Surviving the Top 10 Challenges of Software Test Automation

Randall W. Rice
Rice Consulting Services, Inc.

Thursday, 2 May 2002
Track 8: 1:00 - 1:40
Room 251 D - F

Capture/playback tools make it possible to repeat two or more tests identically and compare the results. This article focuses on capture/playback tools, which hold the largest market share of any test tool category, and on examining trouble spots in test automation, dealing with them proactively and perhaps mitigating the risks of tool abandonment. The purpose of this article is to outline the 10 major challenges that the author sees most often in organizations struggling to make effective test automation a reality.

Capture/playback tools capture or record the actions performed during a test session into software-like scripts that can be replayed against the same or an updated version of the software, allowing a comparison of identical test results. A difference in test results may indicate the presence of a regression defect.

For the past six years, I have been asking training and conference audiences, "How many of your organizations own some type of automated capture/playback test tool?" Typically, 80 percent to 90 percent of the audience will raise their hands. However, the typical response drops to about 10 percent to 20 percent of that group when I next ask, "How many of you who just raised your hand would consider automated test tools an integral part of your testing effort?"

Obviously, there is a large gap between the people who own automated test tools and people who actually benefit from test automation. Also, my survey findings have not changed significantly during the past six years. The observation that automated test tools account for a lot of *shelfware* is not new or all that insightful. In fact, the shelfware problem is shared with many types of software tools.

When faced with the dynamic world of automated test tools, many organizations make the best choice they can and then try to make the tool work in their environment. They hope the rest of the organization will embrace the tool as well.

The purpose of this article is to outline the 10 major challenges that I see most often in organizations struggling to make effective test automation a reality. The earlier these challenges are understood, the better prepared an organization will be to deal with them. Plus, the more an organization understands the issues of test automation, the less risk it incurs in time and money when acquiring a tool.

These challenges are presented in order from the least to highest impact on the overall test automation effort. It is also important to understand that even organizations that have developed core competencies in working with test automation struggle at times with these challenges.

"Test automation promises increased productivity and accuracy, which is where the business case must be made. The cost of a single defect ... can offset the price of one or more tool licenses."

1. Lack of Tool Availability

The lack of tool availability is usually the result of the following:

1. The tool is available, but you cannot get the funding for it.
2. There does not seem to be a tool on the market that does what you need or fits in your environment.

The first issue is based in management's understanding of testing and the priority it is given in terms of funding. While the cost of automated test tools is high compared with more common software packages such as office automation products and software development suites, the anticipated value of the tools is in reduced time and greater testing precision. Test automation promises increased productivity and accuracy, which is where the business case must be made. The cost of a single defect in most organizations can offset the price of one or more tool licenses. It is all a matter of where management chooses to spend the money – in defect detection or post-production rework, which is many times more costly than early defect detection.

Not having tools available in a particular environment is more troublesome. Although there is now automated tool support in most environments, this does not mean the support in every environment is great. In older environments, tool support is very limited.

If funding is the issue, consider the following:

- Measure the current cost of defects, especially in post-implementation rework. Use this information to help build a case for faster and more reliable testing using tools.
- Show the value of automated test tools for other groups besides testers such as the value of developers using the tools.

If getting a good technical fit is the issue, consider the following:

- Network with other testers to find information about lesser-known test tools. Online quality assurance forums are also good places to query people about tools in lesser-supported environments.
- Try to find tools that will work between platforms. This will likely require the use of PC-based emulators as opposed to host-based tools.
- Investigate the possibility of building your own tools or, at least, achieving a level of test automation using common scripting commands and comparison programs.

2. Lack of Tool Compatibility and Interoperability

Tool incompatibility and lack of interoperability are seen in organizations that have diverse technologies and applications. The desire is to be able to auto-

mate tests that bridge applications and platforms. This is a big challenge because most automated test tools have proprietary scripting languages and approaches, and, for the most part, competing vendor tools are not interoperable.

A related challenge is to automate identical tests on a number of different platforms. This requires tool compatibility among various computing platforms and the ability to share scripts between tools and platforms.

