Work in Progress: Teaching Game Design and Robotics Together:

A Natural Marriage of Computing and Engineering Design in a First-Year Engineering Course

Adam R. Carberry and Ashish Amresh
College of Technology & Innovation
Arizona State University
Mesa, AZ, USA
adam.carberry@asu.edu; ashish.amresh@asu.edu

***Abstract*—The increased dependence on computer programming in engineering has made it essential for engineering students to learn about programming throughout their undergraduate education. In the same vein, computing students benefit when given an opportunity to learn more about engineering design and systematic thinking. This paper discusses how one college embedded computing and engineering into a combined first-year introductory course. The course fuses computing and engineering using game design and robotics as an offering for both cohorts of students to work together in a multidisciplinary environment. Over the course of the semester, students learn introductory computing and engineering design concepts by designing games and robots using informatics tools to solve design challenges. Interdisciplinary teams consisting of computing and engineering students work together to prototype a game design idea and then bring that idea to life using robots as part of their final project.**

Keywords – game design, robotics, projoect-based learning, problem solving, common engineering core, computing for engineers

I. INTRODUCTION

Understanding of and an ability to program has become an essential skill for engineers to learn since the creation of the computer. The increased reliance on computing to address engineering needs has rapidly increased the number of programming-based courses required to receive an engineering degree. A parallel need also exists for computing students to understand the principles of engineering as it relates to design, problem solving, and applied learning [1]. The ability to think in a design-centered way and approach problems systematically are skills that assist engineering and computing students in problem solving. The intended goal of these two learning objectives suggests a mutual give and take that lends itself ideally to a multidisciplinary approach. Yet, common offerings to address the learning of these skills have taken on a "siloed" or separate disciplinary structure where engineering and computing students learn these skills separately [2]. This has several pitfalls; the most severe being that underclassmen lack the maturity or the far-sightedness to understand why they have to learn the basics of other disciplines if it is not their intended major [3, 4].

The following work-in-progress discusses the creation of a fused multidisciplinary computing and engineering course, similar to that of Xu *et al.* [5], that relies heavily on the principles of game design [6] and LEGO® Mindstorms® robotics [7, 8]. The course was designed and taught by a computing and engineering faculty team – one instructor from each discipline. Students in both engineering and computing enrolled in the course together during their first-year to create a multidisciplinary experience.

II. COURSE DESIGN

In 2009, the computing department merged with the engineering department allowing for an opportunity to fuse each disciplines separate introductory courses. Teams of computing and engineering faculty began offering these fused introductory classes. The fused computing and engineering course is taught as part of a required first-year project spine.

Anecdotally, the faculty found that the courses were not preparing students well enough for the subsequent second-year courses in both engineering and computing, especially in computing. The courses were technically fused, but in practice became essentially two courses taught during one insufficient time block.

A new approach was piloted during the Spring 2012 semester to explore how a combination of game design and robotics would impact the computing-engineering fusion. The course was designed to utilize the strengths of the department, to make use of the flexibility of both media to address computing and engineering content, and to learn how the benefits of learning robotics and game design separately could be applied to one another.

As shown in Figure 1, students started out by learning the principles of programming using a drag and drop tool (GameMaker™) to design games. They then transitioned into another drag and drop tool (ROBOLAB™) to learn the principles of engineering design. The two skills were then combined via a final project. The final project placed the emphasis on game design to prototype the solution for an engineering design problem (i.e. iterate, test and refine the solution). Robotics was then applied to finish the real- world

978-1-4673-1352-0/12/$31.00 ©2012 IEEE

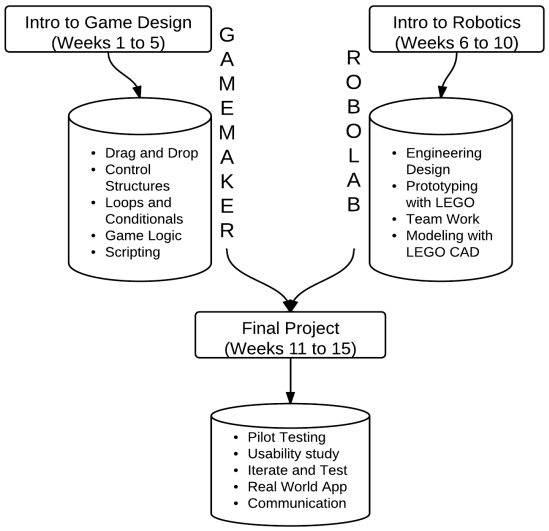

Figure 1. Design of a 15 week fused course

implementation of the solution. Multiple projects for each of the first two sections were used to ensure students had opportunities to display their competency and to work in many different groups. A flipped classroom approach (reverse teaching-homework paradigm) was also used to save class time for hands-on assignments and troubleshooting [9-11]. The overall curriculum design aimed to extract the benefits of gaming and robotics in harmony with one another in a way that showcases the importance of learning both computing and engineering design fundamentals.

