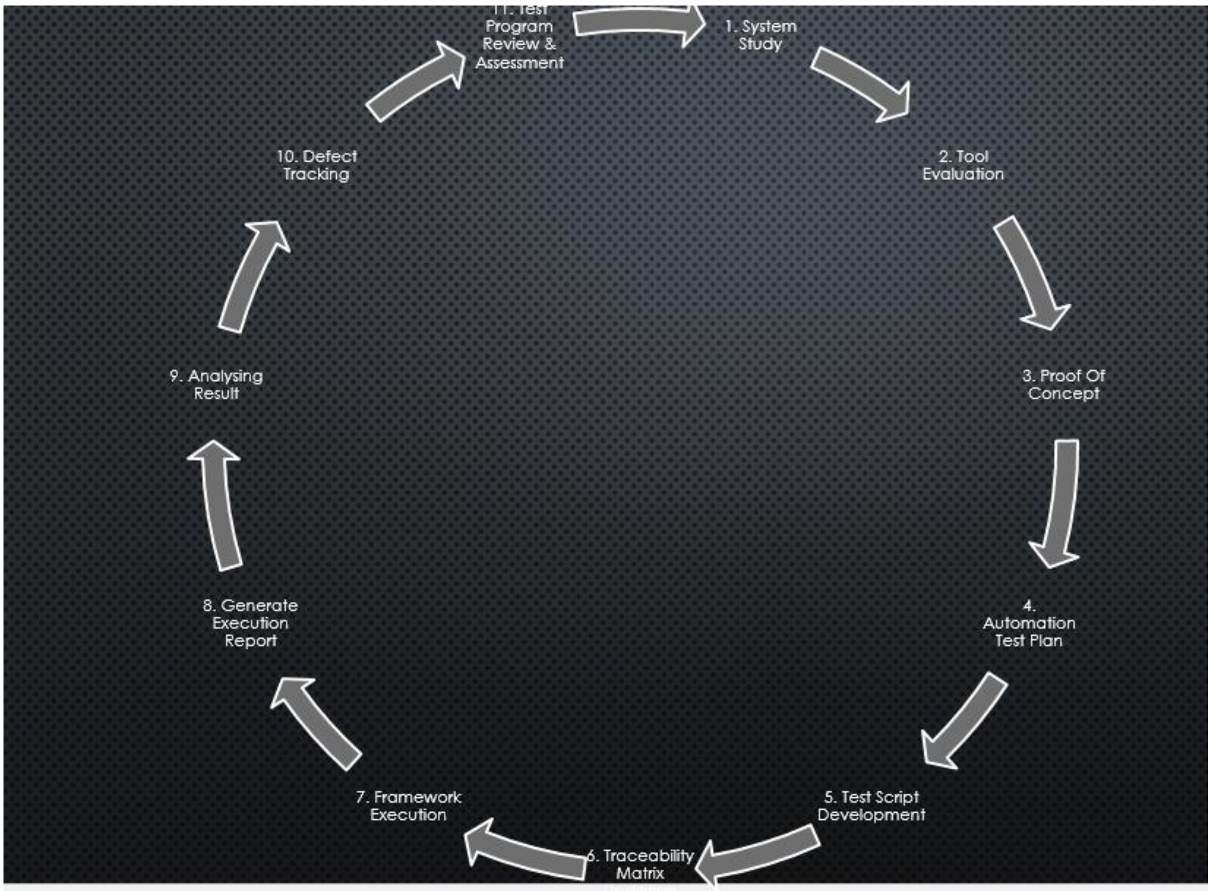
ATLC (automation test life cycle)

It is also called as a step by step process to automate the testing of the application.

It has following stages,



1. System study.
2. Tool Evaluation.
3. Proof of concept (POC).
4. Automation test plan.
5. Test script development.
6. Traceability matrix updating.
7. Framework execution.
8. Generate execution report.
9. Analysing the results.
10. Defect tracking.
11. Test program review and assessment.

System study

System study stage is a process of understanding the application thoroughly.

