Disclaimer

This document in any form, software or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this confidential material is subject to the terms and conditions of your Oracle Software License and Service Agreement, which has been executed and with which you agree to comply. This document and information contained herein may not be disclosed, copied, reproduced or distributed to anyone outside Oracle without prior written consent of Oracle. This document is not part of your license agreement nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

This document is for informational purposes only and is intended solely to assist you in planning for the implementation and upgrade of the product features described. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described in this document remains at the sole discretion of Oracle.

Due to the nature of the product architecture, it may not be possible to safely include all features described in this document without risking significant destabilization of the code.
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Purpose</td>
<td>1</td>
</tr>
<tr>
<td>Introduction</td>
<td>1</td>
</tr>
<tr>
<td>Classifying Mobile Applications</td>
<td>2</td>
</tr>
<tr>
<td>Web-based Mobile Access</td>
<td>2</td>
</tr>
<tr>
<td>On-Device, Native Mobile Access</td>
<td>2</td>
</tr>
<tr>
<td>Specialty Mobile UIs</td>
<td>3</td>
</tr>
<tr>
<td>Challenges in Developing a Mobile Strategy</td>
<td>5</td>
</tr>
<tr>
<td>Multiple Devices, Multiple Programming Environments</td>
<td>5</td>
</tr>
<tr>
<td>Data Management</td>
<td>5</td>
</tr>
<tr>
<td>Provisioning and Security</td>
<td>6</td>
</tr>
<tr>
<td>Oracle Solutions for Enterprise Mobile Applications</td>
<td>7</td>
</tr>
<tr>
<td>A Common Service Layer</td>
<td>7</td>
</tr>
<tr>
<td>A Consistent Programming Approach</td>
<td>7</td>
</tr>
<tr>
<td>A Simpler Declaritive Programming Approach</td>
<td>7</td>
</tr>
<tr>
<td>Oracle ADF for Mobile Devices</td>
<td>8</td>
</tr>
<tr>
<td>Multi-Channel Development</td>
<td>11</td>
</tr>
<tr>
<td>Recommended Approaches for Mobile Enterprise Applications</td>
<td>13</td>
</tr>
<tr>
<td>On-Device vs. Mobile Browser Based Client</td>
<td>13</td>
</tr>
<tr>
<td>Web-based Mobile Access</td>
<td>14</td>
</tr>
<tr>
<td>On-Device, Native Mobile Access</td>
<td>14</td>
</tr>
<tr>
<td>Specialized Mobile UIs</td>
<td>16</td>
</tr>
<tr>
<td>Security and Application Provisioning</td>
<td>18</td>
</tr>
<tr>
<td>Conclusion</td>
<td>19</td>
</tr>
</tbody>
</table>
Purpose

This document is intended to provide an insight into Oracle’s vision for the development of mobile computing solutions for the enterprise. It is intended solely to help you assess the business benefits of, and options for, the delivery of enterprise applications and data to mobile devices.

Introduction

With the recent explosion in the smart-phone and mobile device market, much interest is now focused on the possibilities for both improved workflow and revenue streams that may be afforded by this growth area. In the traditional enterprise application development market, standardization has played a major part in streamlining and simplifying the production of enterprise applications. In the mobile space, however, the wide array of incompatible hardware devices and operating systems impose some significant barriers. Some of the challenges facing developers wanting to work in this space include:

- Wildly differing device capabilities in terms of hardware features and available software APIs.
- A wide range of form factors and screen sizes.
- Variety of input methodologies and user experience expectations; for example devices using multi-touch screens vs. touch-pad driven/non-touch screens.
- Carrier specific constraints and features.
- A large array of mutually exclusive development environments and languages, often tied to specific hardware vendors, for example Objective-C used for Apple devices and Java for most but not all BlackBerry devices.

As a result of the efforts and costs inherent in addressing these challenges, organizations looking to support mobile use cases will often have to severely limit the number of supported platforms and devices.

