Mobile Client Architecture

Web vs. Native vs. Hybrid Apps

Choosing appropriate client architectures is a hotly debated topic of late. Having been party to these debates and also to implementations using all three approaches, it is evident that some of the perceptions surrounding this debate don’t conform to reality. During actual implementation, particularly since the Hybrid approach is a mix of the two other approaches, some benefits and drawbacks intertwine. Additionally, one must view this decision from the multi-dimensional prism of development cost, TCO, multi-platform support, performance, consumer perceptions, longevity, maintainability and brand equity (in case of consumer facing applications). Since marginal adoption is the price one pays for selecting the wrong architecture, it is critically important to choose wisely.

This paper documents some of our findings, articulates our thought process, compares approaches and provides a decision framework.
About the Author

Neeraj Mehta

Education

BE in Electrical Engineering, 1992
MS in Computer Science, 2002
MBA, University of Chicago, Booth School of Business, 2007

Summary of Experience

Neeraj is spearheading the development of reusable frameworks with functional components for building mobile applications for a mobile workforce. He has close to 20 years experience in building technology products and services for telecom, software and market research companies, including the TCS mobile product Charts, which he built from the ground up. He has many interests in the enterprise space, including biometric authentication and mobility applications.
# Table of Contents

1. Abstract ........................................... 1  
2. Executive Summary ............................... 4  
3. Background ....................................... 5  
4. Introduction ....................................... 5  
5. Mobile Client Architectures ...................... 6  
7. Key business criteria for evaluating mobile architectures ......................... 11  
8. Comparative Analysis ............................ 13  
9. Trends ............................................ 15  
10. Conclusion ....................................... 16  
11. References: ..................................... 16
Executive Summary

Enterprise IT is facing a strong demand for innovative mobile applications to dramatically improve customer engagement. Additionally, the demand is growing for mobile versions of existing enterprise desktop and web based applications. In response to this, we see enterprises embarking on several mobile application development projects. One of the first steps for development of any mobile application is selecting the right client architecture. There are three popular approaches today:

**Mobile Web App:** In this approach, the application runs on a mobile browser. The browser only hosts the application's presentation layer that is designed using HTML5. The interface typically looks and behaves like a traditional web site but is designed for the mobile device form factor.

**Native App:** In this approach, the mobile application is custom built for the target device operating system with a compiled programming language like Objective C and using the native SDK. This approach offers the native look and feel, the best performance, and a great user experience.

**Hybrid App:** This approach emerged to address the inability of the Web App approach to access device sensors (like cameras and Bluetooth) while preserving its highly desirable cross-platform support. The Hybrid App approach achieves this by building HTML5-based applications that run in the browser. But unlike in the case of the Web App approach, here the browser is embedded inside a native container app that provides a bridge for the HTML5 pages to access the low-level device functions.

The design of mobile applications must take into account several unique aspects like the device form factor, sporadic connectivity, variable bandwidth, multi-platform support and user experience. On these critical aspects, the three client architectures – Web App, Native App and Hybrid App – differ significantly. This paper discusses how the three client architectures compare along key technical and business drivers like multi-platform support, access to device sensors, performance, native look and feel, app search, app distribution, app upgrades, connectivity and development/testing effort, time-to-market, budget, TCO, consumer perception and BYOD.

A decision framework is provided that assists in the selection of the appropriate client architecture for any mobile application. This decision framework first zeroes in on the Web App or Native App approach based on primary application needs and then fine tunes the decision based on other factors to decide whether the Hybrid App approach is suitable.

TCS' position on the choice of client architecture is that the ideal client architecture for any mobile application depends on the enterprise and application needs. Yet, with a solid Hybrid Application development framework, we lean towards the Hybrid approach more often than not with enterprises, since it enables multi-platform support, has lower TCO and does not limit access to the device hardware.
Background

With the proliferation of high-end mobile devices along with pervasive mobile broadband, there are new opportunities to dramatically improve customer engagement and employee-productivity. While enterprises are acknowledging this and building the capabilities required to enable mobility across the organization, IT organizations are experiencing a strong demand from business units for enterprise mobile applications.

