A Candid Look at a Decade of First-Year Engineering Experiences Programs

Donatus C. Ohanehi
Virginia Tech, dohanehi@vt.edu

Abstract - **The Engineering Education community can reap benefits from an outsider's perspective on First-Year Engineering Experiences programs. Such perspectives from an actively involved outsider may mirror perspectives of other interested observers like parents or industrial sponsors or alumni, perspectives that could be useful to the Engineering Education community in assessing the effectiveness of FYEE. What follows are highlights of observations from a decade of teaching first-year engineering classes as an adjunct instructor. The observations will focus on developments in three areas: "gentle" introductions to computer programming, project-based learning, and the changing structure of instructional teams necessary for teaching large numbers of first-year engineering students. The "gentle" introduction to computer programming started with the "Alice" program, moved to LabVIEW, and currently features a "gentle" introduction to MATLAB. Project-based learning started with traditional hands-on projects where students did all project work outside class. The current model is more formal PBL where significant portions of class time are used directly or indirectly on the project, and class content is significantly dictated by the needs of appropriate projects. Instructional team structures ranged from the traditional solo teaching model, a decade ago, to large-lecture models with workshops led by a trained and mentored team of graduate teaching assistants. The current model employs a regular class size, taught by a team of instructors and graduate assistants, but with a combination of lectures and workshops implementing PBL. The perspective presented here is expected to lead to constructive discussions because the developments described were led by educators who are passionate about student learning, and are taking steps to promote learning. Such educators have the insiders' views on the developments described. It would be interesting to correlate insider, outsider, and student perspectives, and to mine for potential correlations in the tons of student data accumulated in the past decade.**

Index Terms - trends in first-year engineering experiences, hands-on teaching, project-based learning, technology-enhanced learning.

INTRODUCTION

A wide range of developments has occurred in a decade of teaching first-year engineering classes. This paper presents a unique perspective on these developments. Both the author's status and selected focus areas for this paper contribute to the uniqueness. The author has been an adjunct instructor for a decade, teaching numerous first-year engineering classes and a few sophomore-, junior-, and senior-level mechanics and mechanical engineering classes.

The presentation of a unique perspective is meant to complement existing perspectives in the Engineering Education community. The author did not conduct any investigations or interviews but is simply summarizing observations of a highly interested and involved "outsider." While the perspective reported is not expected to duplicate "insider" perspectives, it is expected that this perspective may mirror perspectives of significant stakeholders in the Engineering Education enterprise: parents of first-year engineering students, administrators, industrial sponsors, and alumni who are not in the engineering education field but support educational programs,

Three focus areas for this paper are: "gentle" introductions to computer programming, project-based learning, and the changing structure of instructional teams necessary for teaching large numbers of first-year engineering students.

INTRODUCTIONS TO COMPUTER PROGRAMMING

The current semester's (Summer 2015) statistics on computer programming experience for first-semester engineering students are similar to typical data for the decade. Only 13% claimed that they considered themselves as "having significant programming experience." Thus first-semester classes are geared to students with no programming experience. A decade ago, MATLAB was used in both the first and second semesters. First-semester tests involving MATLAB had low scores and MATLAB essentially functioned as a "weeder" tool for this class.

About 9 years ago, Carnegie Mellon University's program, "Alice," was used as a "gentle" introduction to computer programming [1]. An Alice user simply drags-and-drops icons, and is forced to avoid syntax errors, but learns basic programming constructs by building 3D virtual worlds. Typical students seemed to enjoy using the program, confirming national studies showing the effectiveness of this program and similar in making programming accessible to hitherto excluded populations [1]. However, what seemed like a revolt started in the second year of Alice's introduction. A group of students placed highly negative posters all over campus, and apparently had the support of an Engineering Education faculty member, who was probably attempting to preserve the tried and true traditional approach. The argument was that the program had been stigmatized by being used in middle schools and minority populations, and had never been used directly in professional engineering work.

