MOBILITY REFERENCE ARCHITECTURE

A Blueprint for building Mobile Solutions

Auteur: Dennis Reumer (Arché IT bv)
August 2013
PREFACE

Though this document currently bares my name as author, it cannot take the credit for its content as a whole. This document is a testament of 10 years of (smartphone) mobility and the many discussions with Users, Business Units, Business Decision makers, CIO’s, Enterprise Architects, IT Architects, Product Vendors, Solution Providers, Gartner, Forrester, Compliancy, Auditors, Security Offices, Developers, etc. and etc.

As such I would like to thank all of these people who took the time to listen to my vision, ideas and proposed solutions over the years and have provided me with invaluable feedback and insight into their business and needs and what defines success for them.

Seeing that raw potential, I’ve kept track and have been involved in the mobility space for over 10 years and have always been looking to see how we can achieve more through mobility and find the right solutions and tools to enable myself and the users/companies I worked with to do more than we ever thought possible, changing our lives and businesses.

Dennis Reumer

This document has been realized in cooperation with Enterprise App Store and partners like Blackberry, Capgemini, IBM and Veliq.

In the ever growing world of mobility, organisations need to be able to offer their customers, employees and partners a centralized, consistent, transparent and accessible set of apps, based on a logical back-end. Enterprise apps are apps with a back-end. To be able to develop this efficient and cost-effective, they need to be based on a comprehensive architecture. Questions about this subject are perhaps the most persistent that we see. In our view, the only way of providing such a corporate app store for organisations as single point of entry for mobile apps, is to base it upon a single mobility architecture.

Enterprise App Store strives to provide knowledge, expertise and experience on mobility solutions to help grow the availability and distribution of solutions and has endorsed the quest of Dennis Reumer to put information around mobility architectures into a document. Enterprise App Store has asked and received feedback from the partner network in growing this document into a reference mobility architecture to provide an optimal starting point for organisations to develop their own mobility architecture. As this is an area that is ever evolving, we appreciate your feedback and discussion to further develop the subject in follow-up versions of this document.

Enterprise App Store, Mark Beermann
# PREFACE

# INTRODUCTION

# GOAL

# SCOPE

Approach

Finding the Answers

How to unlock the information?

Data Transport

Data Access Control

Functionality

Devices

Development Environments

Enterprise Infrastructure

Component overview

Reference Architecture Overview

Mobile Solution End-Point Management

Mobile Solution Application Management

End-Point

Connectivity

Enterprise

Malicious Code Protection
INTRODUCTION

Mobility is having a larger impact everyday on our lives and our businesses. It’s reshaping the way we communicate, work and play. From a consumer’s perspective, their mobile is now the first point of contact for anything they do. It’s also changing the forefront of business in a rapid pace (banking, payments, service, etc.) Slowly it’s starting to change thinking on internal business processes. To get started with mobility and change these processes, the right tools and knowledge is needed on how to use these tools to be able to effectively start mobilizing.

Companies that don’t have a vision and strategy for mobility will start loosing business at an accelerated rate in the near future to their competitors that do. Mobility is having the same effect to companies, as Internet web shops are having on retail shops.

Mobile will be the first and most important communication interaction device.

GOAL

This document you are reading now is the second iteration of a Mobile Reference Architecture, that is aimed to provide you with the proper insights, knowledge and tools to choice the right Services and Solutions for your business to successfully implement your mobile vision and strategy. It is aimed at CIO and Enterprise/IT Architects and additionally can serve as background information when defining a vision and strategy for Mobility.

As such the document focuses on mobilization, enabling your company to start unlocking your existing data, information and business processes to mobiles and follows the “Mobile First” paradigm.

By developing all your new services, solutions and products first for mobile, many companies also have solved many problems that apply to any other platform (Desktop, Laptop, Machines) and have allowed them to become more agile, lean and responsive to changes. Due to the restriction mobile puts on you, you’re forced to come up with new models, which in turn are also very effective for your business as a whole, lowering dependencies on other solutions, systems, platforms and information.