If compatibility and interoperability are the issues, consider the following:

- Select tools that have cross-platform capability to the greatest extent possible.
- Consider writing shell scripts and bridging scripts, perhaps in non-proprietary scripting languages such as Tcl.
- Evaluate critically whether the ability to perform cross-platform testing is a firm requirement.

3. Lack of Configuration Management Processes

Test automation is using software to test software. This means that items created using the automated test tools should be subject to the same level of control as any other software asset.

When software configuration management (SCM) is not in place for automated testing, the discipline is missing to work with the tools. The following occurs without SCM for automated test tools:

- Effort is duplicated because different people may each be building similar test scripts.
- Reuse is not realized because people are all creating test scripts for single-use purposes.
- Existing automated test scripts are at risk of corruption if they are modified without the knowledge of the original author.

Here is what is required for effective SCM for test automation:

- A workable process that everyone using the tool can understand and follow.
- A tool to manage the ownership, versions, and organization of the automated test scripts.
- Someone to own the SCM process and ensure that people are following it.

Many of the popular automated test tools have integrated test case and test script management applications. However, you still need the process and the people to make the SCM effort work. You can also build your own test management tool using a database and basic file organization to group related tests into suites.

A related issue is keeping up with changes to applications that are under test. This has been one of the biggest challenges in test automation since its inception. The degree of difficulty in dealing with application changes in automated testing software depends on the tool and the technologies involved. In the object-based world, the more robust tools can be configured to ignore user interface changes as long as the objects still behave the same. However, if the tool uses row-and-column positioning, then each application change will require a change (or many changes) to the automated test scripts.

In character-based applications such as mainframe Customer Information Control System applications, all of the changes that impact the user interface will most likely require maintenance to the automated test scripts that test those interfaces.

"If you do not know which tests are the most important and which tests are the most applicable for automation, the tool will only help perform a bad test faster."

Here are solution strategies for this challenge:

- During your tool search, consider the people and processes that will be required to manage the automated test cases and test scripts.
- If you are in the object-based environment (such as graphical user interfaces), look for tools that accommodate changes to the user interface gracefully. These tools cost more than those that do not offer such flexibility. However, many people have found that the added cost is small compared with the cost on continued manual maintenance of the test software.
- Consider automated test scripts as part of an application's configuration set.
- Involve the prospective test automation SCM person in evaluating test tools and their respective test management offerings.
- Investigate the use of existing SCM tools currently owned by your organization.
- Trace your automated test scripts to functional requirements and defects.

4. Lack of a Basic Test Process or Understanding of What to Test

Most automated test tools do not tell you what to test. Even the tools that have test-case generation features do so at a user-interface level and not at the functional-requirements level.

If you do not know which tests are the most important and which tests are the most applicable for automation, the tool will only help perform a bad test faster. This is a universal principle that I often illustrate with the example of a power tool.

Let us say that I want to build a bookcase. I try cutting the wood with a handsaw, but it is far too slow and laborious. So, I decide to go to the hardware store and buy a power saw. After the purchase of the tool that I can afford, that looks good, or that the salesperson convinces me to buy, I go home and start cutting wood. However, if I do not have bookcase plans or a very good understanding of how to build a bookcase, the saw will just help me make my mistakes faster! To be successful, I will need to first learn enough woodworking skills to understand not only the "what" and "when," but the "why" and "how" of building the bookcase. Then, I'm ready to use the tool effectively.

The tool vendors can train you to use the tool with all of its functionality, but the burden is on you to examine your own applications and determine which functions should be tested and to what extent.

Here are solution strategies for this challenge:

- Create a set of evaluation criteria for functions that you will want to consider when using the automated test tool. These criteria may include the following:
  - Test repeatability.
  - Criticality/risk of applications.
  - Operational simplicity.
  - Ease of automation.
  - Level of documentation of the function (requirements, etc.).
- Examine your existing set of test cases and test scripts to see which ones are

most applicable for test automation.
- Examine your current testing process and determine where it needs to be adjusted for using automated test tools.
- Be prepared to make changes in the current ways you perform testing.
- Involve people who will be using the tool to help design the automated testing process.
- Train people in basic test-planning skills.

