Enhancement in existing agile methodology to counter the issues and challenges in QA process

Ali Ashraf Khanani and M. Ejaz Tayyab
Department of Computing, SZABIST, Karachi, Pakistan
ali_khanani@hotmail.com and ejazshaikh@yhaoo.com

***Abstract*— The title of my research paper is "Enhancement in existing Agile Testing methodology to counter the issues and challenges in QA process". In this research study I have worked on the issues that are faced by QA teams during agile methodology processes and have identified ways of improvement in current agile frameworks that is being used.**

The solution to counter the challenges and issues during testing phase is proposed to improve the time constraint in a testing environment and making testing an easy and effective job in terms of cost and time constraints in an agile environment where deadlines are very strict and time is always short. At the end, there is a survey and interview with domain expert to analyze the results achieved and to prove the hypothesis of my research.

Keywords-Agile development; software; quality control; software quality assurance process.

I. INTRODUCTION

Current trend of organizations and industries has been moved or being rapidly moving towards agile methodology and development model. So first we should know what agile methodology is, it is a model for software engineering that is similar to SDLC models. This agile methodology de manifesto is to promote changes and is used for rapid development. As every process or model has their very own advantages and disadvantages similarly agile methodology also has its advantages and disadvantages [1]. Agile methodology is more focused towards development and promotes rapid development so new things comes around in a software that makes the software worthy and increases the amount of requirements making a software well established and costly for the clients which is an advantage for following methodology due to brace of change. The other advantages of agile methodology is the fast pace environment making the employees to learn new things rapidly making them efficient. Since agile embraces change it also has the ability to satisfy the customer by rapidly delivering the new releases promptly for which customer satisfaction is the most important [2-3]. Agile is an incremental model and works on sprints let's say e.g. if a sprint is of 3 weeks then new requirements are being delivered on those 3 weeks, similarly the n number of sprints is continued. This also gives the advantage of brining in late changes which can be adjusted in another sprint which makes the pattern quite efficient. Communication in agile methodology is on top that leads to cooperation between testers and developers, developers and requirement engineers and vice versa.

Since nothing is perfect similarly there are disadvantages and challenges for agile methodology that needs to worked upon. As the organizations trend towards agile methodology is flowing the measures need to be taken to minimize the challenges and issues with the agile methodology process. The issues and challenges for agile methodology will be discussed in details below.

There are few of the models in agile development methodology which can be used. We will below see what type of models exists in agile methodology. Following below are the model that exists in agile methodology.

- Scrum.
- Extreme Programming.

A. Scrum:

Scrum is the model in agile methodology that focuses on short meetings. The objective of scrum is to initiate a small meeting by a scrum master which is usually the project manager. There are two scrums held, one would ideally be at the start of the business hours in which scrum master initiates a session with the team to ask what would be done by each of the member. And similarly there is another scrum at the end of the day where scrum master takes the status of what was done and what is left behind. There could be various adjustments in timings of scrum which can vary organization to organization. The initial motive of scrum is to get updated on what items team is working, as agile preliminary works on people and resources rather than process and tools. [4-6]

B. Extreme Programming:

Extreme programming is another methodology used in agile development. In this method developers works in teams, while one is developing the other would be doing pair programming at the same way to see if anything is left from the eyes of the first developer – as two eyes are always better than one. [7-8]

Now, considering organizations that shifts towards agile methodology, my goal is to identify problems and issues during testing in an agile development environment and its impact to the testing team, and considering those problems my goal is to provide a solution. Similarly I being part of the production team see many issues that come on a fly in production environments that are not being catered in testing team which becomes crucial at times on the severity of the bug reported, therefore to make the testing process smooth within the agile methodology I will be doing my Independent study in this area to mitigate the risks and issues that comes in real time environment.

II. LITERATURE REVIEW

Agile methodology is the domain under which many researchers are working on streamlining the process but there still seems to be issues and loop holes that need to be covered. Most of the work in agile is done for process improvement in

development area. Most of the work is done is based on survey to the current trends in agile methodology. [9]

As we discussed in section 1 the advantages of agile testing methodology, now we will list the drawbacks and issues that are faced for testing teams in process for agile methodology. After reading numerous papers, meeting with organizations and rigorous literature survey and through my working experience I was able to find out the issues in real world that are faced to quality assurance process in agile methodology. Following below are the list of issues which will be briefly described.

