint testing and project scheduling. The goal was to offer the course only online in spring 2001. However, two of the departments requiring the course, including ME, changed the absolute requirement for EF 2314 to an option; reducing the student demand somewhat and the EF Division was notified that ME's equipment and system administration resources (including Dr. Bohn) would not be available after fall 2001. Simultaneously the decision was made at the college level that the distance-learning model was not working for the majority of students and a structured, in-class, lecture format needed to be re-instituted. With the primary author as the sole course director and developer, plans were immediately made to shift to EF resources completely. For several reasons, including limited financial resources, the decision to shift the system away from proprietary software and hardware, using "public domain" system and application software wherever possible. Linux was chosen as the operating system, PHP[4] as the scripting language, MySQL[5] as the database and GNU C++[6] as the compiler. An undergraduate student, the co-author, was hired to transfer the project from the Sun Solaris platform. By spring '02 with the exception of the practice test, only EF hardware and software resources were used for the course and none of the development tools were proprietary.

Current Version

Responding to student feedback and course results, the system improvements and course changes have been made as follows:

- Projects are compiled, run, and graded as soon as the student submits them.
- When students attempt to submit their first project in the course, they are not permitted to do so until they certify that they have read and understood a summary of the course policies.
- The additional grade penalty for unsatisfactory submission of projects was eliminated.
- The anti-procrastination feature was removed
- The second project deadline for PASS/NO PASS submission was eliminated.
- Project grading has been changed as follows:
    - All projects must be 100% correct with no compile errors or warnings to receive a grade
    - The first two project submissions are for full credit, after that, the score is reduced by 10% with each successive submission until the student has reached a maximum of seven submissions for each project
- There are now 14 projects in the C++ course, five of which are available for partnering.
- A separate, compile-only submission option is available to the students with no limits and no grading. This answers student complaints about differences between their compiler (MS Visual C++) and the GNU compiler with respect to compiler warnings.

- The listserv discussion format has been replaced with phpBB[7], a very nice forum application and a "forum participation" component of the grade has been added.
- The 1-credit Fortran course (EF 2324) has been added with 6 assignments.

The Numbers

1876 students have taken the C++ course with Professor Walker instructing 1122 of those. Since the Fortran course was switched to this format, Professor Walker has taught all of those sections, a total of 159 students.

Distance Learning

The course is still available in a distance-learning mode. An online section of the course is available every semester and 51 students have taken the course in those sections. Officially a student can only sign up for that section if they are not residing within fifty miles of the campus. However, students actually take the C++ course in that mode every semester, although they are enrolled in regular sections. This allows students a level of flexibility which mature students appreciate. They can work, co-op, or even vacation while taking the course. In one case, a pregnant student under doctor's orders to stay home was able to take the course. If they are truly away from campus, they must arrange for an approved test proctor for the three exams. No course or grader changes are made without first evaluating their impact on the distance-learning option for the course.

Lessons Learned

Cheating

It is difficult to discuss the issue of cheating without coming across as a doom-and-gloom prophet, overly condemnatory of this generation of undergraduate students. However, that said, both instructors and administrators were shocked at the initial level of cheating in the course, especially with all the verbal and written warnings the students received about automatic plagiarism checking. It took three semesters of vigorous prosecution of honor code violations before a strong enough reputation was developed and cheating was inhibited. Unfortunately, the instructors' anonymous student-teaching evaluations suffered tremendously. It would appear that the majority of students define cheating by how difficult it is to accomplish. The less effort it takes, the less of an offense it is. Since making a digital copy of someone else's programming project is virtually effortless, it is not cheating, it is "collaborative learning." Unfortunately, the lack of prompt adjudication of offenses contributed to the problem.

