VERSIONONE
Simplifying Software Delivery

Free Resources

Did you know that you could try the world's most popular agile tool for free? VersionOne provides the industry leading tool for teams of any size. Visit the site for more information at:

http://www.versionone.com

Credits:

- **The Cost of a Bug**

  Christian Hargraves

- **Free Resources**

  V. Lee Henson CST

  http://www.versionone.com

Agile Dad

14643 South River Chase Road
Herriman, UT 84096
http://agiledad.blogspot.com

August 2009

Agile Mentor
Agile World

Agile Mentor

About the Author:

CHRISTIAN HARGRAVES

Christian has been an Agile activist since 2001. He presented at Agile 2007, Agile 2008, AgileRoots and to several smaller audiences. Christian has been a consultant and an active mentor in Agile, TDD, general unit testing and automated testing for the past several years.

Christian has been recognized by his peers as an authoritative voice in the realm of Test Driven Development (TDD) and has been invited as a guest contributor to the Agile Mentor Newsletter and Agile Dad Blog.

Tell us what you think?
lee@agiledad.com

this issue

The Cost of a Bug **P.1**
The Cost of a Bug **P.2**
Eye On It **P.3**
Resources & Credits **P.4**

The Cost of a Bug

Three things lead to unhappy customers: late delivery, cost, and bugs. While each of these obstacles has their own dynamics, they are interrelated. For example, late delivery usually means it costs more and fixing bugs generally means more time and money. I would argue that a lot of late projects are late due to bugs. To clarify on this, most of the late projects I have been on were late due to unexpected bugs. And most projects that are over budget are also late. In other words, if bug count can be reduced, it would have a drastic affect on both the cost and the delivery date.

Over the last several years, managers, directors, and executives have asked me to present empirical evidence that Agile Methodologies will cost them less money and take less time. This is a topic that can be discussed for months on end resulting only in opinions or experiences, but no empirical evidence. There are several experience reports out there, but, in the software industry there are many variables that can influence the success or failure of a project and it's not very likely that every organization is in the same situation. How can bugs, cost, and time savings be measured on a new project and do they mean the same thing across different organizations? All of the organizations I have worked at that follow some sort of Agile Methodology have decided to do so because of the hype or because someone read something positive about it.

In other words, in my experience, organizations seem to take the leap based mostly on faith. However, there are organizations out there that insist on hard-facts.

A recent paper published by Microsoft and IBM, showed that practicing Test-Driven Development (TDD) versus general unit testing reduced bug density by 40-90%. They reduced the number of variables in the project by keeping with the same code base, and team. The only thing they describe as being different is practicing TDD over just unit testing.

I presented these findings to our management team and got a question from a couple of them about how this can be correlated to empirical evidence. I guess I somewhat expected this. After all, the most important things to managers are time and money and I needed to present this data to them in either time or money and not in bugs. It seems as though people see bugs as something that are simply part of a software project. They see bugs equal to the time spent eating lunch or checking e-mail. How can bugs be equated to time and money?

Searching 4 Guest Writers

Do you have a topic you are passionate about that you would like to share with the AgileDad subscriber base? Would you like to be officially published in the Agile Mentor Newsletter?

Here is your chance! Select an Agile related topic and a brief snippet of what your thoughts are to lee@agiledad.com via email and we will get back to you with some date related deadlines. You make the commitment, you get published! It really is that easy.

This newsletter reaches loads of people all over the world who LOVE all of the same Agile topics as you do. Here is your chance to get the word out.

Be a leader in your Agile community! Get published in this newsletter. Send your topics NOW!

Cont: The Cost of a Bug

"Bugs Cost a Lot of Time and Money and we should do everything we can to find them as early in the process as possible"

– Christian Hargraves

It seems obvious to me. Bugs take time and time costs money. So I decided to present an actual project in terms of bugs and time. I sent out an e-mail to our QA department and asked them what tasks were involved when creating and tracking a bug and around how much time was spent on each task.

