Mobile App Development for Clinical Trials

Mobile devices and apps are set to become an essential part of clinical trials. While adoption in trials is currently underway, there are still many questions and issues regarding the use of mobile devices and apps in this field. The issues range from sourcing and development methodologies, to validation and deployment strategies. One thing is for sure: the adoption of mobile devices by the general populace is undeniable. More than half of the US population owns a smartphone (57% according to a Pew study – see Figure 1) and Americans are spending an average of two hours per day on their mobile devices. There are an estimated 1.4 billion smartphone devices in existence. That's one in seven people worldwide.

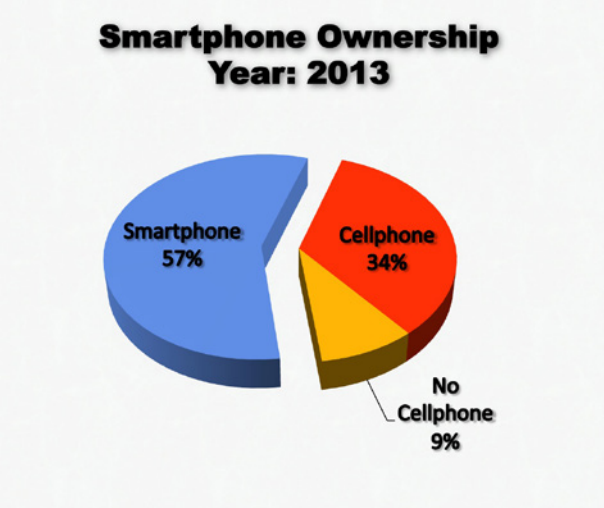

Figure 1. Smartphone adoption figures according to Pew study.

The 'smartphone' nomenclature in use today is somewhat misleading. Mobile phones have in fact evolved to become highly portable personal computing devices. The audio phone capability of the modern mobile device is just one amongst a wide variety of communication possibilities provided. Audio telephony provided by the mobile network operating companies is another way of facilitating two-way transmission of ones and zeros in the same way that Wi-Fi and Bluetooth are. Modern mobile devices are also extremely powerful. For comparison, a modern mobile device is more than 15 times faster than the Cray-1 supercomputer circa 1979 and is, of course, substantially smaller.

A key feature of modern mobile devices is their connectivity to the internet as a default operating state. This 'always connected' capability allows mobile devices to hold two-way data communication with any server in the world. This connectivity, along with high processing speeds, high quality visual displays, and the ability to interface with other devices via protocols such as Bluetooth, make smartphones an ideal choice for capturing participant data in clinical trials.

Application and Deployment

The smartphone's ubiquity brings with it a number of key benefits for clinical trial participant engagement. Firstly, it is likely that the participant will already know how to operate a mobile device and, providing the data capture app adheres to known mobile platform user interface paradigms, will quickly learn how to use the app. Secondly, mobile devices are portable and people generally carry them wherever they go. This gives them a great advantage over desktop or even laptop computers because it means they can be accessed at any time and in any place. Thirdly, mobile devices are always connected to the mobile network operating service and the internet. This permanent connectivity capability is a distinct advantage for mobile ePRO data capture as it allows messages to be pushed to the mobile device in the form of SMS messages, or push notifications via either Google or Apple notification services. These benefits combined give mobile device ePRO data capture an advantage over ePRO data capture on desktop or laptop computers, leading to increased participant engagement and adherence. Interfacing with a mobile device is more convenient.