- Time constraint
- Daily check-ins
- Setup daily test environments
- Rework of testing
- Integration of updates
- Short test executions
- Agile devalues documentation
- Excessive meetings
- A lot of manual testing
- Scrum adding too many people to the situation makes things worse

A. Time Constraint

Time constraint is a big challenging factor in agile development where delivering a project is very crucial in the fast paced environment. [1] The issue here for software quality assurance is the time constraint on which they need to get the project tested. Considering a sprint of 4 weeks this process contains development as well as testing, so it's a burden on testing team when everything gets fallen at the end of the 3rd week. This allows them with only a week to go with the whole testing of new requirements including the bug fixes. This is a very challenging issue for software quality assurance team due to which sometimes the deadlines are unable to meet or some of the requirements are left over to be tested which in terms increase more time or if the issue was not caught due to time crunch it could possibly break in production environments.

B. Daily Check-ins

As we know check-ins is the code that is added to the repository, this is also issue in agile software development where developers do rapid development due to which unstable keeps getting checked-in. The issue here for testing team is with this rigorous amount of check-ins causes hourly environment setups for the new check-in added. This is very hard for software quality assurance to test with code that is highly unstable where they get daily check-ins coming continuously.

C. Setup Daily test environments

First we need to know what we meant here by setup daily test environments. As we got to know above that developers checks-in code daily with lots of iterations which puts in quality assurance people to setup environment with that specific code that is checked-in most recently. So consider, even for a single line check-in by a developer makes to put effort of quality assurance personnel to very high, as he/she needs to get it deployed all over the test-beds multiple times whenever a new check-in is added. The stability of that code is also very important since, let's say for example there is a working environment where all testing tasks are performed, some new fix was added meanwhile by the developers and to cater that fix, testing team is to update the environment with that code. But due to some unforeseen events the new fix didn't work even it failed to start other component causing time loss for testers as all the testing is being stopped until the new fix comes in. These types of situations and delays could occur in agile methodology.

D. Rework of testing

Another issue in agile methodology is that there is a lot of rework of testing. In a sprint of 4 weeks we can have some new requirement changes and some bug fixes. To test only those changes could be pretty straight forward but the issue comes in when adding those new changes a tester needs to test its impact on other components as well, that's what causes rework of testing. If this doesn't get tested there could be a probability of that issue hitting in production. This issue comes up for all the sprints that everything needs to be tested before getting deployed it over to production environments.

E. Agile devalues documentation

Documentation is a very important piece of developing and creating software. As we see complex software contains huge and comprehensive documentations. But agile works differently as it devalues documentation and more focus on people rather than process and tools. This hits testing team as test cases are the major part of test any component. We don't have that comprehensive level of test cases or the new requirements are not rapidly documented. If there is a new requirement then there would be change in test case as well but due to agile devalues it there are not much of test cases that are updated which is another issue for a testing team.

F. Excessive Meeting

Meetings can be in every model whether it is Spiral, waterfall or RAD. But in agile there are excessive meetings, there is a daily scrum in agile [8]. Scrum means short meetings but usually when there are too many people it doesn't remain short and could take up to half hour, whereas, per agile manifesto the scrum time is in between of 15-20min. There are usually two scrums every single day, one in the morning which is used to prioritize what will be done today on the other way there at the day end which discuss what had been done. This can sum up a lot of a time if we calculate half an hour in meetings for each resource.

G. Manual Testing

In agile methodology there is a very little space for manual testing. The reason behind this is that we have very less time and too much to test. In agile, most of the things need to be automated which itself is a costly job for development to automate stuffs.

H. Scrum adding too many people to the situation makes things worse

Having too many people in a scrum makes things worse and more time consuming. The reason behind this is if we have people from development, testing and engineering and the tasks are to be divided in between development and engineering then there is no need for testing people in the

meeting [10]. Involvement of too many people would not solve the problem but would increase the complexity. This eventually sums up more time in meetings due to which it impacts the testing deadlines.

