JOURNAL OF OBJECT TECHNOLOGY

Online at http://www.jot.fm. Published by ETH Zurich, Chair of Software Engineering ©**JOT, 2010**

Vol. 9, No.3, May-June 2010

Book Review

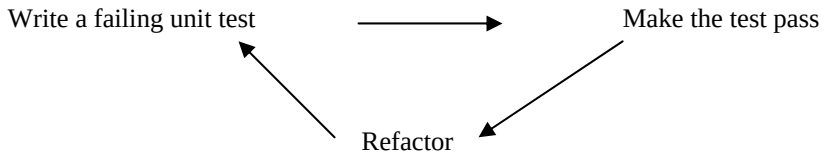
Growing Object-Oriented Software, Guided by Tests

by Steve Freeman and Nay Pryce, Addison-Wesley Pearson, Boston, MA, 2010. 358 pp., $44.99 (paper). ISBN 0-321-50362-7.

Reviewed by Charles Ashbacher, Charles Ashbacher Technologies, Hiawatha, IA, USA, cashbacher@yahoo.com

I have been sold on the software development concept of Test-Driven Development or TDD since the day that I first read about it. Back in the days when programs were a few hundred lines we were writing the code to satisfy our tests, which were nothing more than what the software was supposed to do. Since a single programmer could wrap their mind around the entire program, you wrote the code to satisfy the intermediate and final conditions.

When code modularization arose to help in the creation of large software packages, it was necessary for the testing process to become localized, testing the individual modules and then how they worked in combination. TDD forces you to think thoroughly about the software because you are required to know exactly what it is supposed to do, as you build the test before writing the code rather than after. This also keeps the programmer from falling into a tunnel vision mode, where they write the test that the code will pass rather than the test that the code must pass. The fundamental TDD cycle is



and by making the units small, the granularity of the iterations can be made as small as desired or needed. This allows for the creation of small iterations very rapidly.

This book is primarily an in-depth demonstration of how TDD is used in large projects; in this case you are stepped through the development of a program that will place automatic bids in an online auction. JUnit 4 is used in the construction and management of the tests. The complete development cycle, from the opening sketches of

the design to the logging of the error messages is covered. In a tactic that some may find quaint but that I found endearing, the authors kept the various design sketches and To-do lists in their scratch form. Writing these things in a text processor makes them neater, but to me there is something very appealing about hand-written sketchy notes as that is how software is developed.

The last three chapters are considered advanced topics and have the titles:

- Testing persistence
- Unit testing and threads
- Testing asynchronous code

I certainly would not dispute the labeling of this consideration; these topics are generally (very) hard to code and test. No section of 38 pages could possibly cover these topics in any depth, all that can be done is to skim them, but what is done is done very well.

TDD is an easily grasped development technique that has been proven to be a very effective way to build large software systems. Unlike some other development techniques where the complexity can overwhelm you, it scales well because it is built from small sections and is based on how humans think. This book is an excellent presentation of TDD and a worthy book to introduce the topic to advanced programmers.