This strategy document discusses these general challenges, and then outlines how Oracle approaches this mobile development space.
Classifying Mobile Applications

When considering how to develop a strategy for extending enterprise applications into the mobile sphere, it is worth reviewing the modes of mobile data access which are commonly encountered or requested.

Web-based Mobile Access

Although the potential for the web based delivery of information through to mobile devices has existed for years, it has always been hampered by two factors; primarily, the availability and cost of bandwidth on such devices, but closely followed by the primitiveness of protocol support. The smart-phone revolution has somewhat addressed both of these issues by forcing the carriers to offer unlimited data plans, and by providing more capable browsers. This new generation of mobile browsers is almost completely compatible with the pages served up for desktop browsers. (Except for technologies such as Adobe Flash) and this is further helped by the availability of larger, higher fidelity screens on those devices. Today then, at least a significant subset of the mobile market can be targeted with generic web sites that have no particular concession to mobile browsing. However, when we move beyond basic information delivery into the enterprise space, it is unlikely that this one-size fits all approach will be successful. We will discuss the underlying reasons in the challenges section later in the document.

On-Device, Native Mobile Access

Unlike the browser based connected mode of data access, applications that anticipate the need to operate independently from the network, or that need native services access to the device have to be written as targeted applications for a particular device. These applications may be written specifically for that device OS or for a virtual machine or other framework running on it.

The common use cases for this kind of applications are:

- Applications deployed to specialized, often ruggedized industrial mobile devices designed for offline data gathering, for example stock taking or field service applications

- Mobile workforce applications, for example, sales enablement applications in the insurance and pharmaceutical space, which have traditionally run on commodity laptops. Because the hardware and operating system are standard in these cases, it is possible to re-use certain components from existing enterprise applications or packages for the application. Data synchronization, however, remains as the major challenge.

- Mobile applications that extends Enterprise Applications to mobile users. For example, extending the functionality of an Oracle Financials application to support expense report reporting and approval for managers on their mobile devices. Traditionally users of Enterprise Applications are expected to access these types of functionality from their desktop PC. However, with the general availability of smart-phones and increasingly mobile work forces,
users now expect certain Enterprise Application functionality to be accessible from their mobile device.

Traditionally these use cases are mostly supported through a laptop-based device. However, as smart phones and tablets become more powerful and have a richer set of capabilities such as location awareness, they are increasing capable of supporting these use cases. As matter of fact, a new class of mobile application is emerging that requires extensive integration with on-device services such as PIM (calendar, contact, etc) applications, GPS, integrated smartcard/barcode scanner, etc. Such an application allows end-user to seamlessly navigate between multiple mobile applications without losing context, and enables business applications to leverage data gathered/contained by device-native peripherals and services.

Specialty Mobile UIs

This final category of mobile interfaces encapsulates a wide range of use cases. Some of these cases may overlap with the connected / disconnected scenarios which we have already discussed, but today are all implemented as one-off and highly specialized applications. This portfolio of applications types includes:

- **Games** Although gaming makes up a significant portion, if not the majority of the mobile market, it is an area which is somewhat outside of this discussion of enhancing enterprise applications with mobility.

- **Widgets** In the case of widgets, the developer is tied to specific operating system APIs, as in the case of Android widgets, or to the APIs provided by an add on frameworks such as Yahoo! Widgets. Although widgets are often very small and simple applications, the correct implementation can be challenging, particularly on mobile devices where resources such as memory and CPU cycles need to be used as sparingly as possible.

