Agile Development and Testing by Analyzing Extreme Programming Variances

Anam Mustaqeem, Jahanzeb Yasin, Ali Javed, and Muhammad Nadeem Majeed

***Abstract*—The Agile Software Development is an iterative framework for development and delivery of software. Agile is based on ideas of using small teams in increments; delivering quality software. Compared to traditional linear and sequential methods, the testing strategy in agile is quite different. The concept of testing at the end of development phase is no longer applicable in agile. The pace of testing is much slower than that of development. So in this paper main focus of interest would be techniques and approaches for making the testing agile. The concepts of pair programming and ping pong approach; the types of extreme programming; are merged to find a solution for agile testing. The key challenge could be lack of documentation, but a cohesive team approach would result in greater understanding of the system.**

***Index Terms*—Extreme programming (XP), SDLC, pair programming, ping pong approach, code reviews, tested code.**

I. Introduction

Agile Software Development provides a new and enhanced approach for fast delivery of softwares; acquired within the given time and budget. In comparison with old traditional sequential development methods, agile methodology aimed at providing efficient and quality improved results while being predictive and light weighted. Agile methods aims at overcoming the difficulties arising due to changing customer requirements thereby giving rise to some testing and implementation challenges as well.

Today Agile software development method is being used by most of the organizations, as it provides increased project flexibility and productivity. We can say that in agile a project can start with any activity, and can change between activities at any time. Parallel work is done in agile which helps in fastest delivery of modules. [1]

II. Agile Testing

Agile Testing is quite different from traditional testing methods of sequential processes. Requirements and documentations are not necessary for test processes in agile. Testers are allowed to join the developers and users in their initial plan meetings. During the meeting tester himself notes down the requirements and then matches the developed codes to those requirements.[2]

Manuscript received February 11, 2012; revised April 14, 2012.

Authors are with the Department of Software Engineering University of Engineering and Technology Taxila, Pakistan (e-mail: jahanzeb_yasin@yahoo.co.uk).

III. Literature Review

Developing software within required time and budget is not the total work done; if that software is full of defects it is of no use to the customers. Now a days customer have increased demands of a bug free quality software. The software market is mature enough and user wants a complete quality based product.

Requirements can keep on changing during the software development life cycle and with the change in requirements bugs can be introduced at any stage of SDLC.[3] For locating and fixing bugs two different exceptions are described below

A. Bugs Found Within an Iteration

Whenever a bug is found in iteration under process, it is tracked using the following process

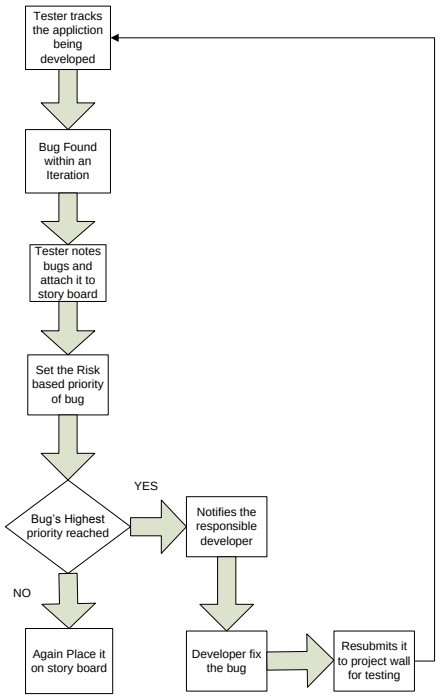


Fig. 1. Bugs found within iteration

B. Bugs Found After an Iteration

Whenever a bug is found on completion of an iteration, it is tracked using the process shown in figure2.

Marick has suggested a new model for Test Development .In addition to documentation; testers use other sources of information while designing tests. Information taken could be from running test cases. The tester is responsible for taking manageable action in response to changed documents or changed codes.[4]

Two imperatives affecting agile testing suggested by Brian Marick are

1) The working software will always be different from the user requirements.
2) Agile methodology focuses on developing working software in quick time based on meetings with users on regular basis.. But the requirements gathered in these meetings should be notified to the tester as well. Otherwise some changes may go unnoticed and thereby can create bugs in the module.

