



Ultimate Shift Left: How to Test (Much) Earlier So You Can Go to Market Much Faster

Author

Guy Arieli is the Chief Technology Officer for Experitest. He has over 18 years experience in test automation with Hewlett Packard, Cisco, and 3Com. Guy founded and sold the largest local Test Automation services company – TopQ to Top Group, a publicly-traded technology group. He administers the largest Test-Automation forum online and is a keynote speaker at events worldwide. Mr. Arieli holds a B.Sc. from Israel's world-renowned Technion.

Table of Contents:

The need: Better enterprise apps, more often

The challenge

The solution: The ultimate shift left

A key challenge to the shift left: object identification

Tips for overcoming the object identification challenge

Object identification with XPath

5 best practices for developing apps that are ready for automation

How SeeTest automation by Experitest revolutionizes testing automation

Executive Summary

With smartphone users in the billions and with most of the time being spent in apps – enterprises know that they need to differentiate themselves and the services and products they provide. Accordingly, the race is on to release value-added apps as quickly as possible and provide a superior customer experience with app performance and functionality that is flawless.

This puts a lot of pressure on developers to get out apps and their latest versions as quickly as possible. The key to expediting release cycles, and ensuring app quality depends on thorough and reliable testing and taking app testing automation to the next level with shift left testing. In this eBook we will provide tips for winning with the shift left approach, as well as the main best practices for developing apps that are ready for testing automation.

The Need: Better Enterprise Apps, More Often

With smartphone users in the billions and with most of the time being spent in apps – enterprises know that they need to differentiate themselves and the services and products they provide.

6.1 billion smartphones to be in circulation **by 2020**¹
85% of smartphone time is spent using **applications**²

Accordingly, the race is on to release value-added apps as quickly as possible and provide a superior customer experience with app performance and functionality that is flawless. Furthermore, they know that they need to do this on a continuous basis, with new releases coming out at a high frequency.

This puts a lot of pressure on developers to get out apps and their latest versions as quickly as possible, with **80% of developers** pushing out **updates at least monthly**.³

*“By 2019, the average update cycle for internally developed mobile apps will be less than four weeks” Gartner.*⁴

(1) Ericsson Mobility Report 2016, [link](#)

(2) Tech Crunch, [link](#)

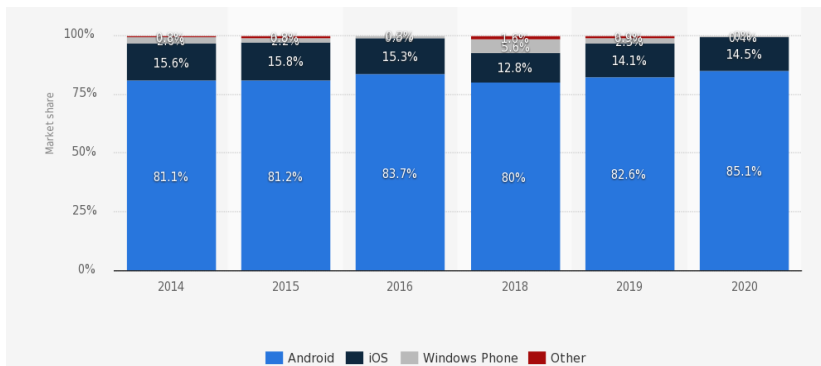
(3) Evans Data, [link](#)

(4) Gartner, Market Guide for Mobile App Test Automation Tools, 2016

The Challenge

“One of the greatest challenges of mobile testing is determining what devices, carriers and OS builds to test on. It’s not possible to test an app on every possible hardware and software combination,” Gartner .⁵

Although most mobile phones run on “only” four major operating systems, that is Android, iOS, Windows Phone, and BlackBerry,⁶ each operating system has multiple versions and vary on many parameters. The variations run in the hundreds.⁷ Needless to say, this makes for a very challenging testing environment.



Mobile operating system market share. Source: [statista](#)

Accordingly, being able to meet growing demands, developers are finding it difficult to properly test their applications.

- Only **5% of developers** ship apps with zero defects⁸
- **20% of developers** ship with between **11 and 50 bugs**⁹
- Testing usually done on **limited subset of devices** and platform versions.¹⁰

The result of the rush and ship strategy can result in irrevocable harm to the customer base, with **many consumers deleting** an app if they find **a single bug**. The implications – lowered revenue and higher cost of keeping existing customers and acquiring new ones.

