Cost Benefits Analysis of Test Automation

Douglas Hoffman
Software Quality Methods, LLC.
24646 Heather Heights Place
Saratoga, California 95070-9710
Phone 408-741-4830
Fax 408-867-4550
doug.hoffman@acm.org

Keywords: Automated Testing, Automation Tools, Cost of Testing, Intangible Costs, Return on Investment, Tangible Costs

Introduction

Many managers today expect software test automation to be a silver bullet; killing the problems of test scheduling, the costs of testing, defect reporting, and more. Automating testing can have positive impacts in many areas, and there are many success stories to provide hope that test automation will save money and solve some testing problems. Unfortunately, there are many more horror stories, disappointments, and bad feelings, even in cases where automation has been beneficial. I have been brought into more than one situation where previous attempts at automating software testing have failed; where large investments have been made in shelfware, and many years of effort creating automated tests abandoned. The purpose of this paper is to provide some practical guidance for understanding and computing cost and benefits from test automation. It describes some financial, organizational, and test effectiveness impacts observed when software test automation is installed. The paper also advises about areas that are difficult or impossible to factor into the financial equations and addresses some common misconceptions management holds about test automation.

There are many factors to consider when planning for software test automation. Automation changes the complexion of testing and the test organization from design through implementation and test execution. It usually has broad impacts on the organization in such things as the tasks performed, test approaches, and even product features. There are tangible and intangible elements and widely held myths about benefits and capabilities of test automation. It is important to really understand the potential costs and benefits before undertaking the kind of change automation implies if only to plan well to make the most of them.

Organizational impacts include such things as the skills needed to design and implement automated tests, automation tools, and automation environments. Development and maintenance of automated tests is quite different from manual tests. The job skills change, test approaches change, and testing itself changes when automation is installed. Automation has the potential for changing the product being tested and the processes used for development and release. These impacts have positive and negative components that must be considered.

Setting realistic expectations in management and understanding where benefits should be derived from test automation are key to success. We can easily provide cost justification for proposed automation if management demands numbers. Tool vendors and experts publishing their test automation strategies provide excellent sources of equations and customer examples justifying almost any approach. In my experience the trick has been to figure out what costs and benefits really relate to the automation at hand, and how to make best use of them. It is critical to keep in mind that the purpose of test automation is to do testing better in some way. Automation is only a means to help accomplish our task – testing a product. The automation itself generally does not otherwise benefit the organization any more than the testing does. Cost benefit analysis provides us with useful information for deciding how to best manage and invest in testing.

There are also many automation areas that have the potential to provide a benefit or be a drawback depending on how they are handled. For example, automated tests may reduce staff involvement during testing, thus saving in relation to manually running the same tests. But, automated tests may generate mountains of results that can take much more staff involvement for analysis, thus costing more to run than manual tests. Often the information obtained from automated tests is more cryptic and takes longer to analyze and isolate when faults are discovered.

Existing metrics techniques such as code coverage can be used to estimate or compute test effectiveness before and after automation. Automated tests can be incredibly effective, giving more coverage and new visibility into the software under test. It also provides us with opportunities for testing in ways impractical or impossible for manual testing, yet conventional metrics may not show any improvements. Automated tests can generate millions of events and sequences limited only by the machine power and time available for running the tests. These tests can find defects in code already 100% covered. Employing random numbers allows sampling of events and sequences, and also allows tests to do new things every time they are run. Automated probes can look inside the product being tested at such things as intermediate results, memory contents, and internal program states to determine if the product is behaving as expected.

Some testing can only be accomplished through software test automation. Computing the cost benefit of a simulated load from a thousand users isn't often necessary. If you need to measure what the system does under such a load automation may be the only other practical way. Testing for memory leaks, checking program internal state variables, or monitoring tests for unexpected system calls can only be done using software automation.

This paper breaks down the many factors and provides examples of how the elements may be viewed, analyzed, utilized, and measured. The goal is to dispel some of the myths about test automation and provide a practical approach to deciding what types of investments are worthwhile.