It is clear that mobile offers many advantages in clinical trials ePRO data capture. How then should a mobile app be developed and deployed to the participants? First, it has to be decided which platform the mobile app should be developed for. Today, mobile deployment will be on either Apple or Android devices. Apple offers a clear advantage in terms of minimising device variety because they carefully control the range of hardware that they manufacture. Apple's operating system, iOS, only runs on Apple hardware, leading to a reduction in compatibility issues. They also have firm policies on operating system upgrades, which leads to less market fragmentation. Google adopted a completely different strategy with their operating system, Android. Android is made available to a large number of hardware manufacturers. As a result, it is available on a bewildering range of hardware devices of varying capabilities. These hardware manufacturers have also occasionally taken it upon themselves to modify the user interface and features - leading to a much more fragmented market both in terms of capabilities offered and robustness. Top-end Android HTC and Samsung devices compete aggressively with iPhone, but there is a huge array of cheaper Android devices of varying capabilities and operating system versions. This makes Android a much more challenging platform to develop for.

For some clinical trials, it is possible to pre-select the mobile hardware and deploy these to the participants. This is by far the safest approach because there is total control over the hardware. Unfortunately, it is also the most expensive option - particularly with Phase III trials where the number of trial participants can run into the thousands.

The alternative is BYOD (bring your own device). Here, the participant is allowed to use their own mobile device, which means there are no upfront costs associated with buying and deploying dedicated mobile devices. The trade-off is a necessary increase in the technical support required. There are literally thousands of different mobile devices in the market that the app will need to operate with. It becomes particularly problematic with Android devices which are fragmented both in hardware specifications, and operating system versions.

Implementation Strategy

Beyond the choice of deployment platform, there are a number of other decisions that must be made regarding app software development strategy. The default de-facto way of developing apps is using the native development tools and languages provided by the platforms. For Apple devices, this means developing in Objective-C using Apple's XCode IDE (Integrated Development Environment), while for Android this means developing in Java using the Eclipse IDE and Google's Dalvik Java Virtual Machine. Native development allows total access to the device's capabilities and delivers the smoothest user interface experience. It comes at a cost, however. Objective-C and Java are very different languages, and XCode and Eclipse are radically different development environments. Developing natively for Apple and Android devices is a duplication of effort, which has an impact on cost, resources, and time.

Another option is to leverage web technologies to produce web apps. All mobile devices ship with feature rich browsers supporting HTML5, Javascript and advanced CSS styling. The idea with web apps is to utilise these technologies to produce web pages that look and behave like an app. This solution is particularly attractive to developers that come from a web development background as it uses familiar web-based solutions. There are a number of downsides to this approach, however. The device must be online in order to load the web page containing the app, and web apps are confined to using browser user interface controls or custom controls created with HTML5 and CSS styling, rather than the mobile devices native user interface controls. This leads to an app that doesn't quite feel or function the same way that native apps do. Web apps cannot be listed in the platform app stores. Instead, a link is provided that starts the browser and loads the web page that contains the HTML5 code for the app. This link can be placed on the mobile device apps' home pages, but it is another aspect that serves to separate web apps from native apps, as there is a clear visual distinction between an app and a web page link. A noteworthy restriction on web apps is that you are confined to the browser environment. This is fine as long as a given project does not require deeper level access to the mobile device's hardware, for example accessing Bluetooth to communicate with medical devices.

The situation can be improved by using a web app wrapping technology such as PhoneGap. These services take a web app and embed it into a generic app chassis that can then be deployed to the platform stores. This means that the web app is now behaving more like a native app – at least from a deployment perspective. Unfortunately, other problems now arise. The browser access provided for embedded web apps has a myriad of issues and idiosyncrasies. Speed of execution is compromised as the Javascript code for such apps is not compiled "on the fly" using a JIT (Just In Time) compiler, but is interpreted instead.

Other pitfalls include cross-platform visual rendering speed and glitch issues, the need to capture mobile-specific events, and implementing workarounds for certain user interactions. Web apps wrapped with PhoneGap have a 300-millisecond latency on element user interactions, leading to an annoyingly sluggish feel with the apps' user interface.