SCOPE

The Mobility Reference Architecture is not separate from the companies Enterprise and IT Architecture and should be integrated into existing Enterprise and IT Architectures. This Mobility reference Architecture is agnostic towards specific concepts like Bring Your Own Device. When using the Mobility Reference Architecture you should apply it to be able to use it for your customers, business partners and employees and not only to solve the need for BYOD or developing customer applications, as most of the components and solutions should be reused to solve the same problems in the same way, so you maximize reuse and lower your cost and maximize your return on investment.
APPROACH

When looking at mobility and how to be able to start using it we need to be able to ask the right questions and provide the (right) answers for them.

The central 3 questions are for mobilization:

1) How can we unlock our information and processes to be used on the mobiles?
2) How can we ensure our information and processes are secure on the mobiles?
3) How can we maintain manageability of the solution, ensuring we can actual offer the actual solution to our users and also be cost effective?

In addition to the 2 questions above, these need to be asked keeping in mind the context of whom you’re actually servicing with the information and processes, as that can have an effect on the decision you make to answer the 2 questions for that particular Persona.

So from a business perspective and for your strategy continue to ask the normal questions:

1) Who?
2) Needs What?
3) Where?
4) When?
5) And How?

And create the right persona to define these and what their needs are. Only then you can start properly answering the 2 central questions.
Finding the Answers

To be able to find the answers for the 3 answers for your company, it's good to know how what you actually need to consider before being able to properly answer these questions. A question must always be asked within the right context.

Appendix A provides an overview of the challenges that define more of the context.

HOW TO UNLOCK THE INFORMATION?

Or better said how can we use our Enterprise Information on mobile.

The fastest way is to take down your firewalls and thereby give access to all mobile devices to come and get it, but even than the information might not be available in a format that the mobile devices recognize or can use.

Additionally the information might come in the right snack size. With the wireless network and cpu/memory constraints of mobile device, they might easily choke on the offered information, even if they can work with it.

And there was a good reason why the Firewall was up anyway; to protect the systems from unauthorized access.

So to unlock we need to look at how we can establish connectivity beyond the firewall and provide the information in an appropriate way to the mobile devices.

The following options give an overview of making things more accessible for mobile, not yet looking at establishing the connectivity.

Using a pragmatic approach you can us the following options to evolve your Mobile Experiences for your enterprise, in order of usability of your ‘apps’

Leveraging the good browser capabilities of mobile devices:

1) Use existing (desktop) web sites
2) Use a VDI solution (Citrix, VMWare, -> (Windows) desktop applications on the mobile)
3) Adapt existing (desktop) web sites to for mobile
4) Create mobile HTML5 web sites (for both existing web sites, but also for existing desktop applications)
5) Create HTML5 Web Applications (on-line or on-device) / Web API’s (Adobe PhoneGap/Cordova)
6) Create Hybrid Applications (Mainly HTML5 Web + Plugins to Native code) (Also Adobe PhoneGap/Cordova)
7) Use a cross Platform development environment (Appcelerator, Qt Creator, Adobe Air, NME,etc.) to reach all platforms
8) Develop Native Applications (XCode, Momentics, Android SDK, Visual Studio)

Each of these options has their own pros and cons.

For option 1 till 4 no adaptions or additional middle-ware is required and existing resources, knowledge and assets can be reused. Drawback for all is that the user needs a pretty decent connection at all times to be able to use them. No connection means no functionality.

Number 1 provides the least user-friendly experience; most websites will be shown too small to be properly used and zooming out and in will not make it an efficient tool for the mobile user, but might work for tablet users, but even then the touch interface can’t replace the mouse. Additionally
internal desktop website tend to use large media resources (pictures, video, sound, etc.) and can load slowly when the mobile device has no or low coverage. It is by far the most cost effective solution though and it should always be included as a starting point for your mobility. Ensure a mobile web portal exists to facilitate easy access to the various intranet sites of your company.

Number 2 is in line with number 1 and isn’t very usable on a phone, if it can’t be used on a bigger screen with a keyboard and mouse. There are some mobiles, that do support this, but the majority will not work effectively. If there is already a VDI solution in place for desktop computers, this is also a very low cost solution to provide direct access to all your existing (desktop) applications. Mind though that connectivity can be an issue and slightly defeats the mobile paradigm, as to properly use it, it requires a desktop size screen, mouse and keyboard. A phone screen will not be sufficient if that phone can’t connect to a big screen. Tablet can work, but if that hasn’t got mouse support and only touch it’s not very user friendly.