Management Perspective

There are several areas in which we should set management expectations: intangible costs and benefits, falsely expected benefits, factors common to manual and automated testing, and organizational impacts. We also need to be careful about how we measure and account for things.

Intangible costs are difficult to realistically assess. Where they can be measured, there is great variation in the financial value we place on them. There are also issues in capturing values to measure how much change is due to automation. Some of the intangibles are generally positive, some negative, but most can be mixed blessings, depending on one's perspective and how they are done. Due to the difficulties attaching objective values to these factors, they are best left out of the ROI computation in most cases.

Some intangibles:

- Hands-off testing. *Although the cost of people is easily measurable, any additional value of computer control is difficult to quantify.*
- Improved professionalism of test organization. *This often increases motivation and productivity, and comes with the new discipline and tasks automation requires.*
- Expansion into advanced test issues. *This can occur because of the new abilities to test and monitor that are a consequence of automation with the improved professionalism.*
- An immediate reduction in perceived productivity of the test organization. *This perception is due to a pause in testing while people ramp up, the lag due to installation of automation tools, and creation of automated tests.*
- Not all members of the test team will want to change. *Some turnover in personnel often accompanies test automation even when some testers continue to test manually.*
- The changes in the quality of tests. *The quality may improve or get worse, but manual and automated tests almost always are different exercises.*
- Numbers of product rolls (test cycles) before release. *Automation often allows faster confirmation of product builds and may encourage more turns. The churning may improve productivity and quality or possibly may cause laziness, lack of attention, and degradation of quality.*
- Test coverage. *It may improve or lessen, depending on the effectiveness of manual testing, the automated tools, and the automated tests.*
  - some testing that can only be done through automation
  - the value of changes in test coverage are difficult to quantify even though we can compute many objective measures of coverage
  - good exploratory testing may cover many more interesting cases than mundane automation
  - manual testing may encourage situations difficult to manage automatically

Management expectations have often been set through the media, vendor hype, conferences, and books extolling the virtues of test automation. Some of the information is quite valid and applicable, but much of it under emphasizes the special circumstances and specific considerations that apply to the projects described and over emphasizes the successes. Test automation is not a silver bullet. It does not solve all testing problems and has to be carefully planned to be even marginally successful. Poorly set expectations can result in a beneficial automation effort getting labeled as a failure.

Some falsely expected benefits[1]:

- All tests will be automated. *This isn't practical or desirable.*
- There will be immediate pay back from automation. *An immediate payback may be seen for some automation (e.g., build tests), but usually the pay back comes much later after*

[1] Kaner, C. (1997) "Improving the Maintainability of Automated Test Suites," *Quality Week 1997*
Kaner, C. (1998) "Avoiding Shelfware: A Manager's View of Automated GUI Testing"

the investment. It takes a lot of work to create most automated tests and the savings usually comes from the running and rerunning after they are created.

- Automation of existing manual tests. *The machine can compound our errors thousands of times faster than we can, and computers don't care that their results are ridiculous or useless.*
- Zero ramp up time. *Automating tests takes time. The tools have to be selected, built, installed, and integrated before testing can take place, and planning and implementing automated tests often take many times the effort of equivalent manual tests.*
- Automated comprehensive test planning. *Automated tools that analyze requirements, specifications, or code don't do everything. They often make significant assumptions and the assumptions should be validated or compensated for. Showing that 'the code does what it does' is not the same as showing that it works per the specification (or per the requirements, or that it meets the users' needs). Automated analysis often misses significant classes of possible errors.*
- Use of capture/playback for regression testing. *This only works in the rare case when the product is so stable that there is very little expectation that any existing tests will need change in the future.*
- One tool that fits perfectly. *Many commercial automation tools are worth the cost and pay back very well. The commercially available tools may fit some organization and project perfectly – it's never been one I've worked on or run into.*
- Automatic defect reporting (without human intervention). *This often is disastrous for the testing organization and development. Problems include duplicate reports, false detection of errors, error cascades (one error triggers many test failures), and unreproducible errors, among others. Even with human review of the reports this feature of some automation tools can require far more effort than it saves.*

