Software Test Automation Solution Buyer’s Guide

Software Test Automation Solution Buyer's Guide

Why Adopt a Test Automation Tool?

Software testing plays an essential role in application development. Testers seek to identify defects before they appear in production, validate that functionality works as intended, and help deliver a quality product that maximizes customer satisfaction. But despite its importance, the rapid pace of application development often means that time for testing is limited.

There are many reasons to adopt a test automation tool. Perhaps you’re looking to:

- Return faster results from testing, to avoid becoming a bottleneck in the release cycle
- Extend test coverage while controlling costs
- Reduce repetitive tasks and allow testers more time for deep exploration of the application under test

To ensure that you select the best automation tool for your needs, it’s important to consider the following areas:

1. Goals
2. Readiness
3. Resources
4. Tool Analysis
5. Vendor Selection

This Buyer’s Guide is designed to help you understand these key areas so that you can select the right solution for your organization.

Section 1: Goals

To be successful, any project must begin with clear goals. To develop these goals, first define the problem that you want automation to solve and identify the benefits that you hope to gain by automating. Use the questions in this section to help you define a limited number of key goals, which you should then be able to summarize in a brief statement.

1 What problems are you trying to solve by automation?

Obvious problems that point to a need for test automation include finding defects in production, or delayed releases due to defects. But there may be other issues that automation can help resolve, such as a lack of motivation, in the testing team due to repetitive tasks, errors in data entry or step completion during testing, delays in resolving defects due to communication issues between testers and developers, rising costs of testing including costs to execute tests and costs to produce test artifacts, inability to test at scale, and reduced time for exploratory and UX testing.

2 What benefits do you hope to gain?

Make a list of the benefits you expect, and then prioritize them. Possibilities include more complete testing within an allotted time period, increased customer satisifcation by not releasing repetitive bug fixes, and reduced cost of development since defects are identified faster. You might also expect to see a reduced cost of application support due to fewer defects released.

3 What are the risks of automating? Of not automating?

Whether you automate or not, this choice will have an impact on your organization. Possible risks of automating include a failure to get a return on investment (ROI) for the time, effort, and money spent on automating. Money includes licenses, hardware, training, as well as the lost opportunity cost.

Risks of not automating could include any of the logical outcomes that could occur from the problems you identified not being resolved – i.e., increasing defects in production which lead to support calls and decreased customer satisfaction, turnover among staff or lack of output from unmotivated staff, lower quality in functionality and UX due to a lack of time for manual testing, etc. Use a risk analysis matrix to identify the most likely risks and to prioritize them according to the potential impact on your organization.

4 What is the scope of testing to be automated?

Do you plan on automating your functional testing, regression testing, end-to-end testing, performance testing? The test automation pyramid suggests the ratio of unit, interface, and UI tests that your project should have, but this may need to be adjusted based on the needs of your particular application. A stable legacy application may benefit from a higher number of UI tests, for example. The scope of testing will place requirements on the automation framework that you will need.

5 What assumptions need to be validated?

One of the biggest challenges in creating a plan is to uncover the underlying assumptions. Refer to the next section for questions that can help uncover assumptions and ensure that they have been addressed in the automation framework that you choose.

Section 2: Readiness

Review current state of your release cycles to verify that your organization is ready to evaluate and adopt an automation solution.

How stable is your overall application?

It should have some measure of stability to gain the most ROI from automating your integration tests, end-to-end tests, and functional UI tests

How many defects are being identified in UAT?

- How many defects are making it through to production?
- How many support calls are you getting that are related to defects?

Is there a trend in the number of defects per release?

A rapid or sustained increase in the number of defects suggests a greater need to adopt a test automation strategy.

What types of defects are occurring?

- User interface/user experience?
- In backend processing, integration, security, performance?

Are you satisfied with how you are capturing and tracking defects?

- Is an automated defect tracking solution in place? If not, how do you plan to handle defects that are identified in a future automation scenario?

How many test cases are being performed in each cycle?

