# How to Deploy an Enterprise Test Automation Strategy for Mobile Applications

## TABLE OF CONTENTS

- Understanding the Mobile Environment ................................................................. 3
  - Mobile is Fragmented .......................................................................................... 3
  - Mobile is Dynamic .............................................................................................. 4
  - Mobile is Local .................................................................................................... 5
- Best Practices for Mobile Testing ............................................................................. 6
- Adapting Your Software ALM for Mobile ................................................................. 6
- Platforms .................................................................................................................. 7
- Networks .................................................................................................................. 8
- User Experience (UX) ............................................................................................. 8
- Mobile Is Not Simple ............................................................................................. 9
- Addressing Mobile Business Needs ......................................................................... 9
- ALM integration ...................................................................................................... 9
- Extend, don’t integrate .......................................................................................... 9
Device Agnostic ........................................................................................................... 10
Test on real devices ....................................................................................................... 10
Cloud-based access ....................................................................................................... 11
Collaboration ................................................................................................................ 12
At a Glance: Business Needs and Their Implications on Mobile ....................................... 12
Mobile Testing Phases .................................................................................................. 13
Automated Regression Testing ...................................................................................... 13
Multi-Platform Development ......................................................................................... 13
Multiple Screen Sizes .................................................................................................... 14
Key Takeaways for Automated Regression Testing ....................................................... 14
Exploratory Testing ......................................................................................................... 14
Mobile Experience is Different on Different Devices ..................................................... 14
Mobile Testing is Done on ... Mobile Devices ................................................................ 15
Mobile Testing is Local .................................................................................................. 15
Key Takeaways for Mobile Exploratory Testing ............................................................. 15
Compatibility Testing (Porting) ..................................................................................... 15
Plan for Platforms: Defining Your Compatibility Testing Strategy ................................. 16
Key Considerations for Compatibility Testing ............................................................... 17
Interoperability Testing ................................................................................................ 17
How to test mobile application interoperability ............................................................ 18
Conclusions .................................................................................................................... 18
About Perfecto Mobile .................................................................................................... 19
UNDERSTANDING THE MOBILE ENVIRONMENT

Mobile services and mobile web-based applications are vital to your business. Your customers demand an exceptional mobile experience, while your mobile employees depend on remote access to enterprise services and information to get work done more efficiently.

To take advantage of the huge business opportunities offered by mobile applications, enterprises need to understand the unique characteristics and requirements of the mobile environment. In particular, to ensure the highest possible application quality, they need to be aware of the implications of this environment on their mobile testing strategy.

In today's extremely fragmented and dynamic mobile market, maintaining application quality has become a daunting task. The complexities of multi-platform development, coupled with the ever-growing number of models, operating systems, screen sizes and network technologies, make it practically impossible to keep your mobile apps and services in sync with ever-changing market trends.

In terms of Application Lifecycle Management (ALM), the mobile environment represents a paradigm shift from the world of desktop and web applications for several reasons:

MOBILE IS FRAGMENTED

Desktop and web applications are developed for relatively homogeneous markets. “Win-Tel\(^1\)” has dominated the desktop market for decades, while “Internet Explorer, Firefox and Chrome account for 90% of web browsers. Unlike the desktop world, mobile applications must take into account at least three major operating systems (i.e., iOS, Android and Blackberry and in some cases even Symbian, Windows Phone or others based on the target market) within their ALM process. To make matters even more complicated, each OS has multiple versions. The dynamic nature of Android, for example, which rolls out a major release every 6 months (Honeycomb, Ice Cream Sandwich, etc.) inevitably adds to the fragmentation as some users upgrade their devices while others do not. (Depicted in figure 1a below)

Moreover, as mobile handsets are marketed as a "consumer electronic" device, segmentation is part of the game. To win over consumers' hearts (and pockets) in this highly competitive segment, handset manufacturers offer endless models and combinations of features – from screen sizes, landscape/portrait screens, various screen resolutions and touch to keypads trackballs and sensors. In order to cover 80% of this highly fragmented market, you would probably need several Android handsets from multiple manufacturers with various OS versions, at least two iPhones with at least two different OS versions, around 5 different BlackBerries, etc.