One of the central challenges associated with building mobile applications is selecting the appropriate client architecture. Today, the debate on client architecture revolves around mobile Web App, Native App and Hybrid App approaches.

While each of these architectures has its pros and cons, it is crucial to pick the most appropriate option. Thus it is important to know and compare the actual benefits, drawbacks and use a consistent framework to select the best-fit for every mobile application.

Introduction

For enterprises, a disproportionately large portion of the demand for mobile applications is based on making existing desktop or web-based applications available on mobile devices. Thus, there is an inclination to reuse infrastructure and view mobile applications as merely a new front end. However, the design of mobile applications must take into account several unique aspects like the device form factor, sporadic connectivity, multi-platform support and user experience.

On these critical aspects, the three client architectures – web app, native app and hybrid app – differ significantly. For example, the Native App approach delivers high performance and compelling user experience by leveraging all device capabilities in an integrated way, while providing no way to support the multiple device OS platforms. On the other hand, the Web App approach scores when it comes to supporting multiple platforms, but struggles with performance and does not provide a way to access device sensors like a camera. And, with the Hybrid App approach, it is a mixed bag – it delivers multi-platform support similar to the Web App approach and it offers access to device capabilities like the Native App approach; however, the performance does not match that of Native Apps.

Moreover, there are a wide variety of applications with sets of contrasting needs and characteristics that must ultimately drive the choice of the client architecture. Here are some ways to classify an application based on its needs:

- Customer facing vs. employee facing
- Data acquisition vs. data sharing
- Rarely updated applications vs. frequently updated applications
- Textual interface similar to web sites vs. non-standard graphical user experience
- Standalone vs. connected
- Online only vs. offline support when necessary
- Custom native look and feel for each OS vs. common web look and feel
- Highly sensitive data vs. publicly available content
- Small set of highly motivated user base vs. large set of indifferent (mildly interested) users
- App store vs. web based distribution approach
- Rarely updated app content vs. frequently updated app content
- Paid app vs. free app
- Time to market
- For a limited marketing campaign vs. nearly permanent app associated with the core brand

There exists no single client architecture option that consistently outperforms the other two. Often, given conflicting real world business constraints, it’s difficult to pick the ideal fit every time. Nevertheless, it is imperative that this complex decision still be a well-informed one.

**Mobile Client Architectures**

In the early days of mobile applications development for smartphones, there were only two dominant client architectures, the web app architecture relying on the phone web browser and the native app which involved custom development for the device OS. It was only in 2009 that a third option started to emerge. This was the Hybrid App approach, which attempted to plug the main limitation of the Web App approach: the lack of access to device features such as GPS and Bluetooth.

All mobile applications today use one of these three mobile architectures. Hybrid App architecture is growing in popularity and most enterprises are evaluating various hybrid application development platforms for enterprise-wide adoption.

Often, mobile applications are just supposed to extend the capabilities of existing desktop or web applications. However, in addition to the obvious difference in the form factor and the touch sensitive screen, the user experience expected is quite different. The user is a lot less patient with a mobile application; typically, the user wants to quickly start the app, accomplish a task and be done. Moreover, the user expects a visually stunning, highly responsive and intuitive interface. Thus, mobile applications that simply port an existing desktop or web experience to the mobile without rethinking the entire user experience usually under-perform. Client architecture plays an integral part in the user experience design of the mobile application.

The universe of mobile devices comprises the entire range of phones, from low-end feature phones and tablets. Building mobile apps for feature phones is quite a different challenge and involves a different set of technologies like J2ME, BREW, content adaptation, and XHTML. With the growing rate of adoption of smartphones and tablets across the world, there is a large demand for mobile applications designed for popular mobile devices running operating systems like iOS, Android, Windows Phone/8, and Blackberry operating systems. Moreover these apps need to run on devices with large touch screens, broadband (Wifi, 3G) connectivity, camera and other sensors. Here we focus on the architecture of mobile applications designed for these smartphones and tablets.
**Mobile Web App Approach**

Mobile web apps are designed to run on a mobile web browser. HTML5 is the most popular and promising technology for ‘Write Once Run Anywhere’. Almost all mobile web browsers running on high-end mobile devices support HTML5 to a large extent, and all are trying to achieve full compliance. Thus, it is safe to consider HTML5 as the technology of choice for developing mobile web apps.