It is worthwhile noting a few additional things for management regarding the cost of automation:

- Much of the benefit from test automation comes from the discipline applied to analysis of the software under test and planning of testing. *These benefits are unrelated to automation and would be beneficial for manual testing as well.*
- Introducing test automation often causes an improved professionalism within the test organization. *This leads to better testing, closer relationships with development, and expansion into more advanced areas of software testing.*
- Often the pay back for the investment in automation does not pay off in the current project, and is usually seen in the **next** project that uses it. *This is often many months or years after the investment is made.*
- There are significant negative schedule and performance impacts in testing when automation is introduced.
- Writing of automated tests is more difficult than manual tests and requires a superset of knowledge and experience over manual testing. *Not all members of an existing test group are able to make such changes.*
- Automated tests frequently require substantial maintenance, especially for cheap (quick and dirty) techniques such as capture/playback. *In some cases automated tests may have an expected "life" of only one or two test runs, yet cost three to ten times as much to create as equivalent manual tests.*

- Great care must be taken to gather unbiased figures and maintain balance if we want to use the measures to understand how we're doing. *"Figures don't lie" doesn't apply to what we refer to as software metrics; we can prove any point we want to (and often delude ourselves by doing so).*

Where financial values can be compared there are simple ways to decide about automation. More difficulty is encountered when comparing organizational changes, and comparing test effectiveness is confounded by intangible factors. With of all of the confusing measures, questionable benefits, and unknown factors it is often better to make first order approximations rather than try to measure carefully. The first order approximations can often be measured easily. It isn't really useful to compute ratios of numbers to five decimal places that are approximations, estimations, guesses, or exact measures of factors totally confused by technology in any case. In many cases we have difficulty knowing what value to use even when we have complete information. (Is a test worth five times more because it is run on five systems instead of one? Is it worth five times more because we run it on one system five times? On a system that is five times faster?)

ROI Factors

The return on investment (ROI) is usually computed as the benefits derived divided by the investments made for a given thing. If we are starting a fresh project, we might compute the value of testing and divide by the cost of the testing to compute the return. Sometimes automation is being introduced after manual testing has been done for a while. In this case the cost benefits from automation can be viewed as trade-offs in comparison to manual testing (or by comparing it to the current testing situation). This latter case is more usual.

Financial costs associated with automated testing can be generally described as either fixed or variable costs[2]. Fixed costs of automation are expenditures for equipment, tools, training, etc. that are not effected by the number of times tests are run or the number of tests being run. Variable costs increase or decrease based upon the number of tests that are developed or the number of times the tests are run.

Examples of fixed automation costs:

- Hardware (additional or upgrades to existing)
- Testware software licenses
- Testware software support
- Per seat licenses
- Automation environment design, implementation
- Automation environment maintenance
- Scripting tools
- Tool and environment licenses
- Tool training
- Tool introduction and ramp up

Examples of variable automation cost factors:

- Test case design for automation
- Test case implementation
- Test maintenance
- Oracle creation
- Test case execution
- Test results analysis
- Defect reporting
- Test results reporting
- Test execution savings
- After-hours testing by systems (not people)

[2] The notions of fixed and variable costs used in this analysis do not necessarily conform to generally accepted accounting principles, but are intended to be used for understanding the costs and benefits of test automation.

Some common factors apply to both manual and automated testing. These factors generally increase when automated testing is introduced, but the increase in costs made for automation would be equally beneficial and justified for manual testing, so they should be left out of comparative ROI computations altogether.

