Top Five Secrets for a Successful Enterprise Mobile QA Automation Strategy

From tool selection through choosing the best framework, here are five ways quality assurance teams can tilt the odds of successful digital transformation in their favor.

Executive Summary

Digital approaches to doing business are nearly ubiquitous, driven in part by the spectacular growth in enterprise mobile computing. Over the years, the mobility space has become highly fragmented, with the evolution of numerous platforms, operating systems, device form factors and steep technology changes, from mobile Web, native and hybrid mobile enterprise application platforms (MEAP), through the current rage: responsive Web design (RWD).

However, consolidation from within the platform and among the technology stacks is beginning to take hold. Such consolidation is further affirmation of mobile’s rise as the undisputed channel for business success; as a result, mobile quality assurance (QA) and mobile automation have become pivotal to creating an effective enterprise digital strategy.

This white paper unveils what we believe are the top five secrets for a successful enterprise mobile automation strategy. These recommendations go beyond successful functional testing, to ensuring integrated continuous delivery and a seamless customer experience.

Select Automation Tools Wisely

Automation tool selection is among the most important decisions that directly influence the success of the overall testing strategy. Traditionally, tools have been classified as object-based or image-based. However, QA organizations must also consider other parameters that focus on the long-term vision that influences overall tool selection (see Figure 1, next page).

In today’s digital world, omnichannel experience is critical; as a result, nearly all digital initiatives revolve around RWD. This adds complexity to tool selection, and raises additional technical considerations, such as script re-use across mobile devices and desktops – not just reusability between mobile devices and platforms. Therefore, the new mandate for selecting mobile tools for automation includes desktop support.
Because the tools landscape is continuously evolving, a wise choice for a successful long-term strategy is to not just rely on supported features of the tool but also on technologies that are not too tightly coupled with the tool. While a portion of the QA investment is dedicated to tool license costs, a majority of the effort and expense is often applied to building the framework and automation suites. Therefore, it is imperative to choose technologies that enable re-use of the automation script across tool sets with minimal effort. This approach considerably improves investment confidence and helps overcome the initial barriers faced in mobile automation tool selection.

It is imperative to choose technologies that enable re-use of the automation script across tool sets with minimal effort.

Another issue that strongly influences the tool selection process is tool maturity and scalability for operating system version support. With mobile devices supporting newer OS version upgrades – as well as customers' rapid adoption of new OS versions by default – it is no longer sufficient for a tool to support a particular mobile OS; however, it must support the latest OS version. For example, in the case of Apple's iOS, users typically feel compelled to upgrade to the latest version within days if not weeks; as a result, support for the latest version is a “must-have” feature.

Additional selection parameters include open integration with other tool sets in the ecosystem (i.e., Jenkins or ALM), strong community support and a shorter learning curve.

Enable Continuous Delivery and Rapid QA

At a time when “born digital” companies are deploying several software releases daily, traditional enterprises are pushing developers to push code out as frequently as possible. In order to increase speed without compromising quality, the mobile automation strategy must extend beyond functional validation and semi-automated approaches. In a DevOps environment, it is crucial to create an ecosystem that breeds success.
From code deployment and build, to continuous integration and continuous testing, it is important to choose the right tool and integration touchpoint across all stages of the software development lifecycle (SDLC). This ecosystem will power continuous delivery and a fully automated system.

While fully automated systems can accelerate the testing process, shift-left automation is necessary to ensure rapid QA. By introducing application programming interface (API) automation, organizations can ensure that functional defects are identified at an early stage of the lifecycle. API automation also ensures quality and reduces defect resolution time—a necessity in today’s distributed development age, when dispersed teams consume each other’s services.

**Automate Beyond Functional Coverage**

Although functional coverage is among the oldest tenets of a successful automation strategy, this is changing in the digital world. Traditionally, manual test cases are evaluated for automation, based on technical feasibility and QA coverage. However, in the context of digital and mobile environments, it takes a different form altogether.

Given the simplicity of most mobile apps, and the unstable connectivity of mobile networks, it is recommended for QA organizations to create small chunks of automation scripts to address specific business cases, rather than a lengthy flow that covers the complete scenario. Additionally, as most mobile projects are delivered using the Agile model, it is better to choose test cases with maximum coverage against the user story, rather than focusing on coverage against manual test cases.

At a time when success is tied to customer experience, automation must go beyond the appropriate selection of functional test cases and also certify the best customer experience.

**Support a Flexible Execution Environment**

One of the most disrupted areas in the mobile QA ecosystem is the infrastructure or mobile device laboratory. The landscape looks crowded, with long-timers such as DeviceAnywhere, Perfecto Mobile and Mobile Labs—as well as relative newcomers, such as Experitest, Sauce Labs, BrowserStack, eggCloud, AWS Device Farm, HP Mobile Center, and Google Mobile Lab—all vying for customers.

While feature-set and price wars continue unabated among players, it is crucial to ensure the mobile QA strategy is not influenced by the test execution environment. It is imperative, therefore, to decouple the scripting environment from the execution environment and from the strategy supporting the hybrid environment (physical devices plus cloud).
Use the Right Framework

Designing the right framework is vital for the long-term success of the mobile automation strategy. In an Agile environment, the key objectives of the framework design are to speed implementation and reduce maintenance. Efficient implementation of data handlers, page object modeling, reusable libraries, multilingual capabilities, keyword-driven test case implementation, debugging support and exception handling can significantly reduce the effort involved in scripting (see Figure 2).

This structured implementation and strong decoupling of the application from the scripts and the scripts from the external interfaces to the continuous integration or execution environment ensures swift adaptation of the automation suite for changes within the app. This significantly reduces the maintenance effort in an Agile world.

Looking Forward

With increasing focus on Agile environments and the expanded role of mobile apps across the digital world, successful enterprise-wide mobile automation strategies are critical to business success. While the above recommendations provide guidance on the approach, we recommend they be tailored to an individual enterprise context.

All company names, trade names, trademarks, trade dress, designs/logos, copyrights, images and products referenced in this white paper are the property of their respective owners. No company referenced herein sponsored this white paper or the contents therein.
Footnotes

1 Jenkins is a continuous integration server. For more information, see https://jenkins.io/.

2 Automated lifecycle management. For more information, see https://en.wikipedia.org/wiki/Application_lifecycle_management.

3 DevOps is the process of integrating software development and infrastructure.

4 Early in the software development lifecycle.

About the Author

Subbiah Muthiah is an Associate Director within Cognizant’s Quality Engineering & Assurance (QE&A) Center of Excellence (CoE). He is responsible for techno-business development and building quality assurance services for trending themes, such as digital services, connected devices, wearables and telematics. Subbiah has more than 15 years of relevant industry experience with in-depth knowledge across the mobile stack – from the application layer to radio frequency. He holds a bachelor’s degree in engineering. Subbiah can be reached at Subbiah.Muthiah@cognizant.com.

About Cognizant

Cognizant (NASDAQ: CTSH) is a leading provider of information technology, consulting, and business process 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 100 development and delivery centers worldwide and approximately 233,000 employees as of March 31, 2016, 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.