78% of consumers have abandoned a transaction or not made an intended purchase because of a poor service experience.¹¹

On average, loyal customers are worth **up to 10 times** as much as their first purchase.¹²

(5) Gartner, Market Guide for Mobile App Test Automation Tools, 2016

(6) Net Market Share, [link](#)

(7) GSM Arena, [link](#)

(8) Evans Data, [link](#)

(9) Evans Data, [link](#)

(10) Evans Data, [link](#)

(11) American Express Survey, [link](#)

(12) White House Office of Consumer Affairs, [link](#)

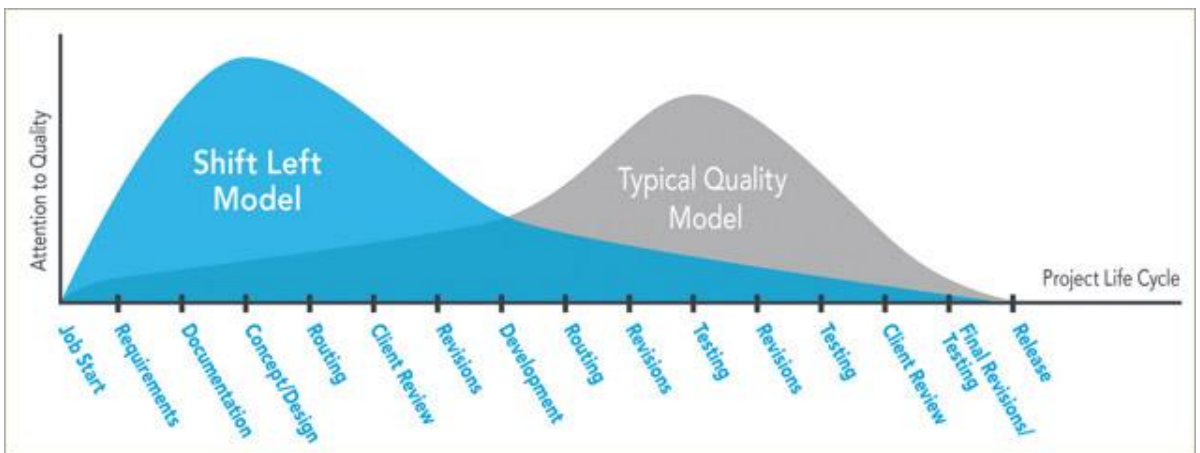
The Solution: The Ultimate Shift Left

Overcoming these challenges to expediting release cycles, and ensuring app quality depends on thorough and reliable testing and taking app testing automation to the next level with shift left testing.

Shift left testing is about building the testing automation into the development stage, where cross-functional teams work simultaneously – as based on the principles of Agile.

Agile enables teams to react quickly to changes in customer needs, where meeting these needs is a pre-requisite for competing effectively in today's demanding market. Shifting left enables faster development and test cycles, a faster time to market, and greater customer satisfaction.

*“There is a strong demand for capabilities in these areas because of the increased complexity of solution delivery, exposure to the market and demand for faster delivery”
Gartner .¹³*



Shift left – quality considerations as a function of the stage in the project, shift-left testing and typical testing model.

A Key Challenge to the Shift Left: Object Identification

This shift left, however, is not without its challenges. A key challenge to this approach is embedding testing automation in the development process and identifying objects in the user interface when attempting to mimic user operation.

This is all the more challenging when developers layer the information during the development phase. This typically results in objects that are not identified uniquely and which are organized in a complex hierarchy. When the app reaches testing, these two parameters make for a more complex and challenging testing process.

Tips for Overcoming the Object Identification Challenge

The key to avoiding this situation is to assign a unique property to every object at the beginning of the development phase. Then, a testing solution should be implemented that can return every element that meets query parameters.

The following are properties that should be assigned to every object to enable the shift left development approach:

- **Is the element visible?** Developers should note whether the object is on screen and visible to the user, since users can only perform actions on objects that are visible. So if an object is covered by a menu or for any other reason, its `@onScreen` property should be changed accordingly.
- **Is the object property consistent?** Object properties should be stable with every new build of the application. Developers should always keep an automation oriented development methodology to ensure every element is easy to query.

Additionally, there are properties that should not be used as an identifier. For example:

- **Is the element dynamic?** Elements that display contents that change frequently throughout the app's operation should not be used as an identifier. For instance, a text field that presents an account balance should not be used as an identifier since it changes constantly with every transaction.

Object Identification with XPath

A tool that is recommended for effective object identification is XPath. XPath is a query language that is used to identify and classify objects in XML files. It is broadly used in web applications, websites, Android application, iOS application, and many more.

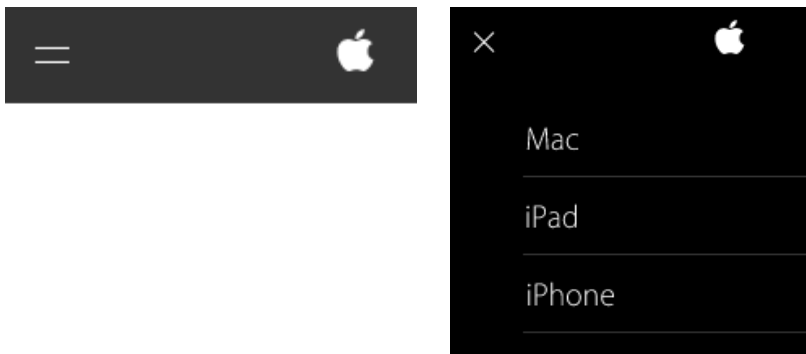
Its benefits include its ability to identify a node, that is – an element in a hierarchy (i.e. the user interface), by addressing each node with many search filters (attributes) such as `@text`, `@id` or `@onScreen`.