\(^1\) Windows - Intel
To allow for the personal touch at work, enterprise IT organizations have started migrating from “one device fits all” to a “bring your own device” (BYOD) approach. Therefore, whether your company is developing an inter-organization application or a customer-facing application, mobile fragmentation must be taken into account.

### MOBILE IS DYNAMIC

"Here today, gone tomorrow" is probably the best way to characterize the pace of change in the mobile market. New smartphones and tablets are released to the market on a weekly basis, each with a unique form factor or set of features. As opposed to the desktop or web market where technologies from last year are still relevant today, in the mobile world last year’s offering is already obsolete. Accordingly, mobile application developers need to constantly “chase” the market in order to remain relevant.
MOBILE IS LOCAL

While web applications may behave differently in different geographies, the factors of location and network are much more important in mobile apps for the following reasons:

- **Networks** – Currently, "network neutrality" does not exist in mobile networks. In other words, mobile operators interfere, prioritize, change and manipulate the traffic to meet their business
needs. In addition, different network technologies (e.g., 3G, WCDMA, LTE, Wi-Fi) might change the way an application behaves.

- **Location Based applications** – Since your mobile device is with you at all times, your location (in the store, downtown, airport, country, near someone, etc.) could be an important input for your application. This is not the case in desktop or web-based applications.

- **Regulatory compliance** - For some industries (such as Healthcare and Finance) in certain countries, regulatory compliance is also a factor for testing. Regulations may stipulate that application activation MUST take place within the proper location/country and activation may be prohibited outside of a particular country.

To address the particular business and technological challenges of the mobile environment, enterprises need to implement a new mobile testing paradigm that will ensure the reliability of business-critical mobile applications.

**BEST PRACTICES FOR MOBILE TESTING**

**ADAPTING YOUR SOFTWARE ALM FOR MOBILE**

Clearly, in organizations with an established ALM framework and skilled resources already in place, a key criterion when “going mobile” is to change as little as possible in the current ALM infrastructure and teams. The basic assumption is that your organization’s existing quality standards and business objectives need to be maintained for mobile applications. There is no need to “reinvent the wheel” and no need to overhaul the existing QA structure within your organization.

The standard ALM process for software applications is depicted in Figure 4 below:

---

**Figure 4 – Standard ALM Process**
Figure 5 highlights the new factors that must be taken into account when testing mobile applications. The key changes introduced to the mobile ALM process relate to the areas of platforms, networks and user experience (UX). These factors must be taken into account in the requirements, planning, test creation and test execution phases.

Another important factor is version timing and some organizations are realizing the difficulties of using the traditional “waterfall” approach for mobile application development. The extreme dynamics of the mobile market renders traditional waterfall development ineffective. The market simply moves too fast to allow for a development cycle of more than few months. Anything beyond that becomes problematic. See above graphs for OS trends to better understand this issue.

**PLATFORMS**

The multitude of mobile platforms (i.e., handset + OS + version) introduces a new consideration to mobile testing strategy. Product management needs to define requirements regarding mobile platforms and determine what represents the market most effectively. Since you can’t cover every possible platform, a critical part of the requirements phase is to plan in advance how many platforms and which variations you wish to test. For example, is one device per OS enough? Is it sufficient to test on the latest version of each OS? If not, how many previous versions is enough? These are critical questions when dealing with mobile applications.
When planning your mobile testing, tests should cover a selected list of reference platforms. In most cases, you should choose a limited list of reference devices you will use for the basic development phases including actual development, regression testing and nightly build sanity testing.

Choosing too many reference devices will hamper your ongoing efforts; 4-6 reference handsets are probably a good number to consider. The reference devices should cover as many important aspects as possible (e.g., platforms, touch/no touch, portrait/landscape, etc.). This will allow faster detection of possible problems. During the QA phase, it is recommended to extend the testing to cover ~10 "major" devices to achieve a better representation of the market.

**NETWORKS**

Quality of the network varies between carriers, states, countries, regions and even areas (especially in populated areas). If your application is sensitive to network latency, requires high bandwidth or short response time, spend time on proper planning of network related testing.

For example, if you are developing an application for the US market, it is critical that your application runs seamlessly on both AT&T and Verizon networks. Since network operators typically prioritize, re-route or manipulate the traffic based on various performance needs, it is important to test on live networks. In addition, different network technologies (3G, WCDMA, LTE, Wi-Fi) might change the way your application behaves.