LabVIEW [2] was introduced next because it was also a drag-and-drop program, with virtually no room for syntax errors. However, LabVIEW was a professional engineering program with object-oriented features and was highly popular in engineering test laboratories. Most students seemed to enjoy the program and some used the program beyond their first years. However, MATLAB was still used in the second-semester, first-year classes, requiring students to learn two completely new programming languages in 2 semesters.

The transition to MATLAB in both first-year semesters was a part of the re-vamped first-year classes, with a focus on project/problem-based learning. In the first semester, a "gentle" introduction to MATLAB was used. Students were required to watch introductory videos, before classes, and the problems' flowcharts with segments of MATLAB codes were provided in class. A key feature in the "gentle" introduction is the two-step approach in employing flowcharts. In the second semester, students were required to understand and create flowcharts as graphical, big-picture means for planning their codes. However, in the first semester, students were merely required to understand flowcharts and be able to produce codes based on provided flowcharts. Thus, students received guided instruction in class, with many opportunities to get programming help (MATLAB help sessions and MATLAB "lounge" sessions) outside class.

PROJECT-BASED LEARNING

A decade ago, numerous opportunities were created to incorporate hands-on activities in class. Teamwork was encouraged. The most prominent opportunities were in design projects [3] which required physical prototypes, and project work was done primarily outside class time. Examples of first-semester project activities included design topics involving toy cars or using Alice software to build games, but second-semester topics were fairly elaborate. For example, one year's project was the design, construction, and testing of "punkin-chunkin" pumpkin launchers [4]. The best launchers in each class were sent to a national competition and some of the entries placed nationally. The classes' testing sessions were run by shifts of instructors, used up entire weekends and one weekday evening. Interestingly, after the punkin-chunkin projects, the following semesters' projects were scaled down considerably, perhaps because of instructor burnout. The following year's projects required strictly paper design including complete sets of Autodesk Inventor CAD drawings and animation, if possible.

At this point, both the first- and second-semester projects were largely add-on's to the classes. Most of the work was done outside class, and class content was not driven by the projects. For some years, to incorporate some MATLAB in the projects, students were required to select one component of their project and use MATLAB to generate it. The most popular student choice was the generation of a decision matrix table using MATLAB.

A slight revival in physical prototypes started when the focus of first-semester projects shifted to sustainability, and cheap simple materials were provided to student teams for designing and producing simple displays promoting sustainability.

The current model is explicitly project-based learning, PBL, [6, 7] in the second-semester course, with course content driven primarily by the requirements of projects selected to furnish exercises in engineering fundamentals. The first-semester course has a project that offers the beginnings of PBL. Another major difference between PBL and traditional offerings is the amount of class times allocated to project activities in the PBL class model. Class times include frequent in-class team reporting and feedback from both the instructor and other students. Teamwork is promoted through discussions of expert recommendations for high-performance teams. Problem solving skills are discussed and applied to the project [8 - 11]. While course content was drawn from ABET criteria, there was concern by some (perhaps those that have leanings towards the traditional model) over the amount of content that had been included.

INSTRUCTIONAL TEAM STRUCTURES

A decade ago, first-year engineering classes were regular class sizes (about 30 students) meeting for 60 minutes, twice a week. Instructors were generally assigned 2 or 3 or 4 class sections. Classes were the classic lecture style, with students required to do homework and work on projects, at "home," outside class. Given very limited interactions between the lecturing instructors and the note-taking students, it was fairly clear that instructors repeated the same lectures. The first class received the fresh version of the lecture, while succeeding classes were not as fresh for the instructor, but had better timing. The repetitive nature of the same lectures given to small groups of students, in succeeding hours, provided a good case for going to large lectures, given that the content was the same, and there was little interaction with the students. Interestingly, new technologies were also emerging for providing some interaction between a lecturer and large groups of students.
Some degree of excitement accompanying technology-enhanced learning tools (clickers and DyKnow software), combined with university-wide calls to cut costs, drove the introduction of large-lecture class sizes [12] in the first- and second-semester classes.

