ASEE's 123rd Annual • Conference & Exposition • New Orleans, LA • June 26-29, 2016

New Orleans JAZZED about Engineering Education

Paper ID #17162

Systems Engineering of Cyber-Physical Systems: An Integrated Education Program

Prof. Jon Patrick Wade, Stevens Institute of Technology (School of Systems & Enterprises)

Jon Wade is a Distinguished Research Professor in the School of Systems and Enterprises at the Stevens Institute of Technology and currently serves as the Director of the Systems and Software Division and Chief Technology Officer for the Systems Engineering Research Center (SERC) where he is leading research in the use of technology in systems engineering education and complex systems. Previously, Dr. Wade was the Executive Vice President of Engineering at International Game Technology where he managed corporate wide research and development. Dr. Wade spent ten years at Sun Microsystems during which time he managed the development of Enterprise Servers. Prior to this, he led advanced development of supercomputer systems at Thinking Machines Corporation. Dr. Wade received his SB, SM, EE and PhD degrees in Electrical Engineering and Computer Science from the Massachusetts Institute of Technology.

Dr. Roberta S Cohen, Stevens Institute of Technology

A Teaching Professor at Stevens Institute of Technology since 2009, Professor Cohen spent 26 years in the Telecommunications industry as a technical and managerial contributor to numerous programs at Bell Labs and Telcordia Technologies. She helped create and is a Fellow of the Telemanagement Forum (www.tmforum.org/), an international consortium of over 900 organizations engaged in digital transformation.

Dr. Nicholas S Bowen, Stevens Institute of Technology

Dr. Nicholas Bowen is an Industry Professor in the School of Systems and Enterprises. His primary focus is developing new graduate programs that combine Systems Engineering & Software Engineering with Cyber-Physical Systems.

He recently retired from IBM after a 31-year career. He held a diverse set of leadership positions across product development (both hardware and software), supply chain and manufacturing, sales operations, research, corporate strategy, leading large teams, and talent development. Nick has led and contributed to many critical projects including saving the mainframe business, taking AIX/Power to the #1 UNIX position, establishing Linux servers in the enterprise market, and was on the team that built the first Bladed architecture for the general purpose x86 market.

Nick received a Ph.D. in Electrical and Computer Engineering from University of Massachusetts at Amherst, an M.S. in Computer Engineering from Syracuse University, and a B.S. in Computer Science from the University of Vermont.

Nick has been on the advisory boards of many engineering schools including Florida International University, North Carolina State University, University of Puerto Rico (Mayaguez) and the URI Research Foundation. He was a founding member of the IEEE Computer Society Industrial Advisory Board.

Nick is an avid sailor having logged over 5,000 miles in the open ocean and is often found competing in sailboat races.

Eirik Hole, Stevens Institute of Technology (School of Systems and Enterprises)

Eirik Hole holdd the position of Profesor of Design in systems engineering and engineering management in the School of Systems & Enterprise at Stevens Institute of Technology. Prior to this, he held systems engineering positions in a number of companies, primarily in the automotive and aerospace fields, in Norway and Germany. He obtained a master's degree in aerospace engineering from the University of Stuttgart, Germany, in 1995.

Systems Engineering of Cyber-Physical Systems:
An Integrated, Experiential Education Program

abstract

Mark Andreessen "*Software is eating the world*". [1]

Elon Musk: "*We really designed the Model S to be a very sophisticated computer on wheels," …. "Tesla is a software company as much as it is a hardware company. A huge part of what Tesla is, is a Silicon Valley software company. We view this the same as updating your phone or your laptop*." [2]

Increasingly, products, systems, and infrastructure in sectors that include transportation (aviation, automotive, rail, and marine), health care, manufacturing, and electrical power generation and distribution are cyber-physical systems (CPS). These systems use integrations of computation, networking, and physical processes to provide key functionality and value. The effective development of these systems often will depend on engineers skilled in cyber-physical systems engineering—trained in the analysis, modeling, implementation and testing of systems in which collaborating computational elements control physical entities and real-time processing plays a critical role.

In this paper, we describe an experiential program for graduate studies that provides engineers with the critically important knowledge and skills needed for the design and development of Cyber-Physical Systems (CPS).

1 background

Based on a 2015 National Academy of Science preliminary report [3]

> "Cyber-physical systems (CPS) are increasingly relied on to provide functionality and value to products, systems, and infrastructure in sectors including transportation (aviation, automotive, rail, and marine), health care, manufacturing, and electrical power generation and distribution. CPS are smart, networked systems with embedded sensors, computer processors, and actuators that sense and interact with the physical world (including people), support real-time, guaranteed performance and are often found in critical applications. As CPS become more pervasive, so too will demand for a workforce with the capacity and capability to design, develop, and maintain them."

While Systems Engineering (SE) education has dealt with the specification, design, development and evolution of many types of systems from very large systems for infrastructure, transportation, health care, manufacturing and military projects to very small embedded systems in these same areas, the particular juxtaposition of capabilities of CPS forces the systems engineer to simultaneously balance the demands of communicating physical entities controlled by real-time or near-real-time computation operating with and on behalf of human and system actors. Moreover, to effectively system engineer CPS, this very broad integration of cyber capabilities must be part and parcel of the entire system development cycle, from inception through specification, design and development, validation and evolution.

