Let’s analyze Elateral project where Automation is already in place. It’s the project with big history. At that moment there are two active different systems. I’ll review MSC system where there were lots of implementation already done and different approaches applied (from testing and development perspective). It’s some kind of excursion to evaluate automation on the project with average numbers used because the approach and workload are changing continuously.

So why automation was really needed on the project, what was automated and how?

The project is a quite difficult with lots of QA environments, a couple of UAT and Prod environments and lots of testing. It’s **long-term** project that has recently celebrated 5th year birthday. It’s quite **stable system**, with lots of functionalities and **repeatable tests** so Automation was really needed for project *to save money*, *to save time*, *to focus manual testing more on exploratory testing* as this type of testing defines quite tricky issues and finally improves the product.  
Some parts of functionalities could be **tested only by Automation** tests (e.g. API part, Performance tests) and UI parts are under automation as well.

Taking in account the points above it looks that appropriate Automation can really increase testing efficiency. Let’s dive more into the details and get numbers for UI automation (smoke and end to end scenarios).

Tests are running on TeamCity on servers that are used together with developer team. Mainly tests are executed at night so there are no conflicts with their usage so we can ignore these type of costs.

The formula to calculate ROI is the next:

**ROI = (Gain from Investment - Cost of Investment) / Cost of Investment**

where: Gain from Investment – cost of manual

Cost of Investment – cost of automation

Smoke tests are executed on QA and UAT environments quite often (12 times per month – average count). It’s minimum number of executions but with Automation it can be performed on daily basis for each deployment as part of CI. No doubts ROI for Smoke part is quite well. Let’s calculate ROI for Smoke tests, include here Framework Setup even for whole application, tests implementation and update, support, tests execution and results analysis. Below is ROI calculation for 1 year.

Smoke tests analysis (1 year)

|  |  |
| --- | --- |
| **Project duration** | **12 months** |
| **Average manual testing cost** | **10 m/h per Smoke test** |
| **Framework Setup + CI configuration** | **120 m/h** |
| **Smoke tests implementation and its update** | **28 m/h per month (average)** |
| **Tests Support** | **12 m/h per month (average)** |
| **Automated tests execution & result analysis** | **24 m/h per month (average)** |

Cost of manual = 10 \* 12 \* 12 = 1440 man hours.

Cost of Automation = 120 + 28 \* 12 + 12 \* 12 + 24 \* 12 = 888 man hours.

ROI = (1440 – 888) / 888 = 62%

We need to adjust this value as automation man hour = 1.3 \* manual man hour (in average). So finally we have:

ROI = (1440 – 888 \* 1.3) / (888 \* 1.3) = 25%

So we see that’s ROI is quite fine even after 1 year and it will grow more in the next years (as there will be no need to setup the framework and configure CI again).

Once the framework was built and smoke tests are automated the next step was to automate end to end scenarios. Those scenarios are not executed so often (as smoke scenarios) but in big time line it should have quite well ROI on the project. End-to-end scenarios are executed 2 times per month (average). Below is ROI calculation for 2 years.

End-to-end tests (2 years)

|  |  |
| --- | --- |
| **Project duration** | **24 months** |
| **Manual end-to-end tests** | **64 m/h per one execution** |
| **Framework Setup + CI configuration** | **Included into Smoke tests part** |
| **End-to-end tests implementation and its update** | **48 m/h per month (average)** |
| **Tests Support** | **24 m/h per month (average)** |
| **Automated test execution & result analysis** | **8 m/h per month (average)** |

Cost of manual = 2 \* 24 \* 64 = 3072 man hours.

Cost of Automation = 48 \* 24 + 24 \* 24 + 24 \* 8 = 1920 auto man hours.

ROI = (3072 – 1920\*1.3) / (1920\*1.3) = 23%

If look at the formula it’s obvious that ROI will be the same for any time-frame because framework setup and CI configuration was included into Smoke tests ROI calculation above.

To keep ROI at high level the team need to have right priorities and not try to automate all possible scenarios but do the progress accordingly:

1. Automate and support firstly more frequent scenarios;
2. Consider expensive scenarios for manual testing;
3. Involve into tests execution and analysis manual testers (to reduce costs).

To sum-up the project really needed automation process as it finally it saved money, improved time to market time, automated API and Performances tests.