

Board 352: Preparing Mechanical Engineering Students for Industry 4.0: an Internet of Things Course

Prof. Hakan Gurocak, Washington State University, Vancouver

Prof. Gurocak is the founding director of Professional and Corporate Education (PACE) program at Washington State University Vancouver. His research interests include haptics, robotics and automation.

Dr. Xinghui Zhao, Washington State University

Dr. Xinghui Zhao is the Director of the School of Engineering and Computer Science, and Associate Professor of Computer Science at Washington State University Vancouver. She received her Ph.D. from Department of Computer Science at the University of Saskatchewan in 2012. She previously received an M.Sc. from the same university, and a B.Sc. from Department of Computer Science, Nanjing University. Dr. Zhao's research interests lie in the general areas of parallel and distributed systems, big data computing, cloud computing, and machine learning. Dr. Zhao is a member of IEEE, ACM, ASEE, and IEEE Women in Engineering, and has been actively contributing to the professional community. She served as the general chair for the 15th IEEE/ACM International Conference on Utility and Cloud Computing (UCC2022) and the 9th IEEE/ACM International Conference on Big Data Computing, Applications and Technologies (BDCAT2022). She also served as the local arrangement chair for IEEE CLUSTER 2021. She was the guest editor for Special Issue on Integration of Cloud, IoT and Big Data Analytics, Software: Practice and Experience (Wiley Press). In addition, she has served on the technical program committee for a number of conferences, and as reviewer for various journals.

Dr. Kristin Lesseig

Kristin Lesseig is an Assistant Professor of Mathematics Education in the College of Education at Washington State University Vancouver. She earned her PhD at Oregon State University and currently teaches elementary and secondary mathematics content and

Preparing Mechanical Engineering Students for Industry 4.0: an Internet of Things Course

Abstract

Smart products can sense their environment, analyze lots of data (big data), and connect to the Internet to allow exchanging data. These capabilities are known as the Internet of Things (IoT) technologies. As they become ubiquitous, smart products provide enormous opportunities for scientists and engineers to invent new products and influence interconnected systems of vast scale. Mechanical engineers will play a significant role in innovating and designing smart products and manufacturing systems of the Industry 4.0 revolution. However, the current mechanical engineering curriculum has not kept pace. In this paper, we present details of a new IoT course for mechanical engineering students. The course contains active learning and project-based learning components. Specifically, a smart flower pot device was integrated into the lectures of the course as an active learning platform. In addition, the course incorporates team projects involving design of smart products. The agile method, often used in software development companies, is introduced to the mechanical engineering students to manage their project development process. The paper concludes with assessment details from the first offering of the new course.

1 Introduction

Today, there are many consumer smart products in our lives such as smart door locks, bike locks, smart kitchen appliances, irrigation controllers, smart thermostats (e.g. Nest), and Amazon Echo, just to name a few. Such physical objects (things) can connect to the Internet for data sharing and control. The technology is known as the Internet of Things (IoT).

As smart products become more ubiquitous, the STEM workforce demands are shifting rapidly, but the current mechanical engineering curriculum at Washington State University Vancouver and elsewhere has not kept pace. Mechanical engineers will play a significant role in innovating and designing smart products and manufacturing systems that are driving the Industry 4.0 revolution. The mechanical engineer of the future still needs the same foundation of technical skills and ability to solve problems. But additional skills are needed to participate in the IoT revolution.

To meet this need, we developed a new modernized mechatronics course that focuses on the IoT technologies, and incorporates project-based learning (PjBL). Our overarching goal was to integrate skills from computer science and mechanical engineering, and bridge the gap in the

mechanical engineering curriculum to better prepare future students for the Industry 4.0 revolution.

We are building on prior work by others using active learning [1, 2], PjBL [3–6], agile software development methods [7–9], as well as existing IoT course materials such as [10–13]. The existing courses tend to target Electrical Engineering and Computer Science students and the *creation* of the underlying IoT technologies, especially low-level software. Mechanical engineers need to develop smart products and systems for Industry 4.0 through *integration* of the IoT technologies *not* creation of them. Thus, we kept this important distinction front and center in our curriculum. Another unique feature is the use of a formal software engineering methodology by Mechanical Engineering students to develop high quality code.