Number 3 improves the experience of number 1, but the form based navigation and confluence of most websites will not provide an efficient user experience overall, but does increase the loading times of the pages, by providing an more optimized experience for the media (smaller pictures, video’s). It is a relative low investment to be made and a huge improvement over option 1.

Number 4 is a strategy to provide a more Application like experience, through any browser. Focused on functionality and less on content. Much of the existing websites architecture can be reused and made smarter. It does require a relative larger investment. It’s cheaper as option 5 and some of the same results might be reached, but it will not provide the flexibility that options 5-8 provide for any future (business) development.

For option 5-8, a change in thinking is required, by placing mobile first. In this case we expect most of the Application assets to be sitting on the mobile device ad the Application on the device communicates with the middleware directly through a RESTfull Web API.

The actual transaction logic of your existing applications will not be handled on the device or through a website, but by making function calls to a hosted solution that can process these calls properly. This creates small queries, which are easily consumed by Mobile and Desktop alike.

Function can be getClientList, createNewClient, addDocument, addContactDetails, etc...

The Mobile application only guides the user through the process and collects and displays the right information, but the actual changes are done in the middleware/backend through the Web API. The Web API can be reused by any type of application, so not only for mobile applications, but also for desktop applications and interactions between systems. It uses the principles of Service-Oriented Architecture and additionally allows these services to be hosted on on-premises cloud or external cloud services.

To ease the creating of these WEB API’s there are a number of Mobile Enterprise Application Platforms that allow these to be created against normal backend and middleware platforms. (e.g. IBM WorkLight, Verivo) These can also include actual development environments, which create complete applications. [http://en.wikipedia.org/wiki/Mobile_enterprise_application_platform]

Option 5 allows a lot of reuse of knowledge and assets to build a stand alone HTML5 application. An HTML5 application uses web technologies to build a standalone application. The difference with a web site is, that a HTML5 application in many cases exists of only one page, which is dynamically changed to show different content. Its therefore self contained and doesn’t need a server to run, just a browser. It’s a cost effective method to create a stand-alone application that can be deployed on many different mobile devices from different vendors and OS’es.
Warning though, even though there is much reuse, the variations of HTML5 and CSS3 support between the different platforms, can cause a considerable overhead when developing, especially to adapt the UI and user experience. Very effective and user friendly application can be created this way, but I will never reach the level of performance and integration as Option 7/8 and will always lack in user experience and effectiveness compared to option 7/8, also as the UI components differ from the Native platform.

Option 6, leverages everything from option 5, but tries to improve the experience, by integrating plugins that use native code, either to provide higher performance for some tasks or to integrate closer with the relevant device. Also in this case a word of caution when applying for cross platform development, the UI and User experience difference between the various platforms and the support for HTML5 and CSS3 variations, can increase the development effort considerably.

Option 7 and 8 apart from providing the ‘best’ user experience, also have benefit of providing the best integrated development tools, making it much easier to integrate with the devices hardware and services (GPS, Camera, Accelerometer, Geo-fencing, etc.) The current state of Web development tools and various open source libraries can easily create a lot of extra dependencies for your applications, which you have no control over (as managed by others). The Development tools provided for option 7 and 8 provide stable, managed and supported environments.

Option 7 also allows you to target multiple devices and OS from a single code base. This time a specific Development Platform takes care of the heavy lifting. Most of these platforms are separate from the HTML5 or Native Platforms and have their own programming language and API’s, which means if you choose to use these tools, you can’t easily port these applications to either HTML5 or Native and you’re depending on their tools and in some cases run-time (Adobe Air). As most of these cross-platform development environments create actual native code, they offer high performance and also provide the look and feel of the relevant platform in many cases.