If your teams are geo-distributed, all of them will need access to devices in different part of the globe. If this is the case, don’t “bet” your quality on your ability to send those teams devices and accept that roaming will work nicely. To put it bluntly, it won’t. There are no shortcuts here - you will need to test your mobile application on real devices on the real networks.

**USER EXPERIENCE (UX)**

Usability testing has always been an essential part of application testing. In the mobile environment, due to the multitude of factors related to user interface (e.g., screen resolution, size, orientation and touch/no touch), this type of testing takes on added importance.

The visual aspect of mobile testing is more important than desktop testing. Market statistics indicate that many of the problems normally detected with mobile applications relate to the screen resolution and orientation. The amount of “free space” on the average mobile screen is much smaller than the amount on the average desktop application. For example, devoting 10% of the screen to a single button (or any other UI element) is normal in mobile but would look ridiculous in a desktop application. Rearranging UI elements under different resolutions and orientation have a greater effect on the mobile user experience than on a desktop application running on a standard (normally much larger) desktop screen resolution.

For this reason, the subjective level of validation (particularly on visual “looks good” testing) is much greater in mobile application testing. This type of testing needs to be performed manually.
MOBILE IS NOT SIMPLE

It is very important to understand that developing software for mobile is not fundamentally different from any other software development process. Mobile applications have long passed the “simple” bar and are today at least as complex as web-based or desktop applications. If anything, the factors described above introduce a new level of complexity to the game. For that reason, it is almost critical that you treat mobile development and QA as an evolution of your existing investment based on your current development methodologies and not as a “side project”. You have probably invested a great amount of money and resources in your ALM infrastructure, methodologies, staff training and processes. The best approach is to take advantage of this existing framework when going mobile.

ADDRESSING MOBILE BUSINESS NEEDS

The diverse and dynamic mobile environment has introduced new business needs and potential pain points that must be addressed by your mobile testing strategy:

ALM INTEGRATION

For enterprises that have already invested in building an ALM infrastructure, implementing tools and workflows, and training your QA team, the goal is to preserve and leverage these assets by extending your existing ALM framework to support mobile testing. This will help to eliminate the costs associated with maintaining, managing and synchronizing separate systems for desktop and mobile testing.

Tip: Eventually, your executive level will want to see progress reports and understand why the version for mobile is not yet released. When planning your mobile ALM, be sure that every important management feature in your “non-mobile” environment will also be available in the “mobile” environment. By managing mobile, desktop and web application testing through a single integrated ALM platform, organizations can centrally manage and track all application projects, leverage the skill sets of their teams and ensure consistent workflows and processes.

The extension of mobile testing within your existing ALM enables centralized and complete test management and cross-organizational visibility for your mobile apps. Using a single platform for managing and automating the application lifecycle helps to ensure mobile quality and performance while reducing total cost of ownership.

EXTEND, DON’T INTEGRATE

If you don’t want to find yourself eventually managing two disparate tracks, choose an approach that “extends your ALM to mobile” rather than “integrates mobile testing within your ALM”. While this might seem like no more than a “semantic” issue, the approach here is critical. The goal is to maintain everything you have today - users, processes, authorizations, reports, source control, versions, repositories, etc. When you extend, all of this seems logical. However, when you integrate, maintaining two source controls may seem “acceptable”, managing two sets of users might be “unavoidable”, etc.
Consult with your existing ALM vendor to see what solutions are available in order to enable you to maintain decision-making control over the processes.

**DEVICE AGNOSTIC**

Due to the fragmentation of the mobile market, it is increasingly difficult for enterprises to ensure the quality and compatibility of their applications across platforms, devices and networks. There are simply too many test parameters and not enough time to create a separate test case for every possible variation.

The best way to address this challenge is to implement device-agnostic test automation. Device-agnostic scripting allows enterprises to build test cases that can be easily maintained, reused and ported to multiple mobile platforms. This approach should allow you to maintain business logic level scripting of your application. By leveraging device-agnostic scripts, you can execute the same test case across several different devices with minimal script maintenance. This also means you can run the same set of scripts on every new version of the application.

Normally, there is little “real magic” involved in device-agnostic scripting, and you can always choose to create scripts which are specific to a particular device. However, in terms of best practices, device-agnostic scripting is a recommended methodology for test script development and your solution provider should support it.

