JOURNAL OF OBJECT TECHNOLOGY

Online at www.jot.fm. Published by ETH Zurich, Chair of Software Engineering ©**JOT, 2003**

Vol. 2, No. 2, March-April 2003

Book Review

Test-Driven Development: By Example

by Kent Beck, Addison-Wesley, Boston, MA, 2003. 216 pp., $29.99(paper).
ISBN 0321146530.

Reviewed by Charles Ashbacher

I will never dispute the basic premise of this book, namely that code development should be done in small steps, each of which is immediately verified. That is the way that I code, but not to the extremes that Beck does. It is easy to follow the examples, but sometimes you groan at the tiny size of the steps.

As many people have emphasized for many years, the main ingredient in the creation of correct code is logical and proper thought. Test-driven development or TDD forces designers to think more at the proper time. As you write a segment of code, you must be thinking about a test that will verify that specific change, rather than a test that must cover several layers of interacting changes. This requires that the code be looked at from more than one perspective and to greater depth.

TDD is done according to the following rhythmic set of actions:

Repeat until code is complete:

1) Add a test.
2) Run all tests and see the new one fail.
3) Make a little change.
4) Run all tests and see them succeed.
5) Refactor to remove duplication.

Note that the usual sequence of code then test is replaced by the reverse, test then code. In other words, you determine what the test should be first and then write the code that will pass that test as well as all previous tests. This may appear unusual, but is the way programming has always been done. The code specifications are a set of broad tests that the software must satisfy, so the difference is in degree rather than in kind.

Do I think that TDD is an effective way to create software? It is clear that the reduced pace and greater thought will prevent the inclusion of many of the smaller bugs. However, I do see a problem with catching bugs due to the interactions of software in several layers. The focus of TDD is on such small changes, that one could easily be

Cite this book review as follows: Charles Ashbacher: Review of “Test-Driven Development: By Example”, in *Journal of Object Technology*, vol. 2, no. 2, March-April 2003, pp. 203-204.
http://www.jot.fm/books/review7

blinded to bugs caused by larger forces. Furthermore, the high level of refinement espoused by Beck would no doubt reduce the speed of development and may drive up the cost. I also can imagine the incredulity of a manager when they see that I am running a test that I already know will fail.

All that aside, this is a development strategy that holds promise, particularly in the most refined level of development. Therefore, I recommend it to people who code at the most detailed levels, but question the value to those who perform integration tasks.