eBook

Test Automation: An Enigma that Continues to Haunt

ZUCI SYSTEMS

You have just completed developing a critical customer application, and it is ready to be used live. Your team is confident that your application has been built with the expected usability, functionality, performance, and security levels. Does that mean it need not be tested? Not quite. Software testing is an essential step for ensuring the quality of any software solution.

At a high level, testing can be classified as either manual or automated, depending on how it is performed. Manual testing involves running test cases manually to identify errors, whereas automated testing uses a programmatic approach for testing. With the increase in the scale and complexity of applications, the need for automated testing is more significant than ever now.

We will discuss the early stages of software testing and its evolution. We will also look into the various challenges faced by organizations in adopting test automation and how to overcome those.

CHAPTERS

01

The SDLC Stages

The software development process is a long-term project that depends on developers' commitments. Team leads and analysts optimise development cycles to outline, design, develop, test and output to ensure quality and efficiency.

The software development cycle is a model provided to the management detailing the productive output of the project. The development cycle goes through the following 7 phases for effective results:

PLANNING STAGE

Here, developers plan for project feasibility and future references.

ANALYSIS STAGE

They analyse the project by gathering all the details required for the software development. Then, they highly focus on discussing the pros and cons of the project.

DESIGN STAGE

Here, designs are created for the overall application and specific aspects, such as user interfaces (UIs), system interfaces, network requirements, and databases.

DEVELOPMENT STAGE

Developers write the code and build the application according to the specifications provided. Here, programming languages like Java, SQL, and several others are used to write these codes. Developers use different tools such as compilers, debuggers, and interpreters in this stage.

TESTING STAGE

Testing is performed to clear bugs in the code and defects in implementation and fix the errors. Then, retesting is performed to remove obstacles and move forward to the next stage.

IMPLEMENTATION AND INTEGRATION STAGE

After testing is complete, the overall design is checked and analysed. Developers optimise the error-free application in its final stage.

MAINTENANCE STAGE

Developers move to the maintenance stage for productivity. They take responsibility for the deployment of the software if it is required.

These software development lifecycles go through the whole development process to output error-free results. If any stage fails, developers analyse the set to rectify mistakes in that stage. Most organisations in the software-development field follow this practice.

Quality assurance (QA) Test automation plays a vital role in developing applications or software products. At times, application testing is not given enough importance. We will now explore challenges faced by organisations in setting up a test automation team as well as its benefits and challenges.

- **What are the challenges faced in these organisations in a testing team?**
- **Are they governing the software testing for money-oriented motives only (or profits)?**
- **What difficulties will they face in choosing and adopting test automation tools?**
- **How much importance should be given to test automation?**
- **What are the steps to be taken to eliminate obstacles?**

02

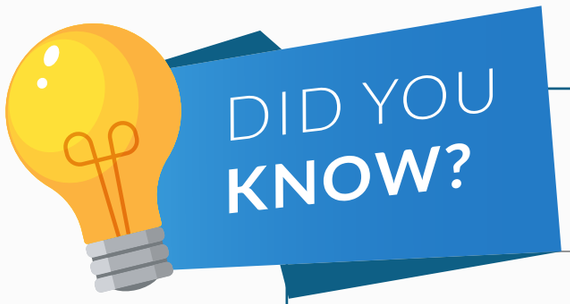
What is Software Testing?

Testing is the process undertaken after a product or software is manufactured. It is to examine whether the software is working correctly. There are many ways of testing software. The two primary approaches are manual and automated testing. Testing also varies depending on the nature and criticality of the software. Testing reduces bug occurrence in a software product or application, lowers development costs, and improves customer satisfaction.

What Is Test Automation?

Test automation is the practice of running test cases, managing test data, and reporting results automatically to improve software quality. It tests the actual result with the expected one. Test automation helps maintain the quality standard of a product or software and helps pass the testing phase rapidly.

Test automation is also eliminating human error while performing manual testing. Without automation, the number of feedback cycles increases, leading to delays in feature releases. Various test automation metrics, including cycle time and test coverage, help understand the effectiveness of test automation.



The most common test automation metrics are

- Requirements coverage
- Percentage of tests passed/failed
- Percentage of defects found
- Percentage of automated test coverage
- Percentage of broken builds
- Percentage of defects in production

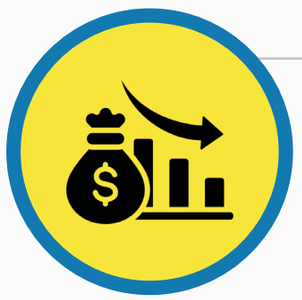
03

The Benefits Of Test Automation

While there are certain limitations to test automation, there's no doubt that it has become a necessity. With the advancements in testing technology, the benefits of automation will prove to be a critical differentiating factor in quality assurance.