- **Website Implementations** On platforms like the iPhone, there has been an explosion of application development effort around creating specialize applications that implement a portion of website functionality into a native mobile applications. Many examples of this application class exist, everything from the social media application space such as Facebook, LinkedIn and Yelp, to banking and news applications such as the BBC iPhone application. All of these applications are highly tailored to the form factor and user interaction model of the targeted device, and will almost always have some integration with device services such as GPS or camera. They often surface only a subset of the functionality of the full parent web site. However, because its content and user interface are optimized for mobile use, these applications are often extremely popular. This class of application provides a sticky presence for the vendor on the mobile device, which establishes a more immediate and intimate connection with the user than a simple browser bookmark. As such, the primary driver for creating these applications is not necessarily a functional one but is more of a marketing tool.
• **Data Collection Applications** As was discussed in the section on disconnected applications, the newer smart-phones are beginning to penetrate into the market formally dominated by specialized industrial devices. Data entry applications using these devices will often be a combined hardware and software combination. An increasingly common example being a point of sale system implemented using a sleeve for a phone which provides both barcode scanning and card reading capabilities.
Challenges in Developing a Mobile Strategy

The range of use cases outlined already has hinted at some of the problems inherent in establishing a common approach for all mobile development within the enterprise. For the connected browser based applications, the promise of single source application delivery is quickly dashed by the inconsistent implementation of web browser standards such as CSS, JavaScript and so forth across devices. However, more significant perhaps is the actual usability of a website designed for a large desktop browser screen on a relativity tiny display. The user will frequently have to scroll just to view the whole screen, and data entry is often painful.

The reality then for this connected mobile application use case is that significant design effort needs to be invested into the presentation of functionality to the user on that specific device. For example a single data entry form on a desktop browser may need to be drastically simplified, split up onto multiple panels or pages, and rearranged to better suit the navigation gestures on the device. Although bandwidth and data consumption is less of an issue than it was, even with 3G/4G/xG phones the download speed is a significant influence on the design and page size, as well as occasionally by lack of local caching.

One traditional approach is to leverage “trans-coding” technique that allows developers to define rules on how to split up a desktop browser page for mobile display. This approach requires the desktop browser page to be mostly static and simple, and relies on a complex and often outdated device database to determine how to split up the page. Trans-coding typically results in sub-optimal breakage of the desktop page, leading to significant usability issues.

For the offline-capable and specialty use cases, the significant challenges are around the range of skills needed to create the applications, the problems of provisioning and security, and of course, data synchronization.

Multiple Devices, Multiple Programming Environments

From the development standpoint, the large range of possible mobile devices presents one of the greatest challenges. Each of the major smart-phone families uses different programming languages, APIs and development environment. This factor will generally constrain organizations into supporting only one or two device types with separate pools of highly specialized developers for each.

Data Management

When building mobile extensions to existing enterprise applications, a key part of the design has to address data management across the application. In the always connected scenario, the mobile specific UIs can continue to use the enterprise data-stores and services without additional issues. In the disconnected cases, however, there are a range of additional problems to tackle.

For example, how much data needs to be stored on the mobile device for offline use? What subset of data is selected? For a data acquisition use case, this may not be a significant problem.
However, for a mobile sales force application, decisions will have to be made about exactly what customer, product and historical data should be replicated onto the device for offline access. Additionally you may have to consider if all of the information within a business object is relevant, and perhaps create specialized lightweight versions for mobile use.

All of these design decisions and the infrastructure to support data selection and synchronization can add significantly to the development costs of the application.

Provisioning and Security

As soon as enterprise information, be it in the form of email messages or a local database on the device as part of an application, there is a risk of unauthorized access. Mobile devices are by nature easy to be stolen or lose. As such a large part of any broader mobile enablement strategy has to be around the problems of device provisioning, remote de-provisioning and general security. Most enterprises already have an existing security infrastructure and mobile device/application provision infrastructure. The challenge is to provide an integrated strategy that ensures the security infrastructure works seamlessly with the device/application provision infrastructure.
Oracle Solutions for Enterprise Mobile Applications

Oracle, having recognized and experienced the challenges of expansion into the mobile space, has defined a set of solutions which can address the key enterprise mobile enablement use cases at minimum cost. The key drivers of Oracle’s solution set in this area are:

A Common Service Layer

Mobile distribution channels should be able to share common infrastructure and services with the rest of the enterprise. The developer should not have to create a separate and specialized data or services layer to service the expansion into the mobile space. Accordingly the core service development frameworks recommended by Oracle, ensure that the repurposing of an existing service for use in the mobile scenario is a simple task. So, for example, the developer can declaratively expose an existing service in multiple modes including publication as Web Services or RESTful APIs.

