Transforming Test Automation:
An Unconventional Approach to Shift-Left by Removing Scripting from the Equation

Conventional test automation approaches are often suboptimal because of their dependency on user interface readiness and the high cost of script development and maintenance. A scriptless approach can help overcome these challenges by providing greater agility to test automation teams.

Executive Summary

In a quest to achieve accelerated software delivery, organizations are increasingly adopting Agile development methodology. However, maximizing the benefits of Agile software development requires testing to be performed concurrently with development, or in some cases before development itself. Agile-specific requirements of frequent builds for sustaining rapid sprints, minimal documentation and the need for active stakeholder involvement make development and maintenance of automation scripts challenging. In our view, an unconventional approach that eliminates scripting, but continues to provide the power and flexibility of a scripted approach, can avoid these glitches in Agile automation and improve efficiency.

This white paper advocates an unconventional, scriptless approach for organizations to overcome automation challenges and realize return on investment (ROI) more quickly, by significantly reducing the cost of test development and maintenance.

Script-based Test Automation Challenges

Historically, test automation has been perceived as a process in which tests drive an application through its user interface (UI). Our experience with UI-based test automation finds that the typical bottlenecks limiting ROI are in the development of test scripts. Scripted approach limitations include:

- **Test scripts are developed in a tool-specific language**, which non-technical users and business stakeholders do not understand.

- **There is often a steep learning curve before mastering the required technical skills.** Time lost in this learning process can have a significant negative impact on testing cycles.

- **Varying test developer competencies lead to inconsistent scripting standards**, which
adds to the overhead that project teams often face when searching for operational efficiencies.

- **UI-based test scripts lag behind development because of dependencies on a stable graphical user interface (GUI) object base before test development can begin.**

- **The uniqueness of the test development language** adds an additional dependency on niche skills that are often rare to find.

**Automation Challenges**

Traditional test automation techniques are found to be ineffective in meeting the business demands of rapid application delivery and faster time-to-market. In response, businesses are increasingly adopting Agile development, which introduces a new set of challenges.

- **Instant obsolescence:** Test automation is often vulnerable to instant obsolescence. In any Agile project, each new build has a good chance of breaking existing automation scripts as changes are welcomed in a highly adaptive environment. Project teams often find that test scripts become shelfware, as noted author and object-oriented guru Robert Martin explains (see sidebar).

- **Maintenance challenges:** Considering that development teams rely increasingly on process automation, automation components need to change frequently and invariably in response to changes in the application under test. The resultant expenses incurred in maintaining these scripts erode the advantages that automation can bring.

- **Lack of early feedback:** In traditional QA methods, the scripts are developed only after the application is built and stabilized. Early feedback, which is essential for reducing the cost of quality in Agile development, is therefore seldom available.

- **Considerable upfront costs:** Considerable upfront investments are needed due to complex scripting environments and tools, investments in the test infrastructure, and hiring and training of skilled resources.

- **Change of focus:** The primary job of the testing team has moved from testing the functionality of the application to technical development and implementation of scripts.

- **Lack of stakeholder involvement:** Finally, the key bottleneck in adopting traditional QA methodologies is the difficulty faced by functional and business teams in understanding automation scripts. Additional effort is required to make them accessible to non-technical users, leading to diminished collaboration between the functional experts and automation teams.

**Defining Scriptless Automation Testing**

Scriptless automation testing refers to the development of tests in an easily understandable and user-friendly manner that makes it easier for all stakeholders to contribute. Aimed at reducing testing effort and maximizing productivity of automation testers, it eliminates scripting, enabling business and functional experts who do not have any programming skills to create automated tests.

**Components of a Scriptless Testing Model**

Scriptless testing depends on the use of keyword abstractions as inputs to build test steps and sequences dynamically. For example, a test script for a set of user actions is represented by a sequence of easy-to-understand keywords. These keywords are then automatically translated to concrete low-level scripts in memory.

---

**Quick Take**

**Robert Martin on Automated Tests Becoming Shelfware**

“One of my clients had to run over 10,000 acceptance test cases, and running them through the UI took him several days using dozens of machines. Furthermore, the tests were written in a code-like language that made them difficult to understand. Over time, he lost track of what each test was trying to verify; all he knew was that all 10,000 tests had to pass. Even tiny changes to the UI caused dozens, if not hundreds, of tests to fail or become inoperable.” - Robert Martin (excerpted from “The Test Bus Imperative: Architectures that Support Automated Acceptance Testing,” IEEE Software, Vol. 22, Issue 4, July-Aug 2005.)
Tests can thus be translated to a spreadsheet and saved in a comma separated values (CSV) format. Therefore, these simple tests can be created from low-fidelity prototypes or wireframes, and the automated test case design can begin even before the application is developed. A simple low-level abstraction of the actions and inputs of the objects on screen is illustrated in Figure 1.

The listed object names are meta-names that serve as placeholders. The meta-names accelerate early test design by allowing non-technical testers to define test navigation flows and business components before the application is developed. With the traditional scripted approach, this would not have been possible, as the objects need to be defined upfront. As the test case now consists of a series of actions on objects and their data combinations, any change in an underlying object can be easily incorporated in the script without a lot of overhead.