III. TYPICAL IMPLEMENTATION AND PROPOSED PROCESS

After a thorough literature survey and research, the knowledge was built for agile model and issues faced during software quality assurance process. Before coming up to the solution, to correctly analyze the processes was very important and is also covered in this literature survey. We have followed prototype model to proposed solution. The prototype contains enhanced version of agile model. First, we will discuss the four quadrant used in agile methodology. Testing in agile methodology covers four quadrants that are to be tested, we had already discussed the issues that testing team faces in software quality assurance. First quadrant contains the entire unit testing that needs to be done before checking in the code to the repository. This quadrant can be tested through automated tools, such as, Selenium and others. The first quadrant gives team the ladder to work on the second quadrant which is quadrant 2. The second quadrant consist of testing's such as functional testing, prototype testing and testing the UI. This step covers a lot of automated and manual testing that needs to be done by the testers. To get working on the functional testing of the user interface we need to make sure that the first quadrant is green, that is, unit testing and component testing is completed successfully. Once the second quadrant is completed, there comes third quadrant that potentially contains testing scenarios and usability of components. Whereas, an extra step has been added in the quadrant three to test the performance and load testing of the system to see if the system is scalable. In existing agile methodology performance and load testing belongs to quadrant four but as an enhancement to reduce issues, we have brought in quadrant four to more productive cycle. Quadrant four is used for executing whole set of tests. This includes the execution of artifactory run which will test all the binaries and code is in stable condition before we deploy it over to production since this is the last and final step that needs to be conducted. Once all the four quadrants are completed, we consider the features and bugs in a sprint are tested and ready to handover. Below is the figure that shows the four quadrants:

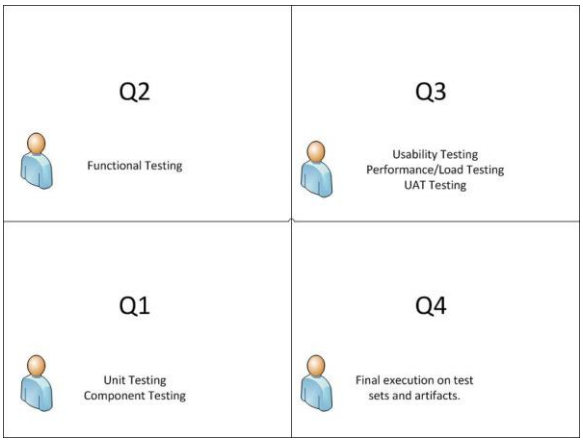

Fig. 1. Four-Quadrant representation

A. Test Approach and proposed strategy

Now, the solution that is proposed to mitigate the issues in software quality assurance is that we divide the testing of these four quadrants in the ownership of testing team and development team. Since agile manifesto says developers should think like testers and they should test as well. Therefore we have proposed a process such that the quadrant one where the entire unit tests and component testing is done should be done by the development team once the developer check-ins the code. The benefit of getting this done at the spot by the development team is if anything breaks in then it is immediately caught before even moving to the testing environment as developer get it fixed. This would save a lot of time for software quality assurance team and subsequently for the entire project. Furthermore, issues that could arouse in testing environments which can become showstopper because of bad code checked-in would be mitigated. Now, the issue of time constraint daily check-ins could considerably be minimized since the unit testing part is moved to the ownership of testing team and we could expect daily check-ins that are stable and won't hurt testing environments with showstopper issues. The approach for quadrant two will remain the same as functional testing and user interface testing would be done by the test engineers. But the benefit they would get is that they would directly start from quadrant two once the code gets checked-in since testers would expect that the checked-in code has already gone through various unit testing and component testing. Approach for third quadrant also remains the same as testers need to manually find out potential bugs from scenario testing and user acceptance testing. But to reduce the challenge of rework of testing and daily check-ins we have introduced a concept of pods where there will be multiple environments as pods and each of them would have a separate version installed. Let's say, if we have ten pod environments so we could avail ten different versions running on ten different pod environments. This will facilitate the testing team to easily track the issue introduced in some release to see whether this is an environment issue or a code issue. Adding pods in the infrastructure will reduce the time spent in troubleshooting the issue to see if it's a bug in a system or if that's some environment related issue. Setup time will drastically reduce for similar check-ins. Once testing team is done with all the measures of testing they would perform performance and load testing on the software and test it before releasing it to quadrant 4. Once the testing team gives a go ahead for all test cases completed the next quadrant i.e. quadrant 4 starts which hold off the final execution or final run of aritifactory to validate all the binaries and jars are stable. This is another quadrant that is being handed over to the development team and now they owned it. If the execution of the run is green which means we are ready to tag the final release but if it fails at some point then it holds off until the execution is green.