I got back a pretty sizable list with different time values for each task. I then compiled the list by grouping similar tasks into single tasks and taking the lower estimates for argument sake. QA also delivered several different bug workflows; four of which are listed below:

Workflow 1 - Bug is valid and gets fixed (60% of the time)

Task	Time Taken in Minutes
Find steps to reproduce bugs	20
Create a bug in the issue tracker	10
Get bug assigned to developer	5 x 4 people in meeting = 20
Developer and QA meet to discuss bug	10 x 2 people = 20
Developer fixes bug (commit & deploy code)	30 minutes
bug gets verified & closed	20
Total	**120 Minutes**

Workflow 2 - Bug is valid, but does not get fixed (30% of the time)

Task	Time Taken in Minutes
Find steps to reproduce bugs	20
Create a bug in the issue tracker	10
bug is discussed an decided not to fix	10 x 4 people in meeting = 40
Total	**70 Minutes**

Workflow 3 - Bug is found, but cannot be reproduced (5% of the time)

Task	Time Taken in Minutes
Find steps to reproduce bugs	20
Create a bug in the issue tracker	10
Get bug assigned to developer	5 x 4 people in meeting = 20
Developer and QA meet to discuss bug	10 x 2 people = 20
Developer fixes bug (commit & deploy code)	30 minutes
Developer tries, but cannot repro	60
QA and dev meet again about no repro	10 x 2 people = 20
Total	**170 Minutes**

Workflow 4 - Defect is sent back for rework (5% of the time)

Task	Time Taken in Minutes
Find steps to reproduce bugs	20
Create a bug in the issue tracker	10
Get bug assigned to developer	5 x 4 people in meeting = 20
Developer and QA meet to discuss bug	10 x 2 people = 20
Developer fixes bug (commit & deploy code)	30 minutes
Tester regresses defect, but not fixed	30
Tester updates issue in defect tracker	5
Developer and QA meet to discuss bug	10 x 2 people = 20
Developer fixes bug (commit & deploy code)	30 minutes
Defect gets verified & closed	30
Total	**245 Minutes**

The problem with these numbers is that they aren't extremely accurate and they don't cover all scenarios. For example, what if a customer found the bug? Also, the scenarios listed and the percentages of times they occur in our organization probably aren't the same as they are for other organizations. The times for each task are likely different for each organization as well. The important thing about these numbers is that they came from our within our organization. So what do the above scenarios tell us? Well, they tell us that it costs at least 70 minutes just to report and track a single bug.

I then proceeded to find an actual project in our department. This particular product had about a four year history and had at least some unit tests against it. However, the product was definitely not developed with TDD. I found that the product had a little over 3100 bugs filed against it since its inception. The rest is a simple numbers game. Take the total number of bugs, multiply that by the percent of times the given scenario happens, multiply that by the number of minutes spent on each bug (Total Bugs * Percentage * Minutes per Bug = Total Hours) and you have about how much time was spent on those bugs. So I did the following:

% Time	Bugs	Minutes	Total Hours
60%	1860	120	3,720
30%	930	70	1,085
5%	155	170	439
5%	155	245	633
		Total Hours	**5,427**
		Total Effort Days	**678**

That basically means we spent almost two effort years tracking and fixing bugs on a four year project. That seems about right considering there were about 12 effort years spent on development and 6 effort years spent on testing. That means about 11% of the total project on tracking and fixing bugs. Actually, if anything it seems a bit low which is what I wanted.

The research paper states that 40%-90% of the bug density was reduced simply by doing TDD. Let's say we could have reduced our bug count by 50%. That means, we can simply half the total hours and days down to 2714 hours and 337 days respectively. Wow! 337 effort days!? That's a lot of time.

What if the project had as much success as reducing the bug density down by 90%? That's a savings of over 600 effort days simply by doing TDD.

Now I had the empirical evidence management was looking for. Bugs cost a lot of time and money and we should do everything we can do to find them as early in the process as possible.

I now have managers calling me and asking me to help train their teams on TDD! For once, I am no longer the only person in the organization pushing for TDD.

EYE ON IT
Upcoming AgileDad Engagements:

August 6th Agile Roundtable SLC

August 3rd – 7th VersionOne Quarterly Service Sessions

August 24th – 28th Agile 2009
Chicago, IL
Http://agile2009.com

AgileDad
Private Course Offerings:

Private course offerings are available including:

Agile For Executives
Agile Boot Camp
Agile Business Requirements
Agile Meeting Facilitation
ScrumMaster Certification
Scrum Product Owner Certification
Managing & Grooming The Product Backlog

On-Site Agile Coaching and Consulting are also available.

For more information or to book a course today, please contact:

Lee Henson – 801-898-0295
lee@agiledad.com

All courses are offered through VersionOne and participating partners.