Testing as a Mental Discipline: Practical Methods for Affecting Student Behavior

Stephen Frezza, Ph.D., C.S.D.P.  Mei-Huei Tang, Ph.D.
Gannon University Department of Computer and Information Science
{frezza001 | tang002}@gannon.edu

Abstract

Software testing plays a key role throughout the development of high-quality software systems. The question often raised is how to motivate and encourage students to perform test design early in their development process. This tutorial presents practical methods for introducing students to Testing as a Mental Discipline and related exercises that help students see the value of changing their development behavior with respect to testing.

1. Introduction

Testing is traditionally defined as “the execution of a program with the intent of finding errors.” This core definition, though useful, does not provide educators with an effective means to communicate either the goals or the value of software testing, e.g., testing and test design, as part of quality assurance, should also focus on bug prevention. Similarly, the related mantra to ‘test, then code’ has found a resonant home in the agile software development community (e.g., TDD: Test-Driven Development).

2. Tutorial Content

This tutorial focuses on “Testing as a Mental Discipline” (TMD) as the most effective form of ‘test thinking’. In teaching this behavior, the presenters have developed a series of exercises that aim not only of communicating the value of TMD, but also of allowing students to experience the benefits of TMD behavior in their development assignments. The goal is to demonstrate the benefits of Testing as a Mental Discipline (TMD), and methods to both measure and encourage this behavior that can be applied in programming, design, testing and other software engineering contexts.

The tutorial will begin with a brief introduction to test thinking, and motivate how this applies to different software development phases. This is followed by an exercise for participants to help them understand and apply the five phase metric for test thinking. At this point, the tutorial will include discussion and interaction on the difficulties of TMD, and how to overcome these in development activities. The latter half of the tutorial will focus on practical methods for affecting TMD behavior in student exercises, and will include a series of short exercises that can be applied in a variety of software development teaching/training contexts. The tutorial will conclude with a brainstorming session on other ways to motivate and encourage TMD development behavior.

While the tutorial will utilize several short presentations on core material, the majority of the time will be spent on exercises to model, measure and motivate TMD behavior. These exercises will include, but are not limited to: Measuring test thinking, Test design first, Reflection, Postmortem, and Producer-consumer.