Volume-3, Issue-7, July-2014 • ISSN No 2277 - 8160

Research Paper | Engineering

Automation of Software Testing in Agile Development - An Approach and Challenges with Distributed Database Systems

P. Suresh Kumar Associate Professor, Department of CSE, Varadhaman College of Engineering, Hyderabad.

Dr. N. Samba Siva Rao Professor and Principal, Department of CSE, Sumathireddy Institute of Technology, Warangal.

ABSTRACT *Many Software concerns are adopting Agile development as a flexible way to develop new software applications. An important part of any software application is testing. Agile development have similar aims as traditional software testing, but the structure of the team is different, testers need to support quality infusion through the entire team. Test automation and selection of test automated testing tool can help project teams deliver more effectively, and in shorter timescales. Automation test cases are helpful to capture and store the test documents, test cases etc. by allowing them to regularly getting a satisfaction and saving some time for the organization. Agile development is all about working piece of software, individuals, close collaboration of customer and quick respond to change. The challenges in testing of Distributed Database Systems are visible in the tools for automatic test case execution. This paper addresses some of these challenges and also highlights every aspect of software testing process in Agile development.*

KEYWORDS : *Software Testing, Agile Development, Distributed Database Systems, Software Test Automation, Agile Testing.*

I. Introduction

The difference between an Agile Development and Waterfall model is shorter feedback loop. Automation of tests can help Agile Development to shorten the feedback loop. In traditional processes the automated software testing is replacing the manual test procedures. In the established way of development, the tests are normally written by the testers. Software Testers test the system and send the feedback without looking and taking into account the current changes and updates into the code. This type of testing involves testing the functionality of the whole application. Hence, the developer gets later and incomplete feedback from the software tester. Automation of Software Testing is a good thing, which can make the testing little easier. It can save money and make the process more effective. In Agile development the automation of testing is used to execute the unit testing, which is helpful to test the smallest pieces of the system. This thing can be done effectively and frequently. It can help the development team to execute the same tests again and again and test the system after each change in the code, and also helpful to make the feedback process short.

The role of test automation is really appreciable in Agile development, and it still has an important place in the world of Agile Development. Test Automation has several advantages like: repeatability, consistency, and better coverage by driving data. The test automation cannot eliminate all of the bugs in the system of its own, for the purpose manual testing is also required. During the development there will be some time when it becomes a regression testing, where the system is checked for errors and then send back to redo and so on.

There can be some issues regarding implementing test automation in Agile development. As the time period of Agile projects or iterations are less than one month and there is not a lot of planning involved in this process and most Agile projects are made on some loose design and it also developed on the basis of some system metaphors. This can create some issues, because test automation is a complex task and it requires some planning. It requires some time to develop these tests. But there is a solution or recommendation for this problem. Test automation should develop the automated testing metaphors in the meantime, when the project team is developing the system design. This will give them a design approach to test the application. As the teams do some planning, before each iteration, then there should be some extra time designated to the automated testing planning and their infrastructure development.

[5] Has shared his experiences while adopting Agile in their company. They faced different problem and challenges at the start. One of the biggest challenges was the software testing. According to him, selection of a proper automation tool is always a difficult task. Despite of the fact that they never wanted to use any manual testing, they have used a combination of both, manual testing and automated testing. The testing was never completely automated. If the automation of acceptance testing will be part of a sprint definition and the "done" criteria, the team must develop a test strategy and automation parallel or in advance of development. This can keep the feedback loop in sync with development the testing activities do not take any time. If the feedback is not in sync with development then it can create some confusion that either the task completed is "done" or not [5].

The philosophy of Agile development is to react to the changes in the requirements so quickly. The testers should also adopt the same philosophy while developing automated test scripts. Automated testers can quickly do the execution by using the external and function for allowing quick changes to use script for future usage. In the Agile development the teams have a close coordination among them, and they communicate with each other regularly. The automated testers should also communicate and interact with the teams to discuss the design principles and they also should decide the scripts to be integrated for regression testing. The regression testing should be there to refine the system.