Examples of common factors:

- SUT analysis
- Test planning
- Base test design
- Defect reporting
- Management reporting of results

Financial impacts from automation can be computed by comparing it with one of two alternatives: manually testing it or not testing (accepting the risk of not knowing). The first alternative is straightforward to compute for a specific test exercise after we have defined the cost factors involved. However, it is usually difficult to identify or measure which cost factors to include in such computations. It becomes even trickier whenever an automated test does not do the same thing as the manual test. There are also some costs and benefits that only relate to automated testing, especially when comparing test effectiveness. Some automated tests have costs associated with new capabilities to exercise and analyze the software under test. Here we are comparing automation costs with benefits of reduced risk (the additional information we get from the automated test). Care must be taken to understand, quantify, and qualify any values assigned to these factors. Often we are best off ignoring subjective factors such as these for the purposes of financial computations because the values we assign are too controversial. (If the benefit is required, then the ROI computation is unnecessary; the investment is required anyway.)

We need also to select the time period (*t*) over which we are computing the return. In the above example, one and four year periods were used. Usually there are natural project milestones that are appropriate to determine the period. Since benefits often begin during the next release, it is natural to select the period based upon releases, either to coincide with the next major release or the one after that. Computing returns for both periods can provide insight into the short and long term benefits.

The fixed costs related to automation are not absolute values. These factors need to be amortized (allocated) over their useful life and adjusted for the time period ***t***. The value of ***t*** should be selected based upon management factors such a accounting pay back periods, time between product releases, ease of ROI computation, life expectancy of tools or tests, etc., and to make computations reasonable, useful, and simple. Amortization of these cost factors is done by multiplying the cost by ***t*** and dividing by the useful life. For example, if a tool costs $25,000 to acquire and is expected to be useful for two years, only $12,500 is a cost in the first year ($25,000 * ½). We should also expect to have expenses of $50,000 over four years ($25,000 * 4/2). The investment costs are also valuable only as long as the item remains in service. If the tool goes out of service before the end of the first year, the entire $25,000 cost is a first year expense. (Similarly, if people leave the department after training, the entire cost of training is lost and does not get amortized over the expected period.)

The biggest advantage automation testing has over manual testing is less expensive test runs. This makes the expected number of runs of automated tests (***n*₁**) and the expected number of manual runs (***n*₂**) during the time ***t*** a critical factor in relative ROI computations. [Note that although n_1 is almost always higher than n_2 when automation is put in place, n_1 could be lower than n_2.][3] These factors need to be chosen realistically if we want to compute realistic, useful values.

Automated tests need to be maintained, so the number of times the test can be run before it needs to be revised (***N***) is also important. One organization I worked with had difficulty running tests because of frequent changes in a GUI. The group was using capture/playback to create the tests, and had measured the effort at about three times the time of just running the test manually. Maintenance consisted of recapturing the test and results, and was done when the test got out of sync with the software. The group seemed to be testing less and capturing more, so we computed the average number of automated test runs before maintenance and found it to be 1.2; four times out of five the tests only worked once before we had to redo them – and the 1.2 runs included the run that captured the test! (We abandoned the capture/replay effort as unproductive.)

The factors can be further divided into costs that are common between manual and automated testing and those that are nearly always increased or decreased. The factors that are common should be left out of the computations since they are neither costs nor benefits of automation. Those factors that increase when we automate are the costs of automation and those that decrease are the benefits. Some factors nearly always increase or decrease, but many factors that change can be costs or benefits, depending heavily on the type of automation and its effectiveness. The following lists provide examples for each.

Examples of factors that change (can be either a cost or benefit with automation):

- Automation environment maintenance (may be incremental cost or reduce overall maintenance cost)
- Test case execution
- Test results analysis
- Defect reporting
- Test results reporting
- Test data generation

Examples of automation benefits:

- Test execution savings
- After-hours testing by systems (not people)

[3] Although it may be convenient (and advantageous toward automation) to compare ***n*** automated test runs against the same number of manual test runs as some people have, I think it is unrealistic and questionable that the events are comparable. Manual tests have built-in variance, and reruns of passed tests are weak, so part of the value of a manual test comes from the fact that it isn't identical each time it is run, while automated tests tend to replicate their behavior every time they are run. So, is it five times as valuable to run an automated test daily as to run the same test manually once in a week? I may agree that there is not less value in running an automated test five times and a manual test once, but I am also sure there is not five times the value just because I ran the automated test that often. Or, do we need to find failures more often to be able to say that running daily is better, and thereby justifying poor development techniques and defective software? When the same number of manual and automated test runs is compared, and with the cost per run much higher for manual tests than automated tests, the benefits of automation will naturally improve the higher the number of runs. If I want to bias the numbers for comparison to favor automation I should run the automated tests 24 hours a day, 7 days a week (possibly on multiple parallel systems) so the number of test runs is quite high.