The education challenge is not trivial. Engineers of cyber-physical systems require systems engineering skills that extend across a very broad range of technologies. Our objective with the Systems Engineering of Cyber-Physical Systems program we have developed is to build a series of courses that can satisfy these needs, in effect integrating the skills of hardware, software and systems engineering, for all of these are required to various degrees by everyone on the CPS team. This same approach may be applicable to all of systems engineering, and this can serve as an exemplar as we update our core SE curriculum.

2 educational gaps to be addressed

The authors of this paper believe that the real issue is not so much what do CPS engineers need to know, but rather how do they need to think and what do they need to know how to do. In addition, it is critical that engineers of all systems, particularly CPSs, have skills in navigating the entire lifecycle, particularly the conceptual front-end of the development program, the verification and validation, through manufacturing and sustainment which are often neglected in traditional, single-discipline academic engineering programs focused on design and implementation. Again, this is reinforced by the findings of the NAS report [3] as noted below:

"Norman Fortenberry, American Society for Engineering Education, described some of the attributes expected of modern engineers: flexibility to manage rapidly evolving technologies; an ability to define as well as solve problems; skill and experience with creativity, entrepreneurship, and public policy implications; and facility with both theory and application."

"… although traditional undergraduate curricula cover the fundamentals of math and science, programming, and problem solving well, they do less well with applications, software engineering, and problem identification."

"… few universities seemed to be emphasizing mission- or safety-critical systems and that hands-on project work tends to ignore properties like fault tolerance and robustness."

"…there is a focus on developing new functions over understanding the tools and techniques needed to test and maintain current systems."

"… project-based learning should be integral to any CPS curriculum. Students need to work on complex interdisciplinary projects that encourage systems-level thinking. Doing so requires test beds that allow for the co-design of physical and computational components that demonstrate the benefits of integrating simulation and experimentation. … design studios, where students can work on integrative CPS projects with multidisciplinary teams, are important."

"Several speakers stressed the value of hands-on projects. Philip Koopman, Carnegie Mellon University, noted that the tools needed to provide students with this experience must incorporate the challenges of large-scale systems and are often expensive and require frequent technology refreshes. Koopman also explained that developing problems that represent the complexity of CPS is difficult. Projects and problems must be realistic and motivating but also incorporate domain knowledge that is accessible to students. There is a risk that problems can become overly complicated—projects must be designed with the right amount of 'messy'."

The National Institute of Standards and Technology (NIST) Foundations for Innovation in Cyber-Physical Systems report [4] as well as the European ARTEMIS Research agenda [5] points out similar needs across many CPS domains. The NIST report identifies 21 barriers and challenges for CPS reliability, safety, and security. In the top rated category of Metrics and Tools for CPS Verification and Validation (V&V), they cite challenges such as the need for increasing coverage of verification and validation while reducing costs, coping with complexity and scale of systems when performing verification and validation, and the inability to apply formal methods at appropriate abstraction levels, especially for a typical engineer.

The Embedded Systems Survey found that university professional development courses came in 8th place with respect to the respondents' self-assessment of effectiveness. Only 18 percent believed that such courses were effective compared to 43 percent for online training courses. The amount of self-reported training per year decreased almost 25 percent from 2012 to 2013 from 11.7 days to 9 days. It would appear that University degree programs and continuing education are missing the mark. It is the authors' belief that academic courses do not provide experiences that can be readily applied at the workplace due to the many differences between the academic and workplace contexts. Providing relevant experiences is a critical aspect of the new pedagogical movement around connected learning such that students are able to make the intellectual leap to directly apply what they have learned in the workplace.

3 program objectives and philosophy

Stevens Institute of Technology is one of several graduate schools in the USA offering Master of Systems Engineering degrees (as well as Ph.D.s in Systems Engineering). The Master's degree is considered a professional degree, typically pursued by students with undergraduate degrees in various engineering fields such as civil, mechanical, electrical and software engineering as well as individuals working in engineering and/or large-scale operations roles.

The Systems Engineering Master's degree blends technical and management training to prepare systems engineers for positions of increasing responsibility. Upon graduation, students are well prepared to address systems integration and life cycle issues, and can apply systems thinking at the system, systems of system and enterprise levels. Among the ten courses taken for this degree are required courses in systems fundamentals (focused on front-end processes such as problem definition and requirements), system architecture and design, systems integration and project management of complex systems. Additionally, students must take a course in either modeling and simulation or design for system reliability, maintainability and supportability, or decision and risk analysis. The curriculum balances theory and practice giving students the opportunity to work on real-world problems in a variety of areas.

Classical systems engineering relies on functional decomposition in its understanding and design of systems. The software-intensive nature of cyber-physical systems challenges this approach, thoroughly blending hardware, software, networking and human interactions, making decomposition irrelevant to the advancement of system understanding. With embedded software systems at their core, cyber-physical systems require a more integrated approach to the systems engineering life cycle.

Our objective was to develop a Systems Engineering of Cyber-Physical Systems program targeting practicing embedded and CPS engineers, to endow them with systems engineering capabilities. This program is projected to include a Master's Degree, beginning with a four-course Graduate Certificate, and a four-hour *executives* workshop, all sharing a common set of materials, with the Master's Degree having specified areas of concentration such as security, autonomy, and high-assurance.