1. Faster time to market

A continuous testing approach means completing the development cycle and taking your product to customers sooner.



2. Cost reduction

Automated testing helps save money, resources, and time during the QA process. While there will still be some manual testing that would need to be performed, the QA team will have time to work on other projects. This will reduce the overall cost of software development, not to mention potential savings from defect avoidance costs.

3. Simultaneous tests

Mature automated tests require little human intervention once it's started. You can run as many tests as you want in parallel. This also saves time and avoids any data errors.

4. Faster feedback

Inherently, manual testing needs considerable time and effort before the testing team can get back to the development team with their results and observations, thereby unduly delaying the release cycle. By deploying a robust test automation framework and appropriate test automation tools, testing can be performed faster, and its results can be shared sooner.

5. Better test coverage

One of the typical challenges that project teams face with manual testing is the ability to ensure high test coverage, given that QA testing teams tend to get the least amount of time in the entire software development life cycle. With a good QA test automation suite in place, even 100% test coverage can be ensured, thereby vastly improving the quality of the software.

04

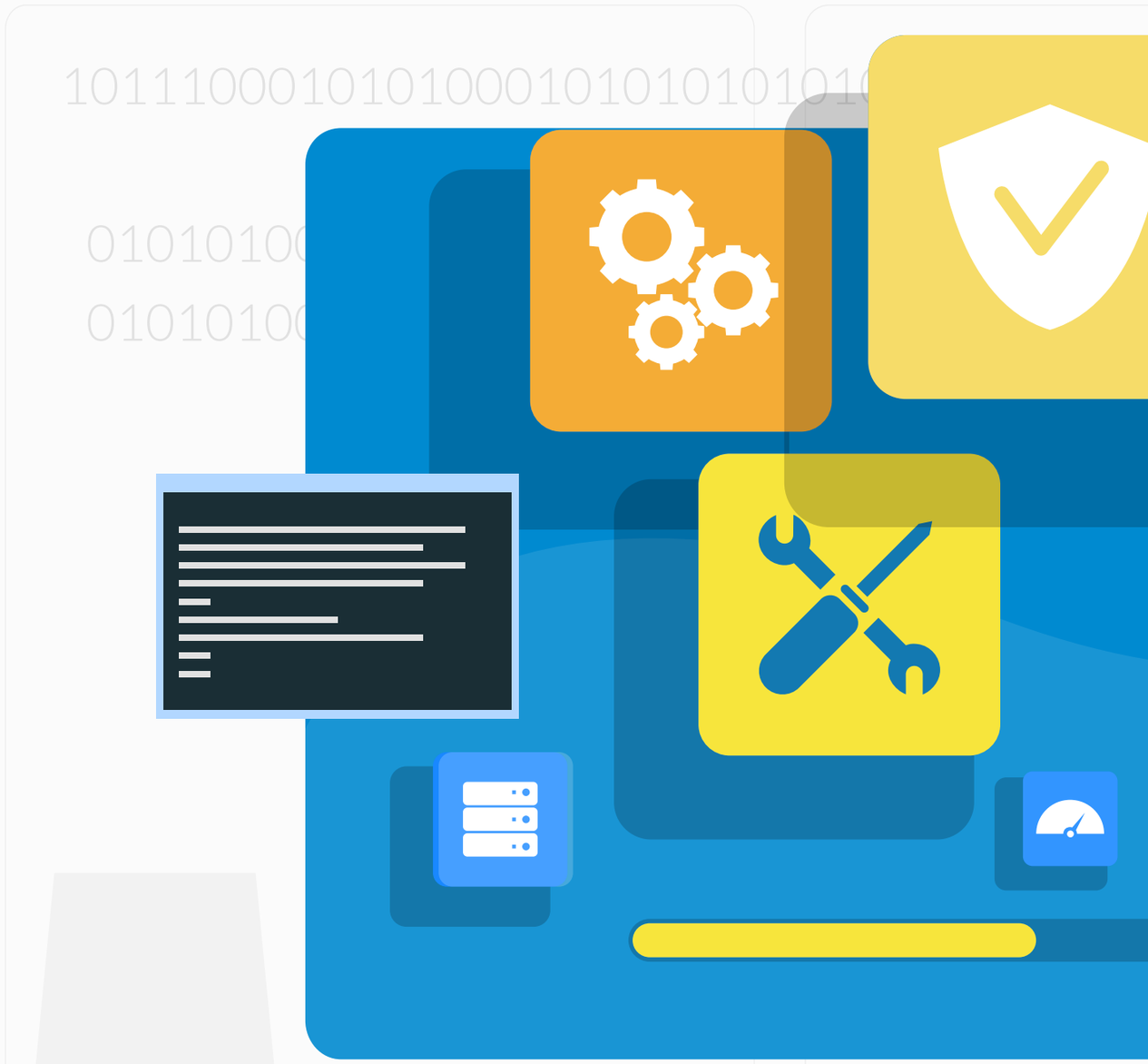
Test Automation Tweens, Teens, And Twenties: What Has Changed

