

Test Automation Framework

Extensive software testing before the actual release is essential for any company to maintain its position among clients and partners. The software has to perform equally well in every platform and scenario and has to be a strategic part of the entire SDLC. Frequent number of revisions makes testing more sophisticated over time, and becomes a larger and larger proposition as time goes by. The solution to this lies in adopting Test Automation, which can be defined as executing a sequence of actions without human intervention. The purpose of the process is to eliminate man-made errors and provide faster results. Companies that opt for Test Automation pass the break-even point for labor cost after just 2 to 3 runs of automated test.

There are many factors responsible for the success of building a Test Automation Framework. The key elements are:

Management Commitment: The management should be actively involved in the Test Automation framework development.

Cost and Budget: Building a framework for Test Automation needs some budgeting.

Actual Process: The actual process should be well-defined, with no ad-hoc testing and a defined guidance for test, test-coverage and criteria for test at every step.

Related resource: To make sure that Test Automation framework development goes on smoothly, there should be a dedicated team.

Realistic Expectation: The management should have realistic expectation as 100% automatic tests is not possible and that all test cannot be automated. Test Automation will bring in results only after several cycles has been executed and there is no immediate return for the investment made for building the Test Automation Framework.

Framework defines the organization's way of doing things – a Single Standard. Following this standard, the project team will achieve -

- A test library design that will help in effective team-communication, library-versioning and Artifacts creation.
- Standard scripting that will result in team consistency during test library design and prevent individuals from following their own coding standards, thus avoiding duplicate coding.
- Sparing the Test engineers from knowing the critical aspects of the code. Implemented libraries and codes can be executed by just invoking the needed libraries.
- Test Automation scripts are separated from input data store and only the input data gets manipulated whereas no modification is needed for the test scripts.
- By developing the libraries, they can be reused again and again, saving time for the entire organization/project team.
- Extensibility and maintenance becomes easy, as the re-usable library can be created as an enhanced feature. By giving right role-based access, the standard process of Test Automation scripting can be maintained.

Test Automation Framework Development Challenges:

Test Automation Framework development involves various challenges that include.

- Clear vision of what needs to be achieved out of this automation. It should address core questions like testing model, types of testing, which areas need to be automated etc.
- Tool identification and Recommendation process is a crucial process, as it means considering critical factors like creating a standard tool evaluation checklist, types of testing, and acquiring multiple tools to perform different types of testing.
- Framework design involves identifying requirements from multiple areas like identification of necessary utility/components, types of input data store to be communicated, communication between the systems and utility/component development, etc.

A Sensible Approach to Automation:

Pre-requisites and Assumptions

It is assumed that the user knows what test automation is all about and he has a planned approach for it. He has investigated "Test Automation Tool - Build or Buy" and has taken a decision of buying a tool or getting an open-source tool.

Step 1 - Identify Testing Scope

Considering the organization's requirements, test automation activities can be performed in three different scopes, which are Enterprise-oriented, Product-oriented, and Project-oriented.

Step 2 - Identify Testing Types

Based on the product/application/module requirement, type of testing that needs to be performed is identified. Priority must be assigned to each type of testing, based on the schedule for product release.

Step 3 – Identify Requirements to be automated

Each requirement has its own actions, validations for testing. All the identified requirements are assigned priority. This would help in identifying "Build-Verification Test (BVT)" requirements that should never fail.

Step 4- Evaluate Test Automation Tool

Identified testing types and requirements act as a base criterion for test automation tool evaluation.

- Checklist
- Identify Tools
- Sample Run
- Rate and Select Tools
- Implementation and Training

Step 5- Identify Requirements that can be automated

This study would result-up in coming with requirements that can be automated.

Step 6 - Design Test Automation Framework

For designing a framework, various elements need to be taken into consideration. Some of them are:

- Actions to be performed
- Database Communication
- Communication with additional automation tools
- Device Communication
- Log
- Error Handlers
- Custom Messages

Based on these, Test automation framework would be designed using the following guidelines like.

- Application-independent.
- Encapsulate the testers from the complexities of the test framework.
- Identify and abstract common functions used across multiple test scripts.
- Decouple complex business function testing from navigation, limit-testing, and other simple verification and validation activities.
- Structure scripts with minimal dependencies - Ensuring scripts executing unattended even on failures.

Step 7 – Design Data Input Store

Types of input data files supported by the tools need to be identified. They can be objects identifier; scenarios/Workflows/Transactions based input, custom message, and Driver. For all the files types, file format needs to be identified and prototyped based on the input data storage.

Step 8 - Develop framework

Framework development is facilitated using the same set of identified tools. Scripting language supported by the test automation tools is used to create the components. Tool extensibility utility/component can be developed using a different language. In addition to the re-usable components, driver scripts and worker scripts need to be created. The approach for developing re-usable utilities/components should include:

- Record/Replay
- Screen/Window/Transaction
- Action/Keyword
- Data Driven

Step 9 – Populate Input Data Store

Data can be populated either manually or in an automated fashion from different data-sources.

Step 10 - Configure Schedulers

Schedulers can be configured to run a worker script (batch script) on a specific time-period.

Key benefits of Test Automation Framework:

- Standard process in Production
- Free from dependencies
- Complete Coverage
- Future Enhancements Support
- Cost Estimation

Conclusion:

Automation is a great idea. To make it a good investment, the testing should adopt the automation in its framework very well. For the success of Test automation, a clear distinction between automation and the process of automation is necessary. The benefits of Test Automation Framework will fuel an interest across domains and find greater acceptance and importance in the industry.

About the Author

CSS Corporation provides [Test Automation Services](#) as a part of [Software Testing](#) and quality assurance lifecycle services. For more information, contact at marketing.css@csscorp.com.

Source: <http://www.tntarticles.com>