Available Online at http://www.recentscientific.com

CODEN: IJRSFP (USA)

International Journal of Recent Scientific Research
Vol. 9, Issue, 5(I), pp. 27068-27072, May, 2018

ISSN: 0976-3031

International Journal of Recent Scientific Research

DOI: 10.24327/IJRSR

Review Article

AS²T WITH CI- A REVIEW

Ramesh Solanki and Shilpa Das

Department of MCA, VESIT

DOI: http://dx.doi.org/10.24327/ijrsr.2018.0905.2181

ARTICLE INFO

Article History:

Received 24th February, 2018
Received in revised form 19th March, 2018
Accepted 16th April, 2018
Published online 28th May, 2018

Key Words:

AS^{2}T, Software, testing, automation, CI, CD, quality, server-side.

ABSTRACT

AS^{2}T is an automated server side testing approach which is much needed as in this age of computers, we are totally dependent on software/applications for all our day to day activities. As customer's satisfaction and reliability are given prime importance, software testing has gained great importance to ensure the Quality of software. In this paper we are going to discuss the benefits of using Automated Server-Side Testing and the importance of using CI/CD techniques to enable seamless continuous integration and delivery.

INTRODUCTION

Software Testing is one of the very important stages of software development process. Testing helps to detect the bugs and anomalies in the software's functionalities. Assuring the quality of software before delivery is very crucial for user satisfaction and helps to maintain a good relation with the clients.

Various types of testing are performed to check the quality as per the complexity of the software, such as frontend testing, backend testing, GUI testing and so on. In this paper we are going to discuss more about the importance of server-side testing or backend testing. Earlier testing was carried out by writing manual test cases considering the test case scenarios by the testers. The testers used to execute the written test cases manually in order to validate the system which was very tedious and time-consuming process.

To overcome the drawbacks of manual testing, automation testing came in picture. Automation testing is nothing but replicating the same old manual test execution process by writing automated test script which will help to achieve the testing goals and eliminate the unnecessary repetitive manual work done by testers.

The automation tools help to execute the automated test scripts and generates the test results without any human interventions. These tools include Selenium, cucumber, Test Complete, QMetry Automation Studio, HP QTP/UFT, etc. These automation tools help to automate the software's frontend/GUI or backend/server-side processes. It also provides an opportunity to other business entities (non-testers) to develop test cases which are needed to be automated.

Using Continuous Integration in software development process helps to identify bugs quickly as soon as any part of code is developed, automated test scripts are executed on that code and it is checked whether any issues exits or not. It makes the whole process faster and more reliable.

Need For AS^{2}T

Benefits: [18, 19] Server-side testing is all about dealing with SQL queries, schemas, triggers and procedures to interact with Database. Backend testing is very important to make sure the business logic implemented in the system is correct. It involves testing systems functional behavior. Almost all big or small applications or software requires some sort of database to hold some data related to business. Testing only the GUI part will not help showing correct information or full filling the business requirements. You know, database is the heart of any application. Moreover, with complex applications, the need to have a stronger and secure database increases. Hence, database testing is equally important for testing an application's performance and functions.

Corresponding author:* **Ramesh Solanki
Department of MCA, VESIT

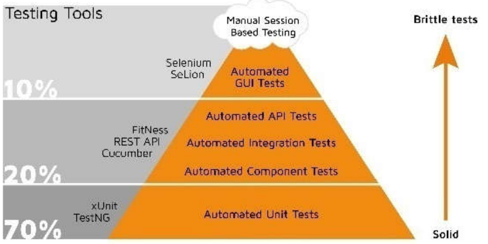

Figure 1 Need of Testing and its evolution

Databases are checked for ACID properties, CRUD operations, their Schema, Business rule conformance. Databases is also checked for Security and Performance.

In back-end testing, there is no need to use the GUI. You can directly pass the data using a browser with the parameters required for the function to get the response in some default format. E.g., XML or JSON. You also connect to the database directly and verify the data using SQL queries

Benefits of AS^{2}T

The common practice of testing a data-intensive application involves utilizing a data examining tool which requires rounds of manual intervention. This limits a lot of productivity, which results in reducing the time-to-market.

Automation has become the most important need of hour in testing world. Considering the time and efforts involved in testing hundreds and thousands of SQL lines and codes manually and expecting to be ahead of release deadline makes no sense. To keep up with the quality and productivity at the same time in today's agile market, experts suggest to implement automation testing for database.