Why automation engineer needs to spend time in understanding the application?

When you go to a doctor, the doctor needs to have complete information about your past and existing health issues, old medical prescriptions, etc. Only then will the doctor be able to understand your new health issues and symptoms. The same is the case with the automation testing engineer. Automation testing engineers need to comprehend the software application under testing inside and out.

The common application details that every automation tester needs to take care of are:

- Which programming languages have been used while developing the application?
- On what platform is the application built?
- Which databases are involved?
- Are there any web services/APIs connected to different parts of the system? How?
- What is the objective of the owner developing this application?
- What are the functions and features expected from the application under development?
- Has manual testing been executed in this application?
- How many manual hours have been spent on this application testing?
- Are there any existing serious issues or unresolved bugs?
- What is the expected delivery date of the application?

These are just a few points and it may vary based on the complexity of the applications. So, make sure you are completely thorough with the application that you are going to test via automation testing.

Tool Evaluation

This is a process of selecting best suitable tool for the automation, this will be done along with the POC.

Create a matrix where we map the parameters against the tool, the toll which gets the highest point is the best suitable. It is only a suggestion; decision is always taken by the management. While giving the points, we should provide supportive document also, some points are on positive scale and some points are on negative scale, the parameters and the cost depends on the project.

Frequently considered parameters,

1. Cost
2. Supported languages
3. Supported technologies
4. Features
5. Integration with third party tool
6. Object identification compatibility

7. Reporting features
8. Debugging features
9. Supported environments
10. Supported browsers
11. Availability of the resources
12. Cost of the resources

Proof of concept

Proof of concept or POC is the stage where we look at the feasibility of the automation on the given project.

Automation Feasibility Analysis

Here the main objective is to find the feasibility of automation and every aspect should be taken into consideration while analysing. The following things should be taken care without fail.

- Which test can be automated and how it can be automated?
- Which module of the application can be automated and which cannot be?
- Which all are the tools that can be used for automation and which will be the best choice.
- Cost, team strength, and expertise also should be taken care.
- In order to find the feasibility, we automate some sample test cases and check whether automation is feasible or not.
- In order to do this, we approximately take 4% of the total test cases of different complexity and from different modules, we convert this sample test cases into automation script using the available automation framework. Sometime we need to design the framework also from the scratch.

Automation test plan

If automation is feasible for the project than we prepare the automation plan, most of the cases it will be part of manual test plan itself, (Test Automation). Or we may prepare separate automation plan document which contains following information.

i. Objective
ii. Scope
iii. Assumptions
iv. Risk and mitigation plan
v. Schedule
vi. Roles and responsibilities of each automation engineer
vii. Process guidelines, rules and best practice of automation
viii. Technical details
Such as, which automation tool, how to install it, plugins information, tool licence information,
automation framework configuration,

Script repository information,
Test case repository information,
Bug repository information,
Build repository information,
System and server related information,
Help documents etc.

Develop Script

This is the actual stage where we convert manual test cases into automation scripts.

In order to do this,

i. Read the test cases and execute it manually at least once.
ii. Develop libraries-methods-pom classes for repeating steps.
iii. Prepare test data if it is required.
iv. Write the scripts as per the test case steps.
v. Execute it locally and ensures that it is working fine (self review).
vi. Send it for peer review.
vii. Fix the review comment.
viii. Send it to approval.
ix. Store it in script repository after the approval after base lining.

Traceability matrix updating

Update the traceability matrix. Traceability matrix is a document where we map the test cases to the requirements to ensure that all the requirements are covered.

Framework execution

We execute all the scripts present in the framework when we get a new build. The new build will be installed only in the lab machine and not in the local machine as part of mitigation plan.

Sometimes the framework is integrated with continuous integration tool such as Jenkins in such cases it will only execute the frameworks automatically in the lab machine after installing the build.

Generate execution report

After executing the framework, it will generate the execution report automatically, which contains list of test scripts with execution status passed or failed.

Analysing the results

Analysing is a stage where we take an execution report and check the status of the executed test cases. We should always give importance for failed scripts, whenever the scripts is failed, we take the respective test case and execute it manually.