5. Lack of Tool Ownership and Acceptance

The challenge with lack of tool ownership and acceptance is that the tool is not applied or is ignored. This is often the result of someone's good intention of buying a tool to make life easier, but the rest of the people do not use it. The following are some of the reasons for lack of tool ownership and acceptance:
- Difficulty using the tool.
- Not enough time to learn the tool and still perform normal work levels.
- Lack of tool training.
- Lack of management support for tool use.
- Lack of tool support, either internally or from the vendor.
- Tool obsolescence.

Here are solution strategies for this challenge:
- Do not cut the tool training. Training does not guarantee success, but without it you are at risk of tool abandonment.
- Have someone in your organization in the role of a *tool smith*. This person's job is to be the resident expert on the tools used for testing.
- Management needs to emphasize that the tool effort is important to them, and that tool usage is a required part of the testing process.

6. Inadequate Tool Training

We discussed the training issue previously in "Lack of Tool Ownership and Acceptance," but this challenge carries its own set of concerns. Some of the key issues are as follows:
- Skipping the vendor's training. The main motivation for this is lack of time and/or money. You will spend more of both without the training!
- Not getting the right training due to the incorrect selection of topics. For example, some tool users will need to learn in detail the tool's test scripting language, while other users will need to learn only the basic tool functionality.
- Inability to apply the training to your environment. This is where you learn to use the tool on the vendor's canned example but have difficulty getting the tool to work on your own applications.
- Trying to learn by self-study. Yes, it can be done, but it takes time and dedication. More often than not, people tend to spend time with only the basic functions and do not gain the benefit of learning the lesser known and perhaps more powerful tool features.
- Not enough time for training. This goes along with the "dive-right-in" approach often seen in information technology groups. When time is scarce, people tend to gravitate toward the easy and basic functions at the expense of not learning the more difficult but more powerful ones.

"Perhaps the greatest challenge seen in management support is balancing high expectations of tool benefits against the time, effort, and discipline it takes to implement the tool."

Here are solution strategies for this challenge:
- Include money in the tool proposal for training at least a core group of people.
- Match people to the most applicable training topics.
- Have tool training performed by the vendor at your location using some of your own applications as exercises.
- Find a skilled local consultant experienced with the tool to sit with your team for about three to four weeks to help get you started in creating automated tests. It is very important that your team does most of the work to accomplish the transfer of knowledge!

7. Incomplete Coverage of Test Types

As you profile your tests and defect types, you will often find a wide variety of test types that need to be performed. These include tests for the following:
- Correctness.
- Reliability.
- Security.
- Performance.
- Usability.
- Interoperability.
- Compatibility.
- Data Conversion.

Although the tool may be very adept at automating many of these tests, there may be test types that the tool simply cannot support. In fact, most organizations are very happy with a coverage level of 80 percent of their existing test case libraries.

Here are solution strategies for this challenge:
- During tool evaluation, prioritize which test types are the most critical to your success and judge the candidate tools on those criteria.
- Understand the tools and their tradeoffs. You may need to use a multi-tool solution to get higher levels of test-type coverage. For example, you will need to combine the capture/playback tool with a load-test tool to cover your performance test cases.
- Manage expectations by reminding people that 100 percent test type coverage is not likely. However, by automating 80 percent of the tests, you have time to deal with the rest manually.

8. Lack of Management Support

Management support is needed in designing and deploying test processes that will support the effective use of test tools, reinforce the role and use of automated test tools in the organization, and allow time for tools to be integrated in the testing process.

Without management support, the entire test automation effort is at risk. If management does not clearly and consistently show their support for test automation, people will be less inclined to show interest in using the tools. This is a major concern, especially considering that overcoming the learning curve of some tools requires dedication.

Perhaps the greatest challenge seen in management support is balancing high expectations of tool benefits against the time, effort, and discipline it takes to implement the tool. Management may become impatient about the lack of tool progress and shift their support to other initiatives.