1. Speeds up the cycle of testing creating a room to test large data sets.
2. Processes complex transactional behaviors.
3. Enables flexible deployment and operational capabilities.
4. Back end testing is not like a Black Box Testing.
5. Full control of Test coverage and depth.
6. In early development stage, many bugs can be effectively found.

Benefits of Automation Strategy

[16] With an automation strategy, one can be certain about achieving below results:

Data Integrity

This makes an easy approach for integration of newly updated data so that the latest state of the data is reflected everywhere.

Accuracy of Business Rules

Databases these days are not meant only to store records. In fact, they support to implement business logics at the DB level. Automation tool can ensure that the implemented business logic is working accurately.

Database Mirroring

This offers considerable database availability checkup. The principal advantage of these operations is that it checks whether any server instances (e.g. Production, Testing, Staging, etc.) contain identical data.

Automated Database Regression Testing Automation tools help in getting the testing team up with the database regression testing requirements in no time, overcoming several challenges they face in manual testing. It ensures the functioning of the applications on a regular basis for the changes made in the database tested.

Types Server-Side Testing

[17]Structural Testing: The structural data testing involves the validation of all those elements inside the data repository that are used primarily for storage of data and which are not allowed to be directly manipulated by the end users. The validation of the database servers is also a very important consideration in these types of testing. The successful completion of this phase by the testers involves mastery in SQL queries.

1. Stored procedures testing
2. Schema testing
3. Database table, column testing

Functional Testing: The Functional database testing as specified by the requirement specification needs to ensure most of those transactions and operations as performed by the end users are consistent with the requirement specifications.

- Checking data integrity and consistency.
- Login and user security.

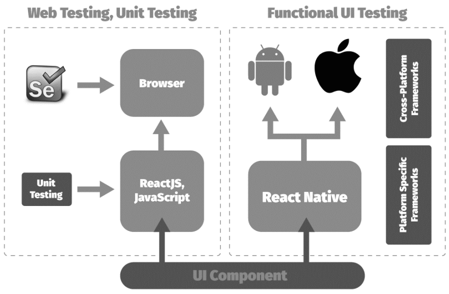

Figure 2 UI Components

Non-functional Testing: Nonfunctional testing in the context of database testing can be categorized into various categories as required by the business requirements. These can be load testing, Stress Testing, Security Testing, Usability Testing, and Compatibility Testing and so on. The load testing as well as stress testing which can be grouped under the gamut of Performance

Testing serves two specific purposes when it comes to the role of nonfunctional testing

Risk quantification- Quantification of risk actually helps the stakeholders to ascertain the various system response time requirements under required levels of load. This is the original intent of any quality assurance task. We need to note that load testing does not mitigate risk directly, but through the processes of risk identification and of risk quantification, presents corrective opportunities and an impetus for remediation that will mitigate risk.

Minimum system equipment requirement- The understanding which we observe through formal testing, the minimum system configuration that will allow the system to meet the formal stated performance expectations of stakeholders. So that extraneous hardware, software and the associated cost of ownership can be minimized. This particular requirement can be categorized as the overall business optimization requirement.

***Load testing**:* A load test is type of software testing which is conducted to understand the behavior of the application under a specific expected load. Load testing is performed to determine a system's behavior under both normal and at peak conditions.

Stress testing: Stress testing is also sometimes referred to as torturous testing as it stresses the application under test with enormous loads of work such that the system fails. This helps in identifying breakdown points of the system.

Continuous Integration helps in identifying the smallest breaks in the application code and fixing them early in the development stage. Hence, each integration is verified

Database testing also refers to Back-end testing is very important. Here, are some important back-end testing tools which help to find issues like deadlocking, data corruption, and poor performance.

Data Factory

Data factory is database testing tool. It works as data generator and data manager for database testing. It has very easy to use interface and capable of managing complicated data relationship.

Data Generator

DTM Data Generator is another backend testing tool. It is used for generating data rows and schema objects for database testing. The tool supports Load Usability and performance testing on the database.

TurboData

Turbodata software tool can be used to generate test data with foreign keys. It allows to use Select, Updates, and Delete SQL commands. It also supports multiple sequential files and relational databases.

ROI on Test Automation

[15]Test Automation is attractive, necessary and expensive too. But it helps to measure the outcome of the whole process which will eventually help to take decision whether to automate or not.

The ROI calculation for automation testing needs to be done in terms of days since automated tests can be run for 24 hours continuously; as opposed to 8 hours that is considered for manual testing. However, 18 hours seems more reasonable since sometimes the test cases are interrupted and do not run for 24 hours.

