indium
Make Technology Work

uphoriX: A Next-Generation Test Automation Platform from Indium Software

White Paper

www.indiumsoftware.com

Enterprises are driving up operational efficiency and getting future-ready through rapid digital transformation and automation. They are adopting a variety of cloud-based and on-premise software applications to streamline and automate functional workflows. As software applications get modernized, it is critical that the testing process becomes "smarter" as well.

Over the last two decades, Indium Software has been ahead of the curve in terms of innovation in the test automation space. Our latest offering, **uphoriX,** a next-generation smart test automation platform, is designed to deliver 10x faster testing release cycles.

Going beyond finding errors, the goal of software testing has expanded to improve product quality and provide **other benefits** such as:

- Save costs and drive financial benefits
- Improve security and compliance
- Enhance customer satisfaction
- Accelerate the development process
- Simplify addition of new features
- Enhance the performance of the software

With time, the software development lifecycle started shrinking to enable faster release of updates and patches. Testers too started automating the testing process to reduce manual efforts for repetitive processes such as smoke and regression testing apart from automated CI CD pipeline. Test Automation has now become the default way of testing and continues to evolve to keep pace with the changing requirements of the developers and the customers.

As a result, the global Test Automation **market size** is expected to grow at a Compound Annual Growth Rate (CAGR) of 18.0%, from USD 12.6 billion in 2019 to USD 28.8 billion by 2024. Test Automation involves automating functional test cases and can be static or dynamic. Static testing can be implemented even at an early stage of the development process when the code is still not logically completed where as Dynamic testing will require the completed product.

Test Automation Framework

Increasing the efficiency and success of test automation are test automation infrastructure or framework that integrate test tools, test scripts, procedures, devices, and the testers. By utilizing the test automation framework, testers can ensure that all the tools and devices work together in a coordinated manner. It allows the reuse of the library of functional script for different test projects, thereby reducing the need to duplicate the development effort.

The test automation framework also ensures standardization and consistency of test scripts and is made available at the time of regression testing as well.

A test automation framework is made up of the following six components of test automation infrastructure are as follows:

1.The System to be Tested: For test automation to be cost-effective, a stable subsystem of the system to be tested is essential.

2.Test Platform: Establish the test platform and facilities such as the network setup including configuration management utilities, servers, clients, routers, switches and hubs where the system will be tested.

3.Test Case Library: Compile libraries of reusable test steps of basic utilities such as ssh (secure shell) from client to server, response capture, error logging, clean up and setup to form the building blocks of automated test scripts.

4.Tools: Test script development requires different types of tools such as test automation tools, tools for traffic generation, traffic monitoring and support. The support tools in turn include test factory, requirement analysis, defect tracking, and configuration management, which need to be integrated with test automation for reporting defects for failed test cases automatically. The test factory tool is used to generate automated test execution trends and result patterns.

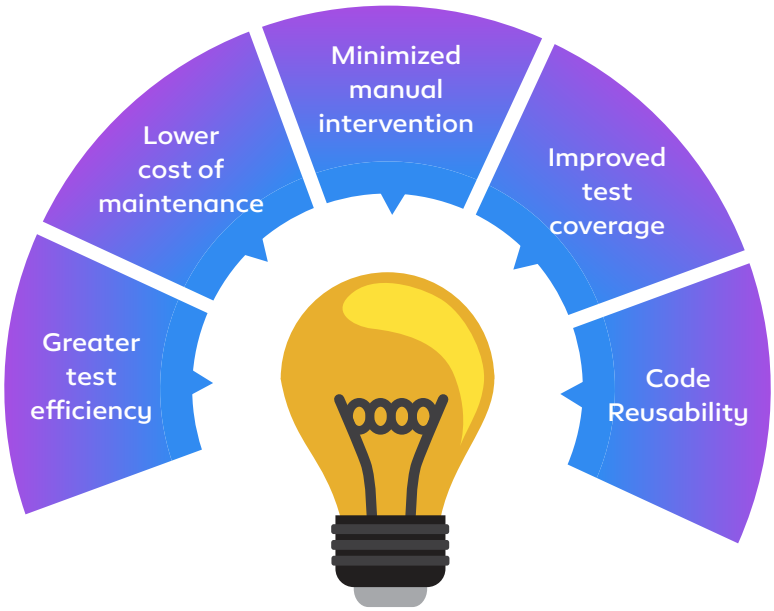
5. Automated Testing Practices: Documentation of how to automate test cases using test tools and test case libraries is important and can enforce consistency across the automated test cases developed by different engineers. It must also list the utilities and guidelines to improve efficiency in test automation and document the maintenance procedure for the library.

