Teaching Scientific Method for Real-Time Software Engineering

David A. Dampier
Department of Computer Science
Mississippi State University
Mississippi State, MS 39762
dampier@cs.msstate.edu

Ronald E. Wilson
INDUS Corporation
1953 Gallows Road, Suite 300
Vienna, Virginia 22882
rwilson@induscorp.com

In examining the literature on software engineering, we have noticed that the prevalent opinion is that people being hired today to develop software have little or no training in scientific methods. The result of this phenomenon is that developers often do not know how to systematically develop software that is reliable and of high quality. This factor is especially important in the engineering of real-time embedded software, where “shoot from the hip” software development can lead to devastating failures. We feel that understanding the process of science is key to ensuring quality and reliability in all systems, especially critical ones. This premise is founded on the belief that by understanding the scientific process, the practice of software engineering would be greatly enhanced and strengthened. Our intent is to demonstrate how the software engineering process and the scientific process are similar, and how university programs can benefit from adding scientific method instruction to their curriculums.

Scientific method is a process through which every advance to the state of the art is built upon known truths and previous advances. The state of the art evolves through an iterative process of building upon and correcting itself. Every scientific discovery, either uncovering some new truth or disproving previous mistakes is a contribution to the state of the art. Software should be engineered in the same way, especially real-time software. If we took the scientific approach to software engineering, there would be wide-spread re-use of artifacts from requirements to code. Systems would be designed and built for evolution, so inevitable changes to the systems would not degrade their integrity and quality. Additionally, mistakes made by one developer would be less likely to be repeated by others. For software engineering to become a true engineering discipline, we must establish, use, and evolve a body of knowledge that can be relied upon to provide standard methods for engineering software. The only way to realize this objective is to train our new engineers on basic scientific methods and establish a clear responsibility to evolve that state of the art in the profession. This can and should be done in graduate and undergraduate courses of study in software engineering, as well as in the ranks of practicing engineers. Only through coordinated effort, can we change the tide of ad-hoc, “code, compile, debug” practice that is rampant in our software industry. Only then can we gain confidence in software for critical real-time embedded applications.