Introduction to Computing: Interdisciplinary Course Design

Vinitha Hannah Subburaj
School of Engineering, Computer Science and Mathematics
West Texas A&M University
WTAMU Box 60767, Canyon
Texas, USA
vsubburaj@wtamu.edu

Anitha Sarah Subburaj
School of Engineering, Computer Science and Mathematics
West Texas A&M University
WTAMU Box 60767, Canyon,
Texas, USA
asubburaj@wtamu.edu

Joseph E. Urban
Arizona State Univeristy
Tempe, Arizona, USA
urban@asu.edu

***Abstract*—This Innovative Practice Category Work In Progress paper presents course design for an interdisciplinary course that can be offered to engineering and computer science students. Engineering disciplines use technology and engineers deal with computing machines to perform day to day activities. The engineers who use such technologies have to be informed users of technology and also are expected to aid in the advancement of technology. Engineers must have sufficient computer skills. An engineering curriculum must help students acquire these skill sets. Employers note that except for a few, most engineering graduates struggle with learning new technologies and putting them to practice. Computer science students who are technically strong are not getting enough exposure to interdisciplinary projects within their curriculum. Computer science students often note not having enough experience working with diverse problem sets inside their curriculum, which is a key for being successful in any industry after graduation. This paper is about introducing an interdisciplinary course across engineering and computer science disciplines that aims at fixing the above two deficiencies. The course is designed for freshman and will enroll students from computer science and engineering disciplines. The curriculum design will aim at teaching computing concepts with interdisciplinary engineering problem sets. This approach is a unique effort as the objectives of the course are completely different from the typical courses offered within the computer science and engineering disciplines. This course will focus on teaching basic computing skills, basic programming, implementing solutions to interdisciplinary problems sets, choice and use of software tools that aid problem solving, and effectively working across interdisciplinary teams. Designing instructional materials and assessment tools to develop this unique mix of skill sets will be addressed in this paper. This effort is just underway; the interdisciplinary course developed will aim at getting used across institutions to serve a similar purpose.**

Keywords—computing, computer literacy, interdisciplinary projects, course design

I. Introduction

Interdisciplinary course offerings enable institutions to assess a wide range of student learning outcomes. Students who are enrolled in quality interdisciplinary courses exhibit better communication and technical skills. These students can take up challenges and confront real world problems. Unfortunately, there are not many interdisciplinary courses designed and offered by institutions that encompass the engineering and computing disciplines. In this paper, we discuss one such effort of designing an interdisciplinary project based course that can be offered across the engineering and computer science disciplines. More specifically we focus on the course structure, instructional materials, and the assessment tools that can used with such interdisciplinary efforts.

Current efforts exist in terms of universities offering freshman level courses that teach computing to engineering students. A literature review shows universities making use of MATLAB [1, 2, 3, 4], Python [5], Scratch [6], LEGO [7,8], and C, as their language of choice in their introductory programming courses for non-computing students. Though most would agree to the advantages of using MATLAB as the first programming language to teach programming to engineers, it does come with shortcomings. Just using one specific software tool to teach computing fails to provide sufficient depth of computing. By using a specific programming tool, computer science often gets perceived as a field of mere programming in the minds of students coming from non-computing majors. Different computing tools that aid the software development process and the application of software systems across different engineering disciplines are rarely addressed in these course designs. Also with the current course offerings, we do not include students from engineering and computer science disciplines to be in the same classroom setting thus failing to have interdisciplinary classroom experience. The course structure and design discussed in this paper addresses the above problems.

In this unique effort, a computer science professor and an electrical engineering professor from WTAMU University will co-teach this interdisciplinary course with students from computer science and different engineering disciplines. The course will be a spring and a fall offering. A pre and a post test will be administered to students taking this course to collect data on different aspects, such as computing knowledge coming in, knowledge of problems from interdisciplinary fields, and experience in using and installing different software packages for solving problems. The initial course offerings will be treated as pilots and the outcomes measured will be used as the deterministic factor to add the course into future computer science and engineering curriculums.