**TEST ON REAL DEVICES**

As each user’s experience depends on the particular device they are using, it is common for users to suffer from bugs and quality issues that are hard to trace and replicate. While simulators and emulators may be useful in some of the testing phases, there is no substitute for using a real handset operating in a live carrier network in diverse geo-locations to reflect the true user experience. Moreover, to detect potential handset interoperability issues, applications and services must be tested over the real end-user devices.

Just take a look at your own handset “update list” in your app store and see how many apps require updates due to “minor bugs fixed”, “now work also on XXX” and so. Again, no shortcuts here. If you don’t test on real devices, your customers will. The rule of “fix cost rises across deployment phases” applies here in just the same way it applies in the normal software industry. Every bug discovered in the field will cost you more (much more) to fix.
Testing on real devices prior to release reduces the number of detected faults in the field. If you need an ROI, take the average cost of "platform related" bugs in your web-based app and multiple it by 10 (at least) to account for the complexity of the mobile environment.

CLOUD-BASED ACCESS

Naturally, it would be extremely costly to procure, deploy and maintain dozens of "cool" devices as soon as they are introduced to the market. In addition, the logistics of managing multiple handsets and offshoring mobile testing (shipments, network availability, regulations, etc.) is a very time-consuming task and will actually cost you much more than the devices themselves. On the other hand, testing only a sample of these devices increases risk (and the cost of faults).

The solution to this quandary is cloud-based access to a large number of REAL handsets and tablets ensuring maximum reliability and quality. Since all devices are managed in the cloud, you no longer have to worry about complicated logistics. There is no need for procurement and devices can be added or replaced within a few hours in response to market needs.

In addition, a cloud-based approach to mobile testing solves workflow issues such as collaboration between teams. From the technology side, a cloud approach relieves the local IT team from handling device specific issues such as device drivers, natural device instability and device management.

A cloud-based approach might also alleviate some security-related considerations. Since most handsets are normally connected to "Wi-Fi" and 3/4G networks, if deployed locally they might require IT to manage special network configurations allowing them to actually work in your specific networking environment.
COLLABORATION

If your development and QA teams are globally distributed, it is likely that some level of collaboration is already part of your ALM environment today. In the mobile world, there are several aspects which turn collaboration from “very important” to “must have.”

Communicating issues on mobile platforms is not as simple as pressing “print screen” and attaching the screenshot to an email. In addition, to enable your testers to reproduce found issues, you will need logs, files and maybe even video. Statistical studies show that the amount of “not reproducible” or “not sufficient information” cases in the average defect tracking system is much larger in mobile applications than on desktop based applications.

Globally distributed teams require real-time collaboration tools that enable a tester to share a device in "live" testing with one or more distant colleagues for their input or approval. If your target market is in the US (for example) while testing is done offshore, you should consider a solution for this type of collaboration.

In addition, sharing live or pre-recorded sessions on specific devices is highly useful for replicating reported bugs on end user devices for improved troubleshooting by your customer support staff.

AT A GLANCE: BUSINESS NEEDS AND THEIR IMPLICATIONS ON MOBILE

<table>
<thead>
<tr>
<th>Fact/Challenge</th>
<th>Pain/Business Need</th>
<th>Implication for Mobile Testing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Already invested in existing ALM infrastructure</td>
<td>Minimize cost of new tools and skill sets</td>
<td>Extend existing ALM to mobile (e.g. HP QTP/QC)</td>
</tr>
<tr>
<td>Fragmented market</td>
<td>Too many platforms, networks, form factors to create separate test case for every variation</td>
<td>Device agnostic automation</td>
</tr>
<tr>
<td>Extreme dynamics</td>
<td>Device procurement and logistics</td>
<td>Cloud-based access</td>
</tr>
<tr>
<td>Many form factors, different end user experience</td>
<td>Users suffer from bugs and quality issues that are hard to trace and replicate</td>
<td>Test on real devices</td>
</tr>
<tr>
<td>Many test factors, geo-</td>
<td>Logistical nightmare to manage</td>
<td>Cloud-based access</td>
</tr>
</tbody>
</table>
## AUTOMATED REGRESSION TESTING

Regression testing is a key element in application development, significantly reducing the risk of faulty deployments of applications and enabling developers to detect faults much earlier in the lifecycle. As development cycles become shorter over time, the need for automation in regression testing is mandatory, particularly when using agile or iterative development methodologies. In any event, mobile dictates shorter development cycles because of the market dynamics. To minimize the gap between software functionality the ever-changing market requirements, shortening the development cycle is a MUST and automated regression testing is helpful in this regard.

