Agile Testing Section Draft

Chris M. Thomas
Division of Science and Mathematics
University of Minnesota, Morris
Morris, Minnesota, USA 56267
thom3706@morris.umn.edu

Keywords

ACM proceedings, LaTeX, text tagging

1. SENIOR SEMINAR PAPER GOALS

The goal of this paper is to explore what entails agile testing and its apparent effectiveness in the practical coding world. Specifically I would like to explore the current implementations of agile testing and see how it has evolved overtime, and how agile testing compares to conventional testing. My articles: [2, 8, 6, 3, 9, 5, 7, 1, 10, 4]

2. KEY POINTS

1. Testing is very costly and not always effective for industry so optimizing testing practices is very important. This was a key point that was brought up in multiple papers, some of which even stated that prior research showed that up to 50% of all development funds was invested in testing [1, 5]. This point is important for understanding the relevance of this paper and this point is also useful for putting testing advantages and disadvantages in a useful context. One article in particular [1] really highlights this but many of the other articles touch on it in one way or another.

2. TDD is the main form of Agile testing This is acknowledged in almost every article I found on the topic and though few mention other forms of testing the bulk of the interest and detail seems to be focused on TDD, usually in its original form within Extreme Programming. TDD is heavily mentioned in the following articles [2, 3, 4, **?**, 5, 10]

3. TDD has inconclusive results tied to it This key point was brought up a lot in papers when explaining TDD itself or the results of TDD experiments. for example some papers say TDD makes a product take longer to produce [6, 3] while others say it decreases the overall development time [5]. Other data points that I ran into that had this same indecisive data problem was whether TDD improved code quality, improved customer satisfaction, and/or created less buggy code. Perhaps the only agreed on data point is that TDD tends to produce a massive increase in code coverage and test size (no articles disputed this point and many reputed it). In particular papers [3, 4, 5] seem to spend some time attempting to explain this phenomena and all agree that some of the major reasons they give for why this data is all over the place is A. TDD is actually hard to implement correctly, B. TDD is a very vague term that is not well defined so many different testing methods are done under the same name.

4. New TDD approaches are evolving to help generate goals to promote customer satisfaction and reduce unneeded complexity. This key point is brought up by only two articles directly, [**?**, 3] but it was referenced indirectly in other articles. Although there is not much backing for what this point is I think it is an important point to bring up as it is showing the future of agile testing.

3. OUTLINE BEGINS HERE

4. INTRODUCTION

Here I plan to talk about the idea of agile testing and why anyone should care. I also plan to lay out a road map here for the rest of my paper.

5. BACKGROUND

5.1 Software Testing

in this portion I will: -Give a basic explanation of software testing -Give the main goals of software testing -Express the importance of software testing -Go into current testing methods

5.2 Agile Programming

in this portion I will: -explain Agile Ideology (make sure to list important tenents like customer communication) -explain basic TDD

6. ANALYSIS OF TDD

6.1 TDD Stances and Data

Currently, TDD is the most popular testing method in the Agile community. Many supporters of TDD claim that it reduces overall time spent on a product, improves test coverage, and increases overall code quality. More importantly there exists data to back up some of these claims. for example, in the experiment documented by [6], it was noted that on average TDD methodologies increased code coverage by 40% compared to conventional testing methods.

UMM CSci Senior Seminar Conference, December 2013 Morris, MN.

Another study, documented in [**?**] also suggested some positive traits about TDD by claiming some research showed that TDD practices could reduce effort by up to 27%. Studies in articles [4, 3, 5] acknowledged multiple studies that empirically showed at least a marginal increase in product quality when TDD was implemented. By looking at this data alone it would be relatively easy to see why TDD has such a positive image in industry and academia.