The formula, which forms the basis of ROI calculation, is:

$$\text{ROI}=\frac{\text{Gain}-\text{Investment}}{\text{Investment}}$$

Investment =Automated test script development time +
Automated test script execution time +
Automated test analysis time +
Automated test maintenance time+
Manual Execution Time

Gain= Manual Execution Time * total number of Testcases (Automated/Manual)*nPeriod of ROI / by 8.

a. Automated test script development time = (Hourly automation time per test * Number of automated test cases) / 8
b. Automated test script execution time = (Automated test execution time per test * Number of automated test cases*Period of ROI) / 18
c. Automated test analysis time = (Test Analysis time * Period of ROI) / 8
d. Automated test maintenance time = (Maintenance time * Period of ROI) / 8
e. Manual Execution Time = (Manual test execution time * Number of manual test cases * Period of ROI) / 8

Period of ROI is the number of weeks for which ROI is to be calculated. Divided by 8 is done wherever manual effort is needed. Divided by 18 is done wherever automation is done.

CI-CD

Continuous Integration (CI)[10, 11] is a software development practice in which you build and test software every time a developer pushes code to the application.

Continuous Delivery (CD) is a software engineering approach in which continuous integration, automated testing, and automated deployment capabilities allow software to be developed and deployed rapidly, reliably, and repeatedly with minimal human intervention.

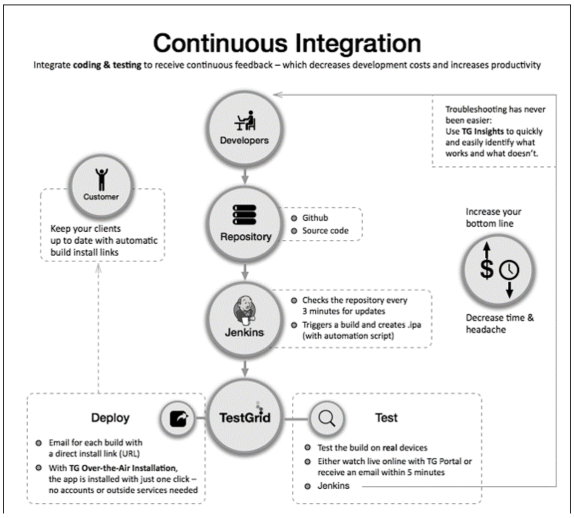
Continuous Integration
Integrate coding & testing to receive continuous feedback – which decreases development costs and increases productivity
Developers
Troubleshooting has never been easier:
Use TG Insights to quickly and easily identify what works and what doesn't.
Customer
Keep your clients up to date with automatic build install links
Repository
Github
Source code
Increase your bottom line
Jenkins
Checks the repository every 3 minutes for updates
Triggers a build and creates .ipa (with automation script)
Decrease time & headache
Deploy
TestGrid
Test
Email for each build with a direct install link (URL)
With TG Over-the-Air Installation, the app is installed with just one click – no accounts or outside services needed
Test the build on real devices
Either watch live online with TG Portal or receive an email within 5 minutes
Jenkins

Figure 3 CI Structure

Advantage of CI [21]

1. ***Early Detection of Bugs:*** Continuous Integration helps in locating bugs early in the development process. Since you can detect at an early stage, you can solve problems quickly and spend quality time building features. Failing to follow a continuous integration process may prove to be costly because you may have to spend long hours trying to find the bugs. The CI server is responsible for monitoring the repository for changes in a build or bug occurrence. But it is the responsibility of the team to monitor for changes.
2. ***Helps in Determining Code-break:*** Through CI, code breaking is detected. The developer makes his commit to the build, and automated testing process takes place to

ensure the new code doesn't break any of the existing codes. If a conflict arises, then he will have to fix it so the new code and the changes that came before and after are synced. This way, you can increase code quality.