The remainder of this paper is organized as follows: Section II highlights the contributions made by other researchers in this field of interdisciplinary course design. This section also describes how the current literature relates with this research effort. Section III discusses the course design with objectives, student team formation, and a tentative schedule. The different methods of assessment that can be used to

978-1-5386-1174-6/18/$31.00 ©2018 IEEE

effectively assess student work in this disciplinary course is discussed in Section IV. Examples of interdisciplinary problems that can be assigned to student teams in this course are discussed in Section V. Section VI summarizes this paper and provides insights into the future.

II. Related Work

Course design that involves learning across different disciplines might at times lead to misrepresentations and confusions. A discipline [17] is described as a particular area of study that has unified tools, techniques, and methods. In higher education, disciplines tend to develop hard shells and specific jargons that are precise and specific to the concerning discipline. The following definitions based on [18, 19, 20] give an overview of definitions used to classify scientific orientations. Unidisciplinary: This approach involves working within a single discipline that works together to address a common problem. Intradisciplinary courses are often referred to as unidisciplinary courses. Crossdisciplinary: This approach involves more than one discipline that tend to work side by side on related problems. The disciplines involved are not involved with each other while solving the problems and the practitioners are confined to their disciplines. Multidisciplinary: This approach involves more than one discipline working independently to solve common problems. With this approach practitioners tend to work only within their discipline, but acknowledge the different facets to a common problem. Transdisciplinary: This approach involves more than one discipline working together on a specific problem with some overlap in the methodology used to solve a common problem. Integration between disciplines happens at a minimal level that may lead to common concepts and theories but there is no complete overlap. This approach can even lead to the creation of a framework that is beyond disciplinary perspectives. Interdisciplinary: This approach involves more than one discipline working integrally to solve common problems. With this approach, disciplines gets synthesized and extended to involve discipline specific concepts that will be relevant to all involved disciplines. Practitioners tend to be at ease with all the involved disciplines.

The goal of the curriculum design course in [9] is to offer students the interdisciplinary theory and practice of robotic engineering, accommodating the fields of computer science, electrical engineering and mechanical engineering. The sophomore level courses (such as kinematics, circuits, signal processing and embedded system programming) and junior level courses (analysis, system-level design and development of a robotic system) enable students to gain design experience to approach the capstone senior level project. Difficulties were discussed in interdisciplinary design for senior engineering students and working on projects of real-world problems.

The reports in [10, 11] emphasize the need for an interdisciplinary course curriculum. Some of the recommendations made to resolve the issues were: conduct credit-based practical and experiential learning, and capstone-type projects as early as freshman year; classes and extracurricular activities must focus on both hard science and soft skills simultaneously from freshman level and continue throughout the entire degree; and include open-ended, interdisciplinary projects undertaken by freshman groups that change composition over time.

In [12] the significance of computer simulation and animation (CSA) has been explained. This paper conducts a comprehensive review regarding the use of CSA as a learning method to teach engineering mechanics courses. However, there are few disadvantages: 1) the CSA modules used cannot be considered as a stand-alone pedagogical resource since they cannot replace conventional classroom instruction and 2) involves only mechanical engineering rather than being interdisciplinary. The paper [13] examines how senior students deal with problem solving in an interdisciplinary environment, during their senior design final project. Students attend this course from various disciplines, which indicates the necessity to accommodate learning objectives and student outcomes while adapting course curriculum to the needs of first year students.

In [14] the need for creating a successful interdisciplinary integrated curriculum is discussed with the help of basic principles (the six A's) identified by Adria Steinberg. In [15] a course incorporates the MATLAB programming language benefiting only students who are enrolled in civil, electrical, and mechanical engineering. However, the computer science students are then excluded from the active learning techniques through laboratory exercises and projects that introduce computer programming and engineering applications. Moreover, there is a real deficit in breadth first CS 1 courses [16]. Students get little exposure to problem solving and computational thinking skills that can be applied across engineering, mathematics, and science fields.

