How Business Apps Go Mobile

Mobile apps have generated a completely new market. According to an InMobi study, "*50% of the average global mobile web users now use mobile as either their primary or exclusive means of going online*" (links.basis.com/fibnd). ISVs with an exclusively business-oriented product portfolio need to offer mobile apps in order to stay competitive. This article provides an overview about the possible approaches and their suitability for business applications.

Mobile Applications are a Must-Have

The market for mobile applications is booming while, in terms of smartphone operating systems, there are not many players left. According to the International Data Corporation (IDC), "*Android and iOS powered 85% of all smartphones shipped in the second quarter of 2012*" (links.basis.com/ahdas).

Seen from a programmer's point of view, though, the mobile apps market still looks pretty unclear. There is no "one size fits all" way to write native apps that would work on all OS platforms. Too different are the specific UI concepts. So, at first sight it seems you would either have to write a separate native app for each platform you intend to cover, or make use of the fact that all mobile devices have Internet access and are equipped with web browsers. But web apps have their downsides as well, which we'll discuss later.

To make things worse, most ISVs do not start with a blank slate; they already have their desktop applications to maintain. In most cases, they just want their mobile app to offer an add-on functionality of their desktop application for specific user groups, such as salesmen, technicians, or management who cannot easily access the company's IT system when they are mobile. Developing several parallel mobile apps that would require working in different development environments while at the same time having to assign resources to the maintenance of their main application, is not a feasible option.

Tools for Cross-Platform Development

With over 100 tools on the market meant to enable or simplify cross-platform development, developers can choose among three basic ways to avoid writing multiple native apps - cross-compiled apps, hybrid apps or web apps (**Figure 1**). We will discuss these ways and, additionally, will see how BASIS' Browser User Interface (BUI) technology fits into the overall picture.

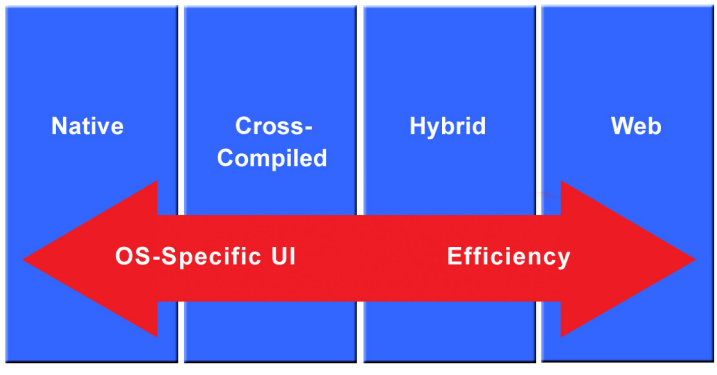


Figure 1. There are four basic ways to write apps for multiple mobile device platforms.

By Patrick Schnur
European Marketing/PR

Native Apps...Snazzy to Run, Cumbersome to Make

Developers design native apps for a given operating system in the respective programming language – Java for Android and BlackBerry, Objective C for iOS, .Net for Windows, and so on. By programming native apps, you make sure that your application can make full use of all hardware resources of the device, and that all interfaces work together uniformly. Therefore, they are best for computation-intensive applications, and are preferred for visualizing 3-D graphics and for games. **Figure 2** illustrates the typical structure of a native app.

The obvious downside are the costs for developing and maintaining several applications in different programming languages meaning a steep learning curve, plus implementing different platform-specific methodologies and compliance with rules and regulations to deploy the apps.

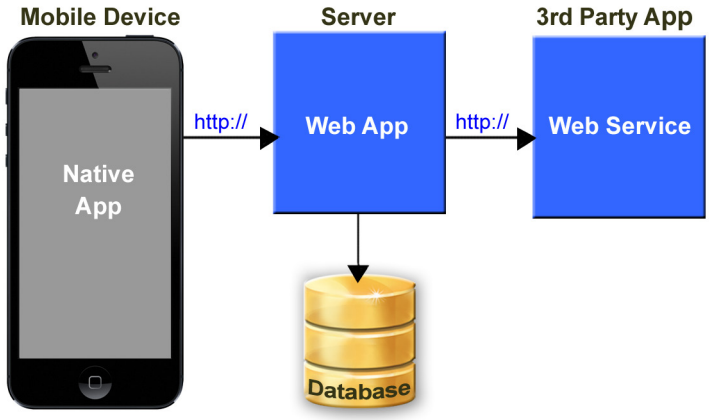


Figure 2. Typical architecture of a native application

Cross-Compiling Apps the Best of all Worlds?