Guiding our activities is this area are the following program principles:

- Emphasize the ability to think and do, rather than just knowing
- Make the curriculum experientially-based, simulating an actual development program experience using contemporary methods, processes and tools (MPT)
- Cover the entire lifecycle including the often neglected areas of:
    - concept generation,
    - validation and verification,
    - system deployment and sustainment
- Focus on the entire process with an emphasis on V&V and mission robustness, not just architecture and design
- Place higher value on practices that provide efficiency and effectiveness in this domain over traditional approaches; e.g., continuous agile vs. phase-based scheduled processes
- Target towards the future using model and computational based approaches

4 course development process

As systems engineers, we developed the SE of CPS using an agile development process with early and continuous verification and validation. First, we met with our primary stakeholder, a corporate engineering executive who wished to fill the educational gaps described earlier in this paper, to ensure that we had a clear understanding of the overall objectives for the program. In parallel with these interactions, market research was performed, as described in this paper to ensure that the problem would address a broad market need. High-level program principles and concept were developed and presented to our corporate sponsor for early validation. Once this was completed, a team of four faculty members was formed to develop the curriculum and serve as instructors for initial pilot of the program. The core courses of this program were to be project based and integrated. A critical element is that the cross-cutting project prototype must require the students to have a lifecycle perspective across these courses. Therefore, the team then constructed a program project that we believed satisfied the requirements of being both representative of CPS systems projects and educational goals, and feasible within the constraints of the target student capabilities and allotted course hours.

At this point, each of the instructors created a set of artifacts (e.g., requirements, concept of operations, architecture, design documents, etc.) that the students would need to bring into their respective course so that they could deliver the desired course outcomes. The team of instructors then iterated on these deliverables until a consistent set of deliverables had been constructed. At this point each instructor developed a spreadsheet relating each of these required deliverables to the educational topics that would need to be mastered to support them. These topics were then sorted into a set of modules in each course. Each course was targeted to include approximately 12 modules. Topics that were not essential for the students' creation of the required course deliverables were deprioritized; some were removed and others were moved to the backend of the course. Deliverables and educational topics were refactored several times to ensure that the educational load was equalized throughout the four courses. Each of these courses and the entire program were documented and then presented to the School and Institute's Graduate Curriculum Committees for review and approval.

The first pilot of the program is seen as a “beta” release in which feedback from the students on strengths and areas of improvement was solicited in class discussion at the end of each day of instruction. In addition, there were formal written evaluations at the end of each course. Instructors were encouraged to sit through all of the course classes in the pilot courses, but were required to attend the prior course reviews to ensure continuity between the courses and instruction.

The program and its course will be updated and improved based on this pilot feedback for a full-program release. It is expected that the program will continue to evolve, be refactored and updated on a regular basis based on future student and instructor feedback.

5 SE of CPS program

To achieve these objectives, we created an experience-based program of four core courses that provide students with experience over the full gamut of system engineering activities. This sequence of courses is meant to mimic an actual professional project environment, using an integrated set of tools. A single project runs through the sequence, which, in our first set of offerings, is the conception, specification, design, development, testing and evolution of a telepresence robot.

Throughout the four course sequence, the theories and practices of systems engineering are presented and applied to the telepresence robot project. And while each course focuses on a specific aspect of CPS engineering, there are a number of threads that weave through all four courses. These threads include:

1) financial analysis,
2) system requirements based on market needs,
3) Model Based Systems Engineering (MBSE) Systems Modeling Language (SysML) architectural models and
4) proof of operation involving fault/hazard/safety analysis.

These threads are introduced early in the program and serve as an anchor while being modified throughout the courses.

The four courses that form the core of the program correspond to the classic systems framework. These courses are:

- Course 1 - Conception of CPS: Deciding What to Build and Why
- Course 2 - Design of CPS: Ensuring Systems Work and Are Robust
- Course 3 - Implementation of CPS: Bringing Solutions to Life
- Course 4 - Sustainment of CPS: Managing Evolution

After their development, it was discovered that the same framework is the basis of the worldwide CDIO movement. “CDIO is based on a commonly shared premise that engineering graduates should be able to: Conceive – Design — Implement — Operate complex value-added engineering systems in a modern team-based engineering environment to create systems and products.” [6] It is the vision of CDIO of an education that stresses the fundamentals with the following properties:

- A curriculum organized around mutually supporting courses, but with CDIO activities highly interwoven

- Rich with student design-build-test projects
- Integrating learning of professional skills such as teamwork and communication
- Featuring active and experiential learning
- Constantly improved through quality assurance process with higher aims than accreditation

Each of the SE of CPS courses is described below.

5.1 conception of CPS: deciding what to build and why

This first course focuses on the conceptual design portion of the lifecycle of CPS. Critical elements include the ideas of systems and design thinking, and elegant design. An Ideation process, as pioneered by the likes of IDEO and other prominent design firms is used to spark the creative process. Ideation is the creative process of generating, developing, and communicating new ideas, where an idea is understood as a basic element of thought that can be either visual, concrete, or abstract. [7] Ideation comprises all stages of a thought cycle, from innovation, to development, to actualization which are traversed rapidly in a matter of minutes. The opportunity is conceived and defined using Kano Maps [8], marketing segmentation and analysis techniques. A Qualify Functional Deployment (QFD) [9] process is used to collect, organize and analyze customer needs, and transform these into product specifications. Concepts are generated, selected and tested. Finally, these concepts are specified using concepts of operation, conceptual design and use case scenarios and technical requirements. A workshop on MBSE, particularly, SysML is used as a means for specification and also to provide a foundation for future modeling work. Lectures are interspersed with individual and group project based activities. The students go through a design review process in preparation for their final report.

This course provides the students with the skills and ability to conceive and develop a Cyber-Physical product concept based on solving an identified problem, satisfying stakeholders' needs in a way that is technically feasible, competitive and profitable. In addition, the students are exposed to working together on multi-disciplinary problems and interacting with external customers and stakeholders.

