Mobile technology isn’t a fad. The emergence and continued expansion of mobile phones, tablets, and their associated applications have created an environment where every major company is racing to deliver the best offering. The need for testing and quality assurance continues to grow as businesses tie their names and reputations to companion apps, and while we continue to see improvement in the design and functionality of these offerings, thousands of stones remain unturned. It’s a technological gold rush, but there are new challenges to consider when testing mobile apps, as well as unique coding languages born from on-to-go devices. Even if the landscape has changed, the need for guides detailing best testing practices remains strong.

In This Mobile Testing e-Guide

Five Sins of Mobile Testing
Discover best practices that, when followed, can develop and produce a successful mobile app that’s user-friendly and reliable. Avoiding these five sins can save valuable time, resources, and strain while developing a finished product that’s functional and reliable, allowing you to bring your mobile strategy to market with confidence and assurance.

Testing Mobile Apps? Consider These Challenges and Solutions
The scope and need for mobile app testing are only going to increase. What does this mean for testers who have traditionally tested web applications? Mobile application testing is not necessarily new, but the increasing set of challenges is new.

Mobile Testing: The First Step—or Two
On many mobile projects, testing is not practiced well—or sometimes not done at all. Many testers from the desktop world are moving into mobile, and there is much they can take from traditional testing into the mobile space. Here are some ideas to get you thinking about testing mobile devices.

Unwrapping the Box: Strategies to Approach Mobile Web App Testing
Mobile web technology has been continuously changing over the past few years, making keeping up challenging. In this article, Raj Subramanian covers the latest trends and changes happening in the mobile web and how testers can prepare for them.

Do Mobile and Embedded Software Really Need Comprehensive Testing?
The smaller the device, the less testing is needed. Right? This insightful article dispels the notion that traditional software testing approaches work for mobile and embedded software.

Hot Quotes on Mobile
Some of the industry’s best and brightest give their opinions on various topics relating to mobile software testing and development.

Additional Resources for Mobile Testing
Great resources to keep you, your company, and your practices at the forefront of mobile innovation.
The demand for mobile devices in all aspects of everyday life is growing exponentially. Whether your organization is adopting “bring your own device” (BYOD) and is rolling out a new internal mobile application or distributing mobile devices to drive employee efficiency, organizations need to meet the on-demand need for mobile apps, websites, and services. The job of IT is now to fulfill its traditional requirements and to meet the demand of mobile growth, without additional resources.

Many enterprises have focused on monetizing and driving revenue by expanding what they do on the web to include mobile. But, some businesses are not prepared to expand to mobile. At times, upper management may make the executive decision to expand its mobile offerings without considering the time and manpower constraints that developers are limited to. Tight deadlines drive IT teams to rush through the development process of mobile apps and websites, leaving greater room for error.

The growth of mobile has presented a new wave of complexity to development because of the increasing device diversity and OS fragmentation. Choosing the best platform to develop your mobile app has become increasingly complicated. The complexities in mobile continue to grow and create new hurdles for selecting and rolling out the right mobile strategies on the right devices. Fragmentation has created the need to build several versions of the same app or mobile website to conform to its variations (like size of the screen, screen resolution, OS, firmware, etc.). To help the process along, there are some key best practices that can be followed in order to develop and produce a successful mobile app that’s user friendly and reliable. Simply avoiding these five sins can save valuable time, resources, and strain while developing a finished product that’s functional and reliable, allowing you to bring your mobile strategy to market with confidence and assurance.

Sin #1: Relying Solely on Emulation Software

In the world of mobile, there are multiple platforms, OS versions, form factors, and carrier/manufacturer customizations. It can be difficult to obtain all of these to properly create scripts and test for bugs or malfunctions. Developers often turn to emulated devices as an answer to their problems. While viable, especially in time and resource crunches, testing teams must take caution and understand that even with the most up-to-date software, certain quirks of mobile devices cannot always be replicated. This leaves room for error, especially while testing the behavior of an app on multiple devices. While emulation software can be helpful for basic functional testing, such as screen resolution, it falls short when trying to test one’s app or website over a real carrier network.