**Native App Approach**

Mobile native apps are built using the native device operating system APIs and SDKs. These are coded using a platform specific language like Objective C for iOS, Java for Android, and C# for Windows phone. One can use the standard GUI components that are part of the platform SDK, easily creating a look and feel that is native to the OS and straightforward. These apps can access all the device hardware including the various sensors and peripherals, if any. These apps are quite fast since the executables are compiled for the specific OS and are run directly on the OS. These come with their development environments including various simulators and infrastructure to do actual device testing.

![Mobile Web App Approach Diagram](image-url)
Hybrid App Architecture

Hybrid app architecture emerged in 2009 but has only become popular recently. This approach achieves the middle ground between native mobile applications and mobile web applications. While mobile web apps attempt to provide platform independence, the price one pays for this is that they do not function when the device is offline and they cannot access device hardware like the camera, Bluetooth, accelerometer, or compass.

The Hybrid App approach evolved to deliver platform independence while providing access to the device hardware and offline operation. This is achieved by building apps using HTML5 pages that run in the browser (e.g. uiwebview) embedded inside a native container app. This app then provides a bridge for the HTML5 pages to access the low-level device functions. The hybrid app is packaged as a native app and thus can be distributed from the app store. They can operate offline since the HTML5 pages are typically inserted inside the app; however, a good hybrid app development framework would allow these pages to be refreshed and update the app without having to update the native app container. Just like the Web App approach, hybrid apps leverage software staff with web technology skills while limiting the need for expertise in the native mobile operating system development.
Key Technical Criteria for Evaluating Mobile Architectures

Multi-Platform Support: The pure Native App approach does not provide multi-platform support. This is in stark contrast to the Web App and Hybrid App approaches that can potentially support any platform, including iOS, Android, Windows Phone, and Blackberry. However, in our experience, due to the non-compliance or atypical interpretations of the HTML5 standards by mobile browsers on different mobile operating systems, a solution with no platform specific code is elusive. Code branches that are specific to the mobile browsers are necessary for true multi-platform support. A good user-interface would also leverage the various JavaScript libraries like jQuery and Sencha Touch, and thus there is also reliance on their standardized implementation across platforms.

The more sophisticated the visual design, the bigger the challenge related to cross-browser compatibility. Additionally, older versions of browsers present their own challenges. The claim ‘Write Once Run Anywhere’ that Web Apps boast of is true, but if the expectation is that there will be no hurdles, then one is likely to be surprised.

Access to Hardware Sensors: One of the main disadvantages of the Web App approach is the inability to access device capabilities. While GPS, touch, contacts, calendar access are among features that have been supported by the newer versions of HTML5, one can still not access features like the camera, Bluetooth, compass or microphone through HTML5/JavaScript. It is safe to assume that slowly HTML5 will support...
most common mobile phone capabilities, but it will always lag behind native apps since device manufacturers will continue to add more sophisticated capabilities to smartphones that HTML5 cannot support, such as barcode scanners or magnetic swipe readers.

With the Hybrid App approach on the other hand, one can access all device sensors. Most popular hybrid application development frameworks provide access to almost all the important device capabilities. But, seamless integration of the device sensor functionalities into the application user interface can be difficult. For example, it would take some platform specific coding using a native SDK to build an augmented reality application. If you want to work with peripherals, then one may have to extend the capabilities provided by the hybrid application development framework through plug-ins.

Of course, native apps are ideally suited to use all the device sensors and various peripherals. It would provide a seamless and native user experience that is responsive.

**Performance:** Mobile web apps and hybrid apps are slower since their code is interpreted by the JavaScript engine running within the browser. Thus, if the user interfaces are graphic heavy or require excessive data processing, the Web App and Hybrid App approaches struggle to deliver the goods. As per PerfMarksII report by spaceport.io, the best iOS and Android smartphones run HTML5 eight times slower than a laptop computer on average; the results are dramatically worse when comparing against an average smartphone. Cross-platform mobile application development is what enterprises want, and although HTML5 may seem like the way to get it, performance of mobile browsers is still nowhere close to where it can be used for demanding applications. Thus, when it comes to computational needs, the Native App approach outperforms the other two approaches by a wide margin.