This is a project-based course that supports the other three courses in the SE of CPS Program. A common program project including a highly integrated, electro-mechanical, computational system, with humans-in-the-loop that is networked and interacts with the physical world is developed throughout the program. CPS integrates the dynamics of the physical processes with those of the software and networking, providing abstractions and modeling, design, and analysis techniques for the integrated whole.

The following are the desired course outcomes:

- Become familiar with systems and design thinking, and elegant design.
- Participate in a Design Thinking Ideation process.
- Conceive and specify an opportunity using Kano Maps, marketing segmentation and analysis techniques.
- Use the QFD process to collect, organize and analyze customer needs, and transform these into product specifications.

- Generate, select and test conceptual models.
- Specify system concept using concepts of operation, use case scenarios and technical requirements.
- Learn Model-Based Systems Engineering fundamentals.
- Experience a design review process for the concept phase.

The syllabus from the initial course offering is shown below in Table 1.

Table 1: Conception of CPS Course Syllabus

	Topic(s)	**Class exercises**
Module 1	Course Overview	
Module 2	Elegant Design	
Module 3	Systems and Design Thinking	
Module 4	Edeation Exercise	Ideation exercise
Module 5	Identifying Opportunities	Apply to project
Module 6	Identifying Customer Needs	Use QFD tools
Module 7	Preliminary Product Specifications	Use QFD tools
Module 8	Concept Generation, Selection and Testing	Use Pugh Matrix [10]
Module 9	Concept of Operations	Apply to project
Module 10	Use Case Scenarios	Apply to project
Module 11	Technical System Requirements	Apply to project
Module 12	MBSE Workshop	Used supplied SysML tools

5.2 design of CPS: ensuring systems work and are robust

Ensuring systems work and are robust educates students on the transition from cyber-physical system concept and preliminary requirements to detailed architecture and design based on prioritized, allocated and traceable architecturally significant requirements. Students create models of system structure and operation, selecting appropriate technology and performing analyses for reliability, performance, safety and security. Since cyber-physical systems operate in real-time, issues of process timing are considered along with potential trade-offs to support space and power concerns. Trade space analyses are also performed.

The objective of this course is to insure that through hands on development of a cyber-physical system prototype, students will learn how to prioritize system architecture activities, create and use system modeling and simulation techniques to achieve systems with the functionality required, buildable on the required schedule, and include the needed levels of performance, reliability, usability, safety, security, evolve-ability and conformity to standards.

The following are the desired course outcomes:

- Understand the nature and role of system architecture and appreciate its strategic importance for the design and support of complex embedded systems and services.
- Decompose a complex embedded system into a set of simpler functional, physical and object-oriented elements.
- Develop a holistic architectural model of a complex embedded system and use it to assess system performance and drive detail design.
- Develop a complete set of design specifications for the subsystems and components of a complex embedded system by systematically flowing requirements down from system specifications.
- Create plans for addressing power, timing and live-ness (concurrency) of the system.
- Create plans for addressing system-level constraints including performance, reliability, safety, security and evolve-ability.
- Develop a trade-space analysis for one or more key system element (possibly CPU or other hardware selection, operating system selection, OTS software, etc.)

The syllabus from the initial course offering is shown below in Table 2.

Table 2: Design of CPS Course Syllabus

	Topic(s)	**Class exercises (Optional)**
Module 1	CPS Engineering Processes	
Module 2	Prioritizing and Allocating requirements	Specification of prioritized, detailed requirements
Module 3	Moving to Architecturally Significant Requirements	Identification of the relevant project ASRs
Module 4	Requirements and Architecture Drift and Traceability	Allocation of ASRs to (gross) blocks and interfaces
Module 5	Complexity in CPS	Plans for addressing project power, timing and live-ness
Module 6	Complexity management: addressing the “ilities”	Plans for addressing other constraints
Module 7	CPS models for embedded system computation	Contrasting models for the project

Module 8	Blocking and Tackling: Interfaces, Heterogeneity, reactivity and autonomous behavior	Interface design via SysML/UML stereotypes
Module 9	Systems engineering models and methods – MBSE intro for Mark	Architectural design for the project – SysML
Module 10	Alternative modeling methods – move to later course	Identification of Object design for the project – UML
Module 11	System partitioning, mapping and technology selection	Trade-space analysis of one or more technologies to be used in the project
Module 12	Domain driven design	Examination of relevant domain-specific standards
Module 13	Evaluating Architectures and Designs	Architecture/design review
Module 14	Final Design Reviews	

5.3 implementation of CPS: bringing solutions to life

This third course focuses on the continuous implementation, integration, testing, analysis, and verification and validation of cyber physical systems (CPS). This course builds on the previous course using a metaphor where the students plan on a successful product launch, using the project to work through continuous integration and test and ultimately bringing a robust solution to life in the form of a working CPS system. We intersperse lectures with individual and group project based activities to ensure that the developed system is functional and robust. We discuss and use the most effective techniques for fault and failure tolerance, analysis, and testing method and principles. We capitalize on simulation and physical systems resources for continuous and automated testing and discuss the balance of testing versus analysis. The students continuously collect evidence of quality, performance measures and traceability information.

This course provides the students with the skills and ability to develop and execute a verification and validation plan with an appropriate balance of test, analysis, reviews that produces measures of evidence given the risk and time constraints. In addition, the students are exposed to working together on multi-disciplinary problems and interacting with external customers and stakeholders.

