Does TDD Really Ensure Quality?

Ben Hughes

There's been some interesting commentary on the National Research Council of Canada's paper titled "The Effectiveness of Test-first Approach to Programming" . The study, carried out on a sample size of 24 IT graduates, adds to the growing body of research on the topic. Though TDD is accepted as an excellent learning tool for quickly understanding the domain in which developers work, the question of whether TDD directly correlates quality in software is still considered unproven by some. This study, while still not conclusive, does show some interesting results - though different results, depending on who's analysing them.

The study's abstract reads, in part:

Test-Driven Development (TDD) is based on formalizing a piece of functionality as a test, implementing the functionality such that the test passes, and iterating the process. This paper describes a controlled experiment for evaluating an important aspect of TDD: In TDD, programmers write functional tests before the corresponding implementation code.

The experiment was conducted with undergraduate students. While the experiment group applied a test-first strategy, the control group applied a more conventional development technique, writing tests after the implementation. Both groups followed an incremental process, adding new features one at a time and regression testing them.

The researchers noted: "The results of the experiment support an alternative theory of the Test-First technique that is mainly centered on productivity rather than on quality."
Our main result is that Test-First programmers write more tests per unit of programming effort. In turn, a higher number of programmer tests lead to proportionally higher levels of productivity. Thus, through a chain effect, Test-First appears to improve productivity.

... We also observed that the minimum quality increased linearly with the number of programmer tests, independent of the development strategy employed.

However, one blogger, Jacob Proffitt, a self-described "passionate developer, sometimes manager, and general all-round techno-geek," probed the paper and blogged his critique of it., proposing that the paper shows a strong tendency toward confirmation bias - i.e. coming to conclusions in spite of the findings of the work. He believes that "TDD’s relationship to quality is problematic at best," citing:

The control group (non-TDD or "Test Last") had higher quality in every dimension—they had higher floor, ceiling, mean, and median quality.
The control group produced higher quality with consistently fewer tests.
Quality was better correlated to number of tests for the TDD group (an interesting point of differentiation that I’m not sure the authors caught).
The control group’s productivity was highly predictable as a function of number of tests and had a stronger correlation than the TDD group.

Jacob proposes that the only facts this study's data tells us are:

The test-first students on average wrote more tests.
Students who wrote more tests tended to be more productive.
The minimum quality increased linearly with the number of tests.
Hakan Erdogmus, editor of IEEE Software Magazine and co author of the original paper, views these points from a different perspective:

A single study, especially a small one like ours, regardless of how well conducted, does not prove or disprove anything. The observations at best shed light to a small part of a large puzzle. In many circumstances, they raise more questions than they answer, hopefully more relevant questions that improve our understanding of the phenomenon under study ... In fact, "proof" is not part of the empirical software engineering terminology. Strength of collective evidence and building refutable theories are the best we can achieve by studying a specific technique. While for certain few practices, notably software inspections, we are now able to state that the evidence is strong. But the jury is still out for TDD.
More so, Hakan told InfoQ this about the wider TDD discussion in context of the breadth of research done so far:

The 23 TDD studies published between 2001 and early 2008 provide somewhat conflicting results, but a big picture is emerging on closer inspection. The differences in findings stem from the multiplicity of context factors that influence the outcome variables measured. On the quality front, the results are more compelling, if not resoundingly in agreement. Of the 22 studies that evaluated some aspect of internal or external quality with vs. without TDD, 13 reported improvements of various degrees, 4 were inconclusive, and 4 reported no discernable difference (including our study). Only one study reported a quality penalty for TDD.
Bookmark digg+, reddit+, del.icio.us+, dzone+ Tags Criticism, TDD, Statistics, Testing
RelatedVendorContent
J2EE Without Application Servers

Free 10-User Agile Project Management Tools

A New Approach to Deploying and Managing Java EE Applications

Delivering a Breakthrough Java ™ Computing Experience

MS Office as an application & integration development platform @ Office Developer Conference Feb 10-13

14 commentsReply

Test First or TDD? by Michael Neale Posted Jan 25, 2008 6:34 PM
Re: Test First or TDD? by Deborah Hartmann Posted Jan 25, 2008 8:29 PM
Re: Test First or TDD? by No Name Posted Jan 26, 2008 5:17 AM

Test First or TDD?
Jan 25, 2008 6:34 PM by Michael Neale

You keep using the word TDD, I do not think it means what you think it means. Headline should read Test First instead of TDD (and remove all references to TDD) and then it makes sense.

eg:
"the control group applied a more conventional development technique, writing tests after the implementation. Both groups followed an incremental process, adding new features one at a time and regression testing them. "

What? so the control group was actually doing TDD? (ie tests were still deep part of the development process).

Reply

Back to top

Re: Test First or TDD?
Jan 25, 2008 8:29 PM by Deborah Hartmann

Good point. I think we take it for granted that the two are interchangeable, because TDD is generally test-first. The inverse is not necessarily true, is it?

Reply

Back to top

Experimental validity
Jan 26, 2008 5:10 AM by Amr Elssamadisy

These experiments were run with undergads. Off the bat, this experiment fails both external validity (a.k.a. generalizability) and statistical validity - sample of 24 students in the same school.

There is also the related infoq article on analyzing experimental data.

Reply

Back to top
Re: Test First or TDD?
Jan 26, 2008 5:17 AM by No Name

I'll try to sort out the confusion.

TDD is always test-first but test-first is not always TDD.