Option 8 Provides the best performance and user Experience, but can be the most costly one (depending on possible reuse of the Web API’s and core code). In this case you need to create an actual mobile application for each individual platform. Though this seems a lot of work, if you have properly setup your Web API’s, the application is mainly a UI and little to no functionality should reside in it. Additionally you can look to consolidate any functionality in the App into a core module consisting of device agnostic c/c++ code, which can be used compiled by most all Mobile platforms (iOS, Android, BlackBerry, Windows Phone).

DATA TRANSPORT

Network Constraints - IPv4

The current mobility is only possible because modern cellphones/tablets/computers are part of the Internet. To be precise the first Internet based upon the Internet Protocol IPv4. This means every mobile device has an IP address based upon the IPv4 protocol. Due to its limitation IPv4 in address space it had been split up in different ranges. Internet accessible ranges and ranges that can only be used internally (10.x.x.x, 192.168.x.x )

This already gives one of the biggest constraints for mobility. With VPN on desktop/laptops systems the internal network can be extended to outside of the physical perimeter of the Company. But
VPN’s are not friendly to the mobility devices batteries and users and not generally available on all mobile devices.

There is a bridge to be crossed from the internal network to the external mobile devices.

Additionally with this setup it will be hard to reach the mobile devices, as we don’t know their automated assigned IP addresses, that can be also from internal addresses of the Carriers and changing between dips in the network.

Currently with mobility there is no real state full connectivity between your internal systems and the external mobility devices.

We will need means to create a bridge and ensure we can cross it and there are a plethora of solutions in the market trying to bridge that gap in very different ways. (more on this later)

IPv6 can solve the issues, but it’s not expected IPv6 will be generally available and the required services on it available in at least 5 years. With IPv6, you can get all your devices on your internal network and keep always know where to reach the device by it’s assigned IPv6 IP address.

Mobile networks

Whether the mobile devices connect through Wi-Fi or through a carrier network, there will be times that the device has no connectivity. The mobile devices might be always on (if your battery is not depleted), they are simply not always connected.

If you rely on the fact these devices always have a network connection and are able to reach you company data through the network, you’ll be in for a few moments of hurt when trying to close that deal, just as a cell tower get’s the fritz.
So after we have fixed the gap in the network, we also must ensure we understand we will not always be connected to the network. Any data you need to close those deals, must to some extend be locally cached on the device, so that on the moment you need it it’s guaranteed there.

It also becomes important to work transaction based, as the connection might drop mid transaction and you might be left with corrupt data if not taking this in consideration.

This should also not just go one way or infrequent, as your data will go stale and out of sync. Selective Syncing of data is essential in mobility if you want to ensure you have the data you need to continue your business processes if the network is acting up.

Current Mobile networks are also due to their nature and paths that the data needs to travel, very much different of the PC/Laptop/company networks. There is less bandwidth available and the latency is much higher and actual delivery are not guaranteed.

This has a large impact on how and how much data we can transfer. A last complication is that the protocols we’re used to on our PC’s/Laptops/Company networks to communicate between the Applications and systems are almost all not available or will require adjustments.

**Eaves Dropping**
As we’re transporting over public networks, there are various ways there might be parties eaves dropping on our data transmissions. Whether it’s through a Man-in-the-middle attack or just Open Wi-Fi networks.

Forward looking: LTE and IPv6 can solve most of these issues, but maturity for these technologies is not expected in the coming 5 years. With IPv6 and LTE, devices could actually be part of you company network and not leave the companies network perimeters even though they are not in the same building, but on a wireless network.

**DATA ACCESS CONTROL**

**Authorization, Identity**
Data access control is at the heart of any IT solution, not build upon open/public data. A standard Data Access model uses the following:
Access to data is restricted by an access control mechanism. For a user to be able to access the data, he/she first needs to provide his/her credentials for authentication to establish the user's identity. Credentials can be in the form of username/password, token/pin, smartcard/encryption keys, etc. The credentials are validated against a central identity provider/repository (e.g. Active Directory, LDAP, etc.). Once the identity is established, it's validated against the Access Control Repository to see if that particular user is allowed to access the data. The identity and Access Control Repository can be the same, but in many cases could also be different. The process is always the same, but the technologies used can vary widely. From plan text username password, to full PKI and encrypted data, that is only unlocked using the user's personal encryption keys, depending on the nature of the data and the need to protect the data from unauthorized access.