6. Administrator: The automation framework requires an administrator who will manage the test case libraries, platforms and tools; maintain the template inventory; provide training support; and help test engineers writing test scripts using the test case libraries.

A framework can be viewed as the integration of set of best practices, not specific to a particular development environment, but applicable across environments due to the use of reusable, maintainable, and stable automation code.

Benefits of a Test Automation Framework

In today's world of Shift Left and DevOps, where quality is becoming integrated with the development process right from the planning stage, testing needs to keep pace with the development process. A test automation framework can enable this by providing the speed and efficiency needed to ensure test accuracy while reducing risks and the test maintenance costs. By using a framework, testers can experience:



Types of Automated Testing Frameworks

Test automation frameworks can be of six common types, classified based on architecture, differing benefits and shortfalls. Knowing this is important for developing a test plan and choosing the right framework:

1.Linear Automation Framework: Also referred to as a record-and-playback framework, functions and steps can be written in a sequential order instead of writing a code and replay automatically for testing. Since there is no need for coding, it does not require expertise in test automation. However, the recorded data is hard-coded into the test script, because of which this is not reusable. Maintenance and scalability are also difficult as it requires a lot of rework in case of any changes.

2.Modular-Based Testing Framework: The application to be tested is divided into different units, functions, or sections, a test script is created for each part, and each tested in isolation. The test scripts are integrated to create larger tests in a hierarchical fashion to represent various test cases.

Creating an abstraction layer ensures that any changes to the individual sections will not affect the overarching module. This approach allows making changes to different modules easy and the test cases are reusable. However, this being hard-coded into the test script, multiple data sets cannot be used. This requires programming knowledge to set up the framework.

3.Library Architecture Testing Framework: Though similar to the modular framework, library architecture framework offers some advantages. The application is not divided into sections but instead, it is based on objectives where tasks that are similar are identified from within the scripts and grouped by function. These test scripts can be reused as they are stored in a library and can be called when needed.

4.Data-Driven Framework: In data-driven framework, the test data is separated from script logic, allowing testers to store data externally. This providers the testers with the freedom to test the same feature or function of an application several times with different sets of data. The test data is not hard-coded in the script. The tester can use external data sources such as Excel Spreadsheets, CSV files, SQL Tables, Text Files, or ODBC repositories to store and pass the input/ output parameters to test scripts. But to run this, a highly-experienced tester proficient in several programming languages will be required to effectively leverage this framework design. This is time-consuming and testers will need to identify and format external data sources and seamlessly connect them by writing a code.

5.Keyword-Driven Framework: Similar to the data-driven framework, in this too the test data and script logic are separated. Along with that, all the functions of the application under test are listed in a table format with corresponding instructions for tests to be carried out in their consecutive order along with keywords. This renders them independent of the automated testing tool used for executing the tests. This doesn't require much scripting knowledge and as a single keyword can be used across multiple test scripts, the code is reusable. However, the initial cost and time of setting up the framework can be high. It can be complex. The keywords need to be defined and object repositories / libraries need to be set up.

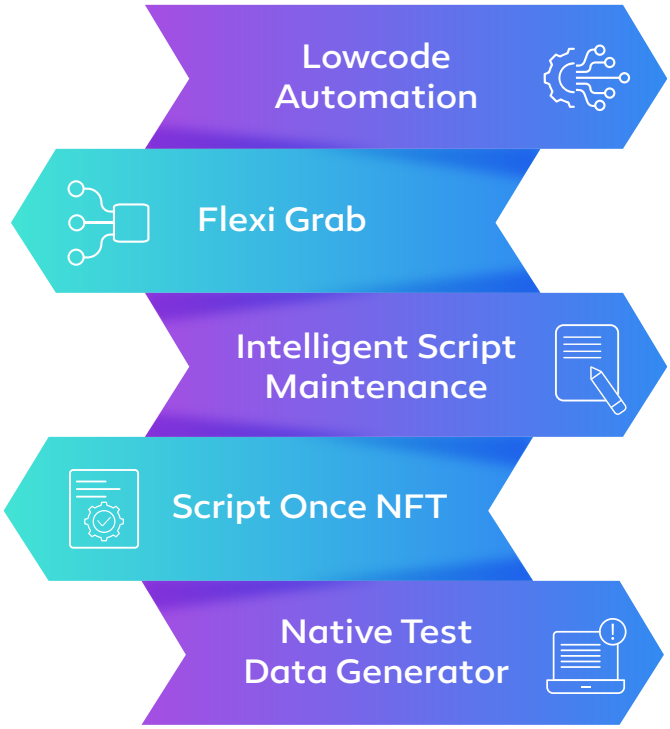
6.Hybrid Test Automation Framework: A hybrid test automation framework brings together the best of all the other frameworks and mitigates some of the weaknesses. It is adaptable and therefore provides the required flexibility essential for agile development.