**Native Look and Feel:** There are several web frameworks that provide libraries that can be used by mobile web apps and hybrid apps to re-create and imitate native mobile interfaces and behavior. However, the effort required to build these interfaces using native code is a fraction of the effort required to mimic the native look and feel. One of the primary reasons to select the Web App approach is multi-platform support. If one wants to have a native look and feel for each platform, then it is difficult to do this, particularly with disparate code bases. Typically, with mobile web apps and hybrid apps, one has a common interface across various platforms and thus one forgoes the look native to the platform. In fact, one of the key differentiating factors of a typical mobile web app or hybrid app is that it often belies its true roots since it looks like a website.

Of course with the Native App approach, one automatically gets the native look and feel.

**Search, Distribution and Upgrades:** Mobile web apps can be hosted on a web server like any website; they do not require any download or installation. In contrast, native apps and hybrid apps are typically hosted in an app store and must be downloaded and installed. Unlike consumer apps, native enterprise apps are hosted in enterprise app stores. Consumer app stores are typically managed by device manufacturers like Apple, Google and Microsoft. One can benefit from the integrated payment system in an app store, but pay up a sizeable fraction of the price for the app to the app store. Mobile web apps, on the other hand, are either free or must rely on their own payment infrastructure. Moreover, particularly for iOS apps, there is an elaborate app validation process that every update must go through, and then the user must explicitly update the app by repeating the process of downloading and installing. With mobile
web apps, just update the web app on the backend, and the next time the user accesses the app, the user gets the latest and greatest. Moreover, the content in mobile web apps is searchable using the standard search engines. Hybrid apps share this advantage as well.

**Offline Capability:** Web apps require connectivity to be operational. This means these web apps stop functioning when the user experiences unexpected loss of connectivity due to network (Radio Frequency) issues or the device deliberately goes off the grid (like when on an airplane). HTML5 has some support for offline functionality, but not all mobile browsers support this in a standard way. With the Native and Hybrid App approaches, one can access the device database and implement a synchronization engine that would allow seamless operation when the device has sporadic connectivity. With the Hybrid App approach, although the interfaces are designed using HTML5, these pages along with the other resources like JavaScript, images, or videos are typically inserted inside the app and thus don’t need to be downloaded from any server.

**Development & Testing:** For a typical native application, roughly 20% of the effort is user experience design, 20% is requirements and design, 40% is development and 20% is testing and miscellaneous.

In our experience, for the first platform, there are only marginal differences in the development efforts for web, hybrid and native apps. The benefits come when support for additional platforms is implemented. When only one platform is targeted, it is often easier and more cost-effective to build a native application, since the integrated environment for native app development and testing speeds up development significantly. It is also possible to automate testing for native applications more easily than for web or hybrid apps. However, with the Native App approach, when a new platform is added, most of the app development needs to be redone, usually by a new team since developers with expertise on the new OS are necessary. Thus, even when the multiple platforms are not built concurrently, there is little benefit that carries over from having built on one platform. We have experienced that debugging native applications is generally easier than debugging mobile web apps and hybrid apps that run within the mobile browsers.

With mobile web apps or hybrid apps, the engineering skill set required is similar to that of web developers and engineers. Workers with this expertise are relatively more abundant than their traditional web counterparts. However, the development is more involved due to the inconsistent compliance and incomplete implementation of the HTML5 standard across mobile browsers. Additionally, debuggers and other tools are non-standard across device browsers. Development and debugging is even more complicated when working with hybrid apps, since there is an added layer of indirection with the browser control being embedded inside a native container, and because of interaction with device hardware through the JavaScript Bridge.
Key business criteria for evaluating mobile architectures

**Time-to-market, Budget and Life cycle:** Before developing any mobile app, it is important to think of the full life cycle of the app from development to when it is retired or replaced. If it is a simple app with native look and feel and only one platform is targeted, then it is typically cheaper to use a Native App approach. On the other hand, if one needs to support multiple platforms from the get-go or in the near future, web app or hybrid app development offers a more cost-effective option. In terms of time-to-market, when starting from scratch for the first platform, the three architectures are competitive; but with subsequent platforms and updates, web apps and hybrid apps have an advantage. In general, at the time of conception of a mobile application, one must look at the likely adoption rates, the change of the device population over time, the update mechanism, and a means of support since these would likely influence the choice of the client architecture.

