Summary:
Software testing of mobile applications uses a lot of the same QA & Testing practices typically used for other software applications. However, there are important distinctions that can define how successful the mobile application is ultimately. This whitepaper discusses some of best practices in developing and performing Quality Assurance on Mobile Applications and outlines strategies for addressing specific aspects of the testing process.

Introduction
The demand for mobility shows now signs of slowing fueled by both consumer and enterprise appetites for mobile applications. That brings up not only the need to create apps but to test them, and ensure that they run on an increasing array of platforms and devices.

While many of the typical Software QA practices also apply to performing QA & Testing on mobile applications, the reality is that mobile applications are quite different in some critical ways. Mobile Applications therefore require some special considerations.

Belatrix Software is a leading Software Services Outsourcing company with its main offices and delivery centers based in Mendoza, Argentina and Lima, Peru. We provide Software Development, Quality Assurance, Localization and Graphics Design Services using the “Agile Near-Shore Model”.
How and Where to Test

It’s important to test the application using the actual device model(s) on which the application will run. This ensures a good level of coverage and also guarantees that the results of the tests will be much more realistic. Only by having the exact device and model can you be 100% sure that the application will compatible and behave the way it’s intended to behave. For example, in the case of iPhone, you cannot go back in versions unless you have the same hardware model multiplied by the number of software versions you want to support. This can get extremely complex and very expensive very quickly.

While helpful for initial testing, none of the alternatives such as emulators, simulators, are 100% reliable. That’s true even for emulators / simulators that come with the official SDKs. It’s not uncommon for companies to want to target all existing devices. However, that’s unrealistic given the actual cost of sourcing, managing and maintaining all the physical devices.

The only realistic approach is to select a list of devices that your company will ensure compatibility combined with a well-managed user Beta testing program.

Aside from assigning resources to focus on the most critical devices, the only cost-effective approach to addressing the other field of devices is to engage groups of Beta testers. These Beta groups test the application on their devices and report any issues or incompatibilities. This will result in a compatibility database that can be used to adapt or fix the application as necessary.

Another important best practice is that companies should, if at all possible, avoid supporting devices or operating systems that the manufacturers or vendors no longer support. Trying to test on unsupported devices and O.S. versions increases costs exponentially.

Good mobile testing practices

- Ensure that you have access to all physical devices on which the application run (Not just emulators or simulators)
- Consider the additional time and/or personnel needed to test all the application’s modules on all the devices.
Have an explicit list of devices on which QA will be performed. Basically this means that the company is committing to supporting certain devices and Operating Systems on which the application will perform as designed. The users will be informed that even though the application might work on other devices, the company will not be held responsible for unexpected issues. This is the approach taken by most leading vendors today, including for example, Skype.

Implement a mixed strategy when you have a scenario where you do not have all the possible devices and therefore cannot do exhaustive testing on each: Use an emulator on the early stages of app development. Keep adding real devices into the testing later down the development cycle. This way you can certify that all the requirements and goals are being covered.

**Interface Testing**

- Validation of buttons, text inputs, labels, etc.
- Validate each screen in a comprehensive manner (does it do what it is supposed to do?)
- Validate navigation flow (i.e. from the list of restaurants screen, can I access bookmarks?)
- What happens when you change

**Usability**

- Is it easy to navigate between screens or does it require weird steps?
- Does it show user redundant information or low value information?
- Can it display text properly, in the language selected?
- ALL (repeat, ALL) user interactions with the system should generate some kind of feedback!
- Clicks on buttons
- Calls over the Internet
- Any potentially slow operation: always assume the worst case!
- Verification of the functionality OFFLINE / ONLINE. Does the user lose information that is sent to a server while in a non-coverage area?
Performance

This is one of the most important factors to verify when developing Mobile apps. Users will not accept apps that take too long to load or perform simple actions. Load and performance testing is a must on Mobile testing. Here are some topics to consider before writing the test plan:

- What if the connection to a server is over 3G instead of WiFi? What about 2G?
- What about when to send / receive much information from a Web service?
- Are the images used the "right" size?
- Optimize: redundant code => CPU cycles => + battery consumption => unhappy users
- Does the App comply with the response times required by users?
- Are there memory leaks?
- Are all resources being freed (GPS, camera)?
- Do you have any tool for running Load tests? (There are several options on the market that could help a lot).

Security

- Any sensitive data being sent or received must be encoded: Encryption + Authentication (HMAC-SHA)
- Consider potential detection of areas in the app that could receive malicious information. Corollary: the server is not necessarily our friend.
- Do not assume good behavior: the world is evil, cruel and vile!
- Any file you saved in the application can reach the hands of unintended third parties.
- Multiuser support without interfering with the data between them.
- Make sure that attacker cannot take your app offline.
- Could an ordinary user take administrator privileges on your app?