Some of the above mentioned problems from the literature review reveals the fact that the fundamental concepts from engineering and computing disciplines are not taught at freshman level. The next section describes the introduction of an interdisciplinary course designed for engineering and computer science freshman that aims at fixing the deficiencies across computing and engineering majors as well addressing the problems mentioned in the literature review. The curriculum design is built in such a way that it will accommodate teaching computing concepts with interdisciplinary engineering problem sets.

III. Course Design and Implementation

The following are some insights into the design of this interdisciplinary course. The course design described in this paper is interdisciplinary as it integrates knowledge and methods from computer science and different engineering disciplines to solve common problems. Newell, et al. [21, 22] in their paper define interdisciplinary as "critically draw upon two or more disciplines and…lead to an integration of disciplinary insights." The interdisciplinary aspects of this course involves students with specialized skills in their disciplines getting the opportunity to work with students from different backgrounds. The course design described in this

paper involves a problem solving approach that integrates concepts from computing and engineering disciplines. Woods [24] in her paper states that the interdisciplinary courses should involve collaborations between students from different disciplines who use their disciplinary knowledge to solve complex and significant real work problems. According to Klein [23], the success of interdisciplinary courses depend on characteristics such as: integrating, interacting, linking, focusing, and blending. This course is designed in such a way that it satisfies the above characteristics. Problem sets identified in this paper have evolved by integrating concepts and knowledge from computing and engineering disciplines. Interaction among interdisciplinary teams are facilitated through project work that are team based. The interdisciplinary problem solving approach helps students with specialized skills to gain knowledge from the other disciplines through a creative design process.

This course is offered as a 3-credit hour to computer science and engineering freshman and has no prerequisites. The first few weeks will consist of traditional lectures that deliver necessary concepts and methodologies from the computing and engineering disciplines in an integrated fashion to provide a more holistic approach. Rather than a serial presentation of topics from each discipline which would lead to a multi-disciplinary approach, this course will present topics that are integrated from computing and engineering disciplines that the students will need to apply in solving problems. There will be a midterm exam administered right after this first few weeks to test the student's learning on the subject material. After which the interdisciplinary teams are formed with students from different disciplines (freshman from computer science, electrical, mechanical, civil, environmental, and pre-eng). Students in the team are assigned with a specific role to play and the roles will be rotated. To learn more about the team formations and role play refer to sub-section 2. Every team will address an interdisciplinary engineering problem from start to finish and will present their solutions during key life cycle points and at the end. What make this approach unique is that along the way, the teams are required to choose appropriate software tools to aid their development process. To make the process less complex from an administrative point of view, students will be required to select appropriate software tools from a list provided by the instructor. For example, to design object oriented programs, students can use software design tools, such as Microsoft Visio and Visual Paradigm for UML to develop with Unified Modeling Language (UML) diagrams. To test a circuit design, the teams can use simulation tools, such as SPICE and CircuitLab. Certain projects might even include Raspberry pi and Python to solve interdisciplinary engineering problems. The students are exposed to several software tools to aid design and development of computing solutions for interdisciplinary engineering problems.

Team teaching of courses raises administrative issues related to faculty course load and performance evaluation. These issues introduce another level of complexity when team teaching is with an interdisciplinary course. Regardless of the issues, there should be a clear understanding with the faculty members and administrators.

A. Course Syllabus Overview

This course is interdisciplinary, which includes students from computer science and engineering disciplines. The main focus of this course is to teach basic computing skills, basic programming, implementing solutions to interdisciplinary problems sets, choice and use of software tools that aid problem solving, and how to effectively work across interdisciplinary teams. This course is project based involving practical implications along with team work. This course will equip students with the needed skill sets to provide computing solutions to interdisciplinary engineering problem sets. Students will be expected to be well organized while working with team members and developing their presentation and management skills.