II. Manual and Automated Testing in Industry

From getting their feedback against our question that how they are managing manual and automated testing in their development process?, we analyzed that most of our sample companies are using both manual testing and automated testing. Some of them are using automated tools for that and relying more on the automation for catching bugs during the sprints. Company A is using the Selenium for continuous integration. They also manually test the product before each official release. The customer teams also verify that the updates work on the customer's handsets before sending them new versions. They do have a regression testing before the final release to assure that no more errors are present in the system.

Company B is not using any automated tools and according to them, they are relying more on manual testing. They hand out prototype builds to a group of internal or external testers. As bugs occur, they report them; the developers try to fix most of them right away, and if there are some harder ones that are difficult to handle by that time, then they are put on the list for the next sprint. New feature requests from stakeholders are taken care of directly, or put on the list in the product backlog.

Company C is using automated and manual testing for quality assurance. They are using tools like JUnit, for automated testing. They use it while the developer is testing his/her own work before sending it to the Team Lead. The Team Lead then review his/her work and is sent to the testing department for more detailed manual testing.

Company D is practicing both automated and manual testing is done, load and stress testing is done using custom made tools specific to the product and also by using Mercury's LoadRunner. Functional testing is normally done manually.

Company E is using both manual and automated testing techniques. All tests, manual and automated, are documented and stored. The automated tests are stored in test suites and run both as regression tests and functional tests. But the emphasis is on automated tests.

From the above feedback we analyzed that most of the companies in the industry are utilizing both of testing ways like: manual and automated testing. The only difference is the use of automation tools. The sample companies were the well known and biggest companies in the industry. When asked about why to use both testing ways? They answered that they cannot rely on one kind of testing, because automated testing is not enough to get a quality product and if they do only manual testing then it can take a lot of time and effort. In the result of that, the project can go out of time and budget. So, that is why they use automated testing to test the application as soon as possible, but still there remain some complex problems that need to be addressed through manual testing. During the sprint the developer writes test cases to test his/her own code.

III. Good Practices in Automated Testing

There are different properties of automation tools to locate the control of the application. When the companies are preparing their automation tools and teams, there can be some good practices that can be helpful in order to perform effective automation. The managers should identify the strengths and weaknesses of their team in a good way, and should manage them in a way that they know that who is good at doing what. They should divide their efforts by assigning the work of the individuals who are better at performing those tasks then their fellow employees. For example, there are some people in the company who are good at script writing instead of test cases and vice versa. Then create an automated test plan. A plan is also worthy for starting any process and to decide that which test should be automated. Automation plan allows identifying the set of tests to automate initially, and also serves as a guide for future tests.

Try to identify your goals for the automation. The following can be some possible goals:

- Speeding up the testing process to allow quick releases
- To allow the testing of the application to occur more frequently
- Improving the test coverage
- To ensure the consistency in the application
- To improve the reliability of the testing process

In the planning, try to document the different types of tests or processes to be automated. It includes the following:

- Regression testing – These tests are run for every build
- Data-driven testing – They require different sets of data over many iterations of the test
- Configuration tests – These are run on many different platforms
- Backend/Non-GUI testing – These tests verify the data which is collected by the application

The selection of a proper tool for test automation is also really important in automation. When you are selecting a test automated tool, you need to check that either this tool supports the technology that is used in your application. Either the tool is technologically updated? It is really hard for companies to train its employees for a new tool and the cost of the training can be underestimated. So try to analyze the strengths and weaknesses of your team and see that if the tool is matched with your employees' skills. Check that if this tool requires users to be programmers or it allows testers at different levels?