A Consistent Programming Approach

As we have seen, in any comprehensive mobile solution, developers may be targeting a wide range of different devices and form factors. This applies to both the connected and disconnected use cases. Oracle has concentrated on providing a common set of tools with the Oracle JDeveloper IDE and a common programming model for use across multiple form factors and capabilities. Furthermore, the programming skills needed for mobile application development are not specialized, but rather extend the existing skill set required for building conventional desktop browser based applications. This is accomplished using components of the Oracle Application Development Framework (ADF).

The Oracle Application Development Framework (ADF) is an advanced runtime framework based on the Java Enterprise Edition standard. ADF provides a layered architecture based on the established Model-View-Controller design pattern which allows multiple User Interfaces to connect to multiple data services using a pluggable binding layer. ADF places a great deal of emphasis on declarative application configuration through metadata, and providing developers with a programming model that abstracts underlying differences in infrastructure. The ADF framework is supported by a rich and comprehensive development environment, the Oracle JDeveloper IDE. Oracle JDeveloper provides highly productive tooling on top of the framework in the form of WYSIWYG screen editors, visual models for process and page flows, drag and drop data binding, and much more.

A Simpler Declaritive Programming Approach

Oracle simplifies the development process for mobile with a two pronged approach. Firstly, all development is consolidated into a single highly standardized development environment using the JDeveloper IDE. Secondly the device specific implementation details are hidden from the...
developer through the use of a declarative metadata driven development process enabled by the ADF framework. The examples of this abstraction are of course seen in the declarative definition of the user interface, and the same approach also gives the developer access to device specific services such as the Personal Information Manager (address book, calendar etc.) data, camera and GPS. All of these services conventionally present a challenge due to differing implementations and APIs on different devices. The Oracle vision is to unify them all behind a generic service abstraction that the programmer can interact with, without having to be concerned with the implementation details on the individual target devices.

Oracle ADF for Mobile Devices

Oracle ADF is able to provide a common service provision and data-binding layer to service a variety of UI types, everything from Swing based desktop applications and Microsoft Office UIs, to rich AJAX enabled web applications across all of the popular browsers. When considering the extension of enterprise applications onto mobile devices, however, there are two specific User Interface technologies which are provided.

Oracle ADF Browser Client

The Oracle ADF solution for web browsers is based on the JavaServer Faces (JSF) UI Standard. ADF provides a range of JSF specification based components collectively known as the Oracle ADF Browser Client. These components support a variety of capabilities found in different mobile device browsers.

At the high end, the framework supplies the ADF Faces Rich Client Components. These components allow developers to produce highly interactive pages targeted at desktop, and are compatible with certain high-end Smart-phone browsers that provide rich JavaScript and CSS support.

Additionally, a second set of mobile-optimized components is supplied to that provide compatibility across virtually all mobile device browsers. These components exploit JavaScript and CSS features to deliver richer user interfaces for smart-phone browsers, but gracefully downgrade to plain HTML for basic browsers that perhaps only support WAP 2.0.

Both of these component sets can be skinned to achieve mobile optimized look and feel, and have adaptive rendering which will automatically handle the subtle differences between browser implementations. Furthermore, mobile components also have mobile-optimized styles built into the component set, which renders a user interface that mimics the device native interfaces.

When defining pages using this component model, the developer is working with an abstract definition of the page, and does not have to worry about exactly how the components will be rendered in markup terms on the target browser. All the components also share the same back end programming model and service interaction layers, so it becomes possible to build a range of user interface versions targeted at specific screen sizes at low cost.
Specifically for the mobile use case, ADF provides the developer with some additional aids which will allow them to re-use this common component set across devices, but at the same time being able to adapt to the capabilities of those devices:

- **Device capabilities are readily available in a declarative fashion.** ADF makes information available to the developer in the form of expression language extensions which can be used to understand how to best present information and functionality to the user.