The best solution to yield more accurate test results before pushing your app to market is to test on real devices. The hybrid approach of using the cloud to run scripts through real, connected devices may best suit developers who need to test against a multitude of handsets but just don’t have physical access to them. Using a service that gives remote access to real devices allows developers to receive more accurate test results than using emulated software. Interactive manual testing of consumer-facing mobile apps can provide an easy, cost-efficient way to test functionality, usability and performance for both native and web-based mobile apps.
“Simply avoiding these five sins can save valuable time, resources, and strain while developing a finished product that’s functional and reliable, allowing you to bring your mobile strategy to market with confidence and assurance.”

Sin #2: Using Jail-Broken Devices
Unlike the Android OS, Apple’s iOS uses a strictly closed operating system. This adds a major roadblock for testers needing access to objects from an app. Many companies adopt jail breaking with trepidation because of the potential legal ramifications with manufacturers.

While the Digital Millennium Copyright Act (DMCA) temporarily allows jail breaking (rooting) of mobile phones, the exemption does not carry over to tablets. The advent of tablets and the legality issues of jail breaking have made it increasingly difficult for testers to extract object recognition data from these devices. Fortunately, there are solutions available where you can tap into real devices on the cloud that can give you advanced and pixel perfect experiences.

When you jail break a device, limitations to testing arise because you have altered the device by deploying an “agent” to control the device’s functionality and touchscreen capabilities. The developer no longer gets the truest experience on how that device would function, opening doors for error. True hardware integration is the most accurate way to test for mobile.

Sin #3: Not Leveraging Automation
While organizations are quickly adding support for mobile, balancing the ratio of effectiveness to efficiency in testing can be daunting.

The cost of producing and testing so many apps can balloon mainly due to device fragmentation. Additionally, building many versions of a mobile app is a strain on resources and time-to-market. Businesses can see this as cost prohibitive to enter the mobile market and are forced to weigh the pros and cons of such an undertaking. However, there are automated mobile testing platforms and tools that provide efficiency and drive down costs through automated testing.

Automated mobile testing can save testers time and increase efficiency by ensuring scripts are repeated correctly, resulting in less error. The process of scripting, testing, verifying, debugging, back to testing, then verifying again can become overwhelming. Mix in the need to test over the spread of several platforms and devices, and the need for automation becomes a no-brainer.

Automation is especially helpful when running the same scripts over multiple devices. Saving time and increasing quality are the biggest benefits of test automation. Automation can be the equivalent of a team of testers working non-stop, around the clock, running scripts on multiple devices, but faster and with less room for human error.

Sin #4: Ignoring the Need to Support Internal Apps
As more companies make an effort to streamline and make their employees more efficient with the BYOD movement, tension has increased for IT departments trying to support devices that are outside the infrastructure and security of the company. Mobile devices often encounter both security and performance issues when integrating a mix of personal and corporate applications, which were not built to coexist and can expose one’s network to malware attacks.
BYOD is a growing reality for the enterprise. Employees want the convenience of using their own mobile devices for both personal and work related functions and organizations are becoming more open to the idea of letting employees use personal devices for work, leaving the compatibility issues in the hands of IT.

The primary concern for enterprise IT and the BYOD culture is ensuring that enterprise apps are functional across multiple platforms and can support the needs of the organization’s users, especially when the applications are tied directly to functions that affect the business’s bottom line. It is critical to be prepared to test and debug at the drop of a hat, possibly over multiple devices. Creating a process of testing your mobile app at all stages of development is essential and will allow developers and QA teams to protect their networks while supporting their employee workforce.

“Without proper testing, the reliability of the apps and services that you offer diminishes, and the future of your development will suffer. Testing is critical to any mobile website or app launch.”

Sin #5: Skipping Testing on Real Devices

Speed is of the utmost importance in app development. The faster you can develop and make your app available, the faster you can generate revenue from the app. This leads many development or QA teams to spend little or no time testing detailed functionality of their app or website.

With the main focus on pushing out apps, organizations are taking risks by rushing through or skipping the testing process altogether. Apps like Google’s iOS Gmail app were not tested, failed, and were removed from the app store. Such failures could have been catastrophic if it had been a less-well-known company. Without proper testing, the reliability of the apps and services that you offer diminishes, and the future of your development will suffer. Testing is critical to any mobile website or app launch.

For those publishing their app for an app store, releasing a product with too many errors may lead to a bad user experience and may affect the success of the app. With millions of mobile apps in stores, users depend on other user reviews to determine whether one is worthy of a download. Piling up negative user reviews from bugs and other incompatibilities can be a brick wall for the success of an app and can affect future revenue growth for the company.