**Total Cost of Ownership (TCO):** The cost of keeping an app updated constitutes a large part of the total cost of ownership once the app has been built and deployed. With frequent updates to the operating systems, it is imperative to retest the app when a major OS release comes about. Testing, updating and upgrading native apps is more effort than it is with web apps. With hybrid apps, however, one must test both the native container as well as the HTML5 based interface. Native apps are also likely to incur associated helpdesk costs for supporting people who have not updated an app. This challenge exists to some extent for hybrid apps, but it is absent in the case of web apps. Typically, mobile applications do not simply mimic their desktop or web counterparts; they tend to focus on a few use cases and improve the way the user finishes a task by leveraging the mobile device in some way. This often leads to changes in established processes, and thus as part of the TCO, one must also include the Organization Change Management ramifications.

**Consumer Perceptions:** Mobile apps are a way for a brand to connect personally with its customer base. Though this presents a great opportunity, a badly designed app can hurt the brand as well. Product managers want the widest reach and thus lean towards web apps or hybrid apps. But, the early adopters and the most loyal customers are the ones most likely to use the applications the most. Building to the lowest common denominator in order to reach the widest audience means that the user experience for these opinion makers is sub-optimal. There are brands that are building both a web app for wide reach and a powerful native (or hybrid) app for heavy users.

**Bring Your Own Device (BYOD):** Earlier, enterprises would provide enterprise-liable devices to its employees. This made the selection of platform for enterprise apps simple. However, with enterprises embracing BYOD, employee-liable devices with different operating systems, form factors, brands, capabilities and features will need to be supported. Thus, most applications that were built or are being built assuming a single platform will have to support other operating systems. As a result, the TCO on these applications and ROI models will need to be updated. Enterprises are realizing this and most enterprises that are aggressively adopting mobility are evaluating hybrid mobile application platforms to adopt as an enterprise standard.
### Comparative Analysis

<table>
<thead>
<tr>
<th>Key Criteria</th>
<th>Web App</th>
<th>Native App</th>
<th>Hybrid App</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multi-Platform Support</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Access to Hardware Sensors</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Access to peripheral SDKS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Performance</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Native Look and Feel</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>App Search through App Store and distribution</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Upgrades</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Backward/Forward Compatibility</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Development Support</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Testing</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Simulators</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Debugging</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Offline Support</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time-to-Market</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Budget</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total Cost of Ownership</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BYOD</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Content Search</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Paid Apps</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Advertising</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Analytics</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Unique, and Interactive Game like Interface</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Graphics</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Media</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Full Support**
- **Some Support**
- **Limited/No Support**
Decision Framework

- App Store deployment
  - Unique/gamelike interface
    - Yes
    - Native App (expensive options for multi-platform)
    - No
    - Targeting multiple platforms?
      - Yes
      - Hybrid App
      - No
      - Frequent updates to the app?
        - Yes
        - Native App
        - No
        - Unique/gamelike interface?
          - Yes
          - Native App
          - No
          - Targeting multiple platforms?
            - Yes
            - Hybrid App
            - No
            - Brand/Consumer facing app?
              - Yes
              - Hybrid App (Lower TCO as other platforms are added)
              - No
              - Native App
            - Tight Schedule and Budget
              - Yes
              - Native App (Cheaper/Quicker)
              - No
              - Large data processing requirement
                - Yes
                - Web App
                - No
                - Hybrid app (Better Performance)
          - No
          - App needs to be searchable
            - Yes
            - Web App
            - No
            - Web App

- Native App
  - Unique look & Feel is a requirement?
    - Yes
    - Native App
    - No
    - Hybrid app (Better Performance)

- Hybrid App (Lower TCO as other platforms are added)
  - Required offline?
    - Yes
    - Web App
    - No
    - App requires device sensors (camera/accelerometer etc)
Trends

Hybrid

All three client architectures have their benefits and drawbacks. It is also clear that the ideal client architecture for any mobile application depends on the specific set of application and business demands. There are, however, certain trends that are predisposing enterprise IT to lean towards the Hybrid App approach.