Prime Technology Group is a big company working in the field of software development. They have a policy for test automation. Their automation approach looks quite interesting. According to them for each Prime Technology Group software product there exist an automated regression test and it at least regression test covers the high risk areas of production. The testers discuss test automation with developers. It is the duty of developers to build unit tests and testers review them or highlight any area that is left uncovered on the basis of risk and available capacity. Developer and tester needs to have a proper communication to get a better test automation approach. They are conducting manual testing before the every official release, to make sure that the program is working accordingly and the results of program are same as required. Also customer teams verify updates work on the customer's handsets before sending them new versions.

The automated tests should be reusable and maintainable. When you are writing the test cases, try to create small scripts rather than combining them to make complex tests, which are difficult to understand, update and debug and also very long. Try to make keeping them reusable, understandable, maintainable, and modular.

The script can make reusable by keeping it small and focused on a single task. For example, if you are testing an application and you write a script for testing a small functionality, then you also use that in the future for testing the same functionality in some other form. Using external variables and external data can be helpful in order to make scripting reusable. If the values for input fields are not hard coded in the script, you can use the same script to test different conditions.

If you maintain a document that contains the detailed descriptions of your script, then it can be helpful to make your script understandable. If you write the description, functionality and purpose of the script then it can help to understand that what types of scripts are available and how to combine them into larger tests. Using of comment can be really helpful in order to make the script understandable. The general rule for adding comments is to use them in variables, functions, subroutines, and branching structures just to give a clear picture that how the internal components work.

IV. Why Do We Need Separate Testing Team

In software development there are different roles, but in this paper, we focused on the role of testers. The testers play a key role in development; developers may lack some skills in development which testers provide. But in Agile development the common mistake is to see testers as junior developers. A good tester has many distinguishing features that make a difference to the developer. Only by understanding this difference, project managers can make a good and successful team.

According to [1], comparison of the characteristics of a good tester and a good developer are shown as below:

All above mentioned characteristics clearly differentiate the tester and developer's role. A good developer thinks theoretically, while a good tester think empirically. Developers take much more time to identify a problem than testers. So, the tester should not be judged according to developer's criteria. Good developers and testers need both thinking, but there are times when each discipline stresses different talents and calls for different emphases [5].

Table-1 Comparison of Good Tester and Good Developer

Good Tester	Good Developer
Get up to speed quickly	Thorough understanding
Domain knowledge	Knowledge of Product internals
Ignorance is important	Experience is important
Model user behavior	Model system design
Focus on what can go wrong	Focus on how it can work
Focus on severity of the problem	Focus of interest in problem
Empirical	Theoretical
What is observed	How it is designed
Comfortable with conflict	Avoid conflict
Report problem	Understand problem

In Agile development, the developers develop an application, and they conduct an integration and unit testing. Then the working is delivered to the customer for testing. Software testing requires a lot of specific skills and challenges, because it is such a creative and intellectually challenging task. A professional tester is required to do this professionally task, so that the task can be performed effectively and efficiently [4, 3]. In Agile methods, testing is usually done as a task that developers do as a part of the development task or as a customer's task. The customer is very closely involved in everyday development in some of the Agile methods and takes the responsibility for acceptance testing. This is a problem and can create some major problems too [2]. If your customer has expertise and skills which are necessary for a tester, and has the capabilities to act as a tester, only then you can think of assigning him the task to test the application. It means that, how someone can test an application, when he does not have any skills and knowledge about testing software? Dynamic Systems Development Method (DSDM) method has recognized the need for basic skills for a tester and [6] recommends that, at least there should be one developer or tester in each team, who has received training in testing [6].

Software testing not only finds the errors, but it also tells us that, at what level we have achieved the quality. Software testing provides information about defects and problems in an application and also evaluates the achieved quality. Metrics are required to evaluate product quality. For Example, the total number of finding faults, fault classification, and the test coverage. Agile methods contain testing activities and also their proper implementation during the development process. But, this does not give any information about the level of achieving the quality of the product, or evaluation of the product. For example, developers write the automated unit tests and always keep running and passing which is a good practice in order to achieve quality. It can be easily tracked by the amount of test cases or the amount of covert methods. But until then it does not give us any information about the achieved quality. For that purpose, we need some metrics like faults, test coverage, and quality of the tests [2].