- How many test cases do you plan to execute in a given cycle? And are you able to meet your goals, or do you often run out of time or resources?
- How often are test cases repeated? The more you repeat a given test case, the greater the potential benefit of automating it. It may be helpful to use a spreadsheet such as the Ranorex test case ROI calculator to estimate the potential ROI from automation.
- Are your existing test cases documented well enough to be automated? The best test cases to be automated have well-defined scenarios, unambiguous pass/fail criteria, and represent actual customer use cases.

How much time do you have for testing in a typical release cycle?

- Is testing currently acting as a bottleneck?
- Is test coverage insufficient?

What percentage of tests in each cycle are regression tests?

- Regression tests are great candidates for automation, since they are often repeated.

What level of test coverage is needed for priority/high-risk features?

Are tests often being repeated for different data values?

- An automated test can rapidly execute tests from a data source such as a CSV file or a SQL database, with a high degree of accuracy.

Are you testing on multiple platforms?

- Performing cross-browser or cross-device testing?
- Does your application include multiple technologies, such as HTML5 plus non-HTML elements, or a combination of cloud-based and backend services?

What reports are needed?

- Typical detail-level reports include the number of test cases executed, along with pass/fail rates and the number and type of defects reported.
- Summary reports may include information like overall costs of each testing per cycle, percentage of tests executed, and defect resolution rates .

How does your current mix of testing match with your needs?

Load testing, security testing, compliance testing, user experience, etc.

Section 3: Resources

Do you have existing resources that you could leverage for automated testing? What additional resources might be necessary for evaluating and then eventually adopting a solution? Below are the types of resources that are typically needed for a test automation project:

1 Hardware

Is there hardware available for testing? In addition to having physical or virtual endpoints available for testing, do you have the necessary support from your operations team?

2 Space

Do you have a dedicated space for a project team to collaborate on the selection process, and for an eventual proof of concept (links below)? Typically, teams work better when they are in proximity to each other. If that's not possible – for example, because key team members are in remote locations – do you have a platform for virtual meetings?

3 Budget

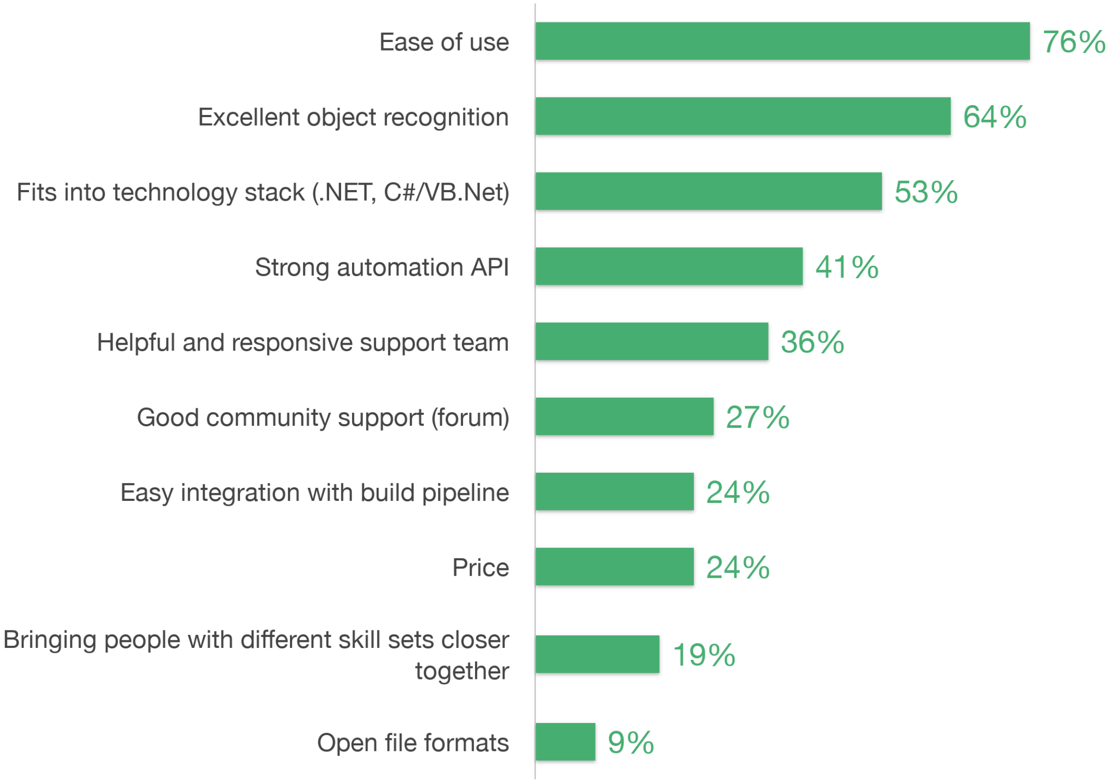
Typical costs may include not just software licenses, but also the hardware infrastructure to develop and execute automated tests, and training for your staff. It may be tempting to turn to an open-source solution because budget isn't available for purchasing a commercial solution; however it is important to compare the total cost of "ownership" including the time to build your own framework, higher technical expertise required for your team, and a longer time required to develop automated tests.