1) Objectives

After completion of the course, students will be able to
(1) Distinguish and appreciate the different areas and applications of computer science and engineering; (2) Understand the fundamentals of computing and engineering; (3) Select appropriate software tools to be used for solving and building solutions to interdisciplinary engineering problems; (4) Analyze, design, build, and test solutions to interdisciplinary engineering problems; (5) Understand how data gets collected, analyzed, and used in computing and engineering; (6) Understand algorithmic problem solving importance; (7) Understand effectively working in an interdisciplinary team setting to solve real world problems; (8) List career opportunities that are interdisciplinary across computing and engineering.

2) Student Team Formation

This course will emphasize team work and collaboration. The class will be divided into groups of three or four and will be assigned a project. Every member in the team is expected to participate and to contribute to the project development. Each student will be assigned with one of the following roles: analyst, designer, programmer, and tester. Every two weeks the roles have to be rotated in such a way that every student in the team gets to participate in all the roles before the end of the semester. Analysts are responsible for gathering information about the existing system in order to determine the requirements for an enhanced system or a new system. Designers describe how to solve the problem and will be coming up with a blueprint of how the different components identified will work together to solve the problem. Programmers will be responsible for translating the design specifications into computer code. Testers will perform thorough and continuous testing throughout the programming stage. Testing is the process that checks to see whether the computer code will produce the expected and desired results, as well as to detect errors, or bugs, in the computer code. Throughout the life cycle of project development, students in different roles have various duties to perform making sure they are occupied and are not idle. Table I outlines the duties that students will be performing during project development.

TABLE I. STUDENT DUTIES

Duties	Analyst	Designer	Programmer	Tester
Requirements gathering phase	1) Analysis of problem statement 2) Gathering of information to come up with a list of requirement specification	1) Identify and list the concepts needed to design a solution to the given problem 2) Identify software tools and techniques to aid design 3) Build a design plan	1) Identify and list the concepts needed to implement a solution to the given problem 2) Identify software tools and techniques to aid implementation 3) Build an implementation plan	1) Identify and list the concepts needed to test the implemented solution 2) Identify software tools and techniques to aid testing 3) Build a test plan
Design phase	Verification and validation of requirement specifications	Convert requirements into design specifications	Start building test cases from the specifications	Start building test cases from the specifications
Implementation phase	Build requirement traceability matrix (both forward and backward traceability)	Verification and validation of design specifications	Convert design into code	Start building automated testing framework in the chosen implementation language
Testing phase	Perform acceptance testing	Perform integration and system testing	Perform unit testing	Implement an automated testing framework to completely test the developed project

3)Tentative Schedule

TABLE II. TENTATIVE SCHEDULE

Weeks	Topics
2	Background material on the field of computing. Applications of computing across engineering disciplines
2	Computing and engineering fundamentals: control structures, recursion, sequencing, sorting, error detection, and data structures, engineering design, fundamental dimensions and units, concept of time, mass, temperature, and force, physics and chemistry laws, and specific topics in engineering
2	Project development life cycle – requirements analysis, design, implementation, testing and maintenance
Midterm Exam Evaluate student knowledge on fundamentals of computing	
1	Formation of interdisciplinary teams and Problem assignment
3	Analysis of problem statement, gathering of information, research on the use software tools and techniques to aid problem solving Design of solution to the given problem, algorithmic problem solving, and generation of blueprints to aid implementation
3	Implementation and testing of the designed solutions with choice of software tools to aid implementation and testing will be focused
2	Wrap-up with evaluations and team presentations
Final Exam	

IV. METHODS OF EVALUATION AND ASSESSMENT

To evaluate the success of this course, traditional assessment methods like quizzes, assignments, and exams will be used. Student performance in these assessments will be used to determine if the stated course objectives in Section III were achieved or not. The different categories that gets included for grading are: in-class participation, quizzes, homework assignments, interdisciplinary project, and exams. Project assessment rubrics will be administered along the way to determine the progress and success of the group projects. Peer-evaluation and soft-skills evaluation will be done periodically to assess effectiveness on working in an interdisciplinary team and communication skills.