The key trend is the fragmentation of the smartphone user base and the emergence/adooption of BYOD by enterprises. In the early days, iOS was a dominant smartphone platform, but as Android and Windows Phone market shares rise, and device fragmentation becomes even more acute as consumers in emerging economies lean towards non-iOS devices, enterprises must look at client architectures that provide multiplatform support cost-effectively, without compromising application performance or user experience. This is why we see that most enterprises have either selected or are seriously evaluating a hybrid application development platform.

Beyond Hybrid

The true limitation, as mentioned in the previous paragraph, is the inability to build compelling native user experiences using HTML5. New tools are emerging that provide Java and C# compilers for iOS, Android, and Windows Phone 8 mobile devices. Additionally, they provide different class libraries for UX on different OS'. With this infrastructure, and careful architecture using Model-View-Controller, one can build apps with significant portions of reusable code. Basically, all non-UX code can be reused while the UX code is platform specific. These tools allow you to build applications with sophisticated native UX for multiple platforms with a convenient tool chain and limited platform specific coding.

Conclusion

Today, the client architectures used for mobile applications can be broadly classified into the mobile Web App approach, Native App approach and Hybrid App approach. Each of these approaches has its strengths and drawbacks. ‘One size fits all’ is not applicable.

The choice of the client architecture for any mobile application depends on the demands of the application and business considerations. There is a distinct difference between the Web App approach and the Native approach. The two key drawbacks of using the Web App approach over the Native approach are:

- Inability of web apps to access the device sensors/other hardware
- Difficulty in building a unique game-like interface with native look and feel.

The key benefit of the Web App approach is cross-platform support. The Hybrid App approach emerged precisely to bridge the gap between the Web App and the Native App approaches. With the Hybrid App approach, one gets cross-platform support without having to forgo access to device capabilities.
The decision framework provided in this paper first zeroes in on the Web or Native App approach based on primary application needs and then fine tunes the decision based on other factors and suggests the Hybrid App approach if suitable.

TCS’ position on the choice of client architecture is that the ideal client architecture for any mobile application depends on the enterprise and the application needs. Yet, with a solid hybrid application development framework, for enterprise applications we lean towards the Hybrid App approach more often than not since it provides multi-platform support cost-effectively, has lower TCO and does not limit access to the device hardware.

References
2. http://docs.xamarin.com/ios/tutorials/cross_platform/building_cross_platform_applications
About TCS Mobility
TCS Mobility delivers best in class mobility services and solutions with complete mobility lifecycle consultation and development service along with customization on top of pre-built solutions to provide the best in class benefits to our customers.

Our passion for providing the very best and comprehensive mobility services and solutions to our customers is realized through our deep expertise in mobility gained through the experience of a strong team that has a career long experience in mobile technologies and a dedicated mobility user experience design team that is committed to leveraging the unique native capabilities of each device platform.

Contact
For more information about TCS’ Mobility contact mobility.solutions@tcs.com

Subscribe to TCS White Papers
Feedburner: http://feeds2.feedburner.com/tcswhitepapers

About Tata Consultancy Services Ltd (TCS)
Tata Consultancy Services is an IT services, consulting and business solutions organization that delivers real results to global business, ensuring a level of certainty no other firm can match. TCS offers a consulting-led, integrated portfolio of IT and IT-enabled infrastructure, engineering and assurance services. This is delivered through its unique Global Network Delivery Model™, recognized as the benchmark of excellence in software development. A part of the Tata Group, India’s largest industrial conglomerate, TCS has a global footprint and is listed on the National Stock Exchange and Bombay Stock Exchange in India.

For more information, visit us at www.tcs.com

All content / information present here is the exclusive property of Tata Consultancy Services Limited (TCS). The content / information contained here is correct at the time of publishing. No material from here may be copied, modified, reproduced, republished, uploaded, transmitted, posted or distributed in any form without prior written permission from TCS. Unauthorized use of the content / information appearing here may violate copyright, trademark and other applicable laws, and could result in criminal or civil penalties. Copyright © 2012 Tata Consultancy Services Limited