In the 2012–2013 World Quality Report by Capgemini, hundreds of developers across several industries were asked if they had a proper standard with mobile app testing before they pushed their app to market. Only 20 percent to 30 percent of developers said yes, unveiling a clear need for proper testing in the market.

The same report also states that the majority of organizations across many industries expect testing budgets to rise between now and 2015 as they realize more attention must be made to not just shift to mobile but to do so with quality results. Starting by avoiding common mistakes, teams can be well on their way to creating efficient applications for use by any target audience.

How do you know if you’ve done it right? If you hear nothing from employees or through comments in the app store, you have successfully tested. Most of the time, the only time you’ll get feedback is when the experience is negative. Still, it seems that many businesses are taking their chances and pushing apps out to market. How risky are you?
Testing Mobile Apps? Consider These Challenges and Solutions

By Mukesh Sharma • QA InfoTech

In January, for the first time ever, mobile apps were used more than PCs to access the Internet in the United States. Of the overall Internet consumption in the US in January, mobile-powered usage accounted for 55 percent, and of this, 47 percent of the traffic was from mobile apps. According to experts, these numbers will only increase.

“Of the overall Internet consumption in the US in January, mobile-powered usage accounted for 55 percent, and of this, 47 percent of the traffic was from mobile apps.”

As varied disciplines gear up for this wave, what does this mean for testers, who have traditionally tested web applications? Mobile application testing is not new, but what is new is the increasing set of challenges in this space by the day.

For example, there are very specific tests that a fast food app will need to run, such as user experience and load handling capacity at lunch times. In certain cases such as location-based apps, real-world testing is becoming inevitable.

Looking at this landscape holistically, mobile app testing in general differs in many ways from a standard web application testing process. With the increasing set of devices, it is a tough call for the testing team to decide whether to source and have all devices internally or to use mobile emulators.

In most cases, having the physical device is very important for the tests being run, at which time deciding how to group the devices to procure one to test from each group is another challenge. This may be a huge overhead up front, but if the team is trained to think along these lines, it will very soon arrive at an optimized strategy and be able to make a quick call on any new device with a considerable market share or potential that enters the market.

The challenges are alike regardless of your size of operations—for example, the Google+ team experienced trial and error before coming up with its meaningful testing strategies. The team gives a succinct list of unit tests, backend tests, UI tests, and monkey tests to run as a high-level approach to mobile app testing.

Another thing to closely watch for is the kind of application that is under test: a native, web, or hybrid mobile app. From there, you can decide what tools to use. In my experience, open source tools such as Selenium, Robotium, Calabash, and Appium provide a lot more flexibility and room for customization than commercial tools do when testing for mobile apps.

The scope and need for mobile app testing is only going to increase. Arriving at important decisions about what kind of app needs to be tested, how the app’s domain will determine what kinds of tests are needed, how to group devices in your testing matrix to arrive at an optimal set, what your device procurement strategy needs to be, and what tools you will use will help you build the much-needed differentiation in this space.
Everyone wants mobile apps and devices. It's the hottest software field. On a few projects, fortunate testers verify and validate mobile software throughout the lifecycle. But on many mobile projects, testing is not practiced well—or sometimes not done at all.

Many testers from the desktop world are moving into mobile, and there is much they can take from traditional testing into the mobile space. Here are some ideas about the discipline to get you thinking about testing mobile devices.

It is this last point—thinking—that testers need to practice the most in any system. To be a good mobile tester, you need knowledge to drive thinking, practice to build skills, and experience. But where should you start when thinking about mobile testing?

Begin by creating a test strategy. Clarify what you will be testing and to what degree: hardware, software, hunting for bugs, verification-checking requirements, validating the system, assessing non-functional qualities, needing to provide some other information stakeholders need, or a combination of these.

For example, how does your app impact battery life? Can the app work without a good connection due to weak or intermittent signals? Does the test strategy need to address Wi-Fi, cellular, Bluetooth, or other communication features of the hardware? And finally, how does (or should) the mobile device’s hardware impact your app testing, such as the sensors, GPS, camera, speakers, or microphone?

Think of these things before you start your test planning, design, implementation, and execution. These considerations work with any of the three types of apps: native (runs with no external connection needed), web apps (needs a web connection to run), and hybrid (mix of both).