In this paper, we present an overview of the curriculum developed for the new course. We provide details of the instructional design elements and assessment results from the first offering of the new course.

2 Overview of the new curriculum

The mechanical engineering program at WSU Vancouver has a senior-level elective course on microcontrollers. This course is part of a 3-course sequence in the mechatronics option track. It is a 3-credit semester course with two 75-minute lectures per week. In Spring 2023, the new curriculum was offered in this course to introduce the IoT curriculum into the mechatronics track and the program.

The curriculum contains 10 weeks of instructional material organized into five modules. The last 5 weeks constitute the class project phase where student teams develop smart products they propose.

Module 1: Overview of Python - (3 weeks) This is an introductory review of Python programming language. Data types, strings, lists, dictionaries, loops, conditionals and functions are reviewed.

Module 2: Data Collection - (2 weeks) This module examines interfacing sensors and actuators to the microcontroller (Raspberry Pi) to explain how a typical mechatronic system is designed. Circuit diagrams are presented for each type of device and code segments are given for hands-on demonstrations.

Module 3: Data Transmission and Processing - (2 weeks) This module starts with an overview of cloud computing. Then, programming details on how to retrieve weather forecast data from the National Oceanic and Atmospheric Administration (NOAA) servers are presented.

Module 4: Data Transmission and User Interfaces - (2 weeks) This module starts with an overview of the MQTT protocol for network communications. Then, programming details of how to build a remote user interface with gauges, digital displays, and buttons are presented for real-time display of data transmitted over the Internet from a smart device.

Module 5: Software Engineering - (1 week) This module starts with an overview of the software process models. The Agile software development method is introduced and its pros and cons are

analyzed. To help students easily manage their projects, Trello software [14] is introduced as a management tool for the class projects.

Class project - (5 weeks) Students work in small teams and propose a smart product to build as their class project. The project requires using the agile software development method and building a prototype device. At the end, student teams present their project to the class.

2.1 Design elements of the course

Active learning - Active learning increases student success in STEM [1, 2, 15]. In this course, we used *Jupyter notebooks* [16] to implement active learning in Module 1. A Jupyter notebook is a free, open-source, web application that allows students to create and share documents containing live code, equations, visualizations and narrative text.

Module 1 - Lectures for the first module gave an overview of Python and were held in a computer lab. In each lecture, students started from an *initial Jupyter notebook* that contained just text explaining concepts as shown in Figure 1a. There were no code examples in this initial notebook. Each student was sitting at a computer with the notebook open on the screen. The instructor showed the same notebook on the projector screen. As the instructor explained concepts, code examples were added to the notebook as shown in Figure 1b. Students were typing these examples into their own notebooks along with the instructor and running them. If there were any mistakes, they got immediate feedback from the Jupyter notebook. The active engagement in the lecture generated many questions from the students.

Each notebook also contained several sections called “Your Turn” with questions for the students to work on (Figure 1a). When the lecture reached a Your Turn section, the instructor paused the lecture and allowed the students to work on the problems on their own. Students were typing Python code directly into the notebook and testing it. Again, lots of interaction took place with the instructor and among the students. Then, the instructor typed the solution to show the results to the class and explained the details. The lecture resumed with the next topic in the notebook following the same approach. At the end of each notebook was a section called “IoT Example.” This section had a problem to show how the programming concepts they just learned are applied to real-life IoT programming situations. Again, the students first worked on these problems for a short while on their own, then the solutions were explained. After the lecture was over, the instructor posted a complete notebook with text, sample code segments, and solutions for the Your Turn and IoT Example sections as a complete set of lecture notes.

Modules 2-4 - Starting with module 2, the class moved to a classroom where 10 stations with the smart flower pots were set up. During each lecture, two students shared one station. The same active learning approach was continued in Modules 2-4 but this time PowerPoint slides were used for the lectures. In the slides, there were “Your Turn” sections. Students started from a skeletal Python code file provided to them and completed the code while trying to run it on the flower pot at their stations. In these modules we used the Thonny Python editor [17] that comes with the Raspberry Pi instead of the Jupyter notebooks. Once again, a very active lecture environment was generated with this approach. After the lecture, the instructor posted complete PowerPoint slides and files containing Python code.