By maintaining a central object repository, any changes made will reflect across test cases, saving considerable time and effort in script maintenance. The underlying scripting engine interprets the keywords and performs the execution in the automation tool.

**Accelerated Benefits**

By packaging a scriptless approach with the standard features of any automation framework, such as data parameterization, modularization and error handling, and an easy-to-use test design and maintenance interface, the automation platform improves ROI. Using an intuitive drag-and-drop feature to design the test steps accelerates test design. With an in situ, Excel-like tabular view and functionality, test data parameterization is simplified. GUI-driven object management involving capturing, ordering and healing eases the maintenance effort, enabling the lean practice of just-in-time object to meta-name mapping.

This approach accelerates the design of the tests for the Agile projects that work in two-week sprints, such that the time invested in test design is almost concurrent with development. The design can start with the wireframes, combined with the navigational flow information. With test data generation techniques that can run in parallel, the lag with development for test automation can be greatly minimized or eliminated.

The elegance of this approach, combined with the simplicity of the implementation in an object-oriented language that leverages the open-closed principle, can result in an architecturally powerful solution. The implementation lends itself to the powerful and much-needed ability to extend the out-of-the-box abstraction of the interactions using the low-level keywords, which can very quickly be exhausted due to the diversity of technologies that have permeated the GUI market. These extensions increase the applicability and usability of the approach to significantly increase the reusability of customization across projects.

**Primary Advantages of a Scriptless Approach**

The scriptless approach addresses most of the Agile automation challenges highlighted above.

- **Business-friendly:** The scriptless and English-like nature of the tests facilitates the involvement of business stakeholders and serves as a catalyst for interaction between business analysts and technical experts. It is easier for the business analysts and functional experts to discuss and review test cases.

- **Eliminates technical debt:** Incorporating smarter framework design upfront can eliminate the future costs that spring from increased script maintenance effort. The technical debt associated with scripts - much like the code developed in agile methodology - is eliminated.

- **Rapid script design and development:** The simpler test design and development process, combined with the intuitive UI, enables quicker development of tests.
**Quick Take**

**Going Scriptless Down Under**

For one of our Australian clients in the information, media and entertainment industry, the scripted approach to test automation introduced challenges such as:

- Increased risk to application quality due to lower test coverage.
- Increased testing turnaround time.

Our Agile scriptless solution delivered significant savings in cost and effort of test design and maintenance. Cognizant leveraged its scriptless solution in the highly Agile engagement. We achieved significant reduction in test design and maintenance efforts that resulted in improved quality through more testing in the same time, and provided faster feedback from automated testing. This helped the customer achieve its objectives to go to market with a higher level of confidence and reduced risk.

**Non-linear Benefits of Using Scriptless Automation in Practice**

![Scriptless in Action](image)

<table>
<thead>
<tr>
<th>Cost/Effort</th>
<th>Time/# of scripts</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Scripted</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Scriptless</strong></td>
<td></td>
</tr>
</tbody>
</table>

**Looking Forward**

Scriptless automation is an unconventional testing approach that begins improving quality and cost savings at the very start of the development phase. By making automation simpler and easier, stakeholders can play an active role during the test automation process. Testers can focus on their primary objectives of test designing and execution, instead of developing a framework and coding that enables rapid testing.

Scriptless automation is, therefore, an ideal solution to shift-left and cater to the requirements and demands of Agile testing.
Footnotes

About the Authors
Venkataraman Moncompu (Venkat) leads the Automation QE&A Practice within Cognizant’s Technology Center of Excellence (CoE). He is an Agile testing evangelist and enthusiast with 17 years of relevant industry experience. Venkat has worked with teams that have mastered the art of Agile software delivery, as well as teams that have struggled through the process of adoption until abandoning the Agile development methodology altogether. He has a master’s degree in engineering from Arizona State University and is a published author. As a renowned speaker in the fields of user experience for driving effective test strategies, Agile development and test automation, Venkat has participated in several conferences, including Pacific Northwest Software Quality Conference, Agile Testing Days and Software Quality Days. He can be reached at Venkataraman.Moncompu@cognizant.com.

Kamala Kannan Sankar Raj plays the role of a product evangelist and quality advocate in Cognizant’s Technology CoE. He has over three years of industry experience in test automation and business development. Kamala Kannan has a post-graduate degree in business administration from Indian Institute of Technology-Kharagpur, West Bengal, India. Kamala Kannan can be reached at KamalaKannan.SankarRaj@cognizant.com.

About Cognizant
Cognizant (NASDAQ: CTSH) is a leading provider of information technology, consulting, and business process outsourcing services, dedicated to helping the world's leading companies build stronger businesses. Headquartered in Teaneck, New Jersey (U.S.), Cognizant combines a passion for client satisfaction, technology innovation, deep industry and business process expertise, and a global, collaborative workforce that embodies the future of work. With over 75 development and delivery centers worldwide and approximately 187,400 employees as of June 30, 2014, Cognizant is a member of the NASDAQ-100, the S&P 500, the Forbes Global 2000, and the Fortune 500 and is ranked among the top performing and fastest growing companies in the world.

Visit us online at www.cognizant.com or follow us on Twitter: Cognizant.