Testing as a Mental Discipline: Practical Methods for Affecting Developer 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

1. Introduction

Testing is traditionally defined as “the execution of a program with the intent of finding errors.”[1] This core definition, though useful, does not provide educators with an effective means to communicate either the goals or the value of software testing. “Testing and test design, as part of quality assurance, should also focus on bug prevention.” [2] 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). [3]

The ‘test first’ approach sources include Boris Beizer’s seminal work Software Testing Techniques, 2nd Ed. [2] wherein he presents an attitudinal progression labeled “Phases in a Tester’s Mental Life:

- **Phase 0** – Testing = Debugging
- **Phase 1** – Test to show that the code works
- **Phase 2** – Test to show that the code doesn’t work
- **Phase 3** – Test to reduce the perceived risk
- **Phase 4** – Testing is a Mental Discipline applied throughout the development lifecycle

These phases we term as “Test Thinking,” with Phase 4 “Testing as a Mental Discipline” (TMD) as the most effective form. In teaching this behavior [4], we 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.

2. The Workshop

This workshop focuses on “Testing as a Mental Discipline” (TMD) as the most effective form of ‘test thinking’. Here, ‘test thinking’ is an extension of the test-first development shown to be effective in many agile development methodologies. The workshop expects to explore TMD successes and hopes from participants, with the following goals:

- Quantify the value of Testing as a Mental Discipline
- Examine methods for measuring “test thinking”
- Attempt to qualify the value of different types of TMD development behaviors
- Explore methods/activities for encouraging TMD development behavior

The workshop will begin with a brief introduction to TMD, and explore how these methods can significantly benefit software. This will be followed by a participatory exercise to help participants apply a five phase metric for measuring test-first thinking.

The workshop will include discussion and interaction on the difficulties of TMD, and how to overcome these in development activities. The latter half of the workshop will focus on exploring practical methods for affecting TMD behavior, and will include a
series of short hands-on activities that can be applied in a variety of software development and/or training contexts.

The workshop will conclude with a brainstorming session on the difficulties involved and other ways to motivate and encourage effective TMD development behavior.

3. Target Audience

This workshop is designed primarily for managers, developers, trainers or university faculty interested in:

1. Incorporating test-first into programming, design, or other software engineering activities
2. Applying Test-Driven Development, extreme programming or other agile methods
3. Supporting quality assurance, testing and/or test-driven processes

Participants will benefit more from the workshop if they come with a basic familiarity with testing terminology, test case design methods, fault models and test result reporting. Experience in practical software quality assurance and/or agile development techniques is an added benefit. (Or anyone interested in fun hands-on activities 😊.)

4. Topical Outlines

1) Beizer’s mental scale… (Short Exercise)
2) Difficulty of type 4 thinking – (What is Test-First Thinking, anyway)
   a. Type 4: Testing as a mental discipline…
   b. Value of Type 4 Thinking
   c. Seeing type 4’s: Test design as requirements validation (Hands-on)
3) Teaching “Test First” (Ways to affect Test-First Behavior)
   a. Minimum Testing Background for developers
   b. Methods
      1. Ask for test cases prior to code (Hands-on)
      2. Reflection on test thinking (Model, ask for the type 4s in test case design)
      3. Postmortem Work (Bug Meeting reports: What bugs, at what cost)
      4. Consumers of software deliverables (Hands-on)
4) Brainstorming:
   a. Other ways/methods to motivate and encourage TMD development behavior.
   b. Difficulties/issues may be/or experienced while applying TMD methods and possible ways of resolving them.
   c. Possible applications of TMD methods.

References