The pressure is on the people who

made the business case for the tools to show progress in a given timeframe. The problem is there are many unforeseen things that can delay or derail a tool initiative. In reality, if people fully knew all of the future problems with any given effort, they would be very reluctant to proceed. While there is a place for optimism in acquiring tools, a heavy dose of realism is also needed to keep expectations in line with what is achievable.

Here are solution strategies for this challenge:

- Communicate that it takes time and planning to build a firm foundation of people, processes, and the right tools.
- When making the case to management for acquiring test tools, present the challenges as well as the benefits.
- Reinforce to management that they carry a great deal of influence in how people will accept automated test tools.
- Keep management informed of tool progress and issues that arise.

9. Inadequate Test Team Organization

Most test organizations learn that automated testing is a new world in terms of how tests are designed and maintained. Most tests require more than just capture/playback. The tool user must also be able to work with the tool's scripting language to accurately replay the test session. It helps if the tool user is comfortable working with coding languages, otherwise, there is a risk that the tool will not be used.

Here are solution strategies for this challenge:

- Add a person to the test team who is a *test scriptor*. This person should be comfortable in working with code and be able to take the basic test that has been designed by a test analyst and convert it into an automated script.
- Start simple with basic scripting concepts and add complexity later.

10. Buying the Wrong Tool

Buying the wrong tool is listed as the No. 1 challenge in test automation because no matter what kind of process or organization you have, if the tool is not a good technical or business fit, people will not be able to apply it.

We know that a good process and organization are also essential for test automation. However, if the tool will not function at a basic level, people using the tool will simply give up trying to use it.

Unfortunately, too few people do adequate research before buying a test tool. Adequate research includes defining a set of tool requirements based on what the intended users of the tool need to accomplish, developing a set of evaluation criteria by which candidate tools will be judged, and taking the experience of other people who have used the tools under consideration.

Here are solution strategies for this challenge:

- Take time to define the tool requirements in terms of technology, process, applications, people skills, and organization.
- Involve potential users in the definition of tool requirements and evaluation criteria.

"Adequate research includes defining a set of tool requirements based on what the intended users of the tool need to accomplish, developing a set of evaluation criteria by which candidate tools will be judged, and taking the experience of other people who have used the tools under consideration."

- Build an evaluation scorecard to compare each tool's performance against a common set of criteria. Rank the criteria in terms of relative importance to the organization.
- Perform a proof of concept (POC) as opposed to an evaluation. In a POC, the vendor often sends their technical team to your site to automate tests using your applications in your environment. Usually, a POC takes about one day to perform. The planning of the POC should be based on the evaluation scorecard. Testers should identify and define the most critical and most common tests they currently perform manually. These tests are often the ones that consume the most time and are the ones that offer the highest payback in test automation.

Summary

These 10 challenges are certainly not the only ones that are seen in test automation, but they are very common and have been the cause for many test automation project failures.

Successful software test automation is possible if fundamental issues are addressed and managed. Success depends on multiple factors that require the coordination of efforts between various groups in an organization. Automated software testing is truly a different way of testing and requires adjustments to current test methods and organizational structures. However, the payback from test automation can far outweigh the costs.◆

Additional Reading

1. Perry, William E., and Randall W. Rice. Surviving the Top Ten Challenges of Software Testing. Dorset House Publishing, Mar. 1998.
2. Fewster, Mark, and Dorothy Graham. Software Test Automation. Addison Wesley Longman, May 2000.
3. Dustin, Elfriede, Jeff Rashka, and John Paul. Automated Software Testing. Addison Wesley Longman, June 1999.

About the Author

Randall W. Rice is a leading author, speaker, and consultant in the field of software testing and software quality. Rice, a certified quality analyst and certified software test engineer, has worked with organizations worldwide to improve the quality of their information systems and automate their testing processes. Rice has more than 25 years experience building and testing mission-critical projects in a variety of environments, including defense and private sector projects.

Rice Consulting Services, Inc.
P.O. Box 891284
Oklahoma City, OK 73189
Phone: (405) 793-7449
Fax: (405) 793-7454
E-mail: rrice@riceconsulting.com