If we have too many automated tests to run in the allotted time, then we may only run each one half as often. I doubt we will miss many defects either way, but the running may be virtually free for the purposes of these computations. What realistically should be compared is the actual cost of running the automated test the number of times it is run on new versions of the software under test, versus the actual cost of running the manual test the number of times we would run it. We cannot reasonably factor any other value for the fact that one test is run more or less often.

Examples of automation costs:

- Hardware (additional or upgrades to existing)
- Testware software licenses
- Testware software support
- Per seat licenses
- Automation environment design
- Automation environment implementation
- Scripting tools
- Tool and environment licenses
- Tool training
- Tool introduction and ramp up
- Test case design for automation
- Test case implementation
- Test maintenance
- Oracle creation

Cost/Benefit Computations

Although there are many factors besides financial costs and benefits, the measurable financial values are the ones used to compute returns on investments in the four equations shown below. The first two examples provide a ratio of efficiency of automated tests over manual tests. The second two provide a ratio of the amount of return on each dollar invested on automation.

(1) $$E_n = A_a/A_m = \frac{(V_a + n*D_a)}{(V_m + n*D_m)}$$ [4]

(2) $$E_{n'} = A_a/A_m = \frac{(V_a + n_1*D_a)}{(V_m + n_2*D_m)}$$

(3) $$ROI_{automation}(\text{in time } \boldsymbol{t}) = \frac{(\text{Benefits from automation})}{(\text{Costs of automation})} = \frac{B_a}{C_a}$$

(4) $$ROI_{automation}(\text{in time } \boldsymbol{t}) = \frac{\Delta(\text{Benefits from automation over manual})}{\Delta(\text{Costs of automation over manual})} = \frac{\Delta B_a}{\Delta C_a}$$

Where:

E_n: the ratio of automated costs over manual costs for the same number of developed tests run the same number of times.

$E_{n'}$: the ratio of automated costs over manual costs for different numbers of test runs.

Subscript *a* stands for automated, *m* stands for manual
V_a: Expenditure for test specification and implementation
V_m: Expenditure for test specification
D_a: Expenditure for test interpretation after automated testing
D_m: Expenditure for single, manual test execution
n: Number of automated (and manual) test executions
n_1: Number of automated only test executions
n_2: Number of manual test executions
N: Average number of runs for automated tests before maintenance is needed

[4] Linz, T. and Daigl, M. "GUI Testing Made Painless. Implementation and results of the ESSI Project Number 24306", 1998. Analysis in Dustin, et. al., *Automated Software Testing*, Case Study: Value of Test Automation Measurement, p. 52^{+} of Addison-Wesley, 1999.

B_a: the benefits from automated testing.

C_a: the costs of automated testing.

ΔB_a: the incremental benefits from automated over manual testing.

ΔB_a (in time ***t***) = Σ (improvement in fixed costs of automated testing times (***t/Useful Life***)) +
Σ (variable costs of running manual tests $\boldsymbol{n_2}$ times during time ***t***) –
Σ (variable costs of running automated tests $\boldsymbol{n_1}$ times during time ***t***)

ΔC_a: the incremental costs of automated over manual testing.

ΔC_a (in time ***t***) = Σ (increased fixed costs of automated testing times (***t/Useful Life***)) +
Σ (variable costs of creating automated tests) –
Σ (variable costs of creating manual tests) +
Σ (variable costs of maintaining automated tests) times ($\boldsymbol{n_1}$/***N***)

Three of the four equations compute values based on comparisons between manual and automated testing. In most situations the return from automation should be computed as incremental return on incremental investment. The manual test numbers are the baseline costs for the test organization – these amounts will be invested whether testing is automated or not. Automation is additional cost with some expected additional returns, which is where the real value of automating shows up. The third equation expresses the ROI conceptually in terms of costs and benefits, but benefits are not computable in the absolute sense, only in relation to some alternative.