The idea behind cross-compiled apps is to write the source code just once and then compile it via code generators to the various platforms. What you get are in principle native apps, but based on the lowest common denominator in terms of functions and the UI. The downside is, again, a pretty steep learning curve and no saving on the deployment hurdles. Cross compiling apps with a code generator as shown in **Figure 3** gives you the lowest common denominator of all native UIs.

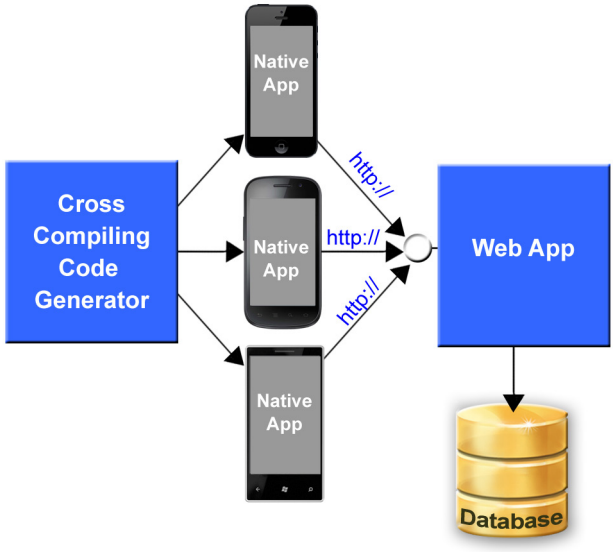


Figure 3. Typical architecture of a code generator

Hybrid Apps Mind the Gap!

Hybrid apps try to combine the advantages of native apps and web apps. Technically, hybrid apps are HTML5- and JavaScript-based web apps connected by containers written in the respective native languages. Hybrid apps can access at least some of the hardware resources of smartphones and tablets, without the need to write completely separate source code for each platform. Such functions are directly using the camera, the contact database, media information, device features and the device's internal database. The native containers needed to achieve this are bundled in a framework such as Phonegap. Hybrid apps generated with frameworks like "Phonegap" allow mobile apps to access some of the hardware resources of smartphones and tablets as illustrated in **Figure 4**.

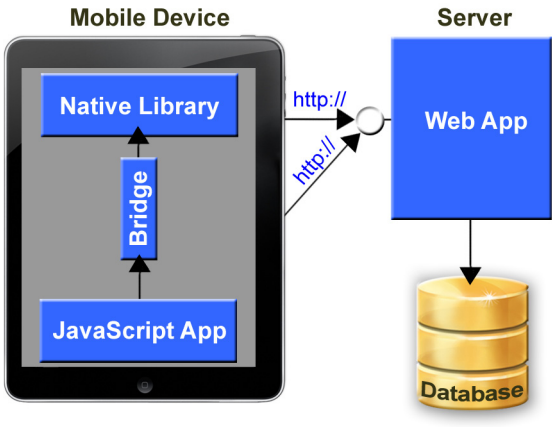


Figure 4. Typical architecture of a hybrid app

Web Applications

Web apps, obviously, are applications which uses a web browser as the client. Since nearly all commonly used mobile devices are equipped with such browsers, web apps are a convenient way to reach potentially all target groups with minimum development effort, and none of the deployment headaches.

But web apps written for the desktop often sport a completely different UI than would be expected for smartphones or tablets, and the user interface is important for user acceptance. Therefore, web apps usually need tweaking to make them more resemble the look-and-feel of native apps which consumers are used to. It is not possible to mimic the native UIs completely, as they

differ from each other significantly, but it generally is possible to adapt to the needs of small screens and gesture-controlled touchscreens. See **Figure 5.**

Where Does BUI fit in?

With the browser user interface, or BUI for short, BASIS offers a fifth approach to developing cross-platform mobile apps. BUI s implemented with GWT (www.gwtproject.org), which cross-compiles Java to JavaScript and enables us to create a browser-based BBj client using standard HTML, CSS, and JavaScript.

