Test automation drivers and challenges
Executive summary
An efficient test automation capability can be a boon for IT transformation programs and software implementation projects looking to shift testing left and save time and money on testing services. At root, the fundamental goals of automation apply across sectors, industries and platforms. Everyone wants to use automation to:

- Reduce time spent on testing. By automating, companies look to complete test execution more quickly, execute multiple tests in parallel and decrease their time to get to market.
- Improve test coverage. Data-driven automation allows for more testing variation than is possible with humans. It also removes the problem of reducing test coverage because of time constraints.
- Enhance test quality. An automated environment removes human error and executes tests in a consistent way.
- Save money. The cost of incremental test execution is reduced, and workers are freed to focus on other tasks.

But these desired outcomes too often can be thwarted by test automation capabilities that are poorly designed, implemented or operated. These hurdles can seem daunting and can be enough to prompt leaders to question the wisdom of committing to an automation program.

The promise of automation is one thing, but how can the obstacles be overcome or avoided altogether so that the promise is realized?

TABLE OF CONTENTS
Executive summary 3
Challenges 4
Getting it right 5
Clearly defined goals 5
The right approach and tools 6
The right things to automate at the right time 7
A model that can scale 8
Continuous innovation 8
Putting it all together 9
Want to learn more? 9
Challenges
Four chief culprits foil companies in their journey to maximize return on investment (ROI) from automation:

- A faulty automation approach
- A failure to operationalize automation capabilities
- A lack of innovation
- High maintenance as a result of the above

Common pitfalls in a business’s approach include:

- Focusing only on test automation that is based at the user interface (UI) level or using the wrong tool for an automation need
- Adopting an “automate everything” mindset that detracts from pinpointing the specific areas that can yield the greatest benefit
- Using a poorly designed automation framework that breaks down when a business attempts to scale the technology

Operations and overhead problems arise when automation planning fails to produce a structured set of metrics and key performance indicators, thus hindering the ability to effectively measure and communicate ROI. Other impediments are the cost of purchasing and maintaining the automation tool; the treatment of test automation as an independent, siloed capability; and the cost of bringing on people with automation strategy skills.

Innovation is an essential component of automation strategy and planning. Its absence is felt when companies limit automation to only test when they rely on old tools to address new problems. Testing and automation teams can also fall into the trap of failing to leverage the open-source community to solve their biggest problems. Chances are, your organization is not the first to encounter a given testing need—someone has probably already solved that problem, and you don’t need to spend weeks re-creating the wheel.

In addition, maintenance costs add up when the right solution isn’t used, when frequent application changes break test scripts, or when an automation team lacks the right level of domain or business process knowledge.

Those are the challenges. What are the solutions?

Getting it right
An essential facet of adopting automation is the ability to properly place it in context within your organization. Test automation is not a “one size fits all” solution. Different circumstances, applications, projects, technologies and teams need different tools, frameworks and priorities for automation. A successful automation implementation can only occur when automation is duly recognized as part of an organization’s testing strategy—when a business maps how automation will fit into its overall testing approach and testing timeline. Manual testing strategy and automation strategy must not be treated as separate components; rather, they should be regarded as complementary puzzle pieces.

For automation to succeed, you need to:

- Clearly define test automation goals
- Choose the right approach and tools
- Identify the right things to automate at the right time
- Have an operations and delivery model that can scale
- Inject innovation continuously

Let’s look at each of these factors.

Clearly defined goals
For all its benefits, automation still must be adopted for a specific purpose rather than simply for the sake of automating. An effective automation project hinges on having the right strategy, the right scope and the right tools. That is, strategy hinges on evaluating the right things to automate, based on the technologies behind those processes and their stability.

To identify more specific goals for automation than to “save time and money,” here are some considerations:

- Are you automating to attempt to alleviate an issue with testing? What is it?
- What should you automate?
- What tools does your organization already have?
- What skills do you have to build and run automation capabilities?
- How can you get the biggest benefit for the least effort?

For example, you could automate the process of creating a policy via the application programming interface (API) because the API is stable and the same script could be used for smoke testing, data creation and regression testing.

- What does success look like?

If you feel like you already have clearly defined goals, make sure to dig deeper and really analyze if those are quality goals.