Equation (1) compares costs of creating and running a manual test with the same number of runs for an automated test. It computes the ratio of the cost of creating and running an automated test ***n*** times over the cost of creating and running an equivalent manual test ***n*** times. A value less than one is advantageous for automation as it shows the cost of the automated test as a fraction of the manual test cost. The equation can be used to compare costs for a single test implemented both manually and automatically, but only applies to groups of tests if there is exact correspondence between the manual and automated tests and all tests are run the same number of times. However, it does not account for overhead or maintenance costs associated with automation. The fixed costs described earlier that are not associated with individual tests, are some of the costs associated with automation that are overlooked by equation (1). Maintenance costs for the automated tests or automation environment are also overlooked by the equation. In order to understand the real cost and value of automation, both of these cost areas need to be considered.

The values used in equation (1) and the equation itself are based upon several assumptions that aren't true for most organizations, and therefore isn't applicable for them. Three of the assumptions in particular are particularly important. Most often, manual and automated tests are not run the same number of times, and every run of a test isn't equally valuable (since the first time is much more likely to find problems than any subsequent time). Also, most organizations don't have resources to create as many automated tests as manual ones, so there are usually fewer new automated tests. Thirdly, many automated tests require maintenance from time to time to keep them working at all, so there are likely to be additional costs incurred before an automated test is run ***n*** times.

Equation (2) accounts for the different numbers of times for running manual test and automated tests. It still expresses the cost of automation as a fraction of the manual costs, so a value less than one shows automation is advantageous. This equation more reasonably accounts for how the tests are run, but still doesn't factor in the potential differences in the number of tests, fixed costs, or maintenance. To represent the ratio for sets of tests the equation assumes that all automated tests are run n_1 times and all manual tests are run n_2 times.

Equation (3) shows how the ROI can be computed in a general form. It is the ratio of the benefits from automation over their costs. A value greater than one indicates that the benefits are greater than the costs. It shows that for every dollar invested in automation we get ROI dollars worth of benefits. The costs to develop, run, and maintain automated tests can be computed. However, it is very difficult to see how to compute the benefits of test automation in an absolute sense.

Equation (4) shows a relative ROI for comparing the added benefits from automation with the added costs from automating. Although the equation is very general, it more reasonably represents the value of automation in relation to manual testing. It allows selection of relevant parameters and tailoring of the components to specific organizational situations and it includes allocation of fixed costs and benefits as well as variable costs and benefits. The equation can be applied easily to projects and groups of tests, and with some care it can be used to show the benefits for individual tests. This is the equation that will be explained and used in the examples.

To use equation (4) we first need to identify all relevant costs and benefits. Values need to be determined for the items for both automated and manual testing, since we are comparing automated and manual testing. Some items may only apply to one or the other, but in some of these situations we have shifted costs by changing our activities. For example, an investment in hardware and tools may be required for automation with no corresponding expenditures when testing manually. In another case, a tool we buy for automating testing may provide a background load that requires additional people to be involved in the equivalent manual test, thus shifting a variable cost in manual testing to a fixed cost in automated testing. (The savings from manual testing are included in the second factor in the equation for ΔB_a, while the costs in automation are included in the first factor in the equation for ΔC_a.)