Let us see how test automation has evolved over the past 20 years and what has changed. In the 1990s, test automation was not prevalent in our industry. Although many people did not readily recognize it, test automation existed from an earlier period itself.

The record and playback methodology performed the first generation of testing, specifically for regression testing. The test data involved in the testing were generally hard-coded. Even a minor error would lead to re-recording the testing manually from the first to complete the actions.

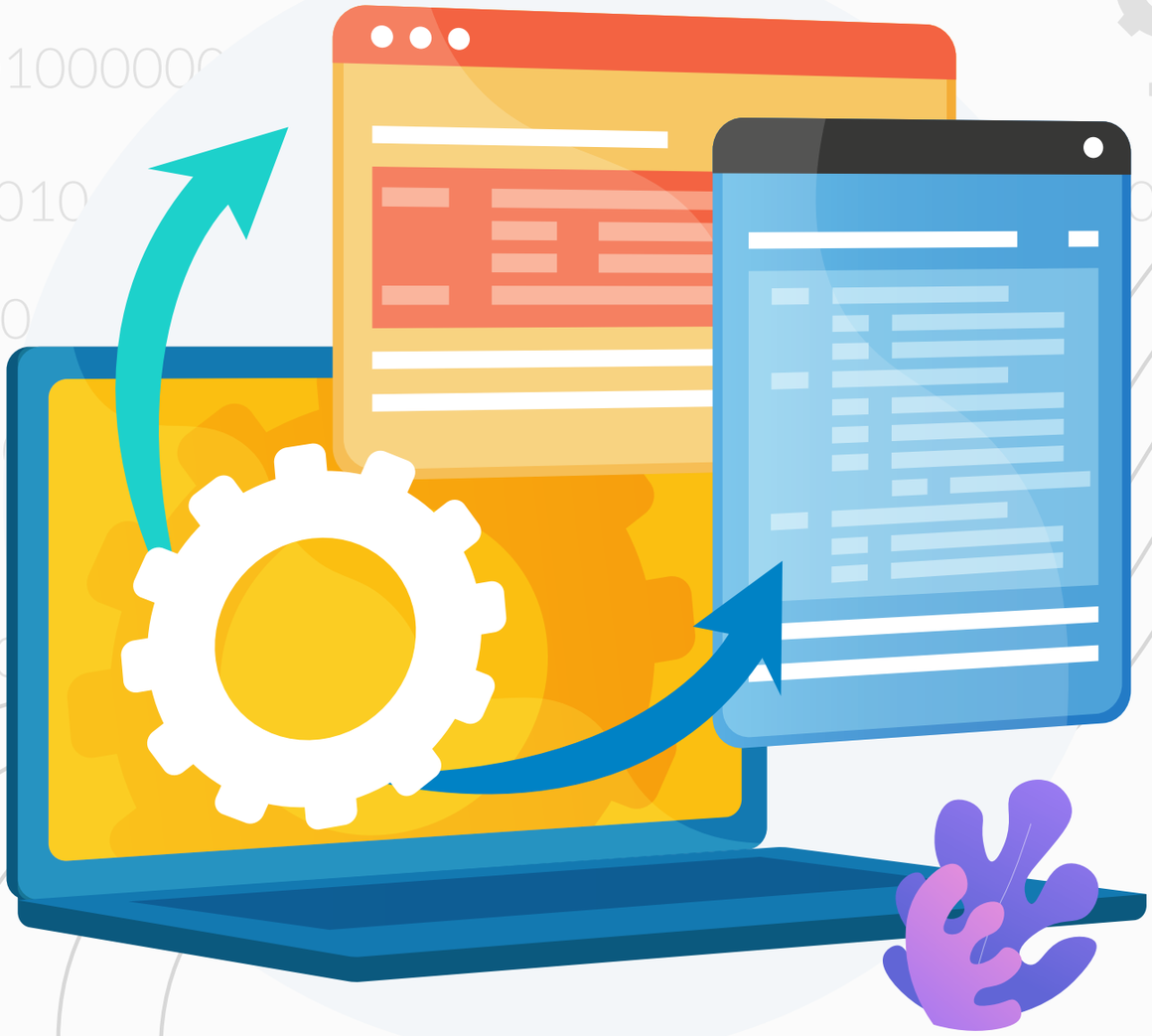
Also, it was an action-based test automation framework that involved manual effort every time. The tester must manually record the steps by clicking the button selecting the drop-down menu, option-box, and submissions.