There are many steps in becoming a mobile tester. Strategy is what gets a mobile tester started. Critical thinking is what makes any tester better at his craft. I am still learning and growing my mobile testing skills and my critical thinking capabilities. You too should keep practicing and learning to be a better mobile software tester.
Unwrapping the Box: Strategies to Approach Mobile Web App Testing

By Raj Subramanian

A couple of years ago, I decided to focus on testing mobile applications. Today, I’ve reached a point where the majority of my testing experience is with mobile web apps, and I find myself spending a great deal of time researching approaches, tools, and networking with other mobile testers to get a better grasp of the subject. Given the past of change, I eventually came to the conclusion that this would never stop; that I would need to continually, constantly invest time and effort into keeping up.

Before we jump into testing them, let’s spend a moment or two on the applications themselves. On a high level we have three types of mobile applications—native, web, and hybrid. Each one has its own advantages and disadvantages. Native applications are specifically designed to run on device OS and machine firmware, while web applications pull resources from the Internet each time it runs (mobile compatible websites). Finally, there are hybrid applications that have a native shell and are implemented using web technologies. I personally have worked with all three types and understand that there are multiple factors in each one of them. A tester needs to be aware of this and think about test objectives, test approach, test design, and test execution specific to the type of application being tested.

The current direction in web evolution is “responsive web design.” The simplest way to explain the concept is to say that every website should render “properly” for every form factor of a particular device. Another way to put it is that responsive design attempts to bridge the gap between a desktop browser experience and the web experience on a mobile device by creating websites that work both on a full-scale desktop computer and a mobile device. To do that, the site will have to change dynamically when viewed on a mobile device. If we can get this conversion to happen automatically, then the programmer only needs to create one site, saving development time.

Of course, this automatic translation can have errors; that is where the new testing challenges come in. Users don’t want to see errors and they demand access to Internet on their device. Here are some statistics of just how wide this difference in use is:

- 28 percent of Internet usage comes from a mobile phone. It is projected to take over desktop usage by 2014.
- 48 percent of users say that if they arrive on a business site that isn’t working well on mobile, they take it as an indication of the business simply not caring.

“48 percent of users say that if they arrive on a business site that isn’t working well on mobile, they take it as an indication of the business simply not caring.”
• 62 percent of companies that designed a web site specifically for mobile had increased sales.
• 90 percent of people use multiple screens to access the web.
• About 91 percent of US citizens have their mobile device with them 24/7.
• The number of unique screen resolutions have risen from 97 in 2010 to about 232 in 2013.
• Only 55 percent of companies are currently conducting any online user experience testing.

To sum it up: If your customers are in a position to expect the software to work on “their” devices, then responsive web design, and a test strategy for it, should probably be on your learning list. The world is heading in this direction and it is here to stay at least until the next game changer appears. Some good examples of responsive web sites are the websites of Microsoft, Disney, and the Boston Globe; there are others listed at www.mediaqueri.es.

“The mantra I follow is to perform ‘Very high-level test cases complemented with exploratory testing.’”

Testing these responsive websites is a challenge as there are different devices, form factors, screen resolutions (over 200 unique screen resolutions), different browsers, and countless permutations. When I wanted to learn to test responsive sites, there was no “book,” so I experimented with different approaches. Here are a few of the things I tried, which resulted in me learning new techniques:

• Networking with mobile testers and consultants both at my company and elsewhere
• Keeping track of new mobile development and testing news and technologies
• Researching different tools to help in testing
• Evaluated how rapid software testing approaches could help out in mobile testing
• Researching production defects. This is to help to focus testing on types of problems the users are seeing in the application
• Attending mobile focused conferences

Another thing to keep in mind is that because mobile is such a vast domain, it’s important to try out different approaches rather than just sticking to a set of scripted test cases and hoping you have met the so-called “100 percent coverage.” These test cases may be effective in covering some aspects of the application, but they rarely cover the aspect of usability and testing from the end user point of view. I have seen this happen time and again throughout my experience as a mobile tester. Don’t fall into the trap of writing detailed test cases for every scenario. By doing this, you will end up consuming valuable testing time that could have actually been spent on exploring the application. The mantra I follow is to perform “Very high-level test cases complemented with exploratory testing.”

The “learn-and-experiment” approach worked well for me while testing my first responsive website. For example, my project implemented totally new pages and functionality that interacted with multiple systems. This being the case, it did not make much sense to waste time in writing detailed test cases as there are only so many interactions we could cover. Instead, we wrote high-level test cases and covered more interaction scenarios by doing exploratory testing.