If .. elif .. else statements


Yet another variation is to add the "else" to the structure of the if statement. The format is:

```

if first expression to test:
    block of code to execute
elif second expression to test:
    another block of code to execute
else:
    block of code to run if all other tests failed

```

Only one block of code will be executed depending on which condition is True. Again, pay attention to the ":" and indentations!

 **Your Turn** ¶

Given x=10 and y=25, check to see if x is greater than y or not. Accordingly, print an output indicating the result.

In []:

Given a variable x, print TRUE as an output if x is less than 10 or greater than 12. Try the code for the cases of x=5, x=20 and x=11

In []:

IoT Example

An IoT device measures water level in a tank. If the water is less than 2 ft, green light is turned on. If the level is between 2 and 3 ft, a yellow light is turned on. If the level is above 3 ft, a red light is turned on.

(a) Initial notebook given to students *without* any Python code.

If .. elif .. else statements

Yet another variation is to add the "else" to the structure of the if statement. The format is:

```

if first expression to test:
    block of code to execute
elif second expression to test:
    another block of code to execute
else:
    block of code to run if all other tests failed

```

Only one block of code will be executed depending on which condition is True. Again, pay attention to the ":" and indentations!

In [52]: `num = 0 # Change this number and rerun the code to try`

```

if num>0:
    print('Positive number')
elif num<0:
    print('Negative number')
else:
    print('It is a zero')

```

It is a zero

In [56]: `x=2
y=3
z=5`

```

# Both test conditions are True but only the first one is executed.
if x > 0:
    print('x is positive')
elif y > 0:
    print('y is positive')

```

x is positive

(b) Python code typed into the notebook during the lecture to complete it.

Figure 1: Example Jupyter notebook for Module 1.

Smart flower pot - The system consists of a flower pot on a motorized rotating base platform (Figure 2). The clear plastic bottom section of the pot is a water reservoir with a submersed pump. The white plastic top part is where a plant can be placed. The smart flower pot contains a light sensor to measure the amount of light the plant receives. It also has sensors to measure the soil moisture, water level in the reservoir and temperature, and humidity sensors for ambient air. All of the electronic components, wiring, and a Raspberry Pi are housed inside the metal pan at the base of the flower pot. Each flower pot is connected to a monitor, keyboard, and mouse to construct a workstation in the computer lab. The smart flower pot was custom designed and built. Excluding the Raspberry Pi, the rest of the hardware cost about \$200 per pot.

The smart flower pot can connect to the National Oceanic and Atmospheric Administration (NOAA) cloud service to retrieve 7-day weather forecast for the location of the pot, take measurements using its sensors, and adjust its actions based on the forecast to periodically rotate the plant and deliver just the right amount of water to keep it alive. Its functions can be monitored over the Internet using a remote dashboard with gauges, digital displays and trend charts.