Now, for the software quality assurance folks the ownerships are for quadrant 2 and quadrant 3 only whereas the quadrant 1 and quadrant 4 will be handled and owned by the development team.

B. Proposed Roles in Agile

Depending on what requirements and what bug fixes are the new set of roles are implied for the programmers and testers.

No matter what tasks are provided in a sprint the following tasks in figure 3 represents what type of tasks would be perform by what person.

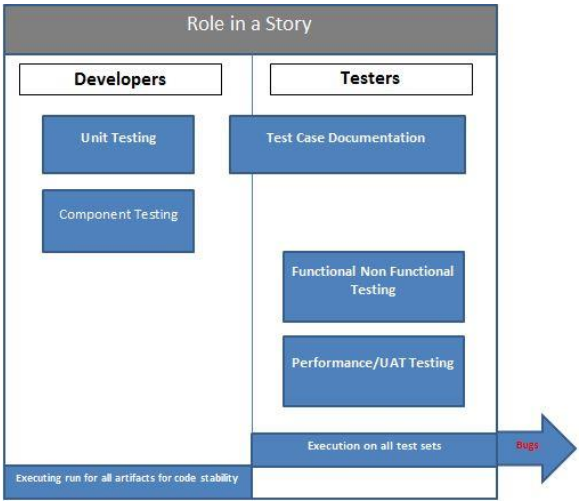

Fig. 2. Represents proposed role in Agile

C. Proposed Test process

The proposed test process contains a more tangible test process defining the flow of testing process. Figure 3.3 represents the flow of test process that is devised where what activities would be done by tester and what activities would be done by the developer is proposed. Below show is the Figure 3.3. briefly describing the proposed test process.

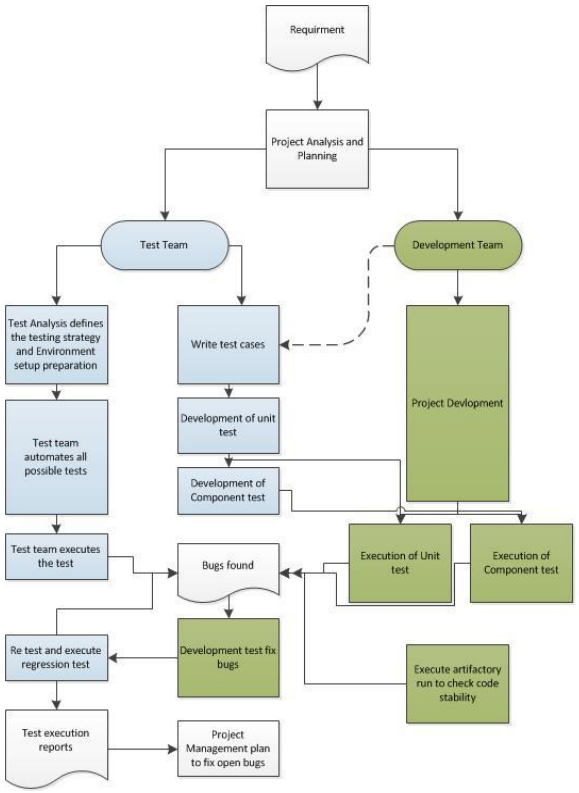

Fig. 3. Representing proposed test proces

For excessive meetings the proposed solution is to add only one representative of each team to be in the meeting instead of all the resources coming in for the meeting. Through this process we have reduced the time savings and efforts. This include minimizing the meeting time, to prove this let's consider the company has 150 resources and each of the employee spends 30 minutes on morning scrum and day end scrum which in total turn out to be 75 hours daily spend in meetings. As per our proposed plan for meetings the meeting time can be reduced to almost 75%. Say let's consider if the number teams in the company are 10 and each member representing the team would add 10 people in the meeting including some managers and project managers taking the scrum. If the total number of population goes up to 30 people attending the meeting daily this would reduce the hours from 75 to only 15 hours, which is a lot of time. The other remaining 60 hours could be consumed in some other tasks.

As per the proposed new roles in agile and the test process the consolidated Figure3.4 represents the proposed process of agile development is enhanced with following figure.

