Volume 3, Issue 10, October 2013 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

Agile Method Software Development Estimation Biases

Michael Scott Brown, Henry Dirska, Michael Pelosi
Graduate School
University of Maryland University College, USA

Mir Mohammed Assadullah
Career Opts
USA

Abstract— This research shows a cost estimation bias toward the Agile development method. Much literature exists showing the true number of projects that exhibit cost savings from using the Agile method. Most studies show about 50% of projects see a cost savings when switching to Agile. This study asked software developers to estimate the number of hours that it would take to complete a software development task using both Agile and Waterfall methods. Results show that there is significant evidence to believe that software developers have a bias toward Agile. The results show that developers underestimate software tasks when it is known that the development method will be Agile.

Keywords— cost estimation; Agile method, software engineering

I. INTRODUCTION

Software cost estimation is an important part of any development effort. Without effective estimations it is difficult to deliver software. The inability to determine release dates, adequately assign resources, and calculate costs are just some of the issues that occur without proper estimations. These issues explain why every software development method has steps to estimate levels of effort. Prior research has investigated biases that can occur when estimating software effort [1]. These biases can cause problems with project planning. Understanding biases benefits software engineering by helping project managers to make them more realistic cost estimations. Making realistic estimations is reminiscent of the old theory that you should ask a developer how long a task should take and then double it. In some cases you should triple it. A number of new software development methodologies that have emerged have provided better ways to build software over the traditional Waterfall model. Agile, Scrum and Extreme Programming, for example, have become very popular lately. But these methods create a new opportunity for software estimation biases. This short research paper examines the effect of using the Agile software development method on cost estimations.

This research makes no claims to the actual cost benefits of Agile methods. We only examine the impact of expecting to use Agile has on cost estimations. This is accomplished by asking software engineers to estimate the cost for using both Agile and Waterfall methodologies to develop software for a fictional set of requirements.

II. LITERATURE REVIEW

A. Agile Development

The Agile software development method has become very popular in recent years. It addresses many of the shortfalls of the traditional Waterfall model [2]. A number of studies have been conducted to examine the effects of Agile methods on software cost and productivity.

TABLE I
COST IMPROVEMENTS OF AGILE

Study	Year	Cost
Shine Technologies [3-5]	2003	49%
Dr. Dobb's Journal [6-7]	2006	60% (increase productivity)
VersionOne [8-9]	2008	38%
VersionOne [10]	2011	49%
Ali [9]	2012	52%

B. Agile Effect on Cost

Before we can determine if a bias exists for Agile software development estimations, we need to have an understanding of the actual effect that Agile development has on cost. A number of studies have been done over the last decade that show that using Agile methodologies improves cost. While some of these studies were not published in peer review journals, these journals often cite them. Table I outlines the percentage of Agile projects within these studies that saw a significant

improvement in cost. For this research we have concluded that about 50% of projects see cost improvements due to using Agile methods. Later in this research we use this figure to compare to cost estimations from software developers.

C. Software Estimation

A study by Jorgensen and Grimstad [1] hired 374 software developers to conduct research on biases in software estimation. In part of the study developers were presented with a fictional set of requirements and asked how long it would take to complete the work. However, there was a slight difference in the descriptions of the requirements. One set of engineers was told that the changes were a minor extension and a second set was told that the changes were new functionality. Other than this minor word choice the requirements were identical. The set of requirements described a Norwegian football, soccer, club that already had an existing software ticketing system. The football club wanted to extend to their partners the ability to sell tickets and then report on how many tickets each partner sold. The requirements included some technical points, such as the requirement that the system was to be written in Java. Developers were also told that the new feature had to be a webservice.

The results of this research show that subtle changes in the language of requirements can create differences in estimations. Developers that were told that these changes were minor had an average estimation of 85 hours. Developers that were told that the changes were new functionality had an average estimation of 107 hours. Jorgensen and Grimstad [1] show that this difference is statistically significant.