In addition to the above evaluation methods, towards the end of the semester, students will be required to complete an anonymous course evaluation survey. This survey will be administered to the computer science students and engineering students separately. After completing this course, engineering students' knowledge will be evaluated on fundamentals of computing, understanding on the different areas of computing, ability to use different software tools during problem solving, and ability to work in an interdisciplinary team setting. In addition, computer science students will be evaluated on their understanding on the different application areas of computing and their ability to analyze and gather information on problems from different domains.

V. EXAMPLE PROBLEMS

The following are some sample problems that will be used with this course. These problems serve as a basis for introducing computing to non-computing majors and introducing the different application domains to the computing majors. These can be modified and extended to include areas that involve computing solutions.

Using Python, design, implement, and test: electronic components, such as resistors, capacitors, op-amps; simple arithmetic and control circuits; dimensional analysis calculator; metric conversation calculator; isometric and perspective projection calculator; problems related to forces in any structures and to solve rigid body subjected to dynamic forces; problems related to fluids in static, kinematic, and dynamic equilibrium; and applications of the conservation laws to 1) flow measurements; 2) flow through pipes (both laminar and turbulent); and 3) forces on vanes.

VI. SUMMARY

This paper has provided the motivation for the development of an interdisciplinary first year course within computer science and engineering curriculums. The paper also included a survey of earlier efforts in exploring interdisciplinary courses that combine computing and engineering problem solving. The interdisciplinary course design described in this paper emphasizes the use of different software tools and packages to aid an effective solution building process. The course is designed in such a way that it is project based involving students from computer science and engineering disciplines. The main objective is to introduce engineering students to different areas of computing and just not programming and to give computer science students an opportunity to come up with solutions to computing problems from different engineering fields. Evaluation and assessment methods were also discussed in this paper. This course will assist engineering and computer science students with introductory computing knowledge and prepare them to face real world problem solving challenges. The next step will be to gather data on the further design and implementation of the discussed interdisciplinary course. Through continuous process improvement, the constituents will help drive course enhancements.

REFERENCES

[1] A. Haubold, "Matlab for First-Year College Engineers," 2007 IEEE Frontiers in Education Conference - Global Engineering: Knowledge Without Borders, Opportunities Without Passports, pp. F1H–7, 2007.

[2] M.-F. López-Pérez, S. C. Cardona, J. Lora, and A. Abad, "MATLAB as a tool as Analysis and Problem Solving Competency Development in Chemical Engineering Degree using MATLAB," Multidisciplinary Journal for Education, Social and Technological Sciences, vol. 3, no. 2, pp. 15–29, Mar. 2016.

[3] A. P. King and P. Aljabar, "Introduction to Computer Programming and MATLAB," MATLAB Programming for Biomedical Engineers and Scientists, pp. 1–29, 2017.

[4] J. L. Vicéns, B. Zamora, and D. Ojados, "Improvement of the Reflective Learning in Engineering Education Using MATLAB for Problems Solving," Computer Applications in Engineering Education, vol. 24, no. 5, pp. 755–764, 2016.

[5] Y. Wang, K. J. Hill, and E. C. Foley, "Computer Programming with Python for Industrial and Systems Engineers: Perspectives from an Instructor and Students," Computer Applications in Engineering Education, vol. 25, no. 5, pp. 800–811, 2017.

[6] D. Topalli and N. E. Cagiltay, "Improving Programming Skills in Engineering Education Through Problem-BasedGame Projects with Scratch," Computers & Education, vol. 120, pp. 64–74, 2018.