As mentioned earlier in the paper, selecting the cost factors is not easy, nor is assigning values to them. The closer one looks at the numbers, the more difficulty one has deciding what part of the costs or benefits to apply. The intangible factors tend to confound the computations if we try to measure costs or benefits too closely. Usually we are trying to answer the question of whether we gain or loose by investing in automation, and we don't need tremendous granularity in the component values to figure that out. After determining the applicable factors we can estimate approximate values based on broad organizational views such as people-months of effort. To compute these values we may substitute alternate computations in parts of the equations to achieve the same results:

$$\begin{aligned}\Delta B_a \text{ (in time } \boldsymbol{t}) = {} & \Sigma \text{ (improvement in fixed costs of automated testing times (} \boldsymbol{t/Useful\ Life} \text{)) } + \\ & \Sigma \text{ (variable costs of running manual tests during time } \boldsymbol{t}) - \\ & \Sigma \text{ (variable costs of running automated tests during time } \boldsymbol{t})\end{aligned}$$

$$\begin{aligned}\Delta C_a \text{ (in time } \boldsymbol{t}) = {} & \Sigma \text{ (increased fixed costs of automated testing times (} \boldsymbol{t/Useful\ Life} \text{)) } + \\ & \Sigma \text{ (variable costs of creating automated tests) } - \\ & \Sigma \text{ (variable costs of creating manual tests) } + \\ & \Sigma \text{ (variable costs of maintaining automated tests during time } \boldsymbol{t})\end{aligned}$$

Generalizing or aggregating the costs is not only faster and easier, but it often leads to more reasonable and usable results. These computations depend upon general amounts of time and costs rather than specific measures of individual test contributions. We can more easily identify the number of people involved in manual testing (and then allocate their time between test design and test execution) than we can catalog all of our tests and then measure and sum up the time spent designing and running each of them. This approach also has the benefit of including much of the real world overhead and inefficiencies that are overlooked when more focused measurements take place. The resulting numbers are much more conservative and defensible, and better represent what happens.

Two simple examples are provided below to show the kind of return we might see from automating some testing. The first example is for automating build tests, which is one area of testing where there is usually a relatively quick payoff from automation. These are run frequently and the tests are simple and relatively stable. The second example is based on automation of GUI tests, which tends to be at the other extreme, with more delayed and smaller payoff. (GUI testing is often where there is a financial loss on the automation investment because maintenance costs may be much higher and more frequent than in the example.)

Automation of build tests examples:

Assumptions and computational values:

- Daily builds and test runs (5 times a week)
- Manual tests take 5 days to design, 2 hours to run
- Only half the manual tests would be run on any given day (1 hour) with the other half run the following day
- Automated tests take 15 days to design and implement, automatically run (zero cost)
- Automation is done with batch scripts and integrated into the build process, requiring $1,000 in added hardware, with a useful life of 3 years
- Automated tests need to be maintained every 25 runs, one day of work required
- Periods of time (***t***) selected: 6 months (125 days) and 18 months (375 days)
- People cost $100,000 per year = $400 per day = $50 per hour

ΔB_a (in time ***t***) = Σ (improvement in fixed costs of automated testing times (***t/Useful Life***)) +
Σ (variable costs of running manual tests $\boldsymbol{n_2}$ times during time ***t***) –
Σ (variable costs of running automated tests $\boldsymbol{n_1}$ times during time ***t***)

ΔB_a (in 6 months) = 0 + (1 hours * 125) – (0 * 125) = ($50 * 125) = $6,250

ΔB_a (in 18 months) = 0 + (1 hours * 375) – (0 * 375) = ($50 * 375) = $18,750

ΔC_a (in time ***t***) = Σ (increased fixed costs of automated testing times (***t/Useful Life***)) +
Σ (variable costs of creating automated tests) –
Σ (variable costs of creating manual tests) +
Σ (variable costs of maintaining automated tests) times ($\boldsymbol{n_1/N}$)

ΔC_a (in 6 months) = (\$1,000 * (6/36)) + (15 * \$400) – (5 * \$400) + (\$400 * (125/25))
= \$167 + \$6,000 – \$2,000 + \$2,000 = \$6,167

ΔC_a (in 18 months) = (\$1,000 * (18/36)) + (15 * \$400) – (5 * \$400) + (\$400 * (375/25))
= \$500 + \$6,000 – \$2,000 + \$6,000 = \$10,500

$ROI_{automation}$(in 6 months) = \$6,250 / \$6,167 = 1.013 [about break even]

$ROI_{automation}$(in 18 months) = \$18,750 / \$10,500 = 1.786 [80% return]

The return for 6 months is just above the investment cost and there is substantial return over 18 months. When the automated tests are run less frequently or require more maintenance there is much lower return. The return is also helped substantially by the low cost of creating and maintaining these tests.