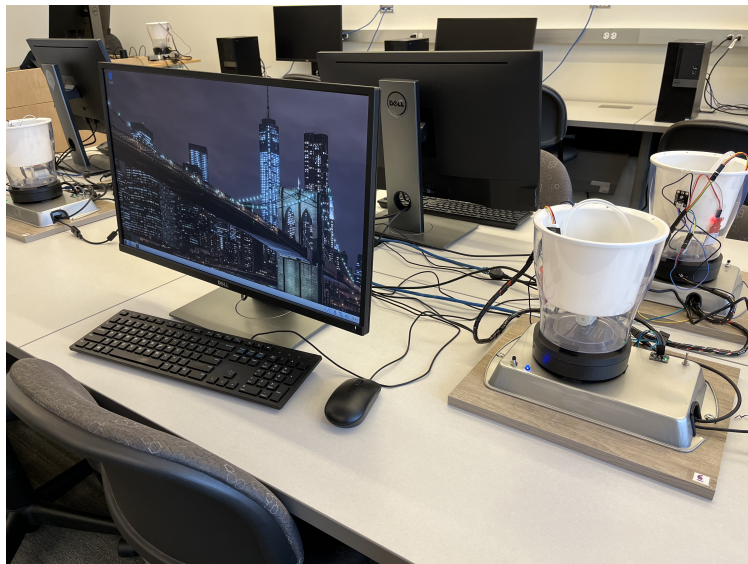


Figure 2: Student station with smart flower pot. It can connect to a cloud service to retrieve 7-day weather forecast for the location of the pot, take measurement using its sensors and adjust its actions based on the forecast to periodically rotate the plant and deliver just the right amount of water to keep it alive.

Project-based learning to frame the curriculum and instruction - In project-based learning (PjBL), students learn the course material from completing a project, which contains and frames the curriculum and instruction [18]. PjBL has been shown to be significantly more effective for student learning in engineering education and in mechatronics courses [3–6].

At the end of the semester, student teams are assigned a class project. Each student team can either choose to build a new smart product or use a flower pot and develop complete control software and remote dashboard for it. Each team submits a proposal to the instructor for feedback. Once a project is approved, parts are ordered by the department staff for the new smart products to be built. Each team is required to use Trello to manage their project, track progress,

and collaborate with teammates. Instructor has access to each team's Trello site to monitor their progress. At the end, each team submits a report, gives project presentation and demo and returns the prototype device to the department.

Agile software engineering methodology - Developing high quality code is challenging, especially for non-Computer Science majors [8]. The mechanical engineering students tend to use an ad hoc approach in code development. The Agile method is systematic and used often by the rapidly growing and volatile Internet software industry for project management [9].

The Agile method was introduced in Module 5. We used Trello [14] as the software platform to implement the Agile method. Fundamental concepts of the method were explained and hands-on demonstrations of how to use Trello were provided to the student teams. Student teams incorporated this method throughout the class project in the last 5 weeks of the course.

Practice problems - Each week practice problems were assigned, but were not graded. Instead, students were provided with solution files as well as recommendations for how to use the problems to enhance their learning and confidence in the material covered. At the end of each course module, students completed a quiz containing exercises similar to the assigned practice problems.

3 First offering of the course

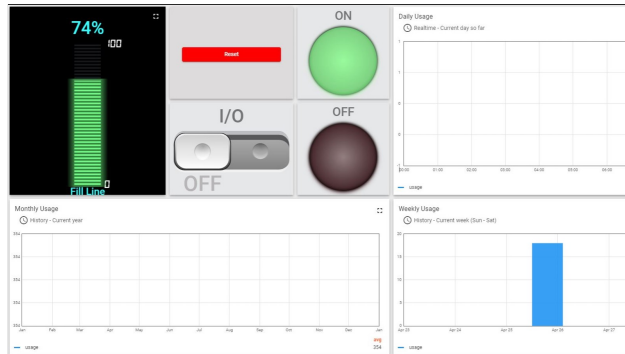
We piloted these materials in an elective mechatronics course. In Spring 2023, the course consisted of 19 students (16 seniors and 3 juniors), six were electrical engineering students and the rest were from mechanical engineering. None of the students had prior experience with Python, but all had some programming experience. After 10 weeks of instruction to cover Modules 1 through 5, the lectures were converted into team meeting times in the same classroom. In the last 5 weeks of the semester, during the regular lecture hours student teams met in the classroom and worked on their team projects and the instructor walked around the classroom to talk to the teams to help and to discuss ideas for their projects.

Five teams were formed by the students. Two teams chose to build new IoT devices while the others chose to use the flower pots as their smart device. The project required each team to develop a functioning IoT device with a remote dashboard for control and monitoring over the Internet. They also had to use agile method in project management and give a presentation at the end. This was the first time engineering students learned about it and used it to manage their team projects.

