Task #1. Determine the payback period based on cost of the manual and automated testing.

The mail goal of automation on my project is to automate user acceptance tests for 170 brand sites of Johnson and Johnson Company. Now these sites are porting to the one platform and it is planned that this platform will be updated once in a 2 weeks. The main reason of automation on our project is to cover all the main functionalities with scripts to run them automatically every time platform updates. We use cross-platform testing (desktop, mobile and tablet devices).

On the average, it takes 40 man-hours to test 1 site manually on one of the devices. On the other hand, usually it takes 120 man-hours to automate these tests using Gherkin scripts. Our approach allows us to run the same tests for all the devices mentioned above. Now we are using new approach to reduce time on writing scripts for every site, so in future this number will be something like 100 man-hours.

We have 170 sites to automate, so if we’d like to **test it manually** it would be:

**170 sites \* 40 man-hours per site \* 3 devices = 20 400 man-hours**

This number for **automation** would be:

**170 sites \* 100 man-hours per site = 17000 man-hours**

Our automation **framework** is custom, so it took 6 month for team of 10 person to set up a framework:

**40 man-hours per week \* 10 members \* 24 weeks = 9600 man-hours**

However, we are still maintaining and upgrading the framework, let’s say we are spending 100 man-hours a week to **maintain the framework** and it will last till the end of the project (e.g. 10 years):

**100 man-hours per week \* 4 weeks \* 12 month \* 10 years = 48000 man-hours**

Every tests run for each site continue at average for **3.5 hours**. Tests will run from Jenkins automatically and detailed report will be generating automatically so there will be no time spent on result analysis. Our tests will only decide broke the build or not. One of the main goals of our tests is ability to run it in parallel. We don’t have to wait for one suit to be totally complete to run the new one.

Let’s say that after writing all the tests for all the sites, the platform will be maintained for 10 years. **Every 2 weeks there will be build** of the platform that should be tested.

**Time for manual testing**:

**40 man-hours \* 3 devices \* 4 weeks \* 12 month \* 10 years \* 170 sites = 9 792 000 man-hours**

**Time for automated testing:**

The tests will be running automatically, there will be no need in root cause analysis. We will only support our framework and test scripts. So the only effort that will be spent is **48 000 man-hours for the entire project.**

So on my project ROI formula will look like:

**ROI = (Cost of manual testing – (Framework set up + Scenarios Development + Maintenance)) / (Framework set up + Scenarios Development + Maintenance)**

ROI = (9792000 – (9600 + 17000 + 48000)) / (9600 + 17000 + 48000)

**ROI = (9792000 – 74600) / 74600 = 130%**

All in all, I think that automation on our project is totally economically justified, because supporting of the sites would be too expensive. There should be enormous amount of people in a testing team to test all the sites every two weeks manually, especially on several devices, even if the speed of work increases twice. In addition, the great plus is ability to run all the tests in parallel.

Now we are trying to make 100% coverage. This means that sometimes we have automate testing content of each site. Maintenance of these tests will cost a lot of money for the customer, so I think that it would be better to choose something in the middle: trying to automate the main functionalities of the sites and keep manual testing for GUI testing and content checking.