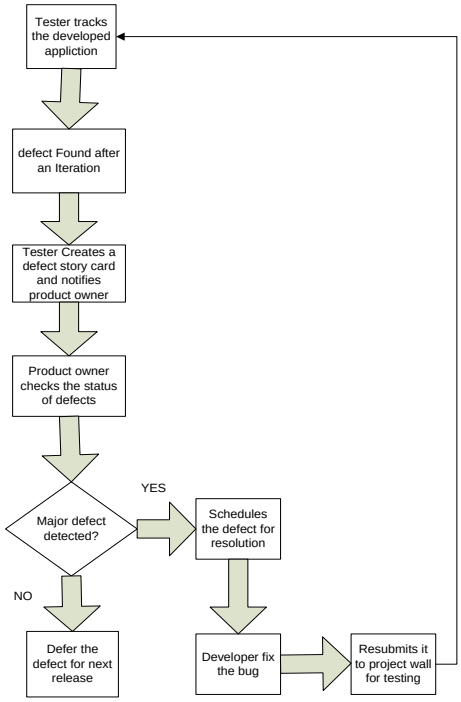


Fig. 2. Bugs found after an iteration

IV. Extreme Programming

Extreme programming (XP) is a well known agile practice. XP tends to manage project tasks, project manuals and documentation to reduce cost and support changing requirements.[5] XP is based on an iterative and incremental approach normally executed in small cycles. Xp brings whole team together thereby increasing productivity and creating a cohesive working environment. Idea behind teamclustering is to get enough feedbacks, so that team can highlight their weak points and tune them accordingly. This is why XP is normally said to be a people oriented process rather than process oriented [6].

Xp has its various dialects. Here focus of interest would be

1) Pair programming
2) Ping Pong Programming

A. Pair Programming

Pair programming is a concept used within XP, In pair programming, two people work in collaboration on the same algorithm, code or designing task. The two involved participants sit at one working computer side by side. One of the people is responsible for writing the code or designing the algorithm while the other person sits beside him and keep reviewing the coding. The first person who is coder is generally known as “Programmer” while the other one who reviews the code is known as “navigator” or “observer”. This practice improves the software development process. [7]

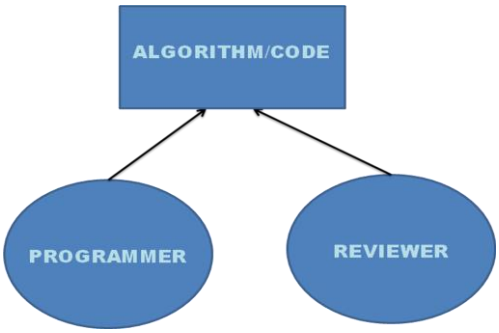


Fig. 3. Pair programming

1) Code reviews

In Code Reviews, Reviewers sit down by taking a dedicated time out to review someone’s code, everyone of these reviewers have some comments related to the code but not all of them will work on that code on daily bases, Everyone seems to be engaged and involved in discussion at that time but once review meeting is done not all of them will actually be a part of coding process, which eliminate the effective feedback loop.

2) How code reviews are related to pair programming

Code reviews are done to analyze the quality of code whereas pair programming is designed to produce a reviewed code.

In pair programming, Driver and Navigator are continuously writing and reviewing the code thus increasing feedback loop effectiveness, which eliminates the code review overhead in later stages.

Introducing pair programming and its variants in agile software development and testing increases the code qualities which ultimately decrease the chances of code review meetings in later testing phase.[8]

B. Ping Pong Programming

Both members of the pair write code. One of them writes a test code and the other writes the actual code(product code).

For example A writes a test code for product code which is written by B.A aims to produce a failing test code whereas B aims to pass that test code.

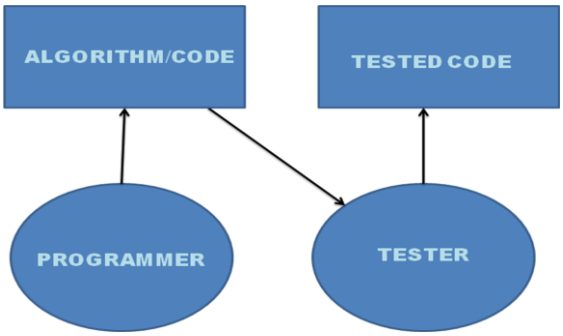


Fig. 4. Ping pong approach

1) Tested code

Ping pong approach is designed to produce a tested code. There is a difference between tested and reviewed code. A reviewed code is such a code which is clean and is according to reviewed code standards. Tested code is such a code which actually passes all necessary test cases. Any code which is tested code does not necessarily assure that the code is clean and reviewed.

Keeping in mind the facts that

1) There is always a room for improvement.
2) There are multiple ways to perform a certain task.