4 Champions

Is there staff with test automation expertise? Successful automation projects are typically driven by one or more “champions.” If you have insufficient experience on the team, what is the availability of training for existing staff using existing resources?

RANOREX STUDIO CUSTOMER RESEARCH

Which of the following were important features when you were evaluating test automation?



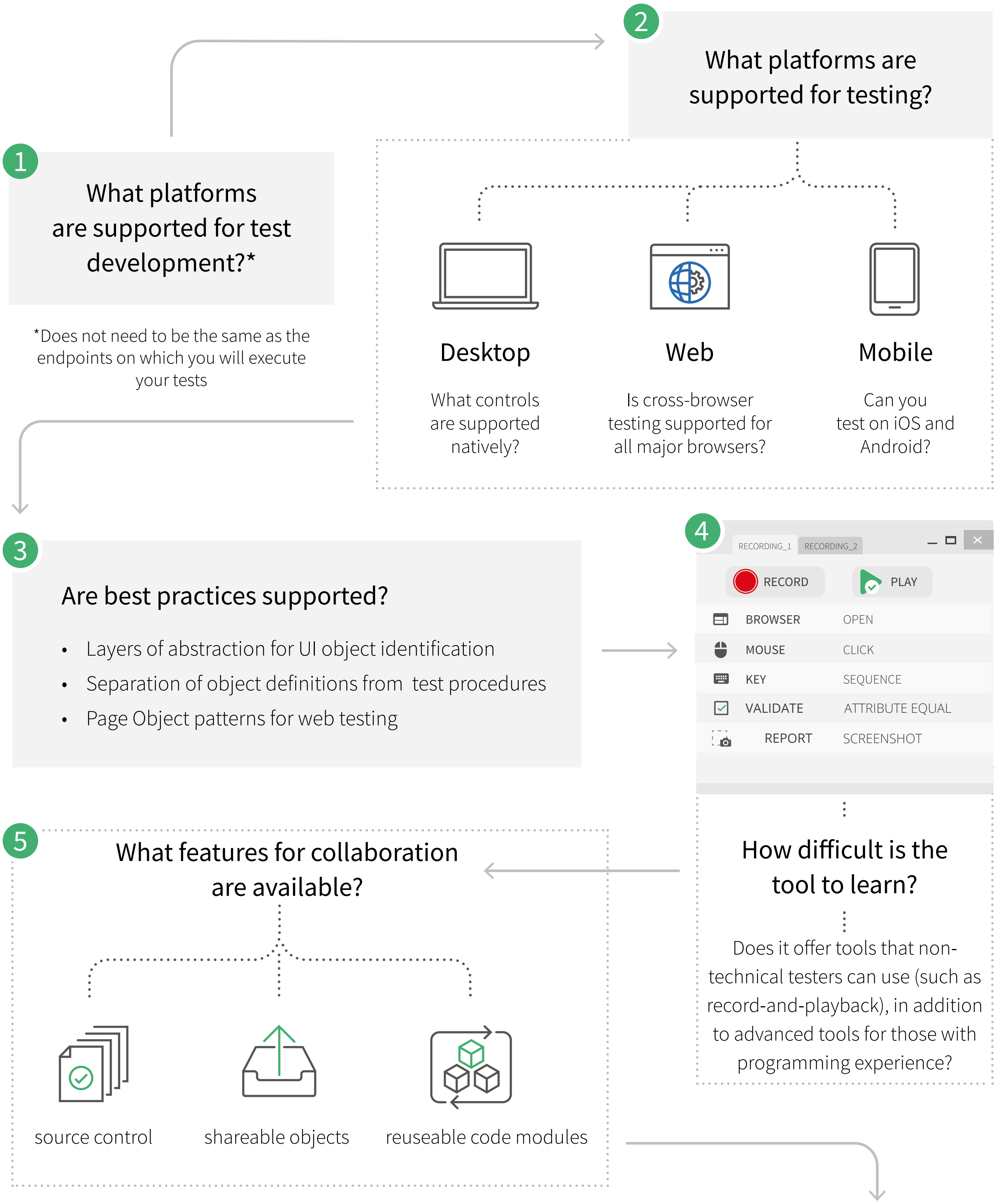
Source: TechValidate survey of 614 users of Ranorex Studio