In the world of mobile applications, regression testing takes on added importance. Mobile regression testing is critical to ensure application quality. Mobile apps are not simple and, similar to the desktop world, developers still insert new bugs in the process and magical leprechauns still insert faults into the code 😊. This means that if your enterprise uses regression testing in other areas, such as desktop or web development, excluding regression testing in mobile is a major risk (i.e., more faults found on field = $$$). Moreover, if your existing regression testing is automated, your regression testing for mobile MUST be automated as well.

Naturally, your mobile automated regression testing must integrate with your ALM environment for scheduling, reporting and statistics.

## MULTI-PLATFORM DEVELOPMENT

Developers tend to develop and test on one platform, and then assume the rest are compliant. With mobile, one platform is usually not sufficient. Depending on your market, you will probably need to support iPhone, Android (multiple versions), Blackberry and maybe more (e.g., Win Phone, Symbian, etc). The various combinations of device and OS that need to be tested are simply “too much” to ask from the developer. These are precisely where problems will inevitably be found in later phases. While everyone is talking about unified code and an optimized approach, the reality is that different platforms require different tests. There is no way around it and regression testing is the primary way to ensure quality within multi-platform development for mobile.
MULTIPLE SCREEN SIZES

Even if you’re developing exclusively for Blackberry or Android or iOS, you still have to account for multiple devices with diverse screen sizes. While it is true that most development environments today include tools to develop for multiple screen sizes, this is far from a bulletproof solution. In reality, those tools are not flawless and creative developers will find ways to overwrite these tools (all for good reasons). Accordingly, problems such as misalignment, overlap controls, unclear or truncated texts are very common.

KEY TAKEAWAYS FOR AUTOMATED REGRESSION TESTING

- If it is/was important for your web-based apps, it is also important (and maybe more so) in mobile.
- Regression testing in mobile is very important. Spend the time and resources to make sure you have a good and stable automated regression environment.
- Perform regression testing on several devices to best represent your target market.
- Do not create another regression environment for mobile. Make the effort to integrate mobile regression to your existing environment rather than duplicating the processes. The “hard part” is not the execution, but the knowledge management around your enterprise with reports, statistics, alerts, executives, etc…
- Automated regression testing is not easy. Start with a few robust test cases, and then add more cases in a gradual manner.

EXPLORATORY TESTING

Exploratory testing is essential for checking attributes that cannot be objectively defined and hence cannot be automated, such as the “look & feel” of the application. This type of manual testing is important for desktop and web-based application, and even more important in mobile applications.

As noted, the user experience is at the heart of everything mobile. Users purchase their mobile first and foremost as a personal device and only later as a “value-added” device. Consumers expect apps to work flawlessly and are less forgiving about faults.

MOBILE EXPERIENCE IS DIFFERENT ON DIFFERENT DEVICES

Different screen sizes, different OS versions, different platforms, different networks. All of those affect the user experience. Often, deviations in the user experience cannot be detected automatically and, even worse, they are subjective by nature (e.g., “does this look good” type of testing). A manual element is needed here.

The physical real-estate of a mobile screen is much smaller than the average desktop monitor. The result is a much less forgiving environment for buttons which are slightly misaligned, misplaced, a bit longer than the required text field entry and so on. Mobile UI elements take MUCH more relative space on the device screen than on the desktop.
MOBILE TESTING IS DONE ON … MOBILE DEVICES

When performing a test on the mobile device, a key challenge is what to do when you find a problem. You have a device in your hand, but you have no way of recording it, reproducing it or communicating it. This is especially problematic for teams that are distributed around the world. For example, if the testing is done in region A and the developer resides in region B, the tester needs to effectively communicate the problem on the device to the developer in region B.

There is also a logistical problem in managing and maintaining the devices. The very “cool” devices, especially the latest ones, tend to somehow disappear (often with their cables, chargers, and SIMs. Their batteries can drain out exactly when you need do the testing, or the SIM can get canceled because bill was not paid. These logistical issues require full-time attention.