- **The ADF components have some specific rendering logic for optimizing the UI experience on mobile devices.** For example, the charting components can automatically render as lightweight .PNG images rather than using the Flash engine. Typically Flash is not an available technology on mobile phones, plus of course .PNG images are a much better delivery mechanism when bandwidth is limited. Other components have specific alternative layouts optimized for the narrow screens of feature phones.

- **Skins for iPhone provide a close match to the native iPhone look and feel.** So even though we are dealing with browser hosted connected application, a very familiar user experience can be provided to the users of that device.

**Oracle ADF Mobile Client**

The ADF Mobile Client provides an application implementation that is actually deployed to the mobile device as a native application. Because the application is running locally on the mobile device, it can have direct access to the services and the local data store on the device. ADF Mobile Client directly renders native and mobile-optimized user interface on the device. This ensures familiar user experiences across all applications on the mobile device, and consistent performance regardless of network condition.

*Develop Once, Deploy to Multiple Devices*

The ADF Mobile Client allows developers to write once, and deploy to multiple device platforms through a framework runtime that runs as native application or modules on the device. This framework runtime provides a common services layer that abstracts developers from the complexities and variations between different mobile device platforms. This minimizes the need to create the same application multiple times simply to support multiple devices.

*Familiar, Declarative Development Experience*

Unlike most of the native development tools that rely on code-driven approach, ADF Mobile Client supports a declarative, meta-data driven approach, where developers use wizards and WYSIWYG UI Designers to create the application. Furthermore, development experience is very similar to web application development, which allows developers to extend their existing development expertise to mobile application development. This dramatically lowers the learning curve and technical skills threshold for mobile application development, as well as shortens the overall development cycle to better meet ever-changing mobile user requirements.
Tight Integration with Device-Native Services and Functionality

ADF Mobile Client can directly invoke device native services such as GPS, barcode scanner, integrated camera, etc. This allows developer to leverage the full capabilities of the mobile device. Furthermore, functionality developed using device-native tools such as the Calendar application can also be directly invoked by the ADF Mobile Client.

Fully Extensible and Customizable

The ADF Mobile Client allows developers to extend and overwrite out-of-the-box functionality in each layer of the framework with custom modules. This enables developers to leverage any new device services through native interfaces, and support any functionality not provided by the framework itself. Developer is no longer constrained by framework capabilities – a requirement especially critical given the rapid pace of innovation in the mobile technology space.

Seamless Integration of Online and Offline Data Access

The ADF Mobile Client supports both online and offline access of data. Web Services provides real-time data access channel, while a local database and data synchronization services supports offline data access. A common mobile application scenario calls for real-time data access where network is available, but falls back to local data access when user is out of coverage. ADF Mobile Client can easily support this scenario.

Two server-side services support data synchronization between server applications and ADF Mobile Client applications:

- Oracle Mobile Server: Provides proven data synchronization infrastructure and application management capabilities.
- Mobile Transaction Replay Service: Replays application transactions asynchronously between client and server applications. This enables ADF Mobile Client application to leverage server-side application components, without having to replicate the same logic on the client.
Oracle ADF Mobile Architecture

ADF Mobile provides a complete architecture for supporting a variety of mobile enterprise applications. Whether mobile requirement calls for on-device, offline capable application, or real-time access through mobile browsers, ADF Mobile provides complete set of choices to mobilize enterprise applications.

Multi-Channel Development

Enterprise application users demand access to critical enterprise data through the most convenient and appropriate channel available to them. Oracle supports the following client types:

- Desktop or Laptop Browsers
- Offline Desktop/Laptop Clients
- Mobile Devices
- Office Productivity Tools such as Microsoft Excel

Key features of Oracle’s Multi-Channel Development support are:

- Common business logic layer, which provides a consistent and integrated set of components for delivering critical enterprise application services.
• Common infrastructure layer that abstracts the application from device-specific technologies and operating system, and provides a robust runtime execution environment.