Student Academic Maturity

There is no question that the course developers overestimated the undergraduate students' ability to self-direct their learning, at least in programming language subject matter. While the students loved the idea of finding everything required for the course on the web and not having to attend class, a significant and vocal number still held the instructors completely responsible for their success or failure in the course. This had a great deal to do with the termination of official

interest in undergraduate engineering distance learning courses. We also made the classic pedagogical error of overestimating the students' level of appreciation for the course material. However, we did confirm that a significant number of students enjoyed the flexibility the course offered and were able to succeed.

Student Reading Habits

A significant number of students did not read the material provided them by their instructors including course administration issues. This information was available in multiple places on the website and/or directly emailed to the individual students by the course administrator. Even when a feature was added requiring the students to certify reading and understanding the course policies before submitting the initial project, many students did not read them. It is the authors' opinion that many students are reading all text as if it was a web page with hot links to other information; if it isn't highlighted in some fashion indicating the availability of interactivity, it isn't important and is skipped. There has even been some consideration given to having a graded quiz based solely on course policies. Unfortunately using the Forum and direct emails does not solve the problem.

Class Participation

The discussion listserv was added early on with the assumption that it would encourage active student participation. The rules were simple; students could discuss course material as long as they did not trade actual code related to a specific project assignment. The publication of "pseudocode" was allowed as was any formulae necessary. Students who successfully submitted a project were encouraged to distribute additional data sets for others to use in testing projects prior to submission. They were also encouraged to share programming errors that they encountered and how to solve them. They could ask questions about the programming projects. It did not work that way. Instead, the students did not keep up with the discussion threads and/or did not check the archives before sending messages asking for help or clarification that had already been given. This immediately turned off the good students who stopped reading and answering the questions and the listserv just became flooded with problems and no solutions unless the instructors participated. Students living off campus without broadband access constantly complained about the amount of listserv traffic they had to download when they checked their email. In short, the listserv was not being used. During summer 2003, a five percent class participation grade was added based solely on proper listserv participation. The results were positive and the practice was repeated in the fall. However, moving from the listserv to a phpBB forum provided a quantum improvement.

Future Development Goals for the OMEN System

The major pedagogical goal is to provide streaming audio-visual instruction asynchronously to the students so that they can view/review the lectures on their own schedules. All other goals are course administrative in nature, providing easier initial course setup, project changes, attention to individual student problems, style grading, and additional programming language options such as MATLAB.

Conclusions

Automated programming project grading is a valuable tool to help reduce the instructional resource requirements in programming courses. It allows the quantity of projects assigned in the course to be based more on the students' learning needs than the instructors' work loads. Until a method of automatic style grading is found, human intervention is necessary for that component of project grading. Programming instruction delivery in a purely asynchronous, online mode is not effective for the majority of engineering undergraduate students. However, a significant number of students can and do successfully use the option to provide more flexibility in their schedule. It is the author's opinion that, given the current rate of online course and computer technology development, it would be advantageous for introductory engineering courses to include a few small, online instructional components so that the students are gradually introduced to the concept. Finally, sophisticated, effective system and application software to provide and administer online instruction is available in the public domain. An excellent example is the current Whiteboard project.[8]

---

[1] Recent name change to Engineering Education (EngE) Department
[2] http://www-rp.me.vt.edu/bohn/
[3] http://ei.cs.vt.edu/~eags/Curator.html
[4] http://www.php.net/
[5] http://www.mysql.com/
[6] http://gcc.gnu.org/
[7] http://www.phpbb.com/
[8] http://whiteboard.sourceforge.net/

THOMAS D. L. WALKER, Associate Professor of Engineering Fundamentals at VPI & SU. He earned his BSEE degree from Purdue University and his MSME from the Naval Postgraduate School. His primary academic and pedagogical interests are in the areas of distance/asynchronous learning technologies and methods, object-oriented programming, and object-oriented modeling.

REVEREND JOSEPH K. GOODMAN, Undergraduate engineering student in the Bradley School of Electrical and Computer Engineering at VPI & SU. Reverend Goodman's interests and experience include the area of network design and administration, interactive web server design and programming, and public licensed software.