TDD is the name of the micro process "red-green-refactor" which requires testing first.

Reply

Back to top
Re: Experimental validity
Jan 26, 2008 10:39 AM by Dave Rooney

Amr, you beat me to it! :)

I have the same reservations about the conclusions, although I believe it would be difficult to run such a study in a true business environment.

Dave Rooney
Mayford Technologies

Reply

Back to top
Re: Experimental validity
Jan 26, 2008 2:34 PM by Deborah Hartmann

Studying this stuff in any context seems devilishly hard. I've heard the story of the plant where turning up the lights improved productivity. Later, turning down the lights improved productivity. What exactly is it that makes humans productive?

We simplify it for ourselves because otherwise we wouldn't be able to play at all in the process improvement domain. But, really, we need to remember that this is all VERRY complex :-)

For this reason we must do the shortest iterations we can, when doing process improvement, to at least shrink the huge list of variables somewhat.

Reply

Back to top
Re: Experimental validity
Jan 28, 2008 2:57 AM by Guy Coleman

What exactly is it that makes humans productive?

Well, from what you say it should be obvious: flashing lights.

Reply

Back to top
Re: Experimental validity
Jan 28, 2008 8:02 AM by Sameer Alibhai

This article - Research Supports the effectiveness of TDD confirms that this is can be generalized:

The researchers do address this question of validity...

The external validity of the results could be limited since the subjects were students. Runeson [21] compared freshmen, graduate, and professional developers and concluded that similar improvement trends persisted among the three groups. Replicated experiments by Porter
and Votta [22] and Höst et al. [23] suggest that students may provide an adequate model of the professional population.

Reply

Back to top
Re: Experimental validity
Jan 28, 2008 8:15 AM by Deborah Hartmann

I find it vaguely depressing that

...students may provide an adequate model of the professional population.
when so many of us have worked hard for many years to become "skilled professionals" :-)

But I'm not questioning the validity of that statement, just reflecting on what "average" looks like in our industry.

Reply

Back to top
Re: Experimental validity
Jan 28, 2008 10:48 AM by Amr Elssamadisy

Actually, digging a little deeper.... Runeson[21] says:

The conclusion drawn from the study can neither reject nor accept the hypothesis on differences between freshmen, graduate students, and industry people.

While Porter's work is on software inspections (and replicated by Basili) which is not exactly TDD.

Reply

Back to top
Re: Experimental validity
Jan 28, 2008 10:49 AM by Amr Elssamadisy

While Porter's work is on software inspections (and replicated by Basili) which is not exactly TDD.

Sorry, typo - Porter, Votta, and Basili are all authors in the paper.

Reply

Back to top
Re: Experimental validity
Jan 28, 2008 4:04 PM by Hakan Erdogmus

I find it vaguely depressing that

...students may provide an adequate model of the professional population.
when so many of us have worked hard for many years to become "skilled professionals" :-)

No need to get depressed. "Adquate model" does not mean that professionals and students are equally skilled. It only means if a technique improves students' performance, it is also likely to improve professionals' performance.

Regarding Amr's comment, he is right, all student studies have low generalizability (external validity) by definition, including this small study (of which I was an investigator and author). But the argument can swing in both directions: a complex technique may require maturity/skill, and may be ineffective for students while effective for professionals. Or vice versa. So observed effects may be reversed, amplified, or dampened across different groups. Complex techniques tend to be better leveraged by highly skilled people.

Also accepting/rejecting a hypothesis in a single study, or even in multiple studies, is not proof. We didn't prove TDD was effective. Just scratched the surface in terms what factors might be influential or explain the differences between two groups of students. Again, this is just one study folks, with a very specific design centered on the test-first dynamic of TDD (in a way, we were effectively controlling for quality by chosing an inverted TDD dynamic as our control group). So it's impossible to make any sweeping statements about TDD based on it.

Regarding statistical significance, you can have statistical significance with small samples. But significance is not meaningful in isolation regardless of sample size. Sometimes when you have significance, the effect size (measured by a particular statistic) is so small, you may not care. Other times, the effect size may be so complelling that you may care, even if significance is low. Lack of significance is a not good reason to dismiss findings, just as presence of significance is not reason for overinterpretation.

Any way, on reading the paper, some remark it's cautious, conservative, and tentative about interpretation. It also appears that reader biases may strongly color interpretation. We get comments from both camps who attempt to use the findings to advocate TDD ("see it proves TDD") or vice versa ("see it disproves TDD"). It's neither.

Reply

Back to top
Re: Experimental validity
Jan 28, 2008 8:49 PM by Ken Ciszewski

The key requirement to getting software "right" is knowing what it is supposed to do when it works correctly. Writing tests first tends to embed the operational requirements in the testing. The tests become a "self-fulfilling prophecy" of sorts, and the software development becomes an

interative process that works toward passing the tests (which a little like teaching the tests in school). If the tests are thoroughly prepared, the results should be very good.

Reply

Back to top
Skill Distribution
Jan 29, 2008 12:26 PM by Wayne Mack

To me the biggest question is related to skills distribution (Table 5 in document and noted in Section 3.7 p 6-7). The Test-First group had 3 rated as Low, while the Test-Last had zero. This might very well lead to the smaller distributions shown in the Test-Last group. It might be interesting if the study analyzed across similar skill levels, though that leads to extremely small population sizes. This experiment probably requires a repeat with skill levels controlled for before it can be used as a valid evaluation of methodology.