Large class sizes ranged from about 130 to 250 students, meeting in large auditoriums. Large classes met for 60 minutes per week and then for a second class period, students met in workshops (90 – 120 minutes) with regular class sizes of about 30. These workshops were led primarily by graduate teaching assistants. The assistants were trained, mentored, and supervised.

In the current model, there has been a revival of the regular class size of about 30 students. Individual instructors, including trained, mentored, and supervised Engineering Education graduate teaching assistants, were fully responsible for 1 to 5 classes. Each class period, 75 minutes long, is typically a combination of a brief lecture and a longer workshop period. There is generally significant levels of interaction between the instructor and the students, and subsequent class periods are generally different, unlike the sameness in the traditional model. The instructor has more of a facilitator / coaching role and this is more challenging than the traditional lecturing role.

DISCUSSION

For each of the three focus areas discussed, there had been conflicts between Engineering Education people with traditional views and those with more progressive views. The conflicts slowed down or complicated the adoption of the newer methods, but probably provided more balanced solutions. This author did not investigate details of these conflicts. Lack of investigation to dig for details in these issues is a limitation of this paper. This paper is strictly a summary of recollections of historical observations. This paper is expected to stimulate some beneficial debates and to motivate an exploration of the theoretical underpinnings of the observations and trends described.

CONCLUSIONS

The perspective presented should serve in filling in a more complete stakeholder perspective. Papers published by course coordinators represent the "insider" views. A number of questions may be worth exploring. How much of a difference exists between "insider" and "outsider" perspectives? If large differences are anticipated between perspectives, is it worth the effort to communicate more on such issues like the reasons why some strategies (such as LabVIEW use in the first semester) were stopped? Is it practical to communicate more, given that course coordinators are already overloaded? Major steps have been taken in the three areas discussed, and there have been major productive interactions (conflicts!) between apparent traditionalists and progressives (as they appear to an "outsider"), potentially strengthening the resulting teaching models.

ACKNOWLEDGEMENT

The author acknowledges the Engineering Education Department, Virginia Tech, for offering the opportunities that made this paper possible. Inputs from Professor Richard Goff, Virginia Tech, are gratefully acknowledged. However, the author is fully responsible for all errors and biases in this paper.

REFERENCES

[1] Kelleher, C, Pausch, R., "Lowering the Barriers to Programming: A Taxonomy of Programming Environments and Languages for Novice Programmers," *ACM Computing Surveys (CSUR)*, Volume 37, Issue 2, 2005, June, 83-137.

[2] National Instruments, Inc., "LabVIEW System Design Software," http://www.ni.com/labview/.

[3] Dym, C. I., Little P., *Engineering Design: a Project-based Introduction,* John Wiley & Sons, Inc., 2009, 3rd edition,

[4] Pumpkin-chunkin http://www.punkinchunkin.com/

[5] Dynamic Knowledge, Inc. DyKnow. www.dyknow.com

[6] Bell, S., "Project-based Learning for the 21st Century: Skills for the Future," *The Clearing House: A Journal of Educational Strategies*, 83 (2), 2010, 39–43.

[7] Abdulaala, R. M., et al., "Design and Implementation of a Project-based Active/cooperative Engineering Design Course for Freshmen," *European Journal of Engineering Education*, Vol. 36, No. 4, 2011, August, 391–402.

[8] Wankat, P. C., *The Effective, Efficient, Professor: Teaching, Scholarship and Service*, Allyn and Bacon, 2002, 107 – 112.

[9] Brockman, J. B., *Introduction to Engineering: Modeling and Problem Solving*, John Wiley & Sons Inc., 2009.

[10] Savery, J. R., Duffy, T. M., "Problem Based Learning: Instructional Model and its Constructivist Framework," *Educational Technology*, 35, 1995, pp.31 – 38.

[11] Jonassen, D. H., *Learning to Solve Problems: An Instructional Design Guide*, Pfeiffer, 2004, chapters 2 - 4, 8.

[12] McKeachie, *W. J., Teaching Tips: Strategies, Research, and Theory for College and University Teachers*, D. C. Heath and Company, 1994, pp. 195 – 222.