|  |  |
| --- | --- |
| D:\PAL\Logo_4_21_15\Logo_4_21_15\Primary Logo\png_files\EPAM_LOGO_Full_Color_RGB.png | CDP Automated Testing Mentoring Program  2018-Q2 |

Module 1: Introduction to test automation

## home task

As a basis for this home task, take SUT (system under test) of your current project.

1. Think about things below:

* Is it necessary to set up test automation processes for this SUT? Why?
* What should/could be automated for this SUT? Why? How?

Write a short essay answering the questions above.

## Answer

My current project is a long-term project (more than 10 years). For all existing time it used several platforms. During each phase automation played important role. In common there are several production environments on my project. Each environment is used by several customers. And there a lot customers that use common functionality so customer specific features.

Currently automation tests contain smoke test and tests for specific features. Normal flow looks like release with couple of new features for each production environment. Automation tests are created for each feature. Then end-to-end scenarios for new feature are added to smoke test. Before each release smoke test has been run.

Also there is one customer that has automated almost all features. So regression testing is fully automated. It is divided into several sets by the most important functionality parts. Recently regression set was run quite often (almost every day) that allowed find issues very quickly, analyze what caused them and fix. This set includes over 1000 tests and manual testing would take several weeks to test for 1 manual tester. Automation tests goes through all tests for 1 night (up to 12 hours). And it really saves time. So when new feature is created the most priority tests are automated and added to regression. Then all set is run. Automated testers analyze results and check that both new tests and old tests are passed. In case new feature requires changes in old cases automation tester makes corresponding changes. So just on this example we can see that automation is suitable for my project:

* Project is long-term
* Tests are executed frequently
* It will be costly to test application manually
* It constantly improves quality of application as regular run of automation tests finds issues faster than irregular manual testing
* It reduces costs of testing by reducing manual labour
* It’s our customer’s requirement

So in our case we save time and money by frequently running automation tests and as a result quality is improved

The following tests should be automated on our project:

* **Smoke test.** As this type of testing covers main functionality of the application so running automation set for smoke test will save time when new feature is implemented and it’s important to make sure that it didn’t affect existing code.
* **Regression testing.** Before each big release we have a requirement to do regression testing. As I mentioned previously we have different production environments and for one of pool there is fully automated regression set. It allows to find any issues in existing features and of course it leads to continuous quality improvement. There is no such option (regression tests covering) for other pools. So manual testers should do testing instead. It takes 6-7 working days in average to do regression testing manually for one tester on 1 environment. And here can also be some risks. For example, when one tester performs the same tests frequently he can face human error. He can miss something important. In case of automation testing there will not be human error, only environment. It doesn’t mean that manual testers should not do any kind of regression testing. Of course, they should. Our application is used in different environments. For example, cross-browser testing is important. Some UI testing also can be done only by manual testing with quality, etc.
* **End-to-end scenarios for new features.** When new feature is implemented we need to create end-to-end test that will cover main aspects of the feature. It will be helpful to run such cases when new feature will be popular in using and time for manual testing is big. Also such scenarios can be included in regression set.
* **Performance testing (stress and load tests).** Such kind of testing is impossible to perform manually. Automation testing can do that. It allows to estimate how system response to high loading and gives an answer about how many simultaneous user can work with application without notable performance reducing.

Generally speaking I would advise to automate:

* Repetitive tests that run for multiple builds;
* Tests that tend to cause human error;
* Frequently used functionality that introduces high risk conditions.
* Tests that are impossible to perform manually.
* Tests that take a lot of effort and time when manual testing.

Of course there are always present cases that can’t be automated. But the amount of such cases is very small. Attempt to automate such scenarios can take very long time and not reasonable.

New tendency required to switch our project to the new platform. Automation testing will play crucial role there. As almost all functionality is going to be automated.

1. Define and collect information and criteria for calculating ROI (Return of Investments) value for this SUT. Provide rationale of your calculation with description on each step.

## Answer

As a formula for ROI I used simplified one (that is based on man-hours approach) as:

,

CM – cost of manual testing (man-hours)

I – investments into automation (man-hours)

,

FW – time spent for implementing framework;

S – time spent for creating TA scenarios;

E – time spent for tests execution (human job);

R – time spent for results analyzing.

As a basis I took my project. But counted ROI for just 1 customer that have fully automated regression tests as other customers don’t have such option. They have just some features automated.

I took period of approximately 7 years when our team started to work with customer. In the beginning there were frequent releases during the year with plenty of features. For last 2 years number of releases and features reduced so I took average number of features release per release. Let’s say there are 4 releases per year with 3 features of middle complexity. Then we got next figures:

* Cost of manual testing – 24 hours per week, 52 weeks per year, 7 years. **It means: 24 \* 52 \* 7 = 8736**. In this time I included estimations for requirements analysis and clarification, creating check-lists and test cases, performing smoke test for the features, clarifying questions with business analysts, developers and test automation testers and also demo sessions with customer. The most time-consuming part is creating well-documented test cases so automation testers can easily read it and go through each scenario
* Time spent for creating TA scenarios – 12 hours per week, 26 weeks per year, 7 years. **It means: 12 \* 26 \* 7 = 2184.** Usually automation testers are not fully busy during release. When new feature is developing testers can not create scenarios. At the same time they can execute and analyze results for previous releases.
* Time spent for test execution and analyzing. 8 hours per week, 52 weeks per year, 7 years. **It means: 8 \* 52 \* 7 = 2912**. Usually tests were executed at night so most time was spent on result analysis, reporting found issues and fixing failed scenarios if it was issue in the test structure. It took 8 hours per week in average.
* Time spent for implementing framework. **100 hours**. Some kind of framework already existed when our automation team started to automate test scenarios. I added this figure as some additional improvements were added since that time.

To summarize:

Cost of manual testing = **8736**

Investments into automation = 2184 + 2912 + 100 = **5196**

**ROI = (8736 - 5196) / 5196 = 68.13%**

Analyzing ROI result (68.13%) we can say that it is very good figure. It is profitable to invest into automation for such period (7 years.)