The Heimbas (www.heimbas.de) BUI application for care management in nursing homes runs on iPads and all other tablets that are equipped with a JavaScript-enabled browser. During the doctor's visit, all relevant patient information is accessible at the tap of a finger, right at the bedside. **Figure 6** and **Figure 7** shows examples of the app running on iPad, examples of patient care and quality management made easy.

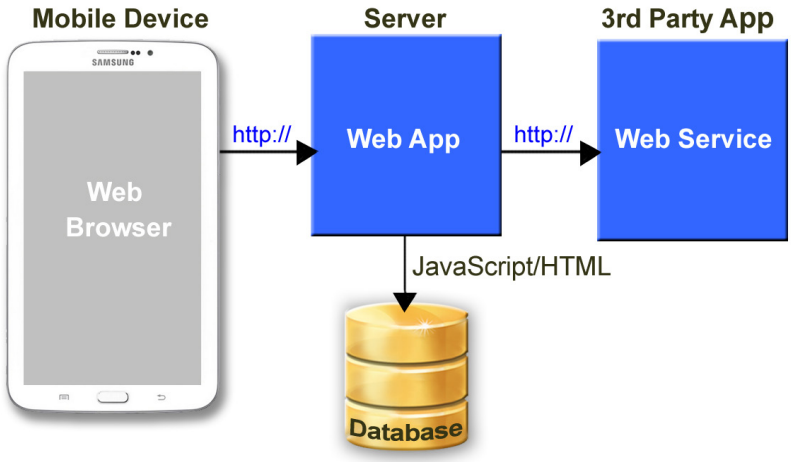


Figure 5. Typical architecture of a server-side web app

Figure 6. Heimbas Care Management running on iPad

Figure 6. Care Management quality issues that need attention

By generating a JavaScript application from BBj code on the fly, developers don't have the worries of developing and maintaining different applications for desktops, the web and mobile devices, they have an all-in-one solution which will work on any device with a JavaScript-enabled web browser. To adjust the UI to the specific needs of smartphones and tablets, BASIS customers use cascading stylesheets, or CSS, which they do with impressive results.

Emque Consultants (www.emque.com) in New York offers a BUI app for the commercial construction industry to which BASIS mocked up some CSS modifications shown in **Figure 7.**

Figure 7. Emque's construction management BUI app with BASIS-supplied CSS

BUI apps can address the GPS sensor of a mobile device with the help of the browser. Logistics service provider Wim Bosman (www.wimbosman.com) used BUI for an onboard tracking system that allows their headquarters in the Netherlands to track their trucks en route across Europe

in real-time (**Figure 8**) and assign incoming orders to the best-equipped and best-positioned truck.

While BUI may not be the best solution if you need to handle rich media or plan to create a blockbuster game, BUI is probably the most efficient methodology available in the market for data-driven mobile solutions. BUI is ideal if you need to extend an existing business application to new target groups or if you have to deploy your application to run simultaneously on multiple platforms. Consider how BUI would save you the effort of building a development team capable of applying several programming toolsets and languages. Now you could enjoy the ease of having only one source code to maintain over the life of your application. Furthermore updating the app is a cinch, just publish a new version to your server and all the client devices will get it.

Revenue Sharing

If you are concerned about the licensing costs for the millions of users who will now use your app, contact your BASIS account manager to work out a win-win revenue sharing plan. We promise not to require more than the 30% revenue share that developers currently provide to the major native app store vendors. ☺