Don’t underestimate the logistics involved in mobile testing. In cases where teams are distributed, you should also consider that the same handsets might be required both for testers, support and development when solving problems.

MOBILE TESTING IS LOCAL

If you are working in a distributed environment and testing is done offshore the target market, you don’t have the devices. This is a logistical problem which should not be underestimated, as it is complicated to keep sending devices overseas. In some cases, it is not even legal (depending on the country). The devices might not work overseas and depending on what you develop, there may be regulatory implications on whether it is possible to run this application on a device which is located on a different continent.

KEY TAKEAWAYS FOR MOBILE EXPLORATORY TESTING

- Exploratory testing needs to be done in a controlled, structured environment.
- The testing environment should provide a logistical solution for the devices to be available upon need
- This environment should provide a clear method of collaboration among teams and send information from the device to other team members (e.g., screenshots, logs, video, etc.)
- Exploratory testing should allow offshore teams to test on devices in the target market
- It should provide a logistical solution for these extreme market demands

COMPATIBILITY TESTING (PORTING)

Unlike the exploratory testing and functional regression testing phases described above, compatibility testing (porting) is a new phase in the ALM cycle designed to meet the unique challenges of the mobile services space. In desktop or web application development, for example, if you have developed an application for IE7 which had a 60% market share, the chances that this will change dramatically within a year are very low.
In the mobile space, this is far from being correct. The devices for which you are developing now will probably have significantly less market share within the next year. Market fragmentation exacerbates this challenge. In terms of Android alone, 50% of the market is controlled by about 10 devices while “others” (which can be around 100 additional devices) control the remaining 50%. In order to reach a reasonable figure of 80%, you will need to test your mobile app on many devices.

**PLAN FOR PLATFORMS: DEFINING YOUR COMPATIBILITY TESTING STRATEGY**

Your product team needs to deliver a list of platforms you need to support - this might vary from market to market. Decide upon the platforms on which you want to develop and, equally important, on what platforms you do not. This decision will help your support staff avoid a flood of calls from frustrated customers. Don’t assume your application will work on other devices - it almost certainly will not.

To address market dynamics, we recommend using an "MMM" testing approach:

1. **"Must" devices** – Select a small number of platforms (4-6 is a good number) and run rigorous regression and nightly testing with many test cases. These tests are also used in the development phase.
   a. Revisit the selected platforms every 4 months and replace 1-2 platforms with newer ones to keep up with the market.
   b. Automation is crucial in this step – without automation, you will enter stage 2 with a relatively low quality of product.
   c. ~50% of the faults should be found in this stage.

2. **"Major" Devices** – Add more platforms (approximately 10 is a good number) and run 50% of the test cases run in the first stage.
   a. Revisit the selected platforms every 4 months and replace 3-4 platforms.
   b. Automation is also very important here, although less critical than in the first stage. As there are more platforms, automation will probably require more effort. A device-agnostic scripting infrastructure is the key for successful test automation in this phase.
   c. With automation, you can expect to find an additional 30% of the faults. Without automation, you should probably expect to find about 10% of the faults.

3. **"Market" Devices** – Add many more platforms (30-40 is a good number) and run 5% of the test cases run in the first stage.
   a. Revisit the selection once a quarter and replace 30% of the offering. This is the most important step. Failure to carry out this step means you will lose contact with the market. Within 9 months, your initial selection of “many” devices will represent only 20-30% of the market.
   b. Automation will be very hard at this stage, due to the differences between the platforms. Invest in a good manual structured testing solution with a reliable collaboration mechanism.
   c. This phase should limit the remaining faults to about 5%.
Note that in mobile, the cost (and time) of fixing a fault post production might increase significantly because of the “app store” effect. For every found fault, you will need to decide whether you wish to re-submit the application to the app store again for re-certification. The distribution process is quite efficient, but the time it can take to release a “fix” might be measured in weeks.

**KEY CONSIDERATIONS FOR COMPATIBILITY TESTING**

When planning your mobile compatibility (porting) testing, logistics are the key factor:

1. Access to handset – including ongoing management of them
2. Structure of testing – especially related to collaboration and fault information delivery
3. Definition of testing – what to test and how

Access to testing devices is probably the most critical aspect of porting testing. You will need a logistical solution which will allow you access 30-40 devices at any given moment with replacement rate of about 10 per quarter. These devices will need to be fully functional and available at your scheduled testing times.