III. COURSES ASSESSMENT

During the Spring 2012 semester, one of four sections consisting of twenty-one students, used the combined game design and robotics curriculum. The cohort of students consisted of eleven declared computing students and ten declared engineering students. The course was assessed throughout the semester using student reflections to determine the effectiveness of this approach. Students provided feedback on the content and identified the "muddiest point", i.e. what was most confusing, each week (source of quotes seen below).

A. Game Design

Game design was received well by the students. The drag and drop style programming of GameMaker made it straightforward and easy for students to learn through the online video tutorials. The majority of students found *"GameMaker [to] definitely [be] an interesting learning experience."* The flipped classroom approach made it easy for the instructors to help students learn the basic skills necessary to program in GameMaker. Individual and group assignments were then completed in class in an environment that allowed students to consult one another. Students generally *"…enjoyed working on assignments in class in collaboration with other students."*

The most common "muddiest point" was the conversion from drag and drop to written script. The instructors agreed that all of the students, particularly the computing students, needed practice with written code, including an ability to identify what a drag and drop function would look like if written. This proved to be a difficult transition for many of the students with the limited time given to learn the skill.

B. Robotics

The easy assembly and familiarity with the LEGO products made it easy for students to design and construct their ideas. Students found the robotics section to "*Overall [be] a pretty enlightening experience"* that gave students *"... a great concept on how to be an engineer."*

The programming was a natural extension of the game design section. ROBOLAB is also a drag and drop program that runs within LabVIEW™ Software; the chosen programming language used in the second-year engineering project courses.

As opposed to the game design section, the robotics section utilized teams of students to complete a series of tasks. The explicit team projects were enjoyed by all of the students.

C. Final Project

The biggest challenge of the game design—robotics curriculum was determining how to combine the two media into a natural final project. While both utilize a drag and drop approach to programming, the two programs do not naturally interact with one another. To temper the compatibility issue, it was decided that students would design a two-player game. First, the students would draw up multiple ideas for their game in written form. Second, the students would create a video game version of the game using GameMaker. The video game would act as a simulation of a physical game. Finally, the teams were tasked with creating a physical version of the video game using robotics.

IV. CONCLUSIONS & FUTURE WORK

Our preliminary results suggest the use of game design and robotics as natural media to include in a fused computing and engineering introductory course. The skills required to complete game design and robotic activities complement and enhance each other. The two media are also viewed as fun to the first-year students.

Future assessment of this course will involve an in depth analysis of how this approach impacts student learning. A comparison will be made between traditional offerings and the fused course. Additional quantitative analysis of computing and engineering design self-efficacy, anxiety, and the perceived values associated with a computing-engineering fused course will expand our understanding of this new appraoch. Research will also identify how best to use a flipped model classroom by iteratively changing the curriculum to improve proficiency with programming.

ACKNOWLEDGMENT

The authors would like to thank the Arizona State University College of Technology and Innovation's Scholarship Support and Enhancement program for funding this research.

REFERENCES

[1] Parnas, D.L. 1995. "Teaching programming as engineering." *Lecutre Notes in Computer Science*. Vol. 967, pp. 470-481.

[2] Aloul, F. and Zualkernan, I. and El-Hag, A. and Husseini, G. and Y. Al-Assaf. 2011. "A Case Study of a College-Wide First Year Undergraduate Engineering Course." *Proceedings of the IEEE Global Engineering Education Conference (EDUCON).*

[3] Feldgen, M. and O. Clua. 2003. "New motivations are required for freshman introductory programming." *Proceedings of the ASEE/IEEE Frontiers in Education Conference.*

[4] Spooner, D.L. and M. Skolnick. 1997. "Science and engineering case studies in introductory computing courses for non-majors." *IEEE Transactions on Education.* Vol 48.

[5] Xu, D., Blank, D., and D. Kumar. 2008. "Games, Robots, and Robot Games: Complementary Contexts for Introductory Computing Education." *Proceedings of the 3rd International Conference on Game Development in Computer Science Education.*

[6] Leutenegger, S. and J. Edgington. 2007. "A games first approach to teaching introductory programming." *Proceedings of The Technical Symposium on Computer Science Education (SIGCSE).*

[7] Williams, A.B. 2003. "The qualitative impact of using LEGO Mindstorms robotos to teach computer engineering." *IEEE Transactions on Education.* Vol. 46.

[8] Behrens, A., Atorf, L., Schwann, R., Neumann, B., Schnitzler, R., Balle, J., Herold, T., Telle, A., Noll, T.G., Hameyer, K., and T. Aach. 2010. "MATLAB Meets LEGO Mindstorms—A Freshman Introduction Course Into Practical Engineering." *IEEE Transactions on Education.* Vol. 53.

[9] Zappe, S., Leicht, R., Messner, J., Litzinger, T., and H.W. Lee. 2009. "Flipping' the classroom to explore active learning in a large undergraduate course." *Proceedings of the ASEE/IEEE Frontiers in Education Conference*.

[10] Toto, R., and H. Nguyen. 2009. "Flipping the work design in an industrial engineering course." *Proceedings of the ASEE/IEEE Frontiers in Education Conference.*

[11] Demetry, C. 2010. "Work in Progress – An innovation merging "classroom flip" and team-based learning." *Proceedings of the ASEE/IEEE Frontiers in Education Conference*.