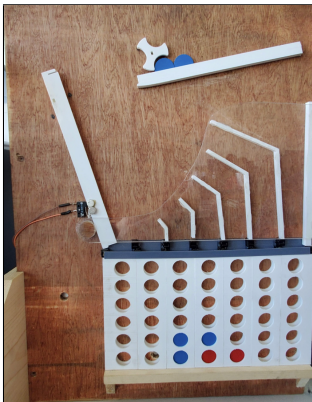
One team chose to develop a smart soap dispenser (Figure 3a). Such a device can be used at airports, hospitals, malls, etc. where multiple devices can be deployed throughout the facility and their usage can be monitored remotely (Figure 3b) by one person. As the dispensers empty out, staff can receive alerts over the Internet to refill them. Another team built a 'Connect-4' game board (Figure 3c). A person sitting by the board can play the game against an opponent located anywhere in the world who is using the remote dashboard (Figure 3d). The dashboard is updated



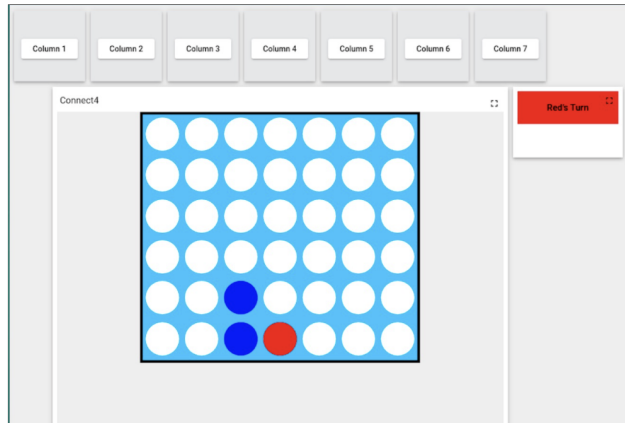
(a) Soap dispenser.



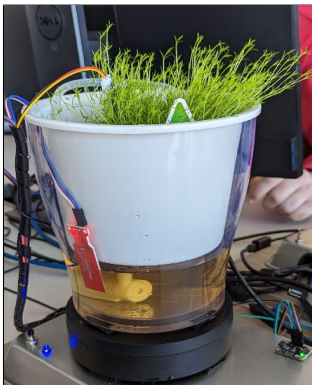
(b) Remote dashboard for soap dispenser.



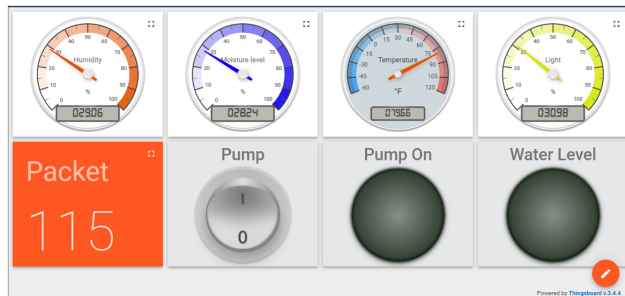
(c) Motorized Connect-4 board to play a chip.



(d) Remote dashboard for Connect-4 game.



(e) Flower pot growing Irish moss.



(f) Remote dashboard for the flower pot

Figure 3: Example team projects.

in real-time showing the status of the actual board. The remote user can play a chip over the Internet, which is delivered into the board via a motorized mechanism. Other teams selected to use flower pots to develop remote monitoring and control software. They ran week-long experiments to keep a plant alive automatically while monitoring various sensor readings remotely (Figures 3e, 3f).

4 Assessment Results

To align our assessments with the IoT content and skills, first we developed a concept inventory of all modules of the new curriculum. After completing each module, students were given a module quiz and a survey to assess the concepts/skills addressed in that module. In each quiz, students were given 1 hour to complete it. Each module required using a computer to demonstrate programming skills. In addition, modules 2-4 required using the flower pots to demonstrate their skills with the hardware and software. The fifth module “Software Engineering” was devoted to introduction of the agile method for project management. Therefore, we observed and assessed student accomplishments for it in the project assignment instead of using a quiz. We analyzed the survey responses for each module and carefully considered student comments. In the following paragraphs, we present a summary of the evaluations.

Module 1 quiz contained 12 programming questions that spanned skills from all major topics in the module leading to a maximum score of 36. Students with the lowest three scores indicated they ran out of time. As shown in Figure 4a, if we consider all scores, the students did well overall (median = 33, st. dev. 4.8). If the lowest three scores are excluded, the median improves even more as expected (median = 34.5, st. dev. 3.6).

