Keynote Address
Ada, Model Railroading, and Real-Time Software Engineering Education

John McCormick
(University of Northern Iowa)

While at the State University of New York at Plattsburgh I developed and taught an undergraduate course in real-time programming. This course became the capstone course in a curriculum called Computer Controlled Systems. The course started off in 1983 as one to improve student skills in low-level and device programming but soon evolved to be much more. Believing that the key to learning this material is working with real equipment rather than computer simulations, I set out to equip a laboratory specifically for teaching real-time embedded system programming. A local binding equipment manufacturing company donated a pile of dusty grease covered PDP 11/23 computers and a number of interesting I/O boards and sensors removed from the binding machinery they were upgrading for their customers. Digital Equipment Corporation (DEC) donated a PDP 11/24 along with a copy of UNIX that could be used to develop control software for the PDP 11/23 targets. Now the problem was finding something on an extremely limited budget ($800) that we could control with our newly acquired used computers. While walking past Plattsburgh Hobbies, I became inspired by the small model railroad layout in the window. After convincing the store owner to accept state of New York purchase orders, we were on our way to building the computer controlled model railroad that became the hallmark of the computer science department.

Students in the real-time course work in teams to specify, design, and implement software to monitor and control the model railroad. For the first 5 years, students wrote their software in C (it came for free with our donation from DEC). I was very disappointed the first year's project results. No team came close to getting their project working. The following summer I spent about 80 hours implementing one team's project to convince myself that it was indeed possible. In subsequent years I supplied students with portions of my code to reduce the amount of work required. Still no team ever managed to finish a project within the time limits of the 15 week semester - even when I supplied over half of the C code needed for their projects.

In 1989 I received a grant from the National Science Foundation to purchase new equipment for the laboratory. I was looking to replace the C language with one that supported concurrent programming. I was sure that I would buy a Modula-2 compiler for the lab. But DEC did not have Modula-2 for the hardware they were proposing I purchase. The salesperson convinced me to have a look at this language called Ada as it met all of the requirements in my request for proposals. She was convincing enough for me to part with over $10,000 (that's with DEC's large educational discount) for a DEC Ada compiler.

The year that the new software and hardware arrived provided a more real life example in system development than I would have liked in a teaching environment. Students in this year's class were developing the control software while I was building the circuitry to connect the new computers to the model railroad. I saw a disaster in the making. I finished the hardware with only two weeks remaining

in the semester. But in those two weeks, nearly 50% of my student teams got their projects working. The only code I supplied was a driver for the digital I/O boards. Needless to say, I was astounded by these results. The only significant change for the students in the course was the programming language. Changing from C to Ada greatly increased the productivity of my students. Over the next few years I gave a little more of my code to the student teams and found that almost 85% of my teams could finish their projects. Needless to say, I have since become a strong advocate of Ada in education.