Within the current Enterprise Architectures, this data access control has been embedded into the Windows systems (Active Directory) and using standard network protocols (Kerberos, LDAP). Authorization for access is regulated on the data is stored in the backend systems or Middle Ware systems.

Unfortunately none of the protocols/methods used within the Enterprise Perimeter are available by default on the Mobile platforms. Also the need to store data locally for a successful mobile solution (see later on) introduces challenges that need to be addressed in your mobile solution. Any Mobile solution will need to address Data Access Control, to ensure unauthorized access is prevented.

**Prevention and minimizing impact**

First line of defence is the Data Access Control, but any solution has weaknesses that are exploited. (Yes are, not could be, not knowing it only obscures the fact) To prevent exploitation of those weaknesses and contain the possible damages if being exploited any mature platform uses protection against unauthorized and malicious behaviour of code. To be able to properly work, the Malware detection software needs to have proper access on code level execution. In itself a tricky feat as by default Malware protection has access rights that
compromises the system integrity completely, to be able to do his work properly. This is the trust we need to give this type of software and which most of the current mobile platforms are unwilling to give. Not having it though is still a lack on the platform and a risk to your solution.

**FUNCTIONALITY**

Data is at the heart of your company, but without functionality around it, it’s just static data without context and purpose. To be able to work properly with data and support business processes, data, functions and workflow are bundled into Applications. Applications have a set context and functionality and ensure we can work with the data and the supported business processes. Depending on the supported business processes the applications can be small and focused or large and spread over various functionalities. With mobility we’ve seen context and functionality be narrowed to provide a very clear context and scope what the applications provide.

For many an App has become synonym for an icon on a mobile device. But please remember that an App is an Application. No matter how the application is delivered or packaged it’s defined by it’s context, scope and functionality, not the delivery or package; this includes web sites and e.g vdi solutions [Remote desktop on phones/tablets]

Keep this in mind when thinking about your mobile strategy; you have options to deliver your business functionality.

**DEVICES**

The actual use of the functionality will take place on the devices. The platform that the devices run and the solutions available for the various device platforms do impart a large impact on the solutions that might be available for your Mobile solution. Additionally the form factor also determines what can work. At the current time we’re seeing 3 various form factors that also come with their own User Interface paradigms:

1) Phone 2.5”-5”
2) 6.5-8” Tablet
3) ~10”+ Tablet

There are still a lot of different form factors popping up, so it’s certainly not a definitive list. Depending on the preference of the users, a real keyboard can be used with any of these devices. In general larger devices tend to have a longer battery life. The battery life the Phone Format is challenging at times.

Devices come with their own platforms, OS, development options attached and can provide their own limiting factors on what solutions you can use to realise your mobile solution. Ensure you inform yourself well before choosing a specific platform or want to use them all, to ensure you know what possibilities are left after choosing your device platform(s).

**DEVELOPMENT ENVIRONMENTS**

An important part of the mobile solution is the development environment. Aside from selecting your infrastructure solutions, this will be one of the most cost associated decisions you’ll be making.
There are multiple parts to this. There is the Device Side and the Enterprise Side. In most cases the Device Side will mainly provide the User Interface and the Enterprise Side the actual processes.

**Device Side**

By choosing a Device Platform (or choosing multiple) you also but into the development options the platforms offer. As most mobile OS’es are propriety so are their Development Environments.

A Development Environment consists of:

1. A runtime language (Java, c/c++, objective-C, JavaScript, Qt, Air, Flash, C#, .NET)
2. Integrated Development Environment (IDE -> Xcode, Visual Studio, Eclipse, etc.),
3. A set of API’s (Application Programmable Interfaces, that expose the platform options, e.g. InApp Purchasing)
4. A UI framework (proprietary in most platforms, Modern UI, Cascades, etc.)