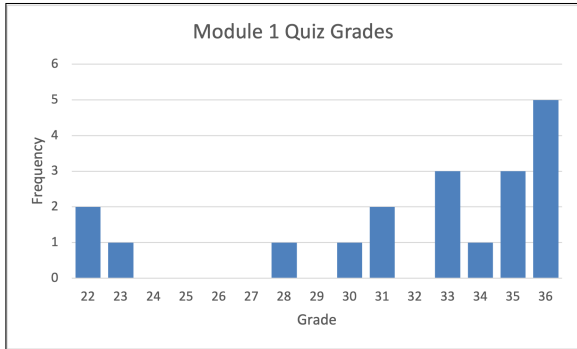
Survey responses indicated that the in-class examples, active learning components and Jupyter notebooks were very successful. Students noted enjoying the ability to learn at their own pace.

Module 2 targeted data collection from the sensors of the flower pot using various programming approaches such as looping and event-driven. It also involved interfacing sensors, motors and controlling them with specific software routines.

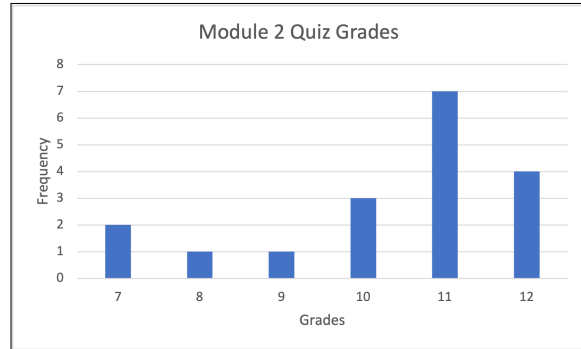
As seen in Figure 4b, overall, students did well in this quiz but there was more spread in the grades (median = 11, st. dev. = 1.61). The programming skills involved in this module are more complex than the previous modules and required application of completely new knowledge to the real device.

The survey results indicated that although students participated in more in-class active learning exercises in this module than the previous modules, they wanted even more hands-on practice.

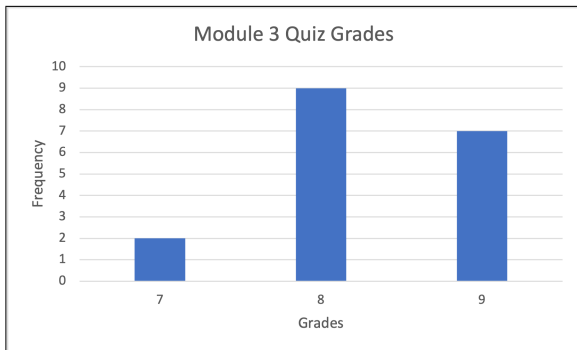
Module 3 quiz contained three programming questions requiring demonstration of skills such as constructing URL queries for the NOAA servers to retrieve weather forecast data, understanding JSON data files, retrieving individual pieces of information from data files, etc. Students wrote programs and ran them on the flower pots. As it can be seen in the Figure 4b, students did very well (median = 8, st. dev. = 0.67).



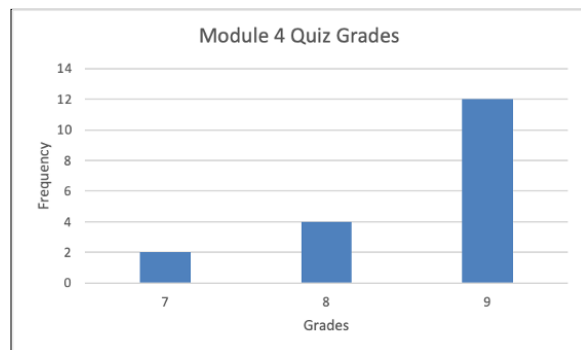
(a) Distribution of Quiz 1 grades. Maximum possible grade was 36.



(b) Distribution of Quiz 2 grades. Maximum possible grade was 12.



