**Feasibility Analysis In Software Test Automation**

In this blog we are going to discuss about the software test automation lifecycle followed by the feasibility analysis that is required. The feasibility study is must before taking any action with the automation since it is the deciding factor for automation sign-off based on the value that is going to be added to the business.

**Life cycle of a Software Test Automation**

Life cycle denotes the subsequent actions in a timely fashion. Below are the lists of things to be considered for test automation life cycle

* Feasibility study
* Test strategy or plan documentation
* Test scripts design
* Test cases execution and results analysis
* Types of Test Automation

**Feasibility analysis & its significance**

As discussed at the start, feasibility study is something without which we can’t judge the outcome of automation efforts. This study helps the test team in forecast on following activities.

* Can the feature be automated or not?
* What type of tool/ test automation framework can be used?
* How much automation is possible?
* In spite of high efforts, what’s the value add in automating?

Below are some key parameters while performing feasibility study, lets look into each one of them in brief

**Feasibility study in tool selection**

We must conduct a careful study before finalizing the tool. The key points to be considered are

Whether or not the tool can detect all the objects of an application

There should be a POC conducted with all the list of proposed tools to automate the applications then decide which one should be of good match by considering the application development technologies and its compatibility with the tool.

Eg: while selenium is used to automate

Also the tool should be able to store

Investment on tool

**Careful study of user story and understand the functional flow**

Before we proceed with automating any functional story, the automation tester must possess the knowledge of the application as well the functionality that is significantly delivered in that user story. Without having this understanding when any test scenario is considered for automation and we may get into several problems and the task either would be left incomplete or likely delayed.

As an automation tester the focus would be to gain the sufficient functional knowledge, get the test cases from the functional (manual) team then conduct feasibility analysis as to know how many scenarios can be automated. If at all some cases can’t be automated there must be a valid rationale and the agreement with the stakeholders must be taken.

The analysis helps the test team to showcase their deliverables, because after thorough examination team will be in a position to understand what will be covered in automation and how long they require to complete them. Failing to conduct this study could lead to irksome state.

**Proper infrastructure provision**

In order to proceed with hassle free automation the test team must not have any issues at infra level, the possible issues that might fall under this category are

* Not having access to create project/ branch in VCS
* Not having access to the application/ DB servers to work independently
* Less software license procurement by the business
* Sluggish network, which will delay the execution and cause failures at times
* Not holding admin rights to perform certain tasks

The above listed problems will prohibit automation activities from point to point. Test team must raise the access requests to all the needed things in the project and gain access.

**Stable environment and major part of development should be completed**

As a test team we should know when to automate a particular application. If the build quality is at an infancy state, we see the basic smoke tests are failing then there is no point in automating the test cases at that stage.

When we go for automation when the major development is in progress, we end up adding lot of maintenance to the test suite as the UI locator elements might get changed often or at times there are some changes expected at functional level, then we may need to re-write the scripts as per new design which is tedious.

**Assessing how much automation can be achieved**

100% automation is fallacy! Given the complexities whether or not they are technical or no-technical that may occur during automation development.

As an automation tester doing this exercise helps the business understand the index of cost and effort saving. Ideally this is achieved by comparing the functional test cases with the automation test cases. This metric defines what percentage of the tests have been automated and accordingly the savings were understood.

More the automation coverage, more the regression coverage as they are directly proportional to each other. Eventually when the regression coverage is more we can believe that sufficient amount of test is conducted and no chances of bug leaks in the market often times.

**Calculating the ROI metrics**

Metrics are key numbers, which will denote the status, progress, outcome in various ways. As a test engineer/ development engineer when we bother about accomplishment of tasks, but the management always look at the numbers extracted as metrics. This is one short and explainable way of presenting data to understand where we are? What have we achieved?

In the same way, in automation world we have certain metrics to be calculated to project the savings to the business, following are couple of quick examples

**Automation savings**

This is to understand how much time in man hours and effort saved because of bringing in automation

TSA=TME-TAE

Where TS= time saved due to automation

TME- time taken for manual execution

TE= time taken for automation execution

**Return on investments**

Certainly we are investing during automation in the means of script development, managing talents and the cost of procuring the infrastructure that’s needed.

Cost of automation investments= cost of talents +cost of script development and maintenance+ cost of infrastructure.

In acronymic form

CAI=CT+CSDM+CI

So, the final ROI would be as below

ROI= Benefits/ Investments

Or

savings achieved through automation over manual testing

ROI= --------------------------------------------------------------------------------

Investment done in automation over manual efforts

**Conclusion**

More the ROI, more the profits and that is what business needed as well. In order to achieve better ROI we should focus on below

* Try to bring in more automation
* Focus of in sprint automation development
* Right choice of tools, which will incur no or as minimal budget as possible
* Develop a generic framework with more reusable components, to avoid extra and often maintenance