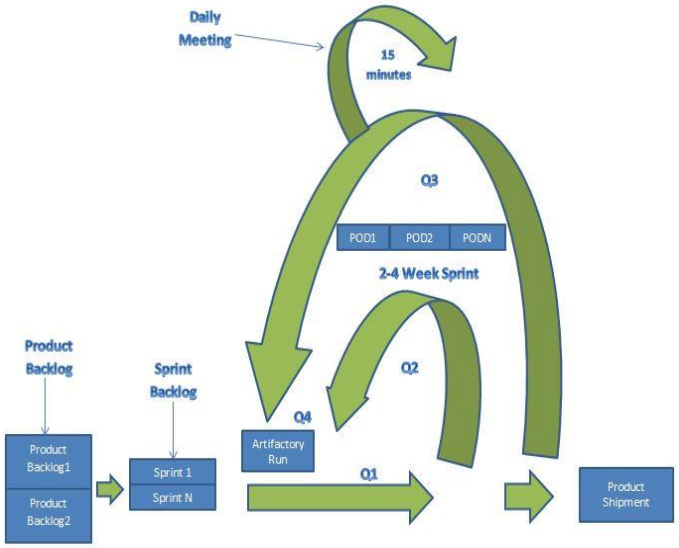

Fig. 4. Representing proposed test proces

IV. SURVEY

A survey was conducted to validate the proposed process. Following figures shows the survey results