Services

- The Mobile Apps should act as the client, not the server.
**Validate** what happens should a service goes down. If it returns malformed responses, control errors, etc.

What if the service takes too long to respond?

What happens if you try to access a service offline?

What if I get disconnected during transmission of info?

Have we included API testing? This is very important since our app could be using APIs and we should make sure that we know these APIs and the way to interact with them. Sometimes the problem is not in the mobile app!

**Low-level resources**

Is the App generating "garbage" of some sort? Temporary files not cleaned, local database growing too much...

Are we correctly using the sensors? Are we freeing the GPS? Do we make more calls to the server than needed?

Are we using too much memory? Or, are we not releasing memory properly?

**Operational**

Are we "backing up" necessary information in the app?

Are we prepared for "shock" mode? If the battery dies, we need a Recovery and Save Plan.

If you upgrade to a new version of the App on the corresponding "Market", is data lost?

What if the user gets a call while using the app? What if there comes an SMS?

Does the battery seem to go down dramatically with the use of our App?

**Compatibility and multi-device environments**

Have you validated the app on devices, OS versions and models agreed?

How does your application behave on different screens and resolutions?

What if the integration server changes? Does it support a seamless transition?
Can our application affect other third-party systems? Can it jeopardize other systems or degrade them?

Localization

The latest HTML5 and mobile web-based apps are optimized to adapt content according to geography. This means your app can be accessed from everywhere in the world, allowing it to work as expected.

Have you considered the following items while you developed your testing plan?

- Date formats
- Text direction (some languages are written from right to left)
- Languages with different set of characters
- Address formats
- Currency conversion

There are also several other things to keep in mind when writing your test scripts. The key here is to always think that your app could be accessed by people all over the world. The goal is to get it to work as if it were in your own country. A proxy service could help on testing different localizations without being present on all the countries. This is a cheap option to test localization. Consider taking advantage of remote real devices as another low-cost option to test localization.

Native vs. Web

There are some important things to differentiate when you test a native app versus a web based app.

Some considerations when testing native mobile apps:

- Testing cycle starts from the installation and launching process
- You may probably need physical access to the device ID
- Your app may have access to other APIs. Make sure those connections are under your testing scope.
Important factors when testing mobile Web-based apps:

- Requires internet connection (WiFi, 3/2G, GPRS, etc)
- Different web browsers render on their own way the pages.
- Make sure performance and load speed is tested (and they are satisfying the requirements).

Native apps developed by cross-platform SDKs -- It is getting more frequent to use cross-platform SDKs to develop mobile apps that work for several devices/OS.

One of them is Titanium. Despite that it brings you a native app, you may want to assure the app is running as it is expected on all the supported devices. Just testing in one of them is not an option!

About Belatrix Software

Belatrix Software helps clients achieve the full impact of their R&D capabilities.

Belatrix develops high quality, innovative software, QA and testing that enables clients to generate best-in-class, revenue producing software products, decrease time to market, and gain competitive edge. The company’s mission is to be a passionate global partner delivering software innovation to enhance our client’s aspirations.

Belatrix's clients include both established Fortune level and emerging, venture backed firms. A rising brand in Nearshore Agile outsourced software development, Belatrix Software has relationships with some of the fastest growing software startups to Fortune 500 companies. Belatrix is headquartered in Mendoza, Argentina with offices in Boston, Naples, Mendoza, and Lima. For more information, please call 617-608-1413 or visit http://belatrixsf.com.

Belatrix’s Quality Assurance Department provides testing and quality assurance services for projects developed within the company (internal projects) or by other companies (external projects). It also supplies mechanisms to
improve existing practices by developing standardized processes for the entire company and by applying ISO™ standards, international standards, and lessons learned from CMMI™-SEI. Belatrix's QA team specializes in high value-added QA services such as Load Testing, Scripted Testing, etc. Because of this we are extremely selective in our recruitment process and only hire top software engineers that have a passion for Quality Assurance.

Some of the Testing Strategies we follow are:

- Functional Testing
- Risk-Based Testing
- Graphic User Interface
- Acceptance, Usability and Accessibility
- Performance (Stress and Load)
- Regression
- Installation and Configuration
- Security
- API Testing (N.Unit and MS-VSTester Edition)
- Automation (Python, VBscript, Apodora, Mercury QTP)
- Smoke Test

Experience how a relationship with Belatrix can give your company a distinctive advantage through high quality Agile software development and quality assurance services. Contact us now!