3. ***Only a small number of changes are lost when the code base reverts to a bug free state:*** We already know that CI helps in detecting bugs and as the tests are conducted automatically, with the developer making frequent checks, it is easy for the code base to get back to old, bug-free state. Only very few/minimal changes will be lost as the integration is taking place frequently. This is a time and money-saving feature, hence a major advantage. When a unit test fails, the team can go back to the original bug-free state as well, without debugging.
4. ***Regular Availility of a "Current" Build:*** Another advantage of CI is the continuous and regular availability of an updated build (give or take a few hours, probably, between the last test) to help with testing, demo activities and even release purposes.
5. ***Enables Continuous Deployment:*** When the result of the development process moves towards deployment through Continuous Integration, it leads to a production-like environment. The automated functional testing executed at regular intervals leads to continuous deployment followed by continuous delivery.
6. ***Enables Automated Testing:*** As mentioned in the previous point, the testing is automated, while the changes are deployed automatically into the production pipeline. Each time an integration is made, it would be verified by an automated build, thereby reducing integration problems and helping the team to develop error and bug free software rapidly.
7. ***Reduces the Risk of a Longer, Time-Consuming Project:*** Using CI process for continuous delivery and automated testing is definitely speeding things up. What used to take weeks can now be finished within hours.
8. ***Increases the Quality of Your Software:*** Through CI, you are eliminating the blind spots completely, helping you release a bug-free and error-free software. The software meeting client requirements, is delivered on time, minus the usual hiccups with performance and with no compromise in.

Why is CI Testing Important?

They reach an agreement with the management taking the developer's promise at face value failing to realize that problems or challenges might hinder with the development progress. In the end, if there is a failure, the management feels that the developers have betrayed their trust, thereby straining their relationship, with the outcome reflected adversely on the product [14].

This situation can be completely avoided through CI process. Each team member has an idea of what the other person has done so far, so there will be no contradictions or confronting situations. Each time a developer makes a commit to the repository, automated test runs will be conducted. This helps in identifying the breaks or bugs in the code, even identifying the person who caused the breaks.

Tools Used for CI

1. ***Bamboo[13]:*** CI tool from Atlassian. Build, tests and deploys tests automatically. Unleash the potential of agile development, detects bugs.
2. ***Jenkins****:* Building and testing software projects continuously and monitoring the run jobs. We will get into Jenkins in detail later.
3. ***Buildbot****:* Contains a master and repository of slaves. The master monitors the codes and assigns tasks to the slaves. Was intended to be an alternative to Tinderbox.
4. ***Travis CI****:* Open source, free and very easy to use. Offers SaaS version to help with free testing.
5. ***Strider****:* Written in NodeJS and JS, uses MongoDB as a backing store. Customization can be done through plugins.

CONCLUSION

In past few years software testing has been given new dimensions by adding few terminologies like automation and DevOps. Automating the test cases helps in saving time and tedious efforts made by the testers also, it ensures the correctness of the system's business as well as functional working. A software development project necessitates the relationship between developers, tester and management. One of the core values of this relationship is trust. The management gives the requirement of a particular application the developers and the testers agree to deliver the completed version of the same on a particular day. Continuous Integration helps in identifying the smallest breaks in the application code and fixing them early in the development stage. Hence, each integration is verified

References

1. https://www.testing-whiz.com/blog/how-to-calculate-roi-for-test
2. https://www.testing-whiz.com/blog/why-is-it-important-to-have-an-automation-strategy-for-database-testing
3. https://www.guru99.com/frontend-testing-vs-backend-testing.html
4. https://www.softwaretestinghelp.com/how-to-perform-backend-testing/
5. https://www.guru99.com/data-testing.html#2
6. https://www.guru99.com/what-is-backend-testing.html
7. https://www.cabotsolutions.com/2016/12/continuous-integration-testing-using-selenium
8. https://nevercode.io/blog/what-is-continuous-integration-and-how-to-benefit-from-it
9. https://dzone.com/articles/9-bene-ts-of-continuous-integration
10. https://dzone.com/articles/9-bene-ts-of-continuous-integration
11. https://nevercode.io/blog/what-is-continuous-integration-and-how-to-benefit-from-it
12. https://dzone.com/articles/9-bene-ts-of-continuous-integration
13. https://www.cabotsolutions.com/2016/12/continuous-integration-testing-using-selenium
14. https://www.cabotsolutions.com/2016/12/continuous-integration-testing-using-selenium

15. https://www.testing-whiz.com/blog/how-to-calculate-roi-for-test
16. https://www.testing-whiz.com/blog/why-is-it-important-to-have-an-automation-strategy-for-database-testing
17. https://www.guru99.com/data-testing.html#2
18. https://www.softwaretestinghelp.com/how-to-perform-backend-testing/
19. https://www.guru99.com/what-is-backend-testing.html
20. https://medium.com/@cabot_solutions/a-guide-to-using-selenium-for-continuous-integration-testing-c2b43f4d4872

How to cite this article:

Ramesh Solanki and Shilpa Das.2018, As2t With Ci- A Review. *Int J Recent Sci Res.* 9(5), pp. 27068-27072.
DOI: http://dx.doi.org/10.24327/ijrsr.2018.0905.2181

****