That being said, TDD also has many detractors as well who would tell you that TDD actually increases time spent on a product while not increasing overall code quality. This side also has a fair bit of research backing its opinion. For example, in [6] the same study that showed that TDD increased test coverage by 40%, it was noted that TDD took significantly more time to implement then conventional testing and did not actually increase code quality. Also, the same study that found research showing the TDD could reduce effort by up to 27% ([4]) found other research that suggested that TDD actually increased the effort by two fold. It is also worth noting that articles [3, 5] also have data and reports arguing that TDD takes more time then conventional testing. What is perhaps troubling about the data given here is that most of it directly contradicts the data given in the first paragraph. This suggests an interesting paradox where even though data given on TDD is statistically significant it is decidedly inconclusive.

6.2 Analysis of Conflicting Data

Even though the data surrounding TDD is inconclusive as a whole it should still be noted that information can still be extracted from it. This is especially true because many of the results were statically significant meaning that some sort of meaningful data was obtained. Because most of the data was statistically sound it may be worth considering not that the data was faulty but perhaps that the studies contained potentially key differences from one another. Thus it may be useful to us to try to understand why the data may be so conflicting as it may lead to useful insights about TDD indirectly.

One of the largest issues for summarizing the effectiveness of TDD is the field itself is very broad. It is important to note that while I specifically defined the original ideas and methodologies of the original idea of TDD, TDD itself has become an umbrella term that has been used to describe a massive array of different testing practices. This diversity may help explain why we are getting differing results because the TDD practice in one study may be extraordinary different from the TDD practice in another study. It is worth noting that in [3] a major issue with many TDD research papers was the absence of how the TDD process was implemented. This is a major issue in studying the field of TDD because it is hard to tell if people used similar or different methods of TDD to obtain their results.

Another issue for understanding the effectiveness of TDD that many TDD researchers have run into is that TDD is actually a difficult process to implement correctly. There were many testimonies throughout various articles [3, 4, 5] of industry professionals finding the TDD process to be much more difficult to implement correctly compared to traditional testing frameworks. Also, many articles on the analysis of TDD at some point brings up the topic of the complexity of TDD. This acknowledgement of complexity is important as it may suggest a result bias in TDD experiments based on the skill level of the participants as less experienced participants may be significantly less likely to implement TDD in an efficient or successful manner while more experienced participants may have more success. In one study at a college, although not significantly analyzed due to a lack of participants, it was noted that Alumni who used TDD produced overall better quality code compared to their conventional testing Alumni counterparts while current students in the college showed almost no difference in code quality between TDD and conventional test methods [6]. This potential bias would wreck havoc on TDD data conclusiveness as a whole as the two major participant types for TDD experiments are industry professionals and college students who have vast gaps in experience difference.

Overall it is very hard to draw conclusions on the effectiveness of TDD. Out of all the data out there there is really only one data point that seems to be incontrovertible which is the fact that TDD always seems to increase overall code coverage. Perhaps another conclusion that can be made is that TDD seems to produce no worse code quality then traditional waterfall testing. Also there are some implications from potential confounding factors that not all TDD methods are created equal and experience may play a large role in TDD effectiveness. Everything else on the effectiveness of TDD seems to be inconclusive.

7. EVOLUTIONS TO TDD

7.1 BDD

Here I plan to explain what BDD is and why it may be better then TDD.

7.2 ATDD

Here I plan to explain what ATDD is and why it may be better then TDD.

8. CONCLUSION

Here I plan to talk about the main take away points of this paper

9. REFERENCES

[1] A. Bertolino. Software testing research: Achievements, challenges, dreams. In *2007 Future of Software Engineering*, FOSE '07, pages 85–103, Washington, DC, USA, 2007. IEEE Computer Society. *Validity: Very Good. This seems like a good article to read on testing in general. If this paper explores testing like I think it will, it may help set the tone of the entire paper and clarify important traits in testing. I think this will be a key background paper.*

[2] B. George and L. Williams. An initial investigation of test driven development in industry. In *Proceedings of the 2003 ACM symposium on Applied computing*, SAC '03, pages 1135–1139, New York, NY, USA, 2003. ACM. *Validity: Very Good. This article is a bit old but well cited and again provides more data of comparing agile testing to other standard testing techniques. Due to this articles age I donâĂŹt think it will ever be a core source. If it has any use i think it will be as a background mention.*