While considering first part of the fact, that there is always a room for improvements; any code which is not according to the coding standards will be difficult to handle when it comes to changing customer requirements or changing project functionalities. Any code which is according to coding standards and has no fat code is efficient but efficiently written and developed code does not actually means it is tested.

If second fact is applied in programming, each task can be implemented in more than one way and some of them are likely to be more efficient. No matter which alternative is adopted a tested code will pass test case but an efficient code will pass test cases more efficiently.

2) Analysis and findings

So far we have differentiated between tested and reviewed code. So to summarize we can conclude that tested code is that which passes possible test cases and reviewed code is such code which is according to the coding standards. Coding standards can be of organization or project wide which means each coding standards can be defined in a project and all codes are reviewed by keeping in mind these standards.

As reviewed code is delivered after the review meetings in static testing (dry run) and tested code is delivered after executing complete dynamic testing phase. In Agile development and Testing, reviewed code is obtained in pair programming and tested code comes out from ping pong programming.. As in Agile environment there are very tight deadlines and there is not much time for complete separate review meetings and dynamic testing so pair programming and ping pong approaches are used separately. Both of these approaches are not applicable on the same code, as it requires more time and waste of resources.

C. Proposed Solution

Actual solution exists in introducing these approaches in merged fashion i.e. combining both of Pair Programming and ping pong in one development cycle. This can produce the tested code as well as reviewed code.

V. Merged Approach

Unlike pair and Ping-Pong approach this merged approach requires three participants. Two of the participants are engaged in original Pair Programming and one remaining participant will be performing a task of ping pong which is of writing a test code.

Now three participants (Driver, Navigator and test coder) will work in a single team. Navigator reviews the overall code and keeps module wide eye on the driver's code. Test Coder will actually code the test code which will be used to test driver's code.

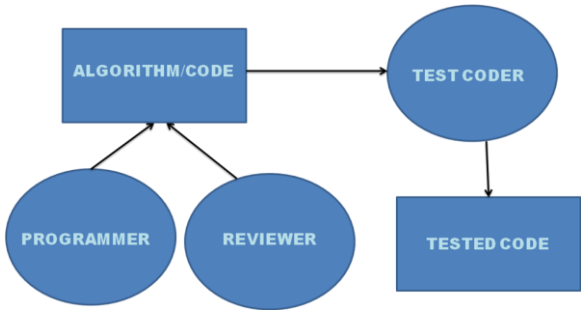


Fig. 5. Merged approach

Effectiveness of feedback loop is increased as tester reviewer and coder are engaged in a single team working on a same single task. Exchange of ideas occurs during this activity. All inputs and outputs are discussed with Test Coder.

In this merged approach Navigator's task is a bit more than that of in traditional Pair programming. Here Navigator also keeps in touch with the Test Coder and updates him with the possible test cases for which test coder will write the code.

Navigator works with both Test Coder and Driver discussing all possible scenarios and approaches, guiding the driver to produce efficient code also helping Test coder in producing a good test code with possible test cases.

A. Analysis

Engaging three participants in a team for close coupled development will produce a better result but with more people management efforts are required, more resources in terms of finance are required as there are three participants introduced for the task of two or one. Overall Development budget is increased but this will save the testing budget. When we talk about more than one person in a working environment it is impossible to deny the fact that people socialize more when they are in a group, so this can be an issue if the team members start socializing more and are drifted away from the actual goal.

B. Implementation of Merged Approach

To achieve this goal both of these approaches were merged and placed for testing. At the initial session total seven participants were called to take part.

RMCS (Remote monitoring and Control system) is a gsm (sms) based remote monitoring device having multiple sensors on it to gather data and send it back to its main server. RMCS has following on board sensors and toggling relays

1) Three temperature sensors termed as t1,t2 and t3
2) Three voltage meters as v1,v2 and v3
3) Three relays for remote toggling devices and switching as r1,r2 and r3
4) Nine Indicators for alarm situations as i1,i2…….i9

In total there are 18 attribute which are sent to server by RMCS after every hour via sms. String which is received by server is of the following form

CMI+:"REC UNREAD","923001234567","RMCS KIT 1","12:12:12 23:00" ,"12/12/12 23:00,r1=0,r2,=0,r3=0,v1=10.v2=20,v3=30.3,t1=100,t2=0,t3=40,i1=0,i2=0,i3=0,i4=0,i5=0,i6=0,i7=0,i8=0,i9=1"

Indicators and relays are both Boolean entities so they can have 0(off) and 1(true) as values.