A well-structured testing process is important for coordination among the various teams. Otherwise, you are liable to find yourself spending most of the time making sure everything is working. Make sure you have a solution that supports offshore testers (if relevant). You will also need a collaboration solution to enable testers to send back to development screenshots, video and log files regarding any faults they discover. Video is extremely helpful for sharing the way to reproduce the problem. Developers will need access to the same device where the problem is found (for reproduction). You should also consider real time online sharing of the handset if you wish to show the developer the exact problem in real time.

**INTEROPERABILITY TESTING**

Since the mobile device is a connected device, its interface with the environment is necessarily different than the average web or desktop application. Therefore, when testing mobile applications, you should consider some scenarios which could affect the quality of your application.
First of all, mobile phones are still, first and foremost, phones. Most modern mobile OS's are configured to give call-related activities precedence over any other activities on the device. This means that if you are in the middle of an important transaction and a call comes in – this call will receive precedence over the transaction.

This scenario, however, extends much further than just calls. Messages, battery status, Bluetooth, other application notifications, OS level notifications and more will “grab” the user’s attention from the application and could cause unintended results.

**HOW TO TEST MOBILE APPLICATION INTEROPERABILITY**

For starters, define how to create the necessary conditions and determine what external factors are required for testing the device (e.g., making a call to the device under test). The problem with interoperability testing is that a slight variation in timing could have a major effect on the results. Thus, it is not sufficient to test the scenario once. For each scenario, you should validate multiple times and keep records of all testing in order to avoid “non-reproducible” test results.

Automation can greatly reduce the risk involved in interoperability testing. Instead of asking a tester to manually test each interoperability scenario a few times, an automated system can do this systematically and record the details of each and every iteration.

**CONCLUSIONS**

To address the complexities of mobile application testing in a highly fragmented and dynamic mobile market, enterprises need to implement a well-designed mobile test automation strategy. This strategy should be composed of automated functional and regression testing, as well as compatibility testing to ensure that each release of your application supports new devices on your target list.

Given the ever-growing number of platforms, devices and networks, device agnostic test automation is required to allow enterprises to build test cases that can be easily maintained, reused and ported to multiple mobile platforms. Moreover, mobile apps need to be tested on real devices connected to live networks in order to reproduce the true user experience and to fix hard-to-replicate issues.

Cloud-based access to REAL handsets alleviates the costs and logistics of procuring and managing new devices as they are released. Devices can be added or replaced within a few hours in response to market needs. In addition, a cloud-based approach facilitates collaboration among globally distributed teams to share a device in “live” testing and to replicate issues for improved troubleshooting.

While your mobile testing strategy must address the new business needs dictated by today’s mobile environment, it is also important to preserve your existing ALM investment. Solutions that allow you to manage the testing of all your desktop and mobile applications using a single integrated ALM platform lower total cost of ownership by leveraging existing tools, workflows and skill sets in your organization.
By implementing a mobile test automation strategy that addresses the unique challenges of the mobile environment, your enterprise will be positioned to roll out high-quality applications that capitalize on the business opportunities in today’s new mobile landscape.

ABOUT PERFECTO MOBILE

Perfecto Mobile is a global leading provider of cloud-based testing, automation and monitoring solutions for mobile applications and websites, utilizing a wide selection of REAL and emulated mobile devices accessible via the web. Perfecto Mobile’s MobileCloud™ Platform enables developers and testers to access and control a comprehensive range of the latest mobile devices connected to live networks around the world. Users can rely on Perfecto Mobile’s handsets and tablets to develop, test, deploy and monitor their mobile applications and services without having to physically obtain them without worrying about logistics, security or range.

Perfecto Mobile and HP have collaborated to develop the world’s first complete solution for automated testing of mobile applications within HP’s QuickTest Professional (QTP) product. MobileCloud for QTP integrates the core scripting language and flow control of QTP with Perfecto Mobile’s comprehensive cloud-based automated mobile testing solution. This ground-breaking solution enables enterprises to naturally extend their existing HP Functional Testing and Application Lifecycle Management (ALM) environment to support mobile applications with both real devices and emulators.