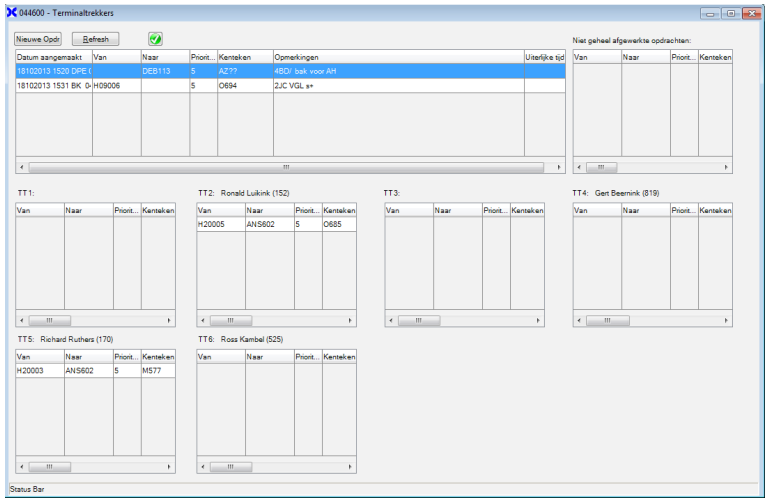
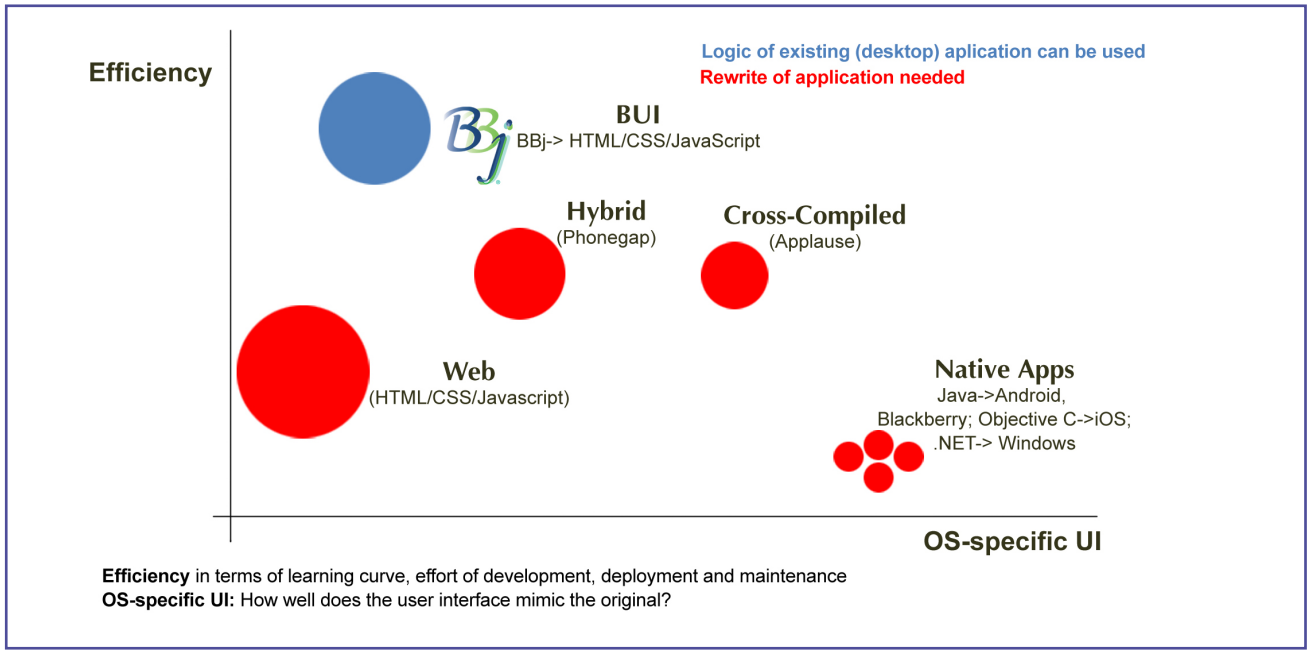


Figure 8. Wim Bosman's onboard tracking app

Summary

Technologically, there are four basic ways of developing mobile apps. While each has its pros and cons, BUI, a superior web app deployment paradigm, is by far the most cost effective and risk-free way to build mobile data-driven business applications, especially those that are derived from pre-existing desktop applications. And perhaps best of all, there's no app store approval process or multiple deployment and update scenarios to contend with. ■



Mobile apps in a nutshell: BUI saves ISVs a lot of time and effort in making mobile apps for various platforms.

Find all the available statistical data about the consumer mobile app market that you could possibly dream of; and in easy to understand graphical format, at links.basis.com/xtvtk

Go to basis.com and CSE search on BUI for a plethora of good reads!