• Channel-specific UI components that support complete set of enterprise application functionality, such as data display, data visualization, application services invocation, etc.

• Rich set of UI rendering technologies that delivers user interface optimized for the target client type, and robust skinning support for delivering highly interactive user interface to web-based applications. For example, when rendering desktop-browser based application, Oracle leverages technologies such as AJAX, Flash, and HTML to deliver desktop browser-optimized UI. However, when rendering user interface for mobile applications, Oracle leverages device-native UI components from the device OS, and delivers highly native user experiences for mobile users.

By supporting a variety of standards-based interfaces, Oracle’s Application Development Framework can easily integrate with any backend Enterprise Applications such as Oracle Siebel CRM, PeopleSoft, EBusiness Suite, and JDEdwards. Using ADF, any developer can enable multi-channel access to virtually any enterprise applications. End-users are no longer tethered to their desktop machines, and can access enterprise applications anywhere and anytime.
Recommended Approaches for Mobile Enterprise Applications

As we have shown, Oracle supports multiple approaches in building mobile enterprise applications, and provides mobile enterprise application developer choices of technologies to create mobile applications most appropriate for their use cases. Picking the right approach for your use case is a critical first step toward building a successful and compelling mobile enterprise application.

On-Device vs. Mobile Browser Based Client

There has been evolution in the mobile browser technologies during the past several years, especially in the smart-phone space. While the old WAP-based browsers are primarily suited to display simple data, it is now possible to build compelling and even native-looking browser-based application on the recent devices. On the other hand, smart-phones now are integrated with a variety of device peripherals and services such as GPS. Mobile browser technologies are still some time away from being able to fully exploit these services.

When considering which technologies to use, first consider the following advantages with a mobile browser-based application over an on-device mobile client:

- Increase re-use of application components: browser-based mobile application can share business logic components with desktop browser-based application, and therefore reduce the overall development effort.
- Less deployment infrastructure: browser-based application only requires a web browser on the client device and an application server that can be reached by the mobile device. On-device application requires additional infrastructure for application provisioning, data synchronization, etc.
- Wider mobile device support: A robust mobile browser-based framework like ADF Mobile Browser supports any device with a basic mobile web browser. On-device framework would typically place stricter limited on supported mobile platforms.

On the other hand, if your application requires one or more of the following functionality, then an On-Device framework like ADF Mobile Client would be a better fit:

- Extensive Device Services Integration: On-device mobile client is necessary for complete integration with PIM (Personal Information Management) applications like calendar, and with peripherals like GPS and barcode scanner. While mobile browsers have started to support interaction with these services, it stills lacks breadth and depth.
- Device Native Look-and-Feel: Mobile application users are typically accustomed to use one type of mobile device, and demand that any application running on the device to look/behaves like a native application. While browser-based application on certain
platforms (e.g. iPhone) can reasonably approximate native application UI, it still misses certain key UI components available to native applications.

- Offline Data Access and Performance: Many mission-critical mobile applications require consistent data access and performance regardless of network connectivity or condition. Support for browser access of on-device data store is still very limited and inconsistent, and therefore on-device application is still critical to access local data stores.

Regardless of whether application requirements calls for browser-based or on-device application, they can be met by Oracle’s mobile application development platform.

Web-based Mobile Access

For the categories of mobile application where web-based mobile access is sufficient, Oracle recommends the following:

- For smart-phones, consumer phones, and small tablet mobile devices, use the ADF Mobile Browser components to build web pages designed for small screen form factor and mobile navigation.
  - To further optimize user experience on mobile browsers, Oracle recommends using styles (Cascading Style Sheets) from the ADF Mobile Browser as a starting point to achieve highly-optimized mobile UI.