- Why do you want to automate a given process?
- Why do you have that coverage goal? Do you really need 95% automation?
- Have you thought enough about it to make sure the build and maintenance won’t outweigh the benefits?
- Are you willing to make an upfront investment in people, tools and time to do it right so it’s scalable in the future?
The right approach and tools

The right approach minimizes maintenance time and costs, a common concern for companies considering automation. While automation can't stop functional changes from happening, the effect of such changes on automation can be minimized if automation teams are integrated with the rest of the organization’s testing teams, and the right approach, tools and innovation are leveraged properly.

The approach would include determining:

1. Type(s) of automated testing
   a. UI
   b. API
   c. Database (DB)
   d. Other
2. When to automate
   a. During development?
   b. Functional testing?
   c. Integration testing?
   d. Combination of the above?
3. Tools to use based on the answers to 1 and 2
4. Integration into the overall testing strategy and timelines
5. Alignment with application development, readiness and functional testing
6. Unique considerations: would you benefit from automation outside of the common methods?
   a. PDF comparisons, validations
   b. Visual testing
   c. Complex machine learning or analytics models that have specialized inputs and outputs
   d. Extract, transform and load (ETL) or conversion testing with complex transformations
   e. Combination of UI, API, DB or more in a single test

Using this information, you can begin to clearly define the automation requirements and move to the next step: feasibility and piloting.

One approach we use for automation includes the following components:

* An automation feasibility assessment that includes an evaluation of application suitability and technical and financial feasibility
* A proof of concept consisting of tool evaluation, an ROI and cost-benefit analysis, and tool selection with the goal to identify when to automate and what to automate to maximize ROI
* Development and execution of an automation suite, which includes setting up tools and the environment and script execution

While that’s a fine structured approach, what if you just need to start with automation tomorrow and iterate to improve and see value? What’s your automation minimum viable product (MVP)? Consider these steps:

* Identify the functionality that is universal and critical, such as logging in, creating an account, etc.
* Create reusable functions to perform those actions
* Put them together in any order required to complete different transactions
* Use those same functions/scripts for as many things as possible — including smoke testing, data creation, regression testing and production checkouts

The right things to automate at the right time

A key concept is to prioritize items that can generate ROI most quickly. Even if you know what you want to automate, it may not be the right time. Unit tests should be automated early as a part of development; API automation should be used shortly after (especially for commercial off-the-shelf (COTS) products) to create smoke tests, data creation and regression tests; and UI tests should be implemented last to ensure user transactions complete as expected. Innovative approaches such as the Ernst & Young LLP (EY) self-healing technology can help move this earlier than traditional automation.

We have found that the ability to introduce pre-built libraries for integration with other tools, which allows a business to minimize ramp-up time and avoid building an automation program from scratch, significantly reduces the initial investment in standing up automation capabilities. These libraries include generic libraries for calculations, data capture and manipulation, and platform-based libraries for interaction with common platforms. These capabilities allow us to help businesses that already have made an initial investment in automation tools, as well as those that need assistance selecting the right tool.

So, what should you automate? Everything? In most cases, no. Trying to automate everything will usually cost you more than you save. Automation candidates can be filtered through any number of criteria depending on your goals, tools and the technical skills of your automation team. These are some of the questions you should ask:

1. Does the test/process require manual intervention?
2. Does the test/process span a long period of time? (Example: multiday business scenarios or annual policy renewals)
3. Does the test/process require manual intervention?
4. Are the related processes/applications going to be around long enough for you to realize ROI? Or will they be retired next month?

Your answers can help you determine the value of automating a given process. Long-running processes or processes that require manual intervention or batch jobs could still be broken apart and automated, for example, but it will be up to you to decide that there’s value doing so. How can you estimate ROI? It’s not as simple as just looking at how much time it would take to do these tests manually. Several aspects of automation detract from true ROI, including framework setup, script development and maintenance. How much depends on factors such as:

1. Stability — helps determine maintenance and scripting ability
   a. High impact on automation priority
   b. Are there outstanding defects that impact functionality?
   c. Are there known major changes expected within the next six months?
2. Business priority — helps determine value
   a. High impact on automation priority
   b. How important is this piece of functionality?
3. Complexity — helps determine effort
   a. Low impact on automation priority
   b. Are there multiple steps to complete a test case?
4. Frequency of use — helps determine reusability, ROI and value
   a. Medium impact on automation priority
   b. How often is this functionality tested?
   c. How often is this functionality used in production?