✓ Validated Published: Dec. 19, 2018 TVID: E99-B67-83E

Ranorex® TechValidate

Section 4: Tool Analysis

After reviewing your goals, readiness, and available resources, the next step is to begin evaluating test automation tools. The diagram below identifies key considerations when evaluating test automation tools.



1 What platforms are supported for test development?*

*Does not need to be the same as the endpoints on which you will execute your tests

2 What platforms are supported for testing?

Desktop

What controls are supported natively?

Web

Is cross-browser testing supported for all major browsers?

Mobile

Can you test on iOS and Android?

3 Are best practices supported?

- Layers of abstraction for UI object identification
- Separation of object definitions from test procedures
- Page Object patterns for web testing

4

BROWSER	OPEN
MOUSE	CLICK
KEY	SEQUENCE
VALIDATE	ATTRIBUTE EQUAL
REPORT	SCREENSHOT

How difficult is the tool to learn?

Does it offer tools that non-technical testers can use (such as record-and-playback), in addition to advanced tools for those with programming experience?

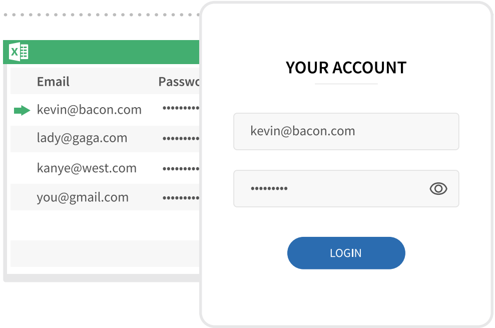
5 What features for collaboration are available?

6

Ease of customization:

- How flexible/customizable is the tool?
- Does it offer a full IDE and/or an API?

7



Email	Passwo
kevin@bacon.com	••••••••••
lady@gaga.com	••••••••••
kanye@west.com	••••••••••
you@gmail.com	••••••••••

YOUR ACCOUNT

kevin@bacon.com

•••••••••

LOGIN

What data sources are supported for data-driven testing?

Can you use CSV files, native Excel files, or SQL data connectors? Will you be able to connect to your production database, if desired?

8

What kind of execution speed should you expect to see from an automated test?

Tests fail if they execute more quickly than the AUT is capable of responding.

A tool should offcer fast execution time, but also be able to fine-tune waits for UI elements.

9

Jenkins

spiraTest

git

JIRA

TC

What integrations are available?

10

Built-in reporting features?

- Can you customize the report?
- Is a license required to access the test run report?

11

Scalability

Can you scale up testing, including distributing onto multiple devices in parallel?

12

Development language

Is expertise in a standard or proprietary language required?

13

Import Options

Is it possible to import test cases from another source? What import formats are supported?