Advantage Uphorix: Speed, Flexibility and Reusability

Uphorix is a smart unified testing platform from Indium Software, a leading global digital technology services company. Apart from the usual benefits of a framework mentioned above, the platform provides all-in-one Smart Quality Assurance (QA) integrating functional testing, performance testing, security and compatibility for faster and frequent release of software/application and address the pain points of building new age applications.

While improving collaboration between QA functions, uphoriX is designed to reduce the overall complexity of the testing processes by automating and increasing the test coverage. This will reduce test creation time as well as maintenance efforts. By enduring 30-40% of the end-to-end testing lifecycle, it can speed up development by 10x and accelerate digital transformation across industries. It automates the tests over the cloud, is compatible with all testing tools and integrates with most of the DevOps tools. This facilitates CI/CD and also supports automation in an Agile Sprint model.

Key Features



Some of the key features of uphoriX include:

- **Low code Automation:** Creating test scripts in automation and a stable framework can take nearly 60% of the time. uphoriX, built on top of Selenium POM, helps kick-start test automation from day 1, making test automation robust and scalable. It can can enable auto-generation of scripts in any chosen format, such as JAVA, C#, JavaScript, Typescript etc.

- **Flexi Grab:** A record and play feature, it is especially useful for UAT and manual testing. As Flexi Grab captures the user interaction on the application such as Textbox values, button clicks, form submission etc., manual defect retesting cycle is reduced by at least 40% to 50% for web based applications. uphoriX uses a browser plugin for capturing the user interaction on the application as Selenium does not have a native record & play feature. The captured script can be organized as scenarios and test cases.

- **Intelligent Script Maintenance:** Applications are constantly changing, because of which many test automation initiatives do not give ROI. To be able to use the affected scripts requires careful analysis and fixing at great effort and time. uphoriX platform's Auto Heal feature intelligently handles application UI element changes and alerts the user of the need for change. Once the user confirms the changes, they get incorporated into the scripts, saving at least 30%–40% of effort on automation script maintenance.

- **Script Once NFT:** Non-functional testing (NFT) is used to assess the performance and vulnerability aspects of an application. This includes:

  - Performance Testing: uphoriX enables re-using the Selenium script generated for automation and run performance tests, speeding up the seamless release cycle times and bottlenecks.

  - Security testing: uphoriX platform supports DevSecOps and ICOVA--Integrated & Continuous Vulnerability Analysis--enables re-use of automation scripts created using the uphoriX platform to run Vulnerability Assessment.

Browserstack can be used to execute the tests over the cloud on real devices in a combination of browser/desktop OS/Android version/IOS version/Windows mobile versions. IACA--Integrated & Automated compatibility assessment facilitates the integration of the automation created using uphoriX platform with CD/CI tools to automatically trigger compatibility assessment for every release.

- **Native Test Data Generator:** QA requires real-world test data to unearth application defects, which can be challenging for testers. D-Gen is an uphoriX native test data generation utility in which the user can configure the data format as a regular expression to generate huge volumes of data. The output can be directed to a text file, database or JSON based output.

Enabling 'Smarter Testing'

uphoriX is a smart test platform that provides the following benefits:

- Parallel testing for faster testing cycle
- Easy to use features, allowing even non-technical users to use it
- Supports several CD/CI tools such as Jenkins, Circle CI, Azure etc,
- Allowing Vulnerability Assessment to be done for every single build, aligning to DevSecOps principle
- Reliable, scalable, robust, and highly secure

uphoriX is a complete platform that allows a single flavor of scripts to perform the complete range of QA services such as automation, performance, security and compatibility testing. Its auto-establishment of Test Artefacts in standard formats resolves technical bias and allows business users to benefit. Enjoy extensive QA coverage through test automation, compatibility across multiplex device/OS/browser combinations, continuous performance and vulnerability analysis. With uphoriX, one of the greatest advantages is that it helps improve synergy between functional (manual, automation) and non-functional testing teams (performance).

To find out how you can benefit from the uphoriX platform, contact us now:

INDIA

Chennai | Bengaluru | Mumbai
Toll-free: 1800 123 1191

USA

Cupertino | Princeton
Toll-free: 1 888 207 5969

UK

London
+91 9980611604

SINGAPORE

+65 6812 7888

Sales Inquiries
sales@indiumsoftware.com

General Inquiries
info@indiumsoftware.com