The students "bring to life" their CPS system that integrates the dynamics of the physical processes with those of the software, simulation and networking, providing abstractions and modeling, design, and analysis techniques for the integrated whole.

The following are the desired course outcomes:

- Develop a V&V plan to define the balance of test, analysis, reviews that produce, measures of evidence given the risk and time constraints.
- Use relevant types of analysis and V&V measures when testing is not feasible to ensure test coverage and requirement-to-test traceability.

- Participate as a team to transform the design into a robust implementation using continuous integration and test driven development.
- Develop test strategies and test cases using CPS-relevant testing methods and principles that are most effective for finding defects.
- Perform automated testing enabled by design for testability using interface-driven test case design in a host and simulated environments.
- Use of simulation, and host testing to maximize the functional testing and assess traceability to the requirements.
- Use physical testing to maximize performance and stress testing with traceability to the requirements.
- Perform physical testing and use fault injection strategies to measure the robustness of the system.
- Produce traceability, analysis results and coverage measures to document the verification and validation evidence.

The syllabus from the initial course offering is shown below in Table 3.

Table 3: Implementation of CPS Course Syllabus

	Topic(s)	**Class exercises (Optional)**
Module 1	Course Overview	Working tool environment
Module 2	V&V Planning	Initial plan
Module 3	Robustness Analysis	Functional mapping to analysis or test method plan
Module 4	Continuous Integration and Test	Implement and integration of project
Module 5	Strategies and Methods for Testing	Testing scripts
Module 6	Automated Testing	Test results and pass/fail analysis
Module 7	Simulation Functional Testing	Apply to project
Module 8	Physical Performance and Stress Testing	Apply to project
Module 9	V&V Measures and Evidence	Test measures and evidence
Module 10	Robustness Assessment	Finalize working system

5.4 Sustainment of CPS: Managing Evolution

This course focuses on managing the evolution of a cyber-physical system after its initial release to the market until its retirement. The course approaches this topic on the following three levels:

1. The foundation is to put in place policies, processes and infrastructure to support, maintain and respond to quality issues for released instances of the system.
2. The second level is to drive the evolution of the system's capabilities and characteristics based on evolving needs and enabling technologies.
3. The third level is to proactively "disrupt" the market by reframing the opportunity and reinventing the system based on internal innovation, or responding to external disruptions in the marketplace or the technology space.

This course provides students with insights into issues pertaining to the life cycle of a cyber-physical system after its initial release to the market/customer(s). The main focus is on understanding how system scope and technical decisions made throughout the first three courses impact the ability to effectively and efficiently:

- support, maintain and respond to quality issues,
- evolve the capabilities and characteristics of the original system based on usage, user feedback and trends in the market and technology space and
- proactively (be the disruptor) or reactively respond to disruptions in available technologies and/or by competing/adversary systems.

The following are the desired course outcomes:

- Develop strategic and tactical concepts for system maintenance and support (M&S).
- Become familiar with life cycle models and create an architecture and "Concept of Operations" for the M&S operations and infrastructure.
- Be able to analyze and assess architectures from the view point of supportability, maintainability and the evolve-ability of a CPS, to create a Technology Roadmap.
- Develop strategies to be a "disruptor", or respond to disruptive changes in market conditions and/or new technologies.
- Make technical updates to their system to fix defects and make improvements based on their maintenance and support plan.

The syllabus from the initial course offering is shown below in Table 4.

Table 4: Sustainment of CPS Course Syllabus

	Topic(s)	**Class exercises (Optional)**
Module 1	Course Overview	
Module 2	Fundamentals of Integrated Logistics Support	Identify Maintenance and Support Elements for project
Module 3	Life Cycle Models	
Module 4	Fundamentals of Risk and Decision Analysis	Risk Analysis Exercises
Module 5	Computer System Warranty	Apply to project
Module 6	SW Release Strategies	Apply to project

Module 7	Maintenance & Support Infrastructures (including instrumenting the system to support M&S)	Identify M&S infrastructure elements for project
Module 8	Technology Road-mapping	Apply to project
Module 9	Obsolescence Planning	Explore Obsolescence strategies
Module 10	Capability Road mapping	Apply to project
Module 11	Architecture Assessment	Architecture Assessment exercise
Module 12	Disrupt or be disrupted/architecture migration	

5.5 program project

A critical component of the SE of CPS core program is the class project. Teams of 3-5 students are formed from the class, with the expectation that each team is composed of people with varied experiences and skills. In particular, it is best to have a mix of hardware, software and program skills, if possible. Since the projects require diverse skills, the students can contribute where they are strong, learn where they are weak, and the project can make good progress during the program. It is expected that each student will spend approximately 100 hours of time engaged in the team project, which results in a total budget of approximately 1,200 to 2,000 hours of total team effort per project spread over four courses. The project needs to have an appropriate balance of realism, yet not require an extensive amount of time to learn the specifics of the application, technology or tools. The project should be cyber-physical in nature requiring human interactions, embedded software and hardware, sensors, real time behavior and be networked, preferably having autonomous behavior and distributed control. The system should have existing libraries and simulators, provide a standard platform for engineers with base capabilities, and have the potential to be able to evolve over time. Finally, it should be compelling and fun for the students.