Section 5: Vendor Selection

The ideal vendor will not just sell you an automation tool, but will be a full partner in ensuring that your test auomation project is a success. Below are areas to consider when selecting a vendor.

Pricing

What pricing options are available? Are licenses permanent or pay-as-you-go?

Licensing

What is included in the license – software updates, support, training, implementation?

Proof of Concept

Does the vendor offer pre-sales support and/or assistance to complete a proof of concept?

Industry Expertise

Does the vendor seem to understand the needs of testers? Does the vendor use their own tool internally for testing?

References

Can the vendor provide you with a list of references – current users willing to discuss their experiences?

Reliability

How long has the vendor been in the industry? What awards or recognition has the tool or vendor received?

Additional Services

Are there services like consulting, training, technical support available?

Technical Support

What technical support options are available? How do current customers rate the technical support team?

Community

Is there an active user community where you can exchange ideas and best practices?

Section 6: Next Steps

Do a proof of concept to prove that the technology works in your environment.

Once you have completed your analysis, but before committing to a solution, you should conduct a Proof of Concept (PoC) involving two or three tools. This will give you the opportunity to demonstrate how these tools actually perform in your environment. You will want to prepare a representative subset of example test cases to automate. Be sure to consider the following questions:

1 What resources do you need to provide, and what will the vendor provide?

The PoC will be conducted in-house, so you will likely need to provide a workspace, access to a test version of your application, dedicated testing hardware, etc. Be sure to allow sufficient time to get these resources in place before beginning the PoC.

2 Who will be involved in the PoC?

It‘s important to identify one or more “champions” who will drive the project. Typical team members include a decision-maker (often the QA manager), product team leader, sales engineer from the vendor, and a combination of manual testers or test automation personnel from your organization who will be essential in analyzing the ease-of-use and effectiveness of the potential tool.

3 How will you document your experiences during the PoC?

Establish a regular review process such as brief daily meetings. Consider setting up a tool for capturing feedback and sharing results, such as a shared spreadsheet with a scoring rubric.

4 How will you know when you are done?

Here is where you "define success." Set a clear beginning and end to the PoC. This may be a set timeframe, such as two weeks, or after automating a number of test cases, running a number of test cycles, etc.

5 What test cases will you automate?

Select realistic scenarios based on the risk analysis that you conducted earlier in the planning process.

Ideally, a successful PoC will be followed by adoption of your selected solution. Ranorex experts are available to assist you with planning a POC and experiencing test automation success.

> "The Ranorex support team is really great. We were facing issues while automating one of the thin client applications at YRC. We explored a lot of software including HP UFT, Smartbear TestComplete, but we were unable to track elements for our application. Ranorex engineers developed a new plugin for us which solved our problem and we completed automating that thin client application. It is a really good experience working with Ranorex!"
>
> *Shantanu K, QA Automation Engineer*
> *Transportation/Trucking/Railroad company 10,001+ employees*
> *Review verified on Capterra*

About Ranorex

From our founding in 2007, Ranorex has been dedicated to empowering software teams with comprehensive UI testing tools that can handle even difficult-to-automate interfaces. Ranorex products are supported by a team of professionals dedicated to your success. Ranorex is a member of the Idera, Inc. family of testing tools.

Rx **Ranorex** Studio

Our all-in-one solution, Ranorex Studio, supports UI test automation across desktop, web and mobile devices. Test automation experts can use Ranorex Studio's full IDE, with its open APIs and tools for intelligent code completion, refactoring, debugging and more. Automation novices can use Ranorex Studio's capture-and-replay tools and built-in methodology to rapidly build reliable, maintainable tests while expanding their automation skills. All members of cross-functional teams can collaborate on solutions by sharing reusable object repositories and test automation modules. Ranorex Studio includes built-in Selenium WebDriver integration for scalable cross-browser testing: execute tests in parallel, on a Selenium Grid or using a cloud platform. Perform data-driven testing with CSV files, Excel files, or SQL data connectors. Ranorex Studio integrates with Jira, Jenkins, TFS, Git, TestRail and many more.