A. General Questions

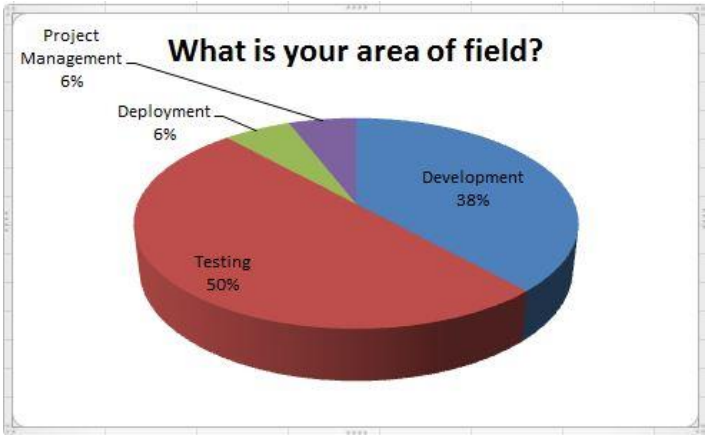

Fig. 5. Represents results for Area of field

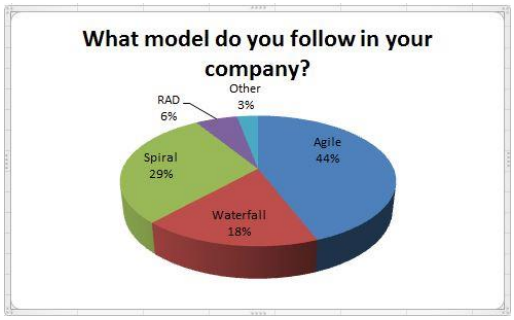

Fig. 6. Represents the model followed in company

B. Process Validation Question

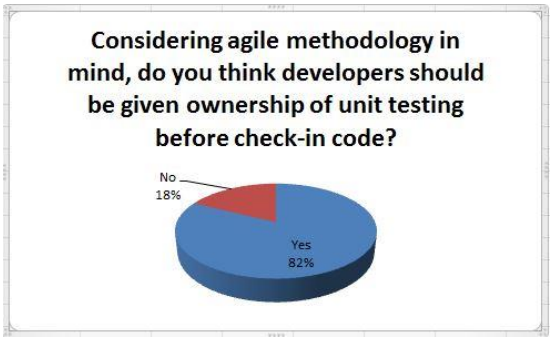

Fig. 7. Daily check-in issue result

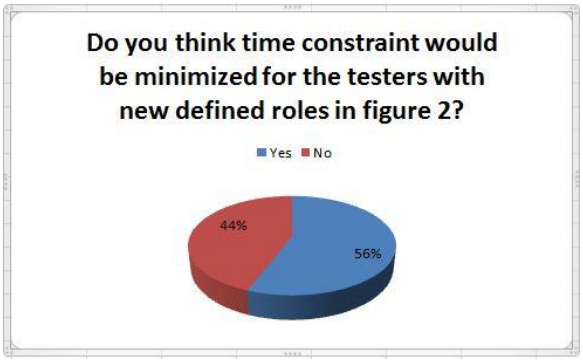

Fig. 8. Time constraint issue results

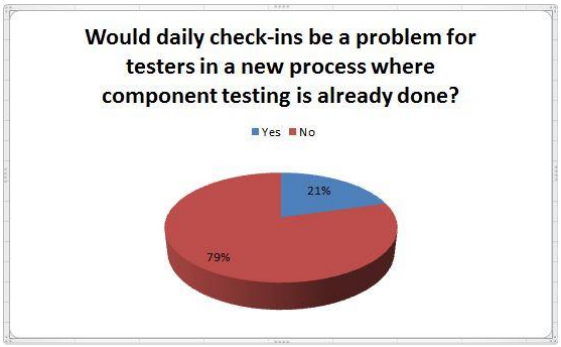

Fig. 9. Daily check-in issue result.

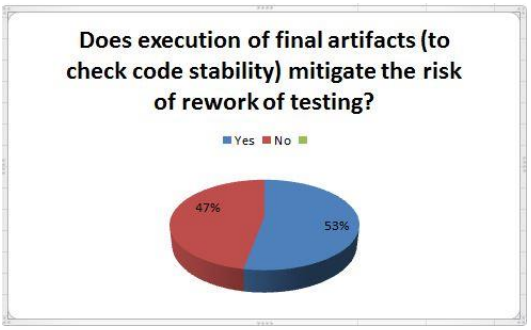

Fig. 10. Rework of testing issue results

Fig. 11. Excessive meeting issue results

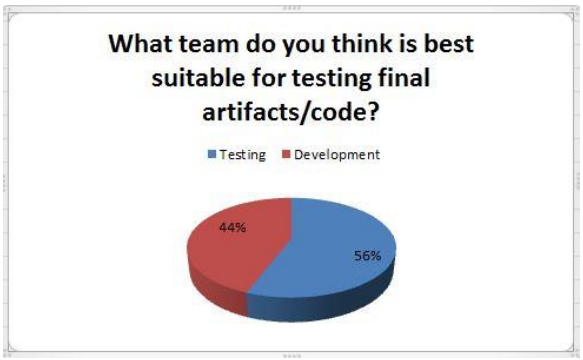

Fig. 12. Testing final artifacts survey

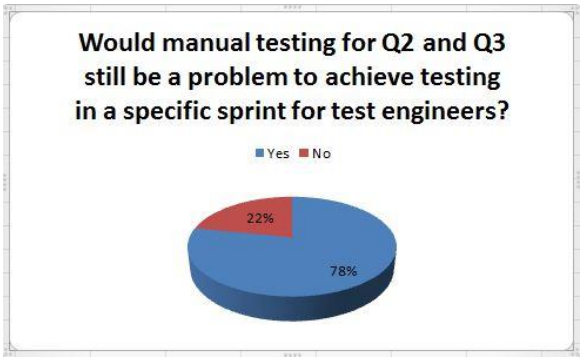

Fig. 13. Manual testing issue results

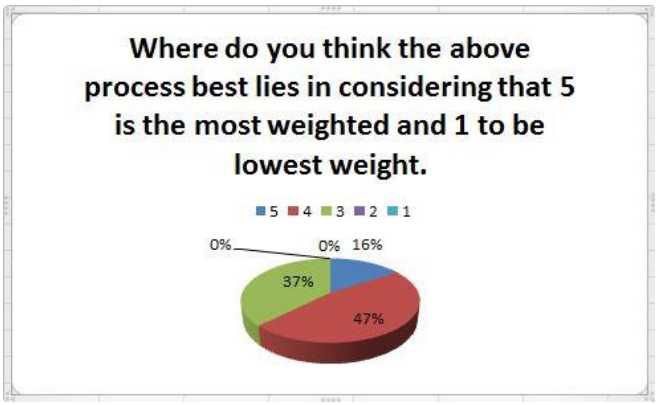

Fig. 14. Proposed Solution weight

C. Consolidated Result

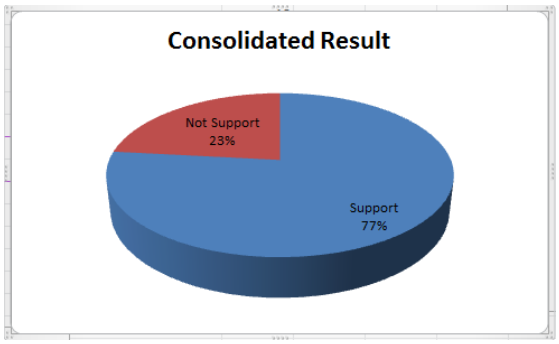

Fig. 15. Shows the percentage of people in support and not support with the proposed process

