Abstract of Briefing at AlaSim 2023

A Mechatronics Virtual Testbed for Investigating Concepts and Practices in Software Engineering Education

Dan Tappan, Eastern Washington University

Computer science (CS) and software engineering (SE) programs have substantial overlap in content. In fact, the curriculum standards from the Accreditation Board for Engineering and Technology (ABET) differ really only in the modeling and simulation (M&S) requirement for the latter. CS programs generally do not cover it at all, and SE programs often have difficulty effectively integrating it into the curriculum. This work presents an innovative approach that has been successful in both environments to expose students to important theoretical and practical aspects of M&S in software development. Its goals are twofold: to provide a flexible simulation toolkit for designing and building prototype software systems based on real-world problems, and (2) to demonstrate the role of M&S in predicting or analyzing performance before, during, and after development.

Students generally do not understand the science in computer science or engineering in software engineering. This toolkit forces them to apply critical thinking to decompose a problem into components with a viable theory of operation (the science) and then design corresponding software solutions (the engineering). Specifically, it provides a vast range of capabilities to build a network of components that interact in a realistic manner to function as a system. The problem domain is mechatronics, which is a multidisciplinary combination of computer science and electrical and mechanical engineering. Underlying all these problems is the simple concept of moving an arbitrary component from one state to another. This action can mapped onto almost anything, from a virtual component like a digital clock to a physical one like a robotic arm. Rich configuration settings allow students to play what-if games with design considerations before committing to a solution, then to investigate them in operation later for test and evaluation.

The toolkit follows the input-processing-output paradigm. Inputs are sensors, which provide data on the state of components; outputs are actuators, which cause an action as described above; and processing is any kind control system that makes appropriate decisions based on the inputs and desired outputs. Students create components, connect them in a network, and exercise them under controlled conditions to monitor and evaluate performance objectively. The network supports a rich set of communication protocols to accommodate most types of common real-world actions in a student-friendly way.

Meaningful output is key to understanding how a complex system functions. This toolkit exports data in a variety of common formats that allow students to prepare detailed reports on questions of interest. This aspect specifically addresses students' weak communication skills.

This work have been deployed several times in the classroom. The main example to present at the conference is a digital fly-by-wire control system for a commercial aircraft. Such a project would typically be far outside the skill set and comfort zone of students, but results show they were very successful at solving the problem and learning from the process.