Based on these criteria, a telepresence robot system was selected. Telepresence robots are a set of technologies that allow video conferencing in such a way that the user feels as if they are actually at the remote site, and, conversely, allows those whom the user encounters to feel as though they are actually interacting with the user. The objective of such interaction is to improve collaboration. This is achieved through the cooperation of several systems, designed to interact in such a way as to increase the ability of the operator to be “omnipresent” without undue difficulty. The omnipresent system of systems, as a stand-in for the remote operator, must be mobile, perceptive of its visual and auditory surroundings, able to communicate visual and auditory information to its remote operator and able to present the live image and voice of the remote operator within its local environment.

The students have been assigned the entrepreneurial task of identifying a new market opportunity and then adapting a base prototype telepresence robot which they build from a kit of hardware parts and software that is available from a class download site to satisfy the required needs. In Course 1, the teams define a market and product concept. In Course 2, they architect and design the system, doing a deep dive on a selected portion of the system, based on how much they believe can be accomplished with their team resources. In Course 3, they implement the system and provide proof that it satisfies their development requirements. In Course 4, they sustain and improve the product, addressing issues that arise from customer use and emerging competition.

The selected prototype technologies include (as shown in Figure 1):

- iRobot™ Create for mobility and robotics platform
- X86 for overall system controller
- Linux® for the operating system
- Skype™, for networking, teleconferencing
- Python™ used as base software language

Student are encouraged to explore the alternatives, but do so knowing that there needs to be value add for these changes. The students are given a box of parts at the beginning of the first course and are required to assemble and have the prototype operational after the conclusion of the second course. However, most students built the prototype earlier than this. The students are also given a laptop in which all of the required software has been loaded. (These students did not have administrative privileges to load software on their work laptops.) Bit bucket was used as a team workspace and as a repository for program project materials. Freedom is given to the students to work on the projects at their convenience, as there are no formal lab or project times. Students may or may not be collocated, just as it may be in the workplace. The deliverables for the project from each course is the starting point for the project in the next course.

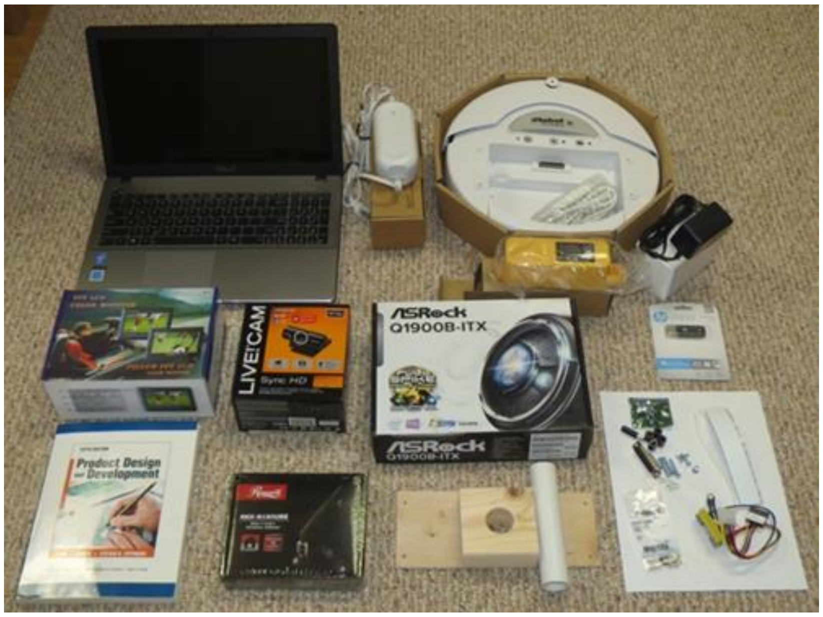

Figure 1. Components Used in Telepresence Robot Project

6 conclusion

The paper concludes with a description of the results obtained from the first pilot of the program and our future directions. As of February 2, 2016, the first four courses of the program have been completed by a pilot group of students. A wealth of information has been collected that is being used to update the program for the next cohort of students. The evaluation feedback has been very positive. Most significantly, the students overwhelming believed that:

- The course was structured to facilitate discussion and participant contribution
- The subject matter has significant usefulness to my organization
- I can apply what I have learned in this course on projects (underway or future) in my organization
- This course will enable me to enhance my career objectives

A general comment was that the students appreciated the opportunity to immediately apply what they had learned in a lecture on their project using an appropriate set of "real world" tools. In fact, the student's acceptance of straight lectures diminished as they grew to be accustomed to active learning. One of the student's requests was to have more examples that involved embedded/cyber-physical systems rather than more traditional systems engineering examples. They also wished to have more end-to-end examples worked out to help guide them in constructing an appropriate methodology for model-based systems engineering (MBSE). Currently, there is not a set of standard practices in place, so this continues to be a topic of research. Also, there were no distinctions in the course about what was considered "software engineering" and "systems engineering" with instructors who are members of each school. As a result, there was terminology that was used in different ways depending on the perspective. This will be resolved with the construction of a glossary of terms that will be used between both the systems and software engineering programs, both of which are contained in the School of Systems and Enterprises at the Stevens Institute of Technology.

One of the most important results was that the project served its purpose of providing a vehicle by which the students experienced an entire project lifecycle and effectively accelerated time by experiencing fairly rapidly the downstream effects of their design and implementation decisions. Students noted this numerous times during classroom discussion of their work. In addition, the project allowed the students to approach the system from both the perspectives of breadth and depth. In particular, Course 2 was an opportunity for the students to do a "deep dive" in an area of interest where they had to do a complete implementation with verification and validation. The student's noted that it was fortunate that they were not informed earlier about the need for a deep dive as this might have caused them not to take the broad view of the entire system in the earlier classes.