[3] S. Hammond and D. Umphress. Test driven development: the state of the practice. In *Proceedings*

of the 50th Annual Southeast Regional Conference, ACM-SE '12, pages 158–163, New York, NY, USA, 2012. ACM. *Validity: Good. This short article seems useful on getting the current scoop on Agile testing. I do not think that it will be a main paper unless it is very dense or relevant but I think it would serve well as a support or background paper.*

[4] T. Hellmann, A. Sharma, J. Ferreira, and F. Maurer. Agile testing: Past, present, and future – charting a systematic map of testing in agile software development. In *Agile Conference 2012*, AGILE 13, pages 55 – 63, 2012. *Validity: Good. This article I think will be a useful summary article on agile testing. This article would make a useful background article. University may not be able to access this article, in that case this article will be ignored. .*

[5] V. Kettunen, J. Kasurinen, O. Taipale, and K. Smolander. A study on agility and testing processes in software organizations. In *PGood/Questionableroceedings of the 19th international symposium on Software testing and analysis*, ISSTA '10, pages 231–240, New York, NY, USA, 2010. ACM. *Validity: Very Good. This article seems very promising as its a recent paper that has research of the advantages of agile testing compared to conventional testing. The writers seem very credible in the field. This paper is likely to be a core paper unless its actually poorly written or doesnâĂŹt mesh well with other papers.*

[6] O. A. L. Lemos, F. C. Ferrari, F. F. Silveira, and A. Garcia. Development of auxiliary functions: should you be agile? an empirical assessment of pair programming and test-first programming. In *Proceedings of the 2012 International Conference on Software Engineering*, ICSE 2012, pages 529–539, Piscataway, NJGood/Questionable, USA, 2012. IEEE Press. *Validity: Very Good. This article seems useful as it is a modern research paper on the effectiveness of agile testing practices. That being said one of the main downsides of this article is that it splits its attention between TDD and Pair programming which is concerning.*

[7] J. T. Sawyer and D. M. Brann. How to test your models more effectively: applying agile and automated techniques to simulation testing. In *Winter Simulation Conference*, WSC '09, pages 968–978. Winter Simulation Conference, 2009. *Validity: Mediocre. This article I think would be useful due to how it covers actual goals and ideas agile testing in detail. This article has a potential to be a core paper due to its potential ability to define the main topic.*

[8] V. Schneider and R. German. Integration of test-driven agile simulation approach in service-oriented tool environment. In *Proceedings of the 46th Annual Simulation Symposium*, ANSS 13, pages 11:1–11:7, San Diego, CA, USA, 2013. Society for Computer Simulation International. *Validity: mediocre. This is a promising article that is very recent. This article I foresee as the best for seeing where agile testing may go in the future. I am concerned how relevant the focus of the article may be to my paper so until I read it I am not sure what the role of this article will be.*

[9] M. Soeken, R. Wille, and R. Drechsler. Assisted behavior driven development using natural language processing. In *Proceedings of the 50th international conference on Objects, Models, Components, Patterns*, TOOLS'12, pages 269–287, Berlin, Heidelberg, 2012. Springer-Verlag. *Validity: Good/Questionable. If I use this article it will be important to analyze it for credibility. This article focuses on english language like BDD testing and the advantages of that. Can be linked to agile testing principles through customer developer interactions. Worried how well this paper will merge with others if it becomes a core paper.*

[10] D. Talby, O. Hazzan, Y. Dubinsky, and A. Keren. Agile software testing in a large-scale project. *IEEE Softw.*, 23(4):30–37, July 2006. *Validity: Good. This article is in many ways a toss up when it comes to usefulness. Its topic seems to be very relevant to my potential topic and the of a military installation would be heavily reliable. Unfortunately the age of the article is limiting. This paper I think will become a backup core in case other papers seem devoid of usefulness. For now though I think this article is going to benched due to its age.*