1. The AT (automation test) are necessary for SUT because of the big regression scope should be run time to time. We have SUT build release scheduled for every mouth, so the tests runs each time, and some hotfix verification (occurs every week) needs to be verified as well. Therefore, average number of the runs is 5 per month.

All the new features, coming to the site should be automated due to long-term maintenance and huge number of runs.

It requires at least two parts:  
-smoke  
-regression   
  
All the runs should be performed due to needs of testing (integrity, regression, access and other reasons based on build changes). If we talk about SUT I am working with – the tests verify to check integrity, regression, UI changes, and some metadata (e.g. references types).

The automation tests brings us time and money saving. Due to once created the automation test can be run over and over again without any additional human involvement (just analyzing is needed).

Let us calculate the ROI for the SUT (system under test) I am working on, using the formula:

,

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.

# SUT calculation

According to the history in VS VCS , the TAF (test automation framework) was created in February, 2016. Let us analyze part of this one – regression pack (2 parts). We assume the 1 week (40 hours) was needed to create the framework and 3 months to create tests for (168\*3 hours).  
Either of parts of regression keeps 4 hours to be executed. Tests configuring in CI keeps not so much time – e.g. 3 hours per month, or 54 hours for framework from the creation. Each build keeps about 1.5 hours for 3 testers to be analyzed 1.5\*3\*388(builds)=1168 hours.

Also, let us improve formula with T – time to spent for framework supporting, which is about 8 hours per month for SUT

Cm for each build is about 10 hours. And for the all builds 388\*10 =3880 hours  
  
From the  we can calculate ROI = 1.03 (at the moment). And it will increase due to annual work items were done.

ROI = (3880 – (1168+54+504+ 40 + 144))/ (1168+54+504+ 40 + 144)=1.03

3880  
FW = 40 h

S = 168\*3=504 h

E = 3 h \*18 months = 54 hours

R = 1.5h \* 3 testers \*388 builds = 1168 hours

T = 8 h \* 18 months = 144 hours

I = FW + S + E + R + T

I = 1910

So the TAF brings positive effort for the SUT.

Due to our SUT is multi-project (contains few subprojects), let us look into the one of subproject, which is just started being live. And let us see the smoke part of it.

In order to see the trend of automation costs increasing / saving in timeline, let’s calculate ROI for two different points:

Current time.

Future (number of the test execution = 300)

It will give us some vision to make a clear conclusion.

# Subproject calculation

The smoke part includes 13 tests, each of it kept about 4 hours to be created. Also average time to execute one of them is 3 mins (0.05 hours). At the moment we have 66 job executions for them. Average time to analyze the build is 20 mins.

ROI calculated for the moment   
ROI = (42.9-83)/83 = -0.48  
FW = 8 h  
S = 4h\*13 = 52 h  
E = 1h   
R = 1/3 h \* 66 = 22h  
I = FW + S + E + R   
I = 8+52+1+22=83 h  
Cm = 0.05\*13\*66= 42.9 h

Let us assume the same scope will be run 300 times (about 1.5 year performing)

ROI = (195-165)/165=0.18

FW = 8 h

S = 4h\*13 = 52 h

E = 5h

R = 1/3 h \* 300 = 100h

I = FW + S + E + R

I = 8+52+5+100=165 h

Cm = 0.05\*13\*300= 195h

And one more calculation – let us find the number of the execution which brings us the first profit

X = number of the execution

FW = 8 h

S = 4h\*13 = 52 h

E = X\*(1/60)h

R = 1/3 h \* X= 100h

I = FW + S + E + R

Cm = 0.05\*13\*X

ROI = ((0.05\*13\*x – (x/3+x/60+52+8))/ – (x/3+x/60+52+8)

X is about 200

So the ROI shows the automation of the testing process value can be negative in the start, but gives a profit with test execution quantity increase.

1. Automated calculation

using <http://www-01.ibm.com/software/rational/offerings/testing/roi/tool/ROI_Rational.html> (costs are randomly assumed)  