(c) Distribution of Quiz 3 grades. Maximum possible grade was 9.



(d) Distribution of Quiz 4 grades. Maximum possible grade was 9.

Figure 4: Quiz grades after completing each module.

Survey responses indicated similar results with great success in using the active learning exercises, Jupyter notebooks and practice assignments. The microprocessor in the flower pots uses Linux operating system. Even though the curriculum does not have anything to do with the operating system, one student found it to be confusing to be using anything other than Windows operating system.

Module 4 quiz contained three questions targeting skills such as building a remote dashboard with real-time updates from the flower pot over the Internet, programming the dashboard and microcontroller for remote procedure calls, etc.

As seen in Figure 4d, the majority of the students demonstrated strong grasp of the concepts through developing working programs and demonstrating them to the instructor during the quiz (median = 9, st. dev. = 0.7).

Once again, survey results showed that the practice assignments and in-class activities to be very helpful. They also thought seeing things happen in the flower pot over the Internet made the lectures more enjoyable. However, further improvements in some logistics, such as uploading the files to the course website a few days before the lecture, could help students study the programming details and come to the lecture more prepared.

Module 5 was about software engineering, specifically learning/applying the agile method for project management. An initial rubric for the skills/outcomes expected from the project was developed and used in the grading (Figure 5).

	Team					total points
	1	2	3	4	5	
smart basic	5	5	5	5	5	5
smart innovative	1	5	5	4	5	5
code design (CSV?)	6	10	10	5	10	10
working	5	5	5	5	5	5
total:	17	25	25	19	25	25
TB design	5	5	5	4	5	5
innovative	3	5	5	2	5	5
working	5	5	5	5	5	5
total:	13	15	15	11	15	15
contributed to team	x	x	x	x	x	
worked effectively with others	x	x	x	x	x	
Trello setup	10	10	10	10	10	10
Trello usage	15	15	15	15	15	15
total:	25	25	25	25	25	25
all headings there	2	2	2	2	2	2
clear writing	3	3	3	3	3	3
grammar/spelling	2	2	2	2	2	2
figures	3	2	3	3	3	3
total:	10	9	10	10	10	10
PowerPoint quality	4	5	5	5	5	5
delivery	3	5	5	5	5	5
demo	5	5	5	5	5	5
total:	12	15	15	15	15	15

Figure 5: Project rubric for the skills/outcomes expected from the project that was used in grading.

All teams completed the projects successfully and demonstrated working IoT devices they built. Most teams came up with innovative ideas and implemented them except for one team with a more basic design. Some grade points were lost due to missing axis units or incomplete figure

captions in reports, and small things missing in the developed software. Overall, there was great excitement in the classroom throughout the 5 weeks of project work and the final presentations were very enjoyable.

5 Conclusions

In this paper, a new mechanical engineering course on Internet of Things (IoT) was presented. The course was designed to bridge a gap in STEM education, specifically in mechanical engineering, to better prepare future students for the Industry 4.0 revolution and for smart product design. The new curriculum focuses on the IoT technologies and brings software engineering methods from computer science into mechanical engineering.

A smart flower pot was custom designed as a platform to be used throughout the course so students could gain hands-on experience with the IoT technologies. The smart flower pot can connect to the NOAA cloud service to retrieve 7-day weather forecast for the location of the pot, take measurements using its sensors, and adjust its actions based on the forecast to periodically rotate the plant and deliver just the right amount of water to keep it alive. The flower pot functions can be monitored over the Internet using a remote dashboard with gauges, digital displays and trend charts.

Active learning in lectures were implemented using Jupyter notebooks. In each lecture, there are multiple “Your Turn” sections where students can try the materials just shown in the lecture using the Jupyter notebooks or a compiler on Raspberry Pi and a flower pot to test their understanding and to try “what-if” scenarios.

The course also contains a project where student teams work on building smart products. The popular agile method used by the tech companies in developing software was introduced to the mechanical engineering students. Using the Trello software platform and the agile method, the teams could manage their project over the 5-week project timeline.