- For full-size tablets such as iPad or Android tablets, use ADF Faces Rich Client components to create rich and highly interactive UI that fully leverage the large screen form factor. The ADF Faces Rich Client components have been optimized to support gestures available from the touch screen user interface of the iPad tablet, ensuring users to be able to fully leverage the rich UI features offered by the iPad.

On-Device, Native Mobile Access

There are three application patterns that support various types of on-device, native mobile applications, running on either mobile devices/smart-phones or laptops:

- **Mobile Client Application with Local Database**: this offers full application functionality even if network connection is not available. Application data is available through a local database on the mobile device. Data is synchronized to the local database when connectivity is available, but generally in limited/short intervals. Local database needs to hold enough data to allow full use between synchronization cycles.

- **Connected Mobile Client Application**: this accesses all of its application data through services interface in real time. No or limited data is cached on the mobile device/smartphone. User interface directly binds to the services interface. The client application directly interfaces with the server application through the services interface.
• **Laptop Client Application with Local Database:** this supports a laptop-based client that’s fully functional without network connectivity. Application data is available through a local database on the laptop, and data is synchronized to the local database in limited intervals where network connection is available.

Each of these different types of mobile clients is comprised of the following technology components:

<table>
<thead>
<tr>
<th>Technology Components</th>
<th>Mobile Client Application with Local Database</th>
<th>Connected Mobile Client Application</th>
<th>Laptop Client Application with Local Database</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADF Mobile Client Application</td>
<td>Required</td>
<td>Required</td>
<td>N/A</td>
</tr>
<tr>
<td>ADF Mobile Client Runtime</td>
<td>Required</td>
<td>Required</td>
<td>N/A</td>
</tr>
<tr>
<td>ADF/Fusion Web Application</td>
<td>N/A</td>
<td>N/A</td>
<td>Required</td>
</tr>
<tr>
<td>ADF Runtime Libraries</td>
<td>N/A</td>
<td>N/A</td>
<td>Required</td>
</tr>
<tr>
<td>J2EE Application Server</td>
<td>N/A</td>
<td>N/A</td>
<td>Required</td>
</tr>
<tr>
<td>SQLite Database</td>
<td>Required</td>
<td>N/A</td>
<td>Optional</td>
</tr>
<tr>
<td>Berkeley Database</td>
<td>Optional (Provides more scalable alternative to SQLite)</td>
<td>N/A</td>
<td>Optional (Provides more scalable alternative to SQLite)</td>
</tr>
<tr>
<td>Oracle XE Database</td>
<td>N/A</td>
<td>N/A</td>
<td>Optional</td>
</tr>
<tr>
<td>Oracle Mobile Server</td>
<td>Optional (Needed for Synchronizing with Oracle DB on server)</td>
<td>N/A</td>
<td>Optional (Needed for Synchronizing with Oracle DB on server)</td>
</tr>
<tr>
<td>Oracle Database (Server)</td>
<td>Required</td>
<td>Optional</td>
<td>Required</td>
</tr>
</tbody>
</table>
These mobile client types support different classes of on-device mobile applications. The typical use cases for these types of mobile applications are as follows:

<table>
<thead>
<tr>
<th>Mobile Application Types</th>
<th>Mobile Client Application with Local Database</th>
<th>Connected Mobile Client Application</th>
<th>Laptop Client Application with Local Database</th>
</tr>
</thead>
<tbody>
<tr>
<td>Applications deployed to specialized, often ruggedized industrial mobile</td>
<td>Field Service, Transportation, Consumer Goods</td>
<td>Inventory Management, Warehouse Management</td>
<td>Public Safety (Police, Fire, and Medical)</td>
</tr>
<tr>
<td>Mobile workforce applications</td>
<td>Pharmaceutical Sales, Insurance</td>
<td>Sales Force Automation</td>
<td>Pharmaceutical Sales, Insurance</td>
</tr>
<tr>
<td>Extension of Enterprise Application Functionality</td>
<td>Expense Reporting</td>
<td>Business Intelligence, Business Approval (e.g. Expenses), Human Resources, Procurement, Project Management, etc.</td>
<td></td>
</tr>
</tbody>
</table>

As smart-phones and tablets become more powerful and functional, more and more users will switch from laptops to smart-phones or tablets.