III. METHODOLOGY

This research uses the experimental research methodology. A set of software developers was asked to estimate a set of requirements. The estimation was for how long a set of average software engineers would take to design, develop and test the changes to a fictional existing system. Developers where actually asked to compute two estimations. One estimation was based on the assumption that the development method would be Agile. A second estimation assumed that the development was done using the Waterfall model. A survey was provided to capture their responses along with some categorical information. The set of requirements used for this research matched the Jorgensen and Grimstad [1] study. Other than a few minor changes the first paragraph was identical to Jorgensen and Grimstad. However, since Jorgensen and Grimstad did not provide the details of the requirements that they put in their study we created our own that matched their high level requirements. Below is the set of requirements that the respondents were given.

A few years ago MOSS Fotballklubb (MFK)—a Norwegian football club—developed a new ticket booking system. The ticket system has been a great financial success and saved MFK a lot of manual work. The web-based ticket system is written in Java and runs on a J2EE platform. The management wants to offer their partners, sponsors etc., an opportunity to sell football match tickets on the partners own web pages. The partners who sell the most tickets will be rewarded by MFK with more free football match tickets. For this purpose, the following software needs to be developed: 1) a web service that the partners can use to book tickets, and 2) a reporting tool that shows how many tickets each partner has sold. Detailed requirements are listed below.

- *A webservice needs to be created to allow vendor systems to query available seats and sell available seats.*
- *Information about the person purchasing the tickets needs to be captured (via the webservice) and eTickets need to be sent via e-mail.*
- *All tickets sold by partners need to be tracked.*
- *A report needs to accept a date range and report the number of tickets each partner sold within the date range.*
- *A screen needs to be created to manage (add, update, delete) partner information.*
- *All database tables need to be created to store data for these requirements.*

The software development team is made up of average software developers. You can assume that these developers have knowledge of the technologies, methods and the source code that the system is created in.

Once the participants read the requirements they were asked to analyze them and determine cost estimates. Table II shows the questions that were asked of the participants.

We received 44 responses in total. Ten of them were removed from further processing due to various reasons such as having no academic or professional experience in programming, giving estimates that were exceedingly small (2-3 hours) or exceedingly large (4,000 hours), or being unresponsive and giving an estimate of zero hours.

TABLE II
SURVEY QUESTIONS

Question	Choices
How many years of academic programming experience do you have?	
How many years of professional programming experience do you have?	
What is your gender?	Male; Female
Do you currently work in the software engineering field?	Yes; No

Question	Choices
What size organization do you currently work for?	Small (<100 employees) Medium (100-1000 em.) Large (>1000 em.) Not working
What software development methods have you used for at least one year?	Waterfall; Agile; Scrum; Extreme Programming; Lean Software Development; Unified Process; Never created software
What types of organizational structures have you worked in?	Functional; Divisional; Matrixed; None
How many hours do you estimate these changes should take by the software development team if they used the Agile software development method?	
How many hours do you estimate these changes should take by the software development team if they used the Waterfall software development method?	

IV. RESULTS

Despite taking extreme cases out of the data set, there remained large variation in the estimates provided. Table III shows the statistics observed.

TABLE III
GENERAL RESULTS

Statistic	Agile (hours)	Waterfall (hours)
Maximum	2,400	2,600
Minimum	30	40
Mean	560.6	717.2
Standard Deviation	583.6	670.9
Median	380	509

This data was sliced multiple ways to provide the observations noted below.

A. Cost Reduction

Recent studies [9,10] have reported 49% to 52% projects seeing a reduction in costs when adopting Agile methodology. Seventy five percent of the 20 responses of this study, that indicated at least one year of experience in Agile as well as in Waterfall methodologies, estimated lesser time with Agile. The remaining estimates were either equal to or higher than Waterfall estimates. A derived variable taking the value of 1 whenever a response indicated less time for Agile than Waterfall, and a 0 for all else was created. A t-test was performed on data for this variable with mean of 75% compared to the average of 50.05%. The p-value using a two-tail Student's t-test is 0.023. This indicates that 75% of the programmers, who have experience in Agile as well as Waterfall methodologies, tend to believe Agile is going to take lesser time (and thus lesser cost) to develop when compared to Waterfall. In reality, as reported by [9,10] only about half the projects experienced lesser costs. This shows that there is a statistically significant difference between the expected value and the data collected in this research.