[7] R. Ross and A. Console, "BattleBots — A First Year Robotics-Based Inter-discplinary Engineering Project," 2017 IEEE International Conference on Mechatronics (ICM), pp. 414–418, 2017.

[8] A. Behrens, L. Atorf, R. Schwann, B. Neumann, R. Schnitzler, J. Balle, T. Herold, A. Telle, T. G. Noll, K. Hameyer, and T. Aach, "MATLAB Meets LEGO Mindstorms—A Freshman Introduction Course Into Practical Engineering," IEEE Transactions on Education, vol. 53, no. 2, pp. 306–317, 2010.

[9] M.A. Gennert, and T. Padir, "Assessing Multidisciplinary Design in a Robotics Engineering Curriculum," American Society for Engineering Education, 2012.

[10] American Society for Engineering Education. (2017). Transforming Undergraduate Education in Engineering Phase II: Insights from Tomorrow's Engineers. Workshop Report. Washington, DC.

[11] "Training Tools for Curriculum Development: Personalized Learning," IBE-UNESCO, Geneva, June 2017.

[12] M. Tajvidi, N. Fang, "Application of Computer Simulation and Animation (CSA) in Teaching and Learning Engineering Mechanics," 2015 ASEE Annual Conference & Exposition, June 14-17, 2015, Seattle, WA.

[13] L. E. Debs, R. Dionne, M. Exter, and M. Shaurette, "Problem Solving in a Multidisciplinary Environment: Observations from a Newly Developed Program," 2015 ASEE Annual Conference and Exposition Proceedings., June 14-17, 2015, Seattle, WA.

[14] M. Clayton, J. Hagan, P. S. Ho, P.M. Hudis, " Designing Multidisciplinary Integrated Curriculum Units," February 2010. ConnectEd: The California Center for College and Career.

[15] J. P. Hoffbeck, H. E. Dillon, R. J. Albright, W. Lu, and T. A. Doughty, "Teaching Programming in the Context of Solving Engineering Problems," 2016 IEEE Frontiers in Education Conference (FIE), pp. 1–7, 2016.

[16] Z. Dodds, R. Libeskind-Hadas, C. Alvarado, and G. Kuenning, "Evaluating a Breadth-First CS 1 for Scientists," ACM SIGCSE Bulletin, vol. 40, no. 1, pp. 266–270, 2008.

[17] A. Ertas, M. M. Tanik, and T. T. Maxwell, "Transdisciplinary Engineering Education and Research Model," *Journal of Integrated Design and Process Science*, vol. 4, no. 4, pp. 1–11.

[18] P. L. Rosenfield, "The Potential of Transdisciplinary Research for Sustaining and Extending Linkages Between the Health and Social Sciences," *Social Science & Medicine*, vol. 35, no. 11, pp. 1343–1357, 1992.

[19] D. Stokols, K. L. Hall, B. K. Taylor, and R. P. Moser, "The Science of Team Science: Overview of the Field and Introduction to the Supplement," *American Journal of Preventive Medicine*, vol. 35, no. 2, 2008.

[20] M. Modo and I. Kinchin, "A Conceptual Framework for Interdisciplinary Curriculum Design: A Case Study in Neuroscience," *Journal of Undergraduate Neuroscience Education*, vol. 10, no. 1, 2011.

[21] W. H. Newell and W. J. Green, "Defining and Teaching Interdisciplinary Studies," *Improving College and University Teaching*, vol. 30, no. 1, pp. 23–30,1982.

[22] W. H. Newell, G. D. Williams, and J. T. Klein, "Interdisciplinary Curriculum Development," *Issues in Interdisciplinary Studies* , 1990.

[23] J. T. Klein, *Interdisciplinarity: History, Theory, and Practice*. Detroit: Wayne State University Press, 1990.

[24] C. Woods, "Researching and Developing Interdisciplinary Teaching: Towards a Conceptual Framework for Classroom Communication," *Higher Education*, vol. 54, no. 6, pp. 853–866, Feb. 2006.