The delivered results from the students from each of the courses were of a very high quality. It was clear that the students were talented, had dedicated time outside of the classroom that was consistent with the instructor's expectations, had learned the course materials, and had developed skills in using the tools necessary to deliver their working systems.

The following are more detailed descriptions of the course results.

6.1 course 1 results

The first course focuses on the conceptual design of a cyber-physical system. The students are presented with a technology, in this case a telepresence robot prototype which has minimal capabilities, and they are asked to find a problem that can be addressed with this technology, and then develop a concept, financial model, concept of operations, use case scenarios and technical requirements. The student deliverables consist of a final report, spreadsheet analysis and a model of their concept using SysML.

While much of the material presented in this course was new to the students, a number of them found it to be “inspiring” and “makes me think outside my comfort zone”. The students desired greater depth in the areas of financial analysis and MBSE techniques. Given the amount of material and available time, one suggestion was to have MBSE exercises available for the students, along with working templates, that the students could use outside of the classroom, or add another day to the course. Finally, the students wanted more explicit expectations of what was required for each course *deliverable*.

The two teams produced very detailed final reports (67 and 84 pages in length) that were both of very high quality. The students leveraged their classroom experience in the Ideation process and systems thinking in the development of credible products. One of the teams had two members who had some experience in writing proposals and, thus, had more experience in working with customers. Their report was a bit more sophisticated in its market analysis, whereas the second team was more focused on the technological aspects of the system. Despite these differences, both teams produced product concepts that were responsive to the needs of a well-defined market, and supported by substantial market data.

Both teams created a solid set of prioritized requirements that were supported through their use of the Quality Function Deployment (QFD), also known as the *House of Quality*. Both used Kano Maps to support this prioritization for their selected markets and use cases. Both detailed the stakeholder needs clearly, developing product concepts and financial analysis to support their viability. While the financial analysis from both teams was rudimentary, this work was revisited in Course 4 and provided some *aha* moments for the students when they realized what they had missed. However, in future classes, the *stubs* for additional factors will be included in the financial spreadsheets.

Finally, both teams created quite complete descriptions of their concepts using SysML as a modeling language. While the teams took different approaches in this work, both were viable. However, the students thought that it would have been more efficient for them to have seen examples of a variety of approaches so that they could have more quickly settled on one. This is clearly a case of the tradeoff between efficiency and the deeper learning that is commensurate with self-exploration. This is an area that will require some experimentation in future classes.

In conclusion, it was apparent from the students’ work and comments that they had achieved the objectives for the course with methods and tools that they can bring to professional work.

6.2 course 2 results

The second course focuses on the architecture and design of robust cyber-physical systems. In this course the system concept and requirements developed in course 1 are transformed into a system design that supports the critical qualities of the resulting product.

Students came into this course with system requirements and trade study results regarding technologies available to satisfy the functional needs of the product they had conceived. To create an architecture, the two student teams needed to boil down their requirements into a set of architecturally significant requirements (ASR). The ASRs represented measurable qualities of the system such as speed of connection and response, power consumed/battery life, dependability (always working software), etc.

Since even small CPSs entail considerable careful design, students functionally decomposed their systems and then examined in depth one area of the decomposition for its architectural needs. Each team developed:

1. Detailed functional and non-functional system requirements for their respective systems
2. A functional analysis of the system's total composition with the identification of a *deep dive* area of capability for which a detailed architectural design would be developed.
3. A physical design for the system as a whole with detailed design decisions for the functional *deep dive* area.
4. A design prototype—a working robot implementing the design though not optimized for the qualities considered most critical to the project (yet).

SysML is used to capture all of these deliverables save the working prototype.

The two teams of students approached their work in very different ways. While one team chose to capture the *live* nature of their system via SysML block and activity diagrams, the other team developed detailed use cases/scenarios to capture the active processes needed within their system. Both teams developed excellent designs that achieved the fundamental goals of their implementations with sufficient flexibility to accommodate changing system requirements over time.

Examining the results of the course against the desired outcomes is instructive.

We sought to have students understand the nature and role of system architecture and appreciate its strategic importance for the design and support of complex embedded systems and services. The final reports of both teams of students in the pilot course demonstrated that they both understood and appreciated the value of architectural thinking for CPS.

Additionally, we wanted to give students the experience of developing a holistic architectural model of a complex embedded system and use it to estimate system performance and drive detailed design. In doing this, students successfully decomposed their complex embedded systems into a set of simpler functional, physical and object-oriented elements and developed a detailed design specification for a subsystem and components of a complex embedded system by systematically flowing requirements down from system specifications. They created, for their deep-dive area, plans for addressing power, timing and live-ness (concurrency) of the system as well as plans for addressing constraints including performance, reliability, safety, security and evolve-ability. Students had developed a *House of Quality* in the previous course from which they derived a trade-space analysis to assist in their design of their deep-dive area.

Student feedback largely indicated that the students benefitted from working with SysML and wanted more hands-on activities using SysML. Having a SysML tool that can be used in a Model-Based System Engineering (MBSE) effort, supporting students in analyzing the system behaviors resulting from their designs on paper, as it were, would be most desirable. But the expense of licensing such a tool and training the students to use it may exceed the budget and time available for this course.

In the next offering of this course, we will focus more energy on addressing the critical design issues surrounding concurrency, multi-core processing, performance and continuous integration throughout development. We will also provide students will hands-on architectural review activities where they will assess several design for *goodness* in critical areas on paper and, perhaps, via a model.