When we removed the qualification of at least one year of experience in both methodologies 71% of programmers believed it would take less time by employing Agile methodologies. When this bigger population was compared with the average of 50.05% the p-value using a two-tail Student's t-test was found to be 0.016. This larger group also shows a statistically significant difference between the expected value and the data collected. This indicates that there is some bias in these cost estimations.

B. Agile vs. Waterfall

There were 20 responses indicating at least one year of experience in Agile as well as in Waterfall methodologies. In order to compensate for individual understanding of the requirements and to focus on the difference in estimation due to the choice of methodology, a ratio of Agile estimate over Waterfall estimate was computed. Twenty five percent of the respondents estimated Agile to take more or the same time in comparison to Waterfall. Whereas the remaining 75% of the respondents estimated Agile taking less time in comparison to Waterfall. Considering 100% to mean that both methodologies would take the same time; less than 100% to mean that Agile would take lesser time, and more than 100% to mean Waterfall would take lesser time, the average was 80% with the standard deviation of 34% and the median 75%.

The 13 respondents that did not have at least a year of development experience in Agile and Waterfall methodologies, also reported similar results. The average for this population was also 80%, the standard deviation was lesser, 25%, and median of 80%. The p-value of a two-tail, two-sample unequal variance (heteroscedastic), Student's t-test was found to be 0.98.

C. Gender

The Jorgensen and Grimstad [1] study also showed that estimations can differ based upon gender. For this reasons gender was asked of respondents. Five of the 34 respondents (15%) were female. The following table shows the percentage of estimates in relation to the mean estimate of the entire survey population. The last row shows the p-value of a two-tail, two-sample unequal variance (heteroscedastic) Student's t-test comparing the female and the male populations. Table IV shows the results by gender.

TABLE IV
RESULTS BY GENDER

	Statistic	**Agile (estimate / mean)**	**Waterfall (estimate / mean)**
Female	Mean	149%	141%
	Standard Deviation	84%	78%
	Median	134%	139%
Male	Mean	91%	93%
	Standard Deviation	105%	94%
	Median	57%	70%
p-value associated with Student's t-test		0.26	0.30

TABLE V
RESULTS BY ORGANIZATION SIZE

	Statistic	**Agile (estimate / mean)**	**Waterfall (estimate / mean)**
Large	Mean	102%	98%
	Standard Deviation	114%	103%
	Median	48%	56%
Small or Medium	Mean	98%	102%
	Standard Deviation	91%	82%
	Median	81%	86%
p-value associated with Student's t-test		0.92	0.89

D. Organization Size

Eighteen of the 34 respondents worked for "Large" (1,000+ employees) organizations. The remaining worked for "Small" or "Medium" sized organizations. Table V shows the percentage of estimates in relation to the mean estimate of the entire survey population. The last row shows the p-value of a two-tail, two-sample unequal variance (heteroscedastic), Student's t-test comparing the programmers who worked for "Large" versus others ("Small" and "Medium") populations.

V. CONCLUSIONS

A. Research Findings

There are a number of conclusions that can be drawn from this research. In some cases data may indicate a finding, but the finding may not be statistically significant. In these cases we propose them as future research. Programmers with at least one year of experience in Agile as well as Waterfall methodologies are more optimistic about Agile costing less time than Waterfall. About 75% of them believe it would take less time, thus less cost, compared to Waterfall, compared to about 50% actual projects that cost less [9,10]. About 71% of programmers in general, with some programming experience but not necessarily experience in both Agile and Waterfall environments, believe it would take less time, and thus lesser cost, with Agile in comparison with Waterfall methodology. Where a significant majority believes following Agile methodology would deliver quicker results, there are a number of programmers that take exception to this belief. Programmers have varied opinions about time estimates by using Agile over Waterfall methodologies as evidenced by a rather large standard deviation. Women estimate the development time to be more than men; 49% more for Agile and 41% more for Waterfall. Other research [1] confirms that women generally estimate higher than men and normally their estimates are more accurate. At the same time, women seem to be more resolute about their estimates as evidenced by the