Assessment results from the first offering of the course are encouraging. A majority of the students could demonstrate their newly gained skills on module quizzes. Survey results indicate overall satisfaction with the use of the Jupyter notebooks, the active learning environment during the lectures, and the practice assignments that supplemented the in-class activities. Project presentations and demos at the end of the course created a great excitement for the entire class. We will be offering the course again with slight changes in the curriculum and will be reporting the results in the future.

Many universities have mechatronics courses, which can be replaced by the new course. Coupled with the inexpensive hardware (\approx \$200 per station) and the free open-source software, the products of this work can be transferred to other institutions increasing the potential for high impact in STEM education. At institutions with large class sizes, scalability can be achieved by holding the lectures in a large computer lab but usually these labs are set up for open access. As a result, the flowerpots may need to be set up before each lecture and taken away after. Another possibility is to hold multiple sections of the course with smaller section sizes.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. DUE-IUSE-2116226. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

- [1] P. Seeling, "Active learning moves programming students from novice to skilled," <https://www.pearsoned.com/active-learning-programming-skilled/>, 2016.
- [2] O. Mironova, I. Amitan, J. Vilipõld, and M. Saar, "Active learning methods in programming for non-IT students." International Conference e-Learning, 2016.
- [3] N. L. Toner and G. B. King, "Restructuring an undergraduate mechatronic systems curriculum around the flipped classroom, projects, LabVIEW, and the myRIO." Boston: American Control Conference (ACC), July 6-8, 2016.
- [4] S. Chandrasekaran and J. M. Long and M. A. Joordens, "Evaluation of student learning outcomes in fourth year engineering mechatronics through design based learning curriculum." IEEE Frontiers in Education Conference (FIE), 2015.
- [5] J. Mynderse and J. Shelton, "Assessment of an improved problem-based learning implementation in a senior/graduate mechatronic design course." ASEE Annual Conference and Exposition, 2015.
- [6] K. Meah, "First-time experience of teaching a project-based mechatronics course." ASEE Annual Conference and Exposition, 2016.
- [7] R. C. Martin, *Agile Software Development: Principles, Patterns, and Practices*. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2003.
- [8] M. Guzdial and A. Forte, "Design Process for a Non-majors Computing Course," in *Proceedings of the 36th SIGCSE Technical Symposium on Computer Science Education*, ser. SIGCSE '05, 2005, pp. 361–365.
- [9] P. Abrahamsson, O. Salo, J. Ronkainen, and J. Warsta, "Agile Software Development Methods: Review and Analysis," *CoRR*, vol. abs/1709.08439, 2017.
- [10] J. He, D. C.-T. Lo, Y. Xie, and J. Lartigue, "Integrating Internet of Things (IoT) into STEM undergraduate education: Case study of a modern technology infused courseware for embedded system course." IEEE Frontiers in Education Conference (FIE), 2016.

- [11] D. V. Gadre, R. S. Gaonkar, S. N. Ved, and N. Prasannakumar, "Embedded systems and Internet of Things (IoTs) - challenges in teaching the ARM controller in the classroom." ASEE Annual Conference and Exposition, 2017.
- [12] P. Bisták, R. Moezzi, F. Semeraro, and G. Nicassio, "Teaching IoT Using Raspberry Pi Based RC-Car," 2020, pp. 1–6.
- [13] C. Rennick, C. Hulls, D. Wright, A. J. B. Milne, E. Li, and S. Bedi, "Engineering design days: Engaging students with authentic problem-solving in an academic hackathon." ASEE Annual Conference and Exposition, 2018.
- [14] Trello.org, "Trello Software for Agile Method," <https://trello.com>, Jan 2024.
- [15] C. Martínez and M. Muñoz, "ADPT: An active learning method for a programming lab course." Proceedings of the 10th International CDIO Conference, Universitat Politècnica de Catalunya, Barcelona, Spain., 2014.
- [16] Jupyter.org, "Jupyter Notebook," <https://jupyter.org/index.html>, June 2019.
- [17] Thonny.org, "Thonny Python IDE," <https://thonny.org>, Jan 2024.
- [18] J. Larmer and J. Mergendoller, "The main course, not dessert," Buck Institute (www.bie.org), 2011.