Automation of GUI tests example:

Assumptions and computational values:

- A new product with all new tests
- 5 people-years developing manual tests, 15 for automated tests
- 1 person maintenance after 1st year for automated tests
- 10 people full time running manual tests, 1 person for automated
- Fixed costs for automated tests of \$90,000 with useful life of 3 years
- Periods of time (***t***) selected: 12 months (250 days) and 24 months (500 days)
- People cost \$100,000 per year = \$400 per day = \$50 per hour

ΔB_a (in time ***t***) = Σ (improvement in fixed costs of automated testing times (***t/Useful Life***)) +
Σ (variable costs of running manual tests during time ***t***) –
Σ (variable costs of running automated tests during time ***t***)

ΔB_a (in 12 months) = 0 + (10 people * \$100,000) – (1 person * \$100,000)
= \$900,000

ΔB_a (in 24 months) = 0 + (10 people * \$200,000) – (1 person * \$200,000)
= \$1,800,000

ΔC_a (in time ***t***) = Σ (increased fixed costs of automated testing times (***t/Useful Life***)) +
Σ (variable costs of creating automated tests) –
Σ (variable costs of creating manual tests) +
Σ (variable costs of maintaining automated tests during time ***t***)

ΔC_a (in 12 months) = (\$90,000 * (1/3)) + (15 * \$100,000) – (5 * \$100,000) + \$0
= \$30,000 + \$1,500,000 – \$500,000 = \$1,030,000

$$\begin{aligned}\Delta C_a \text{ (in 24 months)} &= (\$90{,}000 * (2/3)) + (15 * \$100{,}000) - (5 * \$100{,}000) + \$100{,}000 \\ &= \$60{,}000 + \$1{,}500{,}000 - \$500{,}000 + 100{,}000 = \$1{,}160{,}000\end{aligned}$$

$$ROI_{automation}\text{(in 12 months)} = \$900{,}000 / \$1{,}030{,}000 = 0.874 \text{ [small loss]}$$

$$ROI_{automation}\text{(in 24 months)} = \$1{,}800{,}0000 / \$1{,}160{,}000 = 1.552 \text{ [55\% return]}$$

The return on investment is lower and later than in the first example, but this example includes much more infrastructure in fixed costs and much higher investments in running both the manual and automated tests. By using the alternative forms of the ROI equations we computed it without needing to identify the exact numbers or costs of running either manual or automated tests, and used realistic estimates based upon staffing and job assignments of the test group.

Conclusion

Test automation is not always necessary, appropriate, or cost effective. In cases where we are making decisions based upon an expected return on investment, analysis can direct us to where test automation can benefit us. These returns are best computed by comparing the costs and gains achieved through test automation over manual testing. Good management decisions can made about using automation to improve testing by identifying and estimating test automation costs and benefits. This also helps to identify which factors we need to focus on to make the most of our investments.

When automation is required, either by contract or due to technical constraints, ROI computation may not be helpful. Intangible factors may constitute the bulk of the return, and thus arithmetic computations won’t indicate the real value of automation. Fortunately in these situations we often aren’t faced with questions about the value of automation because it must be employed regardless.

When we want to understand the value of test automation there are several computational approaches we might employ. Some currently published equations have serious drawbacks and management decisions made based on their use may be flawed. To avoid mistakes, misplaced expectations, and disappointment with automation we need to carefully assess the relevant factors applicable to each test situation, determine if there is cost or gain associated with automation, and estimate the amount of it. These factors should be compared between automated and manual testing of the same areas or we should aggregate costs and benefits across sets of manual and automated tests. Alternate forms of equations can be used to provide similar results based on the same ROI computation approaches.