These are the teen stages of the test automation process. An automation framework was used to handle UI elements and changes. Then, it moved to descriptive programming-based automation.



The script-based methodology performed the second generation of the automated testing framework. This method is an object-oriented programming approach. Testers extensively used this method than the first generation; however, good coding skills are still required to create the scripts.

In the twenties, the test automation process was integrated with continuous integration and delivery. Concepts like DevTestOps are becoming popular. Some of the prevalent domain-specific languages are XML(extensible markup language), SQL, HTML, combining DSLs, UML(universal markup language), LATEX, etc. There is a thin-line difference between developers and testers, and sometimes these roles are even interchanged. Test automation as a specialisation has indeed grown to be mature now, and further evolution is in the anvil with a more artificial-intelligence-based approach to test automation.



05

Test Automation Tweens, Teens, And Twenties: What Has Not Changed

Though automation came in the twenties from teens and tweens, some basic questions related to automation remain the same. A few organisations in this industry are practising test automation. Many organisations are still in doubt about test automation as they don't have a clear idea of this field.

There are many open-source community tools, like google selenium tools and a lot to explore, but fundamental questions remain the same.

What are those fundamental questions that puzzle us?

- Is the return on investment (ROI) of test automation based on defects, reducing manual testers on the project or something else?
- Between an application programming interface (API) versus UI, where to automate more?
- Despite test automation coverage, why do we get production defects?
- What reports can we present to the management team to showcase test automation progress and success?
- How early can test automation help us to succeed?

Is test automation a silver bullet for all our QA problems?

What metrics to track for test automation projects?

How do you complete tool x and tool y?

Which is better suited for test automation?

How to track test automation progress?

What are the five metrics to measure the success of test automation?

We should clearly understand that test automation is not about the tool or metrics for compiling your software. We should appreciate the value of automation in improving the quality of the software.

06

What are the enigmas that are observed in your test automation?

These are the enigmas that have been raised by many organisations:

ENIGMA #1

Why do companies spend thousands of dollars on test automation tools only to abandon them later?

ENIGMA #2

Despite having issues in UI, why do companies prefer UI test automation?

ENIGMA #3

Why does only a handful of companies succeed in test automation though it's been in practice for over two decades?

ENIGMA #4

Why is there more emphasis on people over progress?

ENIGMA #5

Why is **test automation** progress a challenge?

ENIGMA #6

Why is 80% of the bugs (variable and more directed at new features) still found manually despite several years of investment in automation?

ENIGMA #7

If automated tests are faster, then why do automation projects fall behind a schedule when there is a change in business tasks that are being automated?

Test automation slows down when any person in an office resigns. This can be resolved by teaching test automation to larger teams. They should be driven by an understanding of implementing test automation tools.

07

What can we do about these enigmas?

There are specific steps to establish and drive test automation practice. Having a well-defined process helps address the problems mentioned above.

Action 1

Identify a test automation champion from the in-house team if possible. If not, engage a consulting firm with prior experience in achieving test automation goals. A top management representative must be involved in this process to ensure focus and timely delivery. Establishing a clear delivery plan and milestones is vital for achieving success in this exercise.

Action 2

Including a holistic view of software development is critical for driving test automation initiatives. This person should be capable of making decisions on his own, based on the needs of his customers and engineering teams. To make such mature decisions, the person must have experience in software development, QA, and product management. A keen eye on customer experience is a must. This ensures that the problems are looked at from all sides before actual implementation.