**Specialized Mobile UIs**

To support the different types of specialized mobile UIs, Oracle’s recommendations are as follows:

- **Games** Oracle does not target the development of games on mobile devices with ADF. In general, games are developed in device-native language to achieve maximum performance and usability.

- **Widgets** Widgets are by nature limited in functionality, and are intended to display limited but focused set of information to end users. Widgets are very popular in consumer application space, but have gained limited acceptance due to its limited scope.

- **Web Implementation** ADF Mobile Browser provides a robust platform to deliver rich functionality and mobile-optimized UI to web-based applications. Furthermore, a web client shell would deliver presence on the mobile devices, as well as integration with device services such as PIM applications and GPS. This client shell provides interface between the web and the native services, as well as provide a user-visible entity in the
application store and on the device itself. This hybrid client that combines web client shell with ADF Mobile Browser deliver the underlying framework to enable Web Implementation type of mobile applications.

- **Data Collection Application** One key requirement for a data collection application is the speed of each data collection action, which is consisted of a trigger event (scanner is triggered), a data entry event (barcode is read), data storage (data is saved to the database), and visual acknowledgement (device shows some data about the item that was scanned). For casual collection of data – for example, a user goes on a site and only needs to scan a few items occasionally and in a location with limited network connectivity; ADF Mobile Browser provides adequate support to enable such an application. However, if the requirement calls for rapid and repetitive collection of data, ADF Mobile Client provides an on-device client and local database to support rapid capture of data.

The classic use case for data collection application is counting inventory in warehouse, retail store, grocery stores, etc.
Security and Application Provisioning

Another key requirement for enterprise mobile application is integration between the security infrastructure works seamlessly with the device/application provision infrastructure. Oracle Identity Management provides a market leading, complete suite of security products that covers virtually every aspect of securing enterprise data, infrastructure, and application. Oracle ADF Mobile is completely integrated with Oracle’s market-leading security products, ensuring a common and consistent security infrastructure for both enterprise and mobile applications.

For application and device provisioning, ADF Mobile adopts the strategy that ensures the mobile application fits in seamlessly with existing device provisioning infrastructure. ADF Mobile works seamlessly with device-native device management solutions such as BlackBerry Enterprise Server by producing deployment artifacts that are natively supported by these device management solutions. ADF Mobile can also work with other, third party management solutions that might already exist in enterprises, ensuring that Oracle customers can leverage existing infrastructure to manage ADF Mobile applications. Lastly, on platforms where there is no native application provisioning support, Oracle also provides key management services such as Application Provisioning and User Management through Oracle Mobile Server.

Oracle Mobile Server and its User Management functionality is completely integrated with Oracle Identity Management and therefore ensuring consistent security across both mobile application and traditional enterprise applications.
Conclusion

Oracle’s Mobile Computing Strategy is formulated based on what the mobile user demands, what mobile application developer needs, and where mobile technology innovations are heading. Mobile enterprise application users demand that the mobile application be natural extensions of the enterprise application, while supporting highly-native user experiences. Enterprise mobile applications also need to support browser-based or on-device applications, depending on target mobile devices, levels of device services integration, and offline data access requirements. Application developers need framework and tools that lower the barrier to entry and cost for developing mobile application. Constant innovations in the mobile technologies are driving greater diversity in device platform that end user leverages, and greater application functionality that mobile user expects.

Oracle’s strategy is to fulfill all of these requirements with a common framework and toolset, thus helping Oracle customers meeting challenges in supporting the ever-increasing mobile workforce and ever-evolving mobile technologies.