“Never hesitate to challenge anything when working in a mobile web project; always ask questions.”

Finally, never hesitate to challenge anything when working in a mobile web project; always ask questions. No one can predict exactly how a new feature would, could, or will be used in the real world, and this is especially true in mobile and the responsive web design world. There are always challenges, and our job as testers is to challenge ideas and question idealistic thinking.
On Android versus Apple:
“Android is so much better thought-out and it’s in a better language. It breaks my heart to say that iOS apps look better on the screen because the fonts are better, the animations are better, the human interface guidelines are more consistent. See, under the hood the Android apps are better and it’s just heartbreaking that they lose it at the very moment before the finish line. On the surface an iOS app looks better, but it’s all superficial.”
– Mark Meretzky

On Mobile Security:
“Right now a lot of applications are just kind of a throwaway thing, but more and more you hear in the world about security issues, whether it’s information being leaked or systems being hacked, and that’s just going to have to follow with mobile devices to be something that’s really examined.”
– Max Saperstone

On Mobile Responsibility:
“Society has a reasonable expectation of trust. For example, no one attacked me when I walked down the street earlier. When I had lunch, the chef didn’t poison me. I don’t normally worry about either of those risks. Companies that provide services to others should be taking reasonable steps to ensure that their services are secure.”
– Erik Costlow

On the Future of Mobile:
“If we kind of follow the trends from what’s happened over the last couple years with the ‘mobility movement,’ I guess I’ll call it, I think wearable and embedded devices are going to be kind of the next thing, right?”
– Ray Potter

On the Importance of Mobile:
“Obviously, over the last couple of years mobile has increased in importance. We call it the mobile imperative. What we’re seeing out there is not only our people demanding access via mobile, but they are demanding quality access via mobile.”
– Rachel Obstler

On Mobile Testing:
“I think you find the right people that are dedicated and are fans of your platform, they make the best testers because they really care about what they’re talking about as opposed to someone who just says they want to do it and then you never see them for like two months.”
– Greg Avola

On Mobile Quality:
“At the end of the day when somebody contacts me who’s angry, my end goal is not only have them leave happy but also have them leave more of a fan than they came in. If I can get them on the way out the door to go write a review, they’re going to write a glowing review because they just had a great personal interaction with me and that’s really, really hard to top.”
– Josh Michaels

On Mobile Applications:
“[Firefly] is a QR coder on unbelievable steroids in that you can take a photo of any everyday objects and all of a sudden now, your ability to act on that object whether you want to purchase it, whether you want to research it, is almost instant.”
– Dave Cotter

“Obviously, over the last couple of years mobile has increased in importance. We call it the mobile imperative. What we’re seeing out there is not only our people demanding access via mobile, but they are demanding quality access via mobile.”

“I think wearable and embedded devices are going to be kind of the next thing.”
Do Mobile and Embedded Software Really Need Comprehensive Testing?

By Jon Hagar • IEEE

"Because we are testing a simple mobile app, our app doesn’t need extensive testing to submit to the app store."

"Because we are testing embedded software, the software is already unit tested by our developers and we have the best software development processes for our small amount of code. As a result, we won’t have bugs in the field."

I focus my energy working with mobile and embedded product development teams, and these quotes are all too familiar to me. On the other hand, we all have read a similar headline in the press: "Large auto manufacturer loses lawsuit because of defective software in an electronic engine controller."

In app store sites, I see hundreds of product reviews with no stars and comments, like “This app is buggy and I can’t figure out how to use it, so I deleted it after twenty minutes of frustration."

Both of these stories are situations the software producers would like to have avoided, so perhaps the attitude that “we don’t need much testing” is wrong. Worse, this lack of attention to quality costs companies millions of dollars every day.

What Are Mobile and Embedded Systems, Anyway?

Mobile “smart” systems are small, handheld devices, usually connected to communication networks and powered by batteries. They share many common features with embedded devices and traditional computers, yet they have limited resources.

Examples of these devices include cellphones and smartphones, tablets, medical devices (such as pacemakers and defibrillators), automobiles and other forms of transportation (like cars, buses, trains, trams, and trolleys), and factory and industrial systems (PLCs, robots, and so on).

Embedded software systems consist of unique hardware or systems with dedicated software that solve specialized problems, often in real time. Embedded systems have the following unique characteristics.

Unique hardware: Software interacts with special hardware, providing interface and control support.

Constrained resources: The systems have limited resources, such as RAM, ROM, stack, power, speed, or time.

