Growing a Software Quality Culture In An Educational Environment

Trudy Howles

Follow this and additional works at: http://scholarworks.rit.edu/other

Recommended Citation

This Conference Proceeding is brought to you for free and open access by RIT Scholar Works. It has been accepted for inclusion in Presentations and other scholarship by an authorized administrator of RIT Scholar Works. For more information, please contact rit scholar works@rit.edu.
GROWING A SOFTWARE QUALITY CULTURE
IN AN EDUCATIONAL ENVIRONMENT

Trudy Howles

Abstract — The technical skills students must acquire in a typical computer science program are often mandated through standards or curricular requirements. How are non-technical skills assessed? Computer science educators must teach and encourage the development of other critical skills needed in the workplace such as personal accountability, a strong work ethic and an ability to deliver on-time and correct work. This paper describes the results of a student survey designed to provoke some thoughts about the evolving work ethic and work culture of today’s students. Along with the survey results, the importance in asking the questions and a brief analysis of how the behavior or activity fits into the quality cycle are presented. Finally, a section on continuous improvement strategies is proposed.

Index Term — Software Culture, Quality-centered, Software Quality, Work Ethic.

Much that is documented about software quality and reliability is not positive. For years, we have heard about substandard products and escalating development costs. Some high profile software defects have taken a toll on the industry. This alone should provide an incentive to find better ways to do business. Compounding this concern is the increasing number of technical jobs that are leaving this country. A recent article in BusinessWeek projected that as many as 473,000 U.S. computer jobs could be lost to other countries by the year 2015 [1]. Viruses, security holes, privacy breaches, and wireless technology have heightened awareness about software reliability, safety and security. It would be naïve, even arrogant, for us to ignore these issues.

New languages, programming paradigms, design tools and development models have been implemented and measured with limited and often arguable results. Many believe that the elusive key to improved software and increased productivity lies not in tools, but with people. Empowered, disciplined and well-trained professionals hold the promise of providing more stability, increased productivity, and product improvements that are not silver bullets.
The data sampling was primarily from first and second year students. Students responding to this survey were a combination of Computer Science, Software Engineering and Computer Engineering majors, all of whom are required to take the same programming course sequence in the first two years.

One hundred sixty-eight students responded. Of this number, 93 were first year students and the remaining 75 were in their second year.

The survey results and a brief analysis of how the behavior or activity fits into the quality cycle are presented. Finally, a section on continuous improvement activities specifically structured for educators is proposed.

REFERENCES