Temperatures and voltages will have floating values.

1) Task

Develop a method which will return an array (String or Object array) having all of these values separated and parsed. This can be stored into database.

2) Constraints applied

1) If any of the attribute's value is null, empty or missing then default value of that attribute should be assigned to that value. (Default values for Boolean attribute 0 and 0.00 for other attributes)
2) If any of the attribute is unreadable then after reading all attributes it should be predicted which attribute is unreadable and it should be reconstructed by assigning default value.
3) End result should be off total 18 attributes

3) Coding guidelines for reviewer.

1) Each variable name should be meaningful
2) Indentation should not be more than two spaces
3) Comment the code where necessary
4) Keep the logic clean

C. Analysis and Findings

Assigned time to complete this task was about three hours but pair programming team managed to complete this task before the assigned time limit reached. As it was studied pair programming not only produces a reviewed code but also it increases the coder's efficiency of coding and finding a fast solution of the problem. After the pair programming TEAM C (Merged approach) completed the task and in the last TEAM B (Ping-pong) finished. However finishing task also depends on the developer's experience and knowledge of development tool. In this case Visual C# .net was used as development tool.

Code quality produced by Pair Programming and Merged approach was significantly higher than that of Ping-Pong but merged approach was also produced a testing code. Actual code produced by Merged approach passed the test code developed by ping pong but actual code produced in Ping-pong was not able to fully pass the test code produced by Merged approach. Both test codes developed in Merged approach and Ping-pong failed on pair programming's actual code.

TABLE I: SUCCESS RATE ESTIMATION

Team	Approach	Members	Finish time	Code Quality	Test Code Development	Test Passed
TEAM A	Pair Programming	Two	2 and Half hours	Good	No test code developed	No
TEAM B	Ping Pong	Two	More than 3 hours	Below average	Yes, One test code developed	Passed its own test code only
TEAM C	Merged Approach	Three	More than 3 hours but less than Ping Pong	Good	Yes, One test code developed	Yes, Passed both test codes with maximum pass results

1) Success rate estimation

Success rate is calculated on the basis of five factors as shown in TABLE1. Following are the five factors.

1) Members: fewer members in team means less resource cost. If a task is completed by fewer resources then it is cost effective. (Two members=maximum cost effectiveness, Three members=25% reduced cost effectiveness, Four members=50% reduced cost effectiveness).
2) Finish Time: less time to finish the task is time effectiveness. less time means high time effectiveness rate.
3) Code Quality: Quality of code measured by good, average and below average. (Good=Maximum, Average=25% less than maximum and below Average= 50% less than max).
4) Test Code Development: if test code is also developed along with the development of actual code than it has maximum test code development rate**.**
5) Test Passed: if the actual code developed in all of the three approaches passes maximum tests will have 100% test passed factor.(0 tests passed=0%, 1 test code passed=50% and 2 test codes passed = 100%)

Total success rate is to be 100% and each factor has equal weitage ie (100%/5 = 20%). So maximum for any factor is 20%. To calculate the success rates we have added all of the factors.

2) Success rate chart of all approaches

This chart is produced with reference to the actual findings chart.

TABLE II: SUCCESS RATE W. R. T ACTUAL FINDINGS

Team	Approach	Members	Finish time	Code Quality	Test Code Development	Test Passed	Success Rate
TEAM A	Pair Programming	Factor value=100%->20%	Factor value=100%->20%	Factor value=100%->20%	Factor value=0%->0%	Factor value=0%->0%	60%
TEAM B	Ping Pong	Factor value=100%->20%	Factor value=75%->15%	Factor value50%->10%	Factor value=100%->20%	Factor value=50%->10%	75%
TEAM C	Merged Approach	Factor value=75%->15%	Factor value=75%->15%	Factor value=100%->20%	Factor value=100%->20%	Factor value=100%->20%	90%

VI. CONCLUSION

If we take out the Tests Passed factor than the success rating would be as shown in Table III.

TABLE III: SUCCESS RATING FOR TESTS PASSED

Team	Approach	Members	Finish time	Code Quality	Test Code Development	Success Rate
TEAM A	Pair Programming	Factor value=100%->20%	Factor value=100%->20%	Factor value=100%->20%	Factor value=0%->0%	60%
TEAM B	Ping Pong	Factor value=100%->20%	Factor value=75%->15%	Factor value50%->10%	Factor value=100%->20%	65%
TEAM C	Merged Approach	Factor value=75%->15%	Factor value=75%->15%	Factor value=100%->20%	Factor value=100%->20%	70%

It is clear from the analysis of Table III that pair programming has a maximum factor values in code quality, finish time and number of team Members but Ping-Pong take

an edge in tested code factor which is minimum for pair programming. Merged approach takes lead because it combines the both pair programming and ping pong.