Apart from the proprietary device solutions, there are a few more device agnostic solutions that are available for most platforms (still proprietary in many cases):

1. HTML(5), CSS, JavaScript (leveraging web standards to deliver the application functionality) (Also good reuse on the desktop)
2. Qt (an open-source platform build on c/c++, can target most mobile platforms currently, also used on the Desktop).
3. Third-Party Platforms:
   a. Adobe Air (Also available on the Desktop)
   b. Appcelerator Titanium (Only for Mobile, compiles to native UI and code).
   c. Service2Media

If you choice a remote desktop solution, you avoid the device side development environment, but then are locked into the Remote Desktop solution platform.

**Enterprise Side**

In mobility the processes and information are usually exposed through Web based API’s (SOAP, JSON) on the Enterprise side. At this time any platform that can build Web Based API’s can be used for a mobile solution and although there are many middleware platforms (MEAP = Mobile Enterprise Application Platform) tailored to mobility to easy the exposure of Web API’s there is no real standardization yet in this eco system.

There are many aspects to consider for the infrastructure on the Enterprise Side, but the development platform isn’t a restricting factor in the Enterprise at this point in time.

The middleware platforms will off course come with their own proprietary development environments.

Small example of middleware platforms:

1. IBM Worklight
2. Verivo
3. Service2Media

**ENTERPRISE INFRASTRUCTURE**

The third costly decision you have to make relate to the Enterprise Infrastructure.

When moving from normal desktop Enterprise applications to mobility, the usage of the applications will be very different from the desktop application (more spread through the day, and a higher request rate to data). Additionally when externalizing you processes to customers, there might be millions of customers instead of a 100 employees wanting to access the application and
data. Additionally, customers expect your applications to be available 24/7 with no downtime for maintenance (they might be traveling worldwide).
This requires reengineering your infrastructure to handle the larger load of traffic and also maybe off-loading/rescaling your current middleware. When starting with Mobility, it might be good to revisit your current Enterprise Infrastructure and Platform choices and revalidate them and reengineer your backbone, middleware, and edge infrastructure.

Mobility requires:
1) High throughput routed front-end infrastructure, directing request to the right internal sources.
2) Able to handled millions/billions of incoming connections.
3) Conserve bandwidth utilization to the mobile devices (even on 4G LTE). Lightweight Web Services (JSON).
4) High Availability.
5) Robustness, modular built up, no domino/cascade effects of failed transactions/mid transactions.
6) Fast return and response times on requests.

In combination with the Data Access constraints and requirements, the infrastructure must also provide the proper Authentication/encryption methods to safeguard the data while it moves back-end <-> Middleware <-> Front-End/Edge <-> Device. Authenticated Web Services, PKI infrastructure.

Component overview
Reference Architecture Overview

Although many of the aspects have been addressed before, each and every component will be addressed again in this chapter and explained what is expected of it.

**MOBILE SOLUTION END-POINT MANAGEMENT**

Even though your information might be secured within the application and you’re properly handling connectivity against eaves dropping, there still will be need to know which end-point are actually using your applications and in many cases, you want to protect yourself against possible attacks to your information on the device itself. This is where End-Point management comes in to play. Its first requirement is to know which endpoints are actually connecting to your infrastructure, for auditing purposes. The second task is to ensure they meet your set compliancy levels for your applications to run on that end-point. (for instance Jail-Break and rooting detection, Password compliancy).
It also provides a basis for any other administrations that need to work with the end-point assets (like Application Management)

**MOBILE SOLUTION APPLICATION MANAGEMENT**

After you have successfully create an infrastructure solution and are able to manage the end-point, you will need a way to manage Applications. Mobile Application Management consist of various tasks that ensure users will be able to properly access and use the company offered applications. Tasks consist of:
- Make applications available to users
- Manage licenses/costs
- (Automatic) application deployment to users
- (Automatically) provide updates to applications
- Disable/remove applications when no longer required or allowed for users
- Administration based on groups that tie into existing User/group administration (thing Active Directory/LDAP)
- Inspect integrity of Apps (Malware detection on apps in the Store)
- Promote Applications to users

To the company and users, this should manifest itself as an Enterprise App Store.

**END-POINT**

**Device**

In this case the Device is the actual hardware, not considering the OS and file system or anything else software related, with the exception of the embedded boot-rom.