On the basis of the discussion in above sections we think that Agile methods should be accompanied by a separate testing team, and they can get more advantages by introducing such team. [2] Also says that Agile development can be benefited through a team of professional testers.

V. Agile Testing and Testing Challenges

In the previous section we described some of the useful testing and quality assurance activities in Agile methods. It is a common misconception that Agile projects do not need a rigorous approach to testing. But if we compare Agile with other traditional approaches, then we will come to know that from a testing perspective Agile methods have lacked in different important aspects of software testing process[2]. In Agile development there is no specific rule defined for tester and often testers are treated as junior developers. A good tester has many distinguishing features that make a difference to developers. In [2] the authors say that Agile development can be benefited through a team of professional testers.

Agile Testing is a software testing practice that follows the principles of Agile software development. Agile Testing involves a cross-functional Agile team actively relying on the special expertise contributed by testers. This allows the combined team to better meet the project's defined business, software usability, quality, and timeline objectives.

Unit tests are the foundations of any Agile projects, effective unit test can be done with automation. The more the team can automate the testing, the faster they can move on to the next developments. But testing in an Agile way is not without its challenges, are left in utilizing the automated tools in some applications. Testing in a complex distributed environment like cloud testing poses significant challenges for a tester to perform unit testing with an automated test generation tool. Moving the application to the cloud will in some cases present some differences in how to implement a certain test or test case depending on the cloud environment. Often, a test case requires the system under test to have a certain internal state as a starting point of the test case. A big challenge for the tester is to achieve the required state prior to test case execution. Another problem with Agile methodologies is that testing is not done in a fixed pattern at the end of the development phase, but instead after each package or integration of packages. Hence a project based on Agile methodology, planned tests must be creative in order to allow adaptation to the changes as well as the schedule.

Conclusions

Examining the role of Automated Software Testing and identifying a set of issues when building and testing complex distributed database systems in a cloud, the focus of our paper, is the least researched area. Very few papers have been published that focus directly on software testing of cloud applications. The main result revealed by our primary study is that for developing and providing a system aimed for the cloud is effective with Agile development process. But it is critical that an Agile tester who is expected to test the quality and performance of cloud applications has a good understanding of what makes a Cloud Computing application and distributed architecture, as well as a good understanding of the tools available and their strengths and weakness for testing different types of cloud applications. To support our work an additional research goal was made to evaluate the applicability and support of various techniques and open source tools for advanced distributed database testing like in a cloud environment.

REFERENCES

[1] Bret Pettichord, Testers and Developers think differently, STQE Magazine, January 2000. | | [2] Juha Itkonen, Kristian Rautiainen, and Casper Lassenius, Towards Understanding Quality Assurance in Agile Software Development, International Conference on Agility Management (ICAM 2005), Helsinki, July 2005. | | [3] Kaner, C., Falk, J. L., and Nguyen, H. Q., Testing Computer Software, Second Edition, Pages: 480, John Wiley & Sons, Year of Publication: 1999, ISBN: 0471358460. | | [4] Myers, Glenford J. Art of Software Testing, Hoboken, NJ, USA: John Wiley & Sons, Incorporated, pages: 234, June 2004, ISBN-10: 0471469122. | | [5] Puleio, M., How Not to Do Agile Testing, In Proceedings of the Conference on AGILE 2006, IEEE Computer Society, PP: 305 – 314, Year of Publication: 2006, ISBN: 0-7695-2562-8. | | [6] Stapleton, J., DSDM: Dynamic Systems Development Method, Technology of Object-Oriented Languages and Systems, 1999, pp.406-406, Jul 1999, ISBN: 0-7695-0275-x. | |