6.3 course 3 results

The third course focus more on the implementation, but brought in additional types of analysis for hazards, faults, failures and safety, which played important parts of both teams implementation. The approach and objectives were heavily based on:

- Develop V&V approach/plan that balances test, analysis, and reviews that produce measures of V&V evidence given risk and time constraints
- Participate as a team to refine requirements and architecture in transforming a design into a robust implementation using continuous integration and test
- Develop test strategies and test cases using CPS-relevant testing methods and principles that are most effective for finding defects
- Use design for testability and layered testing and to enable test efficiency in achieving test coverage and to support test-driven development and automation
- Use balance of simulation and host testing to maximize functional testing to achieve the needed level of test coverage based on risk
- Use relevant types of hazard, fault, failure and safety analyses to produce design constraints for a robust design and implementation
- Perform physical testing to assess performance, stress and robustness of system
- Produce traceability, analysis results and measures to document the verification and validation evidence

These objectives were very much focused on a producing risk-based evidence to support a verified and validate product. What turned out to be very interesting is that the teams approached the objectives from two different points of view. Team 1 was comprised more of engineers that typical work on small project teams of two or three people, and Team 2 was comprised of engineers that often work large program easily greater than 20 people, and often larger. In addition, while we provided guidelines for how they might approach the effort using the SysML, with an inexpensive and humbly-featured SysML tool, they both came up with innovated approaches by developing their own methodology.

Team 1 focused more on a risk-driven approach, which was quite different from that of the Team 2, yet it reflected on addressing the course objectives from a different viewpoint that comes from the types smaller projects in which your team is often involved (based on our discussions). Team 2 developed an elegant modeling methodology and approach, again using the modeling tool in a unique way regardless of some of the lack of feature that might be present in a more powerful modeling too. Their method and approach validated our team's decision to not impose a particular modeling approach.

Finally, both teams incorporated video demonstrating the physical testing results, and with a final remote driving session where the faculty was able to drive the final implementation.

The student’s comments affirm that the principles and concept for the program are on the mark. The completed work to date was of a very high quality, methodologically innovative and reflects positively on the success of the learning approach. The future plans are to improve the implementation of the courses by making the classes even more interactive shortening the lecture time between computer/tools exercises, increasing the CPS examples described in class, refactoring the course slightly to better balance the workload and tuning the program project to enhance its support of learning objectives.

6.4 course 4 results

The fourth and final course in the sequence deals with the managing life cycle of the system after its first release. The main deliverables for the course were:

- A Conceptual Design, a Concept of Operations, and an accompanying economic analysis of a Maintenance and Support infrastructure.
- A Product Roadmap to guide the evolution of new features based on vision, emerging technologies, and market analysis as well as customer/end-user feedback and usage monitoring.
- A Technology Roadmap of enabling technologies needed to deliver on the Product Roadmap
- A presentation to kick-off the concept phase of the next generation system with the aim to *change the game*
- A live demonstration of a key feature to the instructor team where one (or more) of the instructors serves in the role of a remote operator.

The students had little prior exposure to the issues that are associated with operating and sustaining a product. In particular, the business plans that they had constructed in Course 1went from being profitable to losing money based on these impacts. The students noted that this work had given them a far greater appreciation on these issues which they generally did not see in their professional work as they were not exposed to the issues with sustaining products. The students wanted to see more case studies to get a better appreciation of best practices and the pitfalls of making the inappropriate decisions. Rather than having a text book, the students desired a digital library of relevant materials. While the course material was mostly relevant, the warranty approach was too consumer oriented for their domain. They would have preferred a focus on built-in self-test (BIST) and other technical approaches to support serviceability.

acknowledgements

The authors wish to thank their corporate sponsor for their support in the development of this program and particularly to the students in the pilot course for their insights and excellent feedback. We, the instructors, are most often their students. We also are grateful to Steven Avery who developed the telepresence robot prototype, Rob Cloutier who provided his expert insights and served as SysML guest lecturer, Michael Pennotti who originated the four-course program structure, and finally, thanks to our Dean Dinesh Verma, the School of Systems and Enterprises and Stevens Institute of Technology for their support in this innovative program.

references

[1] Andreesen, Mark. “Why Software is Eating the World”, Wall Street Journal, August 20, 2011.

[2] Hirsch, Jerry. “Elon Musk: Model S is not a car but ‘a sophisticated computer on wheels’ ”, Los Angeles Times, March 19, 2015.

[3] National Academy of Science Interim report on Cyber-Physical Systems Education, 2015.

[4] National Institute of Standards and Technology, Foundations for Innovation in Cyber-Physical Systems, Workshop Report, 2013.

[5] ARTEMIS-GB-2012-D.46 – Annex 2, 2013. https://ec.europa.eu/research/participants/portal/desktop/en/opportunities/fp7/calls/artemis-2013-1.html

[6] www/cdio.org/cdio-vision, downloaded January 15, 2016.

[7] Jonson, Ben. “Design ideation: the conceptual sketch in the digital age.” *Design studies* 26.6 (2005): 613-624.

[8] Anne, Martensen, and Lars Grønholdt. “Using employee satisfaction measurement to improve people management: An adaptation of Kano's quality types.” *Total Quality Management* 12.7-8 (2001): 949-957.

[9] Hauser, John R., and Don Clausing. ‘The house of quality.” *Harvard business review* 66.3 (1988).

[10] Wassenaar, Henk Jan, and Wei Chen. “An approach to decision-based design with discrete choice analysis for demand modeling.” *Journal of Mechanical Design* 125.3 (2003): 490-497.