Limited user interface: Embedded systems typically have a restricted or no user interface.

Examples of embedded software systems include software-controlled robotics, avionics systems, control devices, and smart electronics. In fact, users may not even be aware a device has software.
Testing Considerations

What testing approaches should be considered for mobile and embedded systems? The simple answer is to use the same techniques and testing approaches used for developing IT software, PCs, and the web. You’d think that with heavily constrained systems, testing would be dramatically less expensive. On the contrary, testing mobile and embedded systems may cost more and require more effort than expected. Instead, mobile and embedded validation might include the kinds of attack testing listed in Table 1, which focuses on the kinds of errors commonly seen in mobile and embedded systems.

The attacks outlined in Table 1 may appear to be common to other software environments, but specific patterns of attacks in the mobile and embedded space will be dramatically different. The attack patterns of Table 1 are unique to mobile and embedded environments and would also need to be customized for specific types of software, such as medical, transportation, industrial, space, gaming, mobile information, sales, and so on.

“The testing mobile and embedded systems may cost more and require more effort than expected.”

The concept of test attacks to break software is a popular basis for testing and is detailed in a variety of books and standards that go beyond basic requirements verification checking. Test attacks are patterns of testing to find errors in the software based on common failures. Test attack patterns will be customized for the local context, considering factors such as the software under test, who does the test, where the attack is done, available resources, and the goals of the test.

This menu of attack types and test approaches will leave most test teams concerned that to implement a full range of tests will take too much time and money. In order to prioritize the best test approach, I’d recommend considering the following approaches for mobile and embedded testing.

<table>
<thead>
<tr>
<th>Attack Type</th>
<th>Finds</th>
<th>Notes on the Attack</th>
</tr>
</thead>
<tbody>
<tr>
<td>Developer level attacks</td>
<td>Code and data structure problems</td>
<td>Almost a quarter of errors in mobile and embedded can be found by structural testing.</td>
</tr>
<tr>
<td>Control system attacks</td>
<td>Hardware and software control system errors</td>
<td>Many critical bugs in mobile and embedded are centered in the control logic.</td>
</tr>
<tr>
<td>Hardware-software attacks</td>
<td>Communication and interface integration issues</td>
<td>The software works with unique hardware that must be assessed.</td>
</tr>
<tr>
<td>Communication attacks</td>
<td>Digital communications problems</td>
<td>Mobile and embedded systems communicate with hardware, networks, and software with complex interfaces.</td>
</tr>
<tr>
<td>Time attacks</td>
<td>Time, performance, sequence, and scenario bugs</td>
<td>Embedded and some mobile apps have critical timing and performance factors.</td>
</tr>
<tr>
<td>User interface attacks</td>
<td>Problems between man and machine</td>
<td>The usability of devices and software is critical to success.</td>
</tr>
<tr>
<td>Smartphone attacks</td>
<td>Issues specific to smart device configurations, including gaming and cloud apps</td>
<td>Gaming and cloud computing comprise a majority of the apps being deployed.</td>
</tr>
<tr>
<td>Security attacks</td>
<td>Bugs that can expose devices to security threats</td>
<td>Security of devices is increasing in importance.</td>
</tr>
<tr>
<td>Generic functional attacks</td>
<td>Requirements and interoperability bugs</td>
<td>These are the basic checks testers should conduct on mobile and embedded devices.</td>
</tr>
</tbody>
</table>
Using Simple Risk-Based Testing

Although there is no best or single way to test apps, always consider the heuristics behind test approaches, planning, and techniques. Risk-based testing uses product risks identified by a team’s stakeholders to determine the areas most critical to product success. Risks are then used to focus test activities on critical high-risk concerns. If functional and nonfunctional qualities are quickly attacked with cost-effective exploratory testing (not highly scripted testing), data about product trustworthiness can rapidly provide information to the team to aid development and release decisions using an informed agreement approach. Using risk-based testing may result in some functional areas being under-tested or not tested at all, but the full team will get to decide the amount and approach of acceptable testing for the product.

Defining Your Scoping Strategy with Early Testing

Closely related to risk-based testing is the determination of test strategies as part of higher-level test planning. For example, on one mobile app software project the team decided to use agile developer-based testing combined with a risk-based attack during development. As the software became mature, they employed a crowdsourced third-party group of testers to follow a new detailed plan of validation checks, which focused on potential bugs and risks that might likely impact the user of the app. This was performed with a small team of developers, testers, and a crowd team. Once the product received positive customer reviews, the team expanded the product and attack testing. This early and fast feedback testing approach during initial app development not only found critical errors but also was cost- and time-effective. Once the deployed product gained user acceptance, more testing was performed to find latent bugs before users found them.