CONCLUSION

In this paper, we focused on the enhancement of agile testing methodology to fix and cater to the issues in software quality assurance process. We have identified numerous issues which testing teams face in real world environment, such as, time constraints, daily setup environment, rework of testing, missing documentations leading to mishaps in testing, excessive meetings leading to work not done in time and moving down the deadlines, manual intervention and manual testing in a short execution iteration that takes a lot of time. To cater to the issues, we proposed a process using four quadrants which were divided into four important testing parts including unit test, performance testing, functional and nonfunctional testing and overall running an artifactory run to see if the binaries and codes are stable. Those 4 quadrants are then strategically divided between development and testing teams. Q1 and Q4, which included unit testing and running artifacts, are to be in the ownership of the developers whereas Q2 and Q3, that includes functional, non-functional and performance & regression testing comes in the ownership of software quality assurance team. We then conducted survey and interview with domain expert to get the suggested improvements validated.

After preparing the results that we got through our survey and calculating the consolidated results we found that in the favor of our hypothesis with a healthy result of 3.78 out of the scale of 5. Though the population and sample size was not too big and was of 34 but considering the population consisted of targeted experts, with these results we can say that our hypothesis is true.

FUTURE WORK

The future work includes implementing the proposed process of agile model to a known software house. Live demo and training session would be provided to organizations for the validity of the new process and to implantation Validation technique used in this process was survey but the population size of the survey was to thirty four sample size due to time constraints. In future work, we would also do the survey for a large number of sample size to validate the results to know the outliers in the processes.

Survey population for this research is very limited numbers so the results generated may vary with a big population but we could get an intimate from the results we have got. We can conduct a survey with large amount of population in the future to see the probability of people that are in favor of the suggested improvements identified in the process.

ACKNOWLEDGEMENT

The author would like to thank Allah with the grace of God he was able to complete his research. He is also extremely thankful to Ejaz Tayyab for the support and knowledge sharing he provided during the research period.

REFERENCES

[1] Meneely, A, Williams, L, Osborne, J.A. Yonghee Shin, "Evaluating Complexity, Code Churn, and Developer Activity Metrics as Indicators of Software Vulnerabilities," *Software Engineering, IEEE Transactions on (Volume:37 , Issue: 6)* , pp. 772 - 787 , 2013.

[2] Seliya, N, Khoshgoftaar Yi Liu, "Evolutionary Optimization of Software Quality Modeling with Multiple Repositories," *Software Engineering, IEEE Transactions on (Volume:36 , Issue: 6)*, pp. 852 - 864, 2010.

[3] Claes Wohlin Kai Petersen, "The effect of moving from a plan-driven to an incremental software development approach with agile practices," *Empirical Software Engineering*, pp. 654-693, 2010.

[4] Orit Hazzan, Yael Dubinsky, Arie Keren David Talby, "Agile Software Testing in a Large-Scale Project," *IEEE Software*, pp. 30-37 , 2010.

[5] Diana Brown, *Agile User Experience Design: A Practitioner's Guide to Making It Work*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. San Francisco, CA, USA ©2013 , 2012.

[6] B. Henderson-Sellers A. Qumer, "A framework to support the evaluation, adoption and improvement of agile methods in practice," *Journal of Systems and Software*, 1899-1919 2013.

[7] Jeff Sutherland, "Introduction to Agile Software Development: Lean, Distributed, and Scalable," *2012 45th Hawaii*

International Conference on System Sciences, 2012.

[8] Claes Wohlin Kai Petersen, "The effect of moving from a plan-driven to an incremental software development approach with agile practices," *Empirical Software Engineering*, pp. 654-693 , 2011.

[9] Borje F. Karlsson, Andre M. Cavalcante, Igor B. Correia, Emanuel Silva Andreia M. dos Santos, "Testing in an agile product development environment: An industry experience report," *LATW '11 Proceedings of the 2011 12th Latin American Test Workshop, pp. 1-6, 2011*.

[10] Jussi Kasurinen, Ossi Taipale, Kari Smolander Vesa Kettunen, "A study on agility and testing processes in software organizations," ISSTA '10 Proceedings of the 19th international symposium on Software testing and analysis, pp. 231-240, 2010.