Action 3

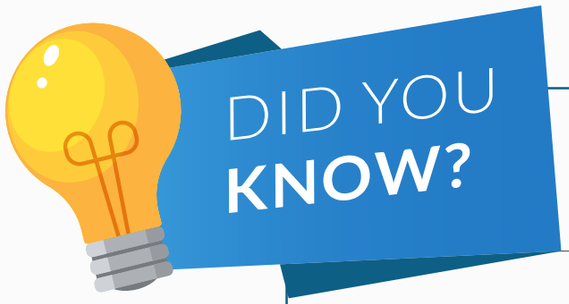
The test automation champion should have a clear vision of what is to be achieved from test automation, and they should be able to look at the roadmap, and plan accordingly. For instance, each application in question could have a varying life span. Some applications may have a more comprehensive roadmap and some have a shorter duration. The test automation framework must consider all these and ensure that any functional and technological changes do not affect the automation suite.

The engineering-savvy champion plans for the near term and the long term. The emphasis needs to be in the near term on the investment required to achieve a certain level of success. And in the long term, how much change will happen that would impact the near-term goals. The clarity of vision is essential for making such balanced decisions. While planning for the test cases that would be automated, one should be aware that some of the test cases would get deprecated because of changes in functionality and/or technology. So, tactical decisions need to be made about which components are more likely to change in the near term. In most cases, it would be UI-related test cases. Such planning will help reduce the wastage of effort and, in turn, money.

Action 4

A good balance between manual and automated testing can help maximise your returns. People think that manual testing can be discarded entirely because of test automation. That is never the case. The best way to address it is to automate more complex areas like API, components, and regression testing, whereas UI testing can be manual. This approach complements the unit testing effort of developers and helps unearth defects and improve the test coverage.

Automating everything is nearly impractical, whether it is an agile approach or the conventional waterfall approach. Striking a good balance between what to automate and what not to ensure the long-term success of automation endeavours.



The most preferred tests for automation are

- Functional tests
- API tests
- Regression tests
- Smoke tests

What are the test automation tools for developing the software?

Setting precise automation testing tools is the answer to the manifesto and the fast-growing standards for software testing. Here are some test automation softwares that were updated in 2022:
Selenium, Katalon Studio, Appium, Cucumber, and cypress.io.

08

Test Automation - Best Practices

1. Define the goals

Establish the goals of automation clearly at the start and continuously monitor the progress to ensure the team stays on track.

2. Cross-functional team collaboration

It is essential for the test team to be constantly interacting with the business team and the engineering team. This gives them a clear understanding of the product requirements.

3. Robust Test framework

Ensuring that the test framework is robust and yet, is easy to use, for the team. This will help improve productivity and also ensure the test scripts are maintainable.

Keys to Approaching Test Automation

- Establish the outcomes of automation efforts
- Define and measure success
- Establish the following roles and responsibilities for each member in the team
  - Who will roll out the test plan?
  - Who defines test cases?
  - How should you communicate among the team?
  - How to monitor test progress?
  - How to communicate test reports?

09

The Way Forward

There is a lot of buzz today around how the latest technologies like AI can help test automation maturity. There are also concepts like DevTestOps picking momentum. But before you can jump on to these bandwagons, it is important to have successfully implemented test automation and gathered necessary metrics, data around it.

Without extensive experience with test automation it is not practical to deploy technologies like AI. For instance, if you want to deploy AI for generating test data it is important to have a huge amount of data collected already which can then be processed by the AI algorithm. Also, the scalability and robustness of the test framework needs to be achieved before moving forward. Once such areas are addressed, your organisation is set for leveraging relevant modern technology tools.

10

Final thoughts

Addressing the aforementioned points can resolve the enigmatic queries that haunt organisations regarding test automation. Focusing on the larger vision and purpose of it, test automation driven by a test champion with a holistic knowledge of software development will go a long way in realising the goals and ROI.

Having a good test automation suite can reduce efficiency even when people leave the office. Automated testing should not be approached just for monetary reasons; values should drive it. Automation should be incorporated into every organisation's culture to realise its potential fully. Talk to QA consultants at Zuci Systems to get started with your test automation journey.

About Zuci

Zuci is a digital organization focused on the craft of assuring quality to software which we have perfected over the years. Zuci enables product teams to improve product quality, fasten release speed and enhance user experiences - with a perfect blend of test planning, engineering perfection, and customer-centricity..

Learn more at **zucisystems.com**