Pulling It All Together

There is no best set of test attacks, number of tests, or combination of approaches that can be generically used for all mobile and embedded systems. A good approach should be to focus on attack-based testing early, frequently consider the risks to refine scope, and adjust test plans. Such a strategy can be done within a project’s cost and schedule constraints because mobile and embedded software is often developed under tightly constrained budgets and very aggressive schedules. For many projects, budget and schedule constraints can be changed as testers provide information on the quality status of the software under test. There are times when schedule changes and increased budgets need to be considered by demonstrating with test data that functional bugs or other software quality problems exist in the product. You may want to avoid releasing buggy software when the team knows about the quality issues. There are times, however, when the team will have to make an informed release decision by accepting the risks.

Focus testing on what is important: Conduct ongoing, risk-based test planning and prioritization.

Coordinate test scope: Define your test scope in a test strategy or plan to gain agreement with your stakeholders.

Employ early testing: During development of the code, use exploratory, nonautomated tests to provide useful information to the team.

Stay agile: Evolve the test plans and strategies to provide more information to your stakeholders and to determine if more or less attack testing is needed.

“The balancing of constraints and test approaches requires highly skilled and practiced testers.”
Many of the newer mobile and embedded teams who don’t learn the testing lessons of the past may not be around in the future.

A major consideration in risk-based and attack-based testing is that the tester must go beyond just checking functionality. If testers only verify requirements, usually with simple “happy path” test cases, many errors may remain in the software. Missed errors can result in unhappy users, resulting in negative feedback. System testing that tries to show the software does not work should be a common practice of mobile and embedded testers within project constraints. Such “break-it” system testing, with a focus on risks and using attacks or complementary test techniques, can provide valuable information to the team and decision makers. The balancing of constraints and test approaches requires highly skilled and practiced testers. This is true in general for most software, but mobile and embedded systems can have significant risk involving safety, hazards, finances, legal issues, or other factors, which should elevate the need for comprehensive testing.

In Summary
Risks of bugs, bad customer feedback, and other undesirable results should be considered by the team when selecting an optimal testing approach to take. Skilled mobile and embedded testers who balance testing to constraints will most likely release products that are good enough and meet expected quality goals. Unfortunately, as we have all witnessed with quality disasters in PC and web app environments, many of the newer mobile and embedded teams who don’t learn the testing lessons of the past may not be around in the future.

Want more than mobile?
In addition to insight into the world of mobile, TechWell.com covers development, testing and QA, project management, requirements, agile, configuration management, DevOps, cloud, people and teams, and so much more!

>> Go to TechWell now.

Mobile Software eGuide Credits:
External:
Authors: Josh Galde, Mukesh Sharma, Raj Subramanian, and Jon Hagar

Quotes provided from interviews with Dave Cotter, Mark Meretzky, Ray Potter, Max Saperstone, Rachel Obstler, Greg Avola, Erik Costlow, and Josh Michaels

Photos: Wikimedia Commons, FreelyImages.com, and Better Software Magazine

Internal:
Producer: Cameron Philipp-Edmonds
Editors: Josiah Renaudin, and Beth Romanik

In addition to insight into the world of mobile, TechWell.com covers development, testing and QA, project management, requirements, agile, configuration management, DevOps, cloud, people and teams, and so much more!
Additional Resources for Mobile Testing

More Information for Software Professionals

AGILE CONNECTION
A TECHWELL COMMUNITY

Bringing you today’s best agile ideas and thought leaders with how-to advice on the latest agile development and methodology practices.

CLICK TO VISIT NOW

STICKYMINDS
A TECHWELL COMMUNITY

The home for software testing and QA professionals provides practical advice on test automation, test management, test techniques, and more.

CLICK TO VISIT NOW

CMCROSSROADS
A TECHWELL COMMUNITY

Where configuration management and development professionals go for answers on SCM, ALM, change management, DevOps, tools, and more.

CLICK TO VISIT NOW

Mobile Dev + Test Conference
April 12–17, 2015
San Diego, CA • Manchester Grand Hyatt

CLICK TO VISIT NOW