Hence to achieve maximum success rate it would be preferred to go for merged approach as it produces maximum results. Even if we increase the team members cost for three members to 50% we can still achieve 65% of overall success rate, which is 5% more than Pair Programming and equal to ping-pong but with one addition benefit of having good code quality.

VII. Future Work

In this complete research work we have limited and kept constant some factors related to team member's experiences and work knowledge. We have engaged all those professional who have been in the software development for not more than 2 years. More success rate can be achieved by acquiring a team which has better communication, more trained and experienced members. Developers with different experience and age can be combined in a team to analyze the overall success rate. Combining different level of experts for a single team can be cost effective as well as will achieve more success rate. If we introduced more experts and relatively low expert people in merged approach that can also effect the overall success rate because it can improve cost factor, believing on the fact that person with high expertise will demand for high salary and a person with less experience will charge relatively less.

Acknowledgment

We would like to express cordial thanks to our technical team who helped us in implementing Ping-Pong, paired and merged programming techniques. Our gratitude to honorable Engr Ali Javed and Muhammad Nadeem Majeed, for guiding us in our work.

References

[1] R. C. Martin, "Agile Software Development, Principles, Patterns, and Practices," Prentice Hall, October 2002.

[2] L. Crispin, "Agile Testing in Real Life – Looking Back at Ten+ Years of Agile," *ISSN 2191-1320*, July 2011

[3] V. E. Jyothi and K. N. Rao, "Effective Implementation of Agile Practices," *(IJACSA) International Journal of Advanced Computer Science and Applications*, vol. 2, no.3, March 2011

[4] B. Boehm and R. Turner, "Using Risk to BalanceAgile and Plan-Driven Methods," *IEEE Computer*, vol. 36, no. 6, pp. 57-66, June 2003.

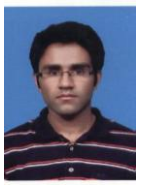
[5] M. Grant, "Introduction to extreme Programming", *ACM*, April 5, 2001.

[6] K. Beck, "Extreme Programming Explained: Embrace Change," *Addison Wesley Longman Professional, Reading, Mass*., 2004.

[7] A. Cockburn and L. Williams," The Costs and Benefits of Pair Programming, "*Software,IEEE Volume:17Issue:4*,Jul/Aug2011

[8] A. D. Radermacher, Gursimran S. Walia," Investigating the effective implementation of pair programming: an empirical investigation," *SIGCSE '11 Proceedings of the 42nd ACM technical symposium on Computer science education ACM New York, NY, USA* ,2011

Engr Anam Mustaqeem is MSc Scholar in Department of Software Engineering at University of Engineering and Technology Taxila. She completed her Bachelors degree in Software Engineering from University of Engineering and Technology Taxila in 2010. She is currently working on QoS support for multimedia transmission over vehicular adhoc networks (vanets). Her area of interest is Software Quality Assurance, Wireless Networks and Adhoc Networks.



Engr. Jahanzeb Yasin is MSc Scholar in Department of Software Engineering at University of Engineering and Technology Taxila. He completed his Bachelor's degree in Software Engineering from University of Engineering and Technology Taxila in 2010. He is also working as a Software Engineer at Agilitron Pvt Ltd.

Engr. Ali Javed is a PhD Scholar in Department of Computer Engineering at University of Engineering and Technology Taxila. He has been a lecturer since April 2008 in the Department of Software Engineering, University of Engineering and Technology Taxila, Pakistan. He accomplished his M.Sc in Computer Engineering from University of Engineering and Technology Taxila, Pakistan in February, 2010. His areas of interest are Video Summarization, Digital Image Processing, Computer vision, Software Quality Assurance, Software testing and Software Requirements Analysis.

Muhammad Nadeem Majeed is PhD Scholar in Department of Computer Engineering at University of Engineering and Technology Taxila. He holds a MS degree in Computer Engineering from Center for Advance Engineering, University of Engineering and Technology Taxila and has 11 years teaching experience. He is currently serving University of Engineering and Technology as Assistant Professor and working on Optimized Vertical handoff algorithms in vehicular Ad-hoc network. His research related to Risk management in IT industry of Pakistan which is aimed to aid different managers and team leads to manage the risk in their software development.