If manual test case itself failed, it indicates there is a bug in the application, in such cases we submit the bug. If the manual test case is passed it indicates the bug is in the script. In such cases we debug the scripts.

After analysing all the test scripts, we prepare the test execution report.

Defect tracking

Defect tracking is the process of tracking the logged defects in a product from beginning to closure, and making new versions of the product that fix the defects.

Test program review and assessment

Test program review and assessment activities need to be conducted throughout the automation testing lifecycle, to allow for continuous improvement activities.

Throughout the testing lifecycle and following test execution activities, metrics need to be evaluated and final review and assessment activities need to be conducted to allow for process improvement.

The various steps necessary for test program review and assessment are outlined below.

- Following test execution, the test team needs to review the performance of the test program to determine where changes can be implemented to improve the test program performance on the next project. This test program review represents the final phase of the Automated Test Lifecycle Methodology (ATLM).
- Throughout the test program, the test team collected various test metrics. The focus of the test program review includes an assessment of whether the application satisfies acceptance criteria and is ready to go into production. The review also includes an evaluation of earned value progress measurements and other metrics collected.
- As part of its culture, the test team needs to adopt an ongoing iterative process of *lessons learned* activities. Such a program encourages test engineers to take the responsibility to raise corrective action proposals immediately, when such actions potentially have significant impact on test program performance. Throughout the entire test lifecycle, it's good practice to document and begin to evaluate lessons learned at each milestone. The metrics that are collected throughout the test lifecycle and especially during the test execution phase help pinpoint problems that need to be addressed.
- Lessons learned, metrics evaluations, and corresponding improvement activity or corrective action need to be documented throughout the entire test process in a central repository that's easily accessible.

- After collecting lessons learned and other metrics, and defining corrective actions, test engineers also need to assess the effectiveness of the test program to include an evaluation of the *test program return on investment*. Test engineers capture measures of the benefits of automation realized throughout the test lifecycle in order to support this assessment.

- Test teams can perform their own surveys to inquire about the potential value of process and tool changes. A survey form can be used to solicit feedback on the potential use of requirement-management tools, design tools, and development tools. Surveys are helpful to identify potential misconceptions and gather positive feedback.

Test Automation Strategies

One of the best recommendations is adopting Agile and DevOps with a focus on automation and shift-left testing.

What is shift-left testing?

Shift-left testing is an approach to software testing and system testing in which testing is performed earlier in the lifecycle. It is the first half of the maxim "Test early and often."

Now that we have discussed how automation can accelerate the best practices of software testing, let's discuss the best of all test automation strategy to look for in the coming years.

Automation Planning

Firstly, we need to plan out the test automation processes. Previously, it was next to impossible, but now, you can think of automating the complete application with the proper test automation planning. This includes estimation, project deadline, test schedules, automation environments, and other planning-related activities.

Automation Architecture and Framework Creation

Scripting knowledge is very much important to automate any application testing, but working on selecting the appropriate automation framework becomes mandatory. We need to be prepared with well-designed automation architecture and integration with the various third party tools for CI, CD, and more, which will help us to execute automation testing. We need to write reusable components and methods.

Automation Test Case Creation and Execution

Once we have decided on the test automation framework, we can start with creating actual automation test cases. Also, priorities can be set to check on the important test cases compared to other test cases based on project complexities. Once test cases are developed, they are taken up for execution on various browsers and platforms to check its stability.

Automation Tests Maintenance and Monitoring

Reporting is a crucial factor in test automation, but reporting via test automation tools can be quite useful. We need to verify the execution logs and reports. Defects are logged into the bug tracking tools along with the screenshots. So, after this, we need to enhance our test cases and test automation frameworks in this phase.

Final Thoughts

We have always known the importance of software testing, but, we had never thought of optimizing our traditional software testing practices. Enterprises have now started investing into software testing services and test automation. We need to go back and give a deep thought on how we can accelerate our software testing processes with the help of automation testing.