Xpath also enables developers to go up or down the tree, regardless of where the aim is.

5 Best Practices for Developing Apps That Are Ready for Automation

Although UI navigation tools may have their limitations, nevertheless there are five best practices that can help us ensure effective and reliable automation of quality assurance processes.

1. **Add a unique identifier:** always use meaningful names to describe elements as opposed to general identifiers such as “button” or “textView.”
2. **Build stateless functions:** that is, functions such as ‘menu,’ should operate regardless of the state or page they are on.



A stateless function – Menu button on the apple mobile website

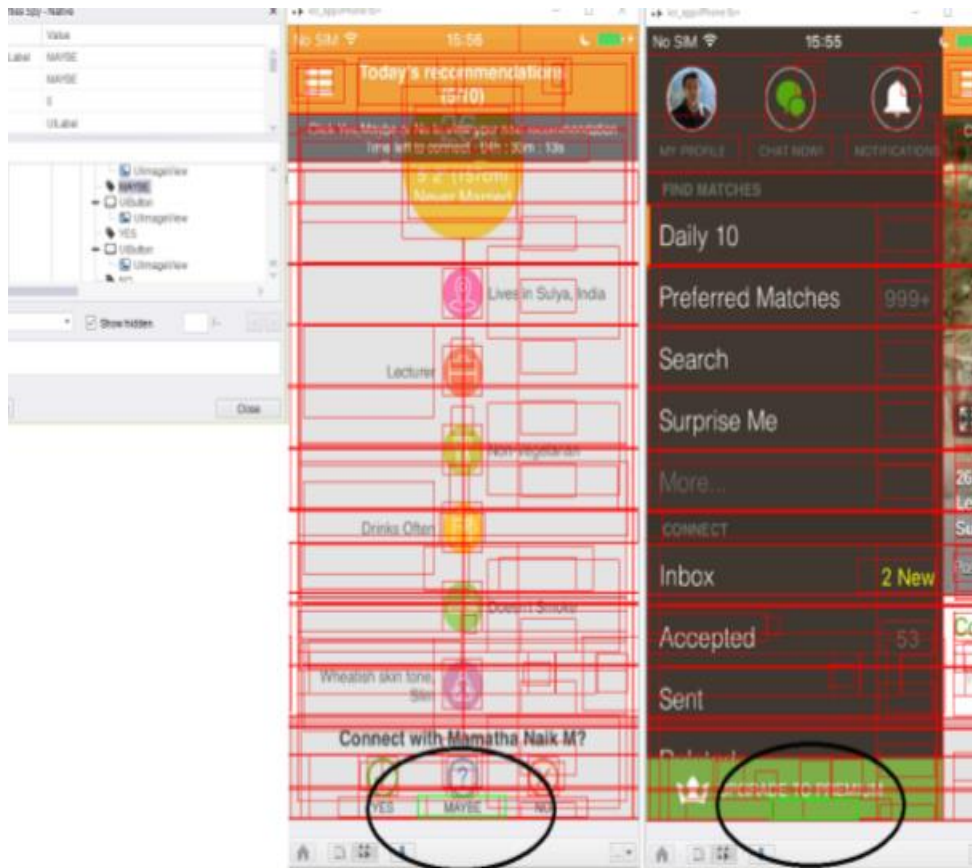
3. **Do not “tweak” UI object events:** instead, let the OS handle as many events as possible, while still maintaining the desired logic. This will ensure maintainability and handling.



An example for “tweaking” an event – delegating a gesture to self.

5 Best Practices for Developing Apps That Are Ready for Automation

4. **Avoid proprietary UI elements:** it is best to define every element to meet your exact needs. Using the OS native property and methods ensures maintainability and prevents crashes when updating the OS.
5. **Avoid object layering:** complex layering makes it almost impossible to define elements in the UI. It is logically incorrect (in terms of the UI) to layer pre-existing elements with new content.



An over complex layering – reflections on the left and on the right present the same object, and the difficulty of identification.

The Cloud Advantage

Being able to shift left and automate testing in a cloud based lab environment drives testing effectiveness and efficiency even further.

- **Agile driven:** a cloud based solutions enables development and testing teams to work simultaneously on the same project even when located in different sites and regions. That enables around the clock operation and constant monitoring.
- **Remote debugging:** enables testers to debug applications as they are being developed at a remote site, yet form within the developer's environment (i.e. Xcode or Android Studio) which bring testing consideration to the development process.

How SeeTest Automation by Experitest Revolutionizes Testing Automation

SeeTest Automation by Experitest is a robust mobile app testing tool that accelerates time to market, delivering five-star apps to consumers while decreasing the testing cycle and increasing ROI.

Experitest provides Automation tools, cloud services and network virtualization tool that combined, provide the upmost advanced testing environment for the mobile industry. It is the only solution available today that supports all peripheral testing requirements, simulates on device cameras, enables mock location and audio insertion, along with many other capabilities – that contribute to a greater than 60% reduction in the manual tests required today – making it the tool for the ultimate shift left.

Trusted by
over 1000
Enterprises



Get started today! [Download SeeTest Automation now >>](#)