lower standard deviation as compared to the one for the male population. This is observed in estimates for Agile as well as Waterfall methodologies. The higher than 0.05 p-values indicate the results to be inconclusive and point the researchers to investigate more in this direction. Organization size did not matter much in estimations. Programmers working for larger organizations estimated about the same in comparison with programmers working for smaller organizations.

B. Future Research

There are a number of areas of future research that can be conducted due to this study. Future research should be conducted to verify these findings. This should include a study in which developers are asked to estimate either Agile or Waterfall, but not both. Making them separate groups could further explain this bias. A number of other classification questions were asked of participants. Gender, organization size and organizational structures were all recorded. While these findings are interesting we cannot make any conclusions based upon our sample size. Future research could increase the sample size and produce new findings. Other classification categories could be added. Another way to examine this problem is to conduct similar research in which the developers actually implement the requirements. Some could implement the requirements using Agile and others using Waterfall. This would allow for comparison between estimation and actual costs. This type of study could quantify the amount of bias, which could be used to adjust cost estimations for Agile projects.

Another interesting observation was the range of estimates for these changes. They ranged from 30 hours to 2,400 hours. This seems like a very large range that we cannot explain. One thought was that participants were considering organizational factors. In some organizations it is more time consuming to get tasks accomplished. Hypotheses could be developed and tested to try to explain this difference. All of these areas are future research.

C. Final Comments

This research shows that software developers have a bias toward Agile development methods when they perform cost estimations. We do not know how extensive the bias is or methods to adjust estimations. We suspect that gender, organizational size, organizational type and methodology experience influence this bias, but our results are inconclusive on that. Our findings can begin to assist organizations in developing quality cost estimations.

REFERENCES

[1] M. Jorgensen & S. Grimstad, "Software Development Estimation Biases: The Role of Interdependence" IEEE Transactions on Software Engineering, vol. 38(3), pp 677-693, 2012.

[2] Agile Manifesto, "The Agile Manifesto" [Online]. Available: www.Agilemanifesto.org, 2013.

[3] Shine Technologies, "Agile Methodologies Survey Results" [Online]. Available: http://www.shinetech.com/attachments/104_ShineTechAgile Survey2003-01-17.pdf, 2003.

[4] L. Vijayasarathy & D. Turk, "Agile Software Developmnet: A Survey of Early Adopters" Journal of Information Technology Management, vol. XIX, no. 2, pp. 1-8, 2008.

[5] V. Vinekar, C. Slinkman & S. Nerur, "Can Agile and Traditional Systems Development Approaches Coexist? An Ambidextrous View", Information Systems Management, vol. 23(3), pp. 31-42, 2006.

[6] S. Ambler, "Survey Says: Agile Works in Practice" Dr. Dobb's Journal, August 3, 2006.

[7] D. Parsons, H. Ryu & R. Lal, "The Impact of Methods and Techniques on Outcomes from Agile Software Development Projects" International Federation for Information Processing, vol. 235, pp. 235-249, 2007.

[8] VersionOne, "3rd Annual Survey: 2008 The State of Agile Development" [Online] Available: http://www.versionone.com/pdf/ 3rdAnnualStateOfAgile_FullDataReport.pdf, June-July, 2008.

[9] M. A. Ali, "Survey on the State of Agile Practices Implementation in Pakistan" International Journal of Information and Communication Technology Research, vol. 2, no. 4, May 2012.

[10] VersionOne, "6th Annual Survey: 2008 The State of Agile Development" [Online] Available: http://www.versionone.com/pdf/ 2011_State_of_Agile_Development_Survey_Results.pdf, 2011.