Given that native development can be expensive and require more resources, whilst web apps do not give the much-coveted native feel or deeper access to mobile device hardware features, is there another solution that provides

native look and feel and hardware access capabilities, combined with the cross-platform development benefits mentioned above? Indeed there is. There are a number of different middleware technologies that produce native apps but allow development in a common language, such as Javascript, with a library that provides access to native controls. The Javascript is often compiled to native code. This is potentially the best of both worlds: a common language and set of libraries that map to either platform's native user interface elements, whilst producing a fast native app for deployment. As an additional benefit, such middleware provides the ability to include external libraries as plug-ins, allowing access to mobile hardware if required.

One particular middleware that has been used to achieve this is called Titanium Pro, part of Appcelerator's middleware offering. It includes an Eclipse-based IDE that is used to develop Javascript code for deployment to Apple and Android

mobile devices. There are, as usual, a number of things to watch out for. The APIs for Titanium can often change to adapt to alterations or fixes in mobile platform operating systems, which can lead to minor incompatibility issues that, nevertheless, need to be resolved. There still needs to be some code that is platform-specific because, although the majority of user interface elements have a common abstraction for each platform, there are some differences in paradigms. Most notably, the user interface for tabs is different between iOS and Android.

Even with the issues stated above, in our experience mobile app development time can be cut by up to a third by using middleware like Titanium, so it is a solution that is often overlooked, yet well worth consideration given the right project.

Figure 2. Anatomy of a mobile ePRO system Connectivity

Connectivity

Interfacing smartphones with biosensors and medical devices is becoming more commonplace in mHealth and clinical trial apps. For example, a clinical trial based on treatment for respiratory problems such as asthma or COPD may utilise a smartphone that pairs with Bluetooth-enabled respirators and spirometers. There are many sensor devices that utilise Bluetooth for tracking biometrics such as temperature, heart rate, blood glucose and blood pressure. Figure 2 highlights some of the implementation and connectivity options available to mobile ePRO apps.

Such device pairing allows the objective collection of usage and biometric data which can then be transmitted to back-end servers for analysis and reporting. This data, when paired with the usual ePRO diary data, provides a compelling and detailed record of how a participant is progressing within the trial.

Persistent Bluetooth connectivity comes at a cost with regard to power consumption, however. To alleviate this issue, a low-energy version of Bluetooth was developed, called Bluetooth LE (low energy). Bluetooth LE uses the same 2.4 Ghz radio frequencies as Classic Bluetooth, but uses a much simpler modulation system and a different set of channels. Most new mobile devices support Bluetooth LE, but not all smartphones currently present in the market do. The Bluetooth Special Interest Group (SIG) predicts that more than 90 per cent of Bluetooth-enabled smartphones will support the low energy standard by 2018. In the meantime, this is a consideration for trials that utilise BYOD, particularly on Android, where Bluetooth LE isn't supported on Android operating system versions below 4.3.

Any deployment scenario where the app requires an interface to a Bluetooth device will rule out a pure HTML5 approach, because standard out-of-the-box web technologies have no means to support Bluetooth at all. HTML5 app deployments can utilise wrappers such as PhoneGap that allow access to native plugins and therefore can provide a library for interfacing with Bluetooth. A similar plugin approach can be taken when using middleware such as Titanium Pro, but the easiest way to interface with Bluetooth and other specialised mobile device features is by developing natively.

Mobile apps are set to play a major role in ePRO clinical trial participant data capture. There are a number of trade-offs to take into consideration regarding development and deployment. More mobile ePRO cases will emerge that need to interface with Bluetooth LE-capable medical class devices as part of their data collection capability, which will reflect on which development and deployment strategies to choose.

Justin Johnson is Chief Executive Officer at FirstApp. He has over 20 years experience in the software industry and is the co-founder of several technology companies ranging from clinical trial software development to virtual environments and merchandising in the games and entertainments industry. In recent years his primary focus has been on user data capture using mobile devices combined with big data processing and analytics. His latest company, FirstApp, specializes in mobile software for mHealth and clinical trials. Email: justin.johnson@firstapp.com