Other factors, depending on your situation, could include frequency of defects (maybe you want to automate more of these things to try to find defects faster) or being more specific about frequency of use to identify those tests that are highly data-driven vs. simply just being used frequently, for example.

Our use of automation pilot programs helps identify areas with the best ROI prospects, then employs a targeted approach in those areas to verify that the chosen automation tools and teams are a good fit. Using all the factors listed above, we created a scoring mechanism to prioritize automation candidates. Adjustments are made throughout the process to maximize the utility of the pilot program.
A model that can scale

An often-overlooked but important aspect of the automation journey is to operationalize your automation capabilities. This means being able to take time to understand manual processes and develop an ROI analysis and model that drives automation priority and operations.

Metrics can highlight the business case for strong automation capabilities, but these cases can’t exist in a vacuum. They must be an integrated part of the overall testing strategy and the overall software development life cycle. If you rush into automation without planning and tracking ROI, you can easily be overcome with maintenance and the fact that humans do more than write automation scripts.

Critical components of operationalizing automation capabilities include:

1. Traceability to existing manual testing
2. Metrics on automation execution for testing, data creation and other uses
3. Robust documentation for automation capabilities and coverage (makes sure existing functions/scripts are reused instead of repeated)
   a. Using agile methodologies does not mean no documentation. If you started using a product or tool without some user guide, you’d be pretty lost, too, right?
4. Coding practices, standards and reviews
5. Comparing time and money spent vs. savings
   a. This is often avoided because true ROI isn’t as easy to achieve as everyone thinks. Maintenance and bad approaches can cause automation to cost more than it saves.

Continuous innovation

Innovation isn’t just a trend or something to aspire to – it’s a requirement in the fast-paced world of technology. Applications have moved at lightning speed, so why are some organizations expecting tools that largely haven’t changed in five or more years to keep up? With all this new technology in machine learning and data analytics, can’t we do things better? More efficiently? Why are we still dealing with the same issues from 10 years ago?

The most aspirational intentions lead nowhere without innovation. We stay on top of the rapidly changing technology landscape so that the most useful tools can go to work for you. We leverage the collective brainpower from leading open-source contributors such as Google, Twitter, Intuit and others to employ solutions that use machine learning, image cognition, and data analytics and modeling to solve complex testing problems without reinventing the wheel.

Examples of EY innovation include:

• Self-healing automation – we built this tool to be compatible with most major test automation tools and frameworks; it uses data analytics to address the top cause of test automation script maintenance: object changes
• Machine learning production feedback – this capability allows the processing of production issues and tickets to understand human language and use it to improve testing
• Truly, fully automated testing – this custom-built EY solution uses model-based testing to automate scripts, and test data identification and selection, execution and reporting
• Test data analysis and simulation – this process allows you to crawl your application databases, understand data relationships, and manipulate and/or copy data from another source to fill data needs

Putting it all together

By articulating your goals and adopting the right approach, the right planning and the right strategy for automation, the theoretical benefits become real. A well-thought-out automation capability not only saves time in testing - it can ultimately lead to an organization getting its products to market faster. The real issue to confront is not whether to automate, but how to do so in a way that provides the greatest benefit.

Want to learn more?

If you have specific questions related to your situation, or would be interested in a demonstration, please reach out to John McEvoy at john.mcevoy@ey.com.
EY | Assurance | Tax | Transactions | Advisory

About EY
EY is a global leader in assurance, tax, transaction and advisory services. The insights and quality services we deliver help build trust and confidence in the capital markets and in economies the world over. We develop outstanding leaders who team to deliver on our promises to all of our stakeholders. In so doing, we play a critical role in building a better working world for our people, for our clients and for our communities.

EY refers to the global organization, and may refer to one or more, of the member firms of Ernst & Young Global Limited, each of which is a separate legal entity. Ernst & Young Global Limited, a UK company limited by guarantee, does not provide services to clients. Information about how EY collects and uses personal data and a description of the rights individuals have under data protection legislation is available via ey.com/privacy. For more information about our organization, please visit ey.com.

For more information about our organization, please visit ey.com.

Ernst & Young LLP is a client-serving member firm of Ernst & Young Global Limited operating in the US.

© 2019 Ernst & Young LLP.
All Rights Reserved.

1904-3108678
US SCORE no. 06385-191US

ED None

This material has been prepared for general informational purposes only and is not intended to be relied upon as accounting, tax or other professional advice. Please refer to your advisors for specific advice.

ey.com