To be able to be a end-point in a mobile solution, the device should provide multiple connection options (Wi-Fi, Cellular) (secondary: Bluetooth, NFC) and properly shield its signals from listening parties (not caves dropping the electronic signals) and finally the boot-rom should provide proper protection against potential hackers, the memory in the device (for Storage and processing) should not be readable when removed from the device and not allow unauthorized tempering with the boot-rom to allow access on a logical level to the device.

**Device OS**

To be able to do something with Hardware an Operating system is needed. This is at the core of your Mobile security and possibilities.

The OS is responsible for handling the hardware, memory, file system and the Application Programming Interface (API).

Any endpoint device is only as secure as it’s OS. Any weaknesses in the OS, compromise the whole of the device and all data contained on it.

A good OS completely segregates all processes running on a device.
This means all applications (even system application, like networking, sound, display) have no access to the code, memory, CPU/GPU, files of any other process that doesn’t belong to them. The only way to exchange information between the various processes (Apps) should be authorized programmatic interfaces and shared storage.

A good OS should safeguard all of the storage on a device against unauthorized access attempts. There are off course various methods, including encryption to secure this. All Operating systems have a primary / kernel / root identity under which most of the core processes run. Access to the root account should not be available.

Actual Applications should all run under a user specific account, which can be protected with a passcode/password. Logical access to the shared data should only be possible using the specific user account. (e.g. via USB cable/Wi-Fi/Cloud)

The OS should provide bulkheads/permissions for access to security compromising features, such as Camera, Microphone, Location Services, Hooks into messaging system (SMS, email, etc.), contacts, agenda and any other privacy and personal information.

Both the user and/or IT management system should be able to determine which permissions are granted to the applications.

The OS should provide hooks and permissions for malicious code scanning Applications to ensure no applications abuse their permissions try to elevate their permissions on the device.

From a Mobile solution perspective any work related management system should be able to set the permissions for work related applications (that store and use company data) and ensure the access to any work related data/applications is protected properly and the malicious code detection/prevention should be able to report on the state of the work data and applications.

Device / App Data

Already mentioned in Device OS, both the device, user and application data need to be shielded from unauthorized access/tempering.

By default the Device OS should be safeguarding any of this data against tempering.

If the device OS provides not enough means to safeguard the data stored on the device, other means must been found to safeguard the data on the device. (Either by not using any permanent storage or encrypting anything stored). As unprotected storage presents a chicken and the egg problem for storing encryption keys and unlocking passwords, take good precautions before storing this type of data or actually don’t store this information on the device and only work with ‘cloud’ based Authentication.

There are various systems that provide different means to protect the Device/App Data.

In some cases, the applications are run within a ‘container’ that is encrypted, either automatically or programmatically through an additional API provided by a solution provider.

Developer Env /API’s
The Developer Environment and API’s from a device perspective are the exposed programmatic interfaces exposed by the Operating System, which allows developers to actually create an application.

It allows the programmers to display data on the screen, interface with the device instrumentation (Network, Camera, microphone, Gyroscope, GPS, etc.) and potential interaction with other applications and data available on the platform.

Although the OS should be handling security itself, the Developer Environment/ API’s could provide either from the OS extra help in securing data and access or can be helped by third-party API/SDK’s that solve the potential security issues for the developers.

Most platforms allow Third-party SDK’s and API’s to perform additional functionality. Never off course beyond the capabilities of the OS platform and or it’s security restrictions.

By default the development environment and API’s should allow safe communications between the device and information systems in the Enterprise.

Through the OS a Mobile management system should be able to set the permissions on certain API’s for an Application to ensure application behave in the predefined behavior.

**Application**

The Application is the bundle of code that makes use of the OS and the API’s to present a User Interface.

As such most of the Device side controls are already handled in the OS/API and Device Data/App Data sections.

A potential malicious code scanner or a Mobile Solution Management system could scan the App packages on intent and validate it doesn’t exceed its expectations or performs unwanted behavior.

**CONNECTIVITY**

Connectivity is broken with Mobility, as the device as not part of the company network (and shouldn’t be). There is no way to be able to reach the device from the company network. Only when the device makes contact to the company network, you might be able to have a conversation with them.

To create proper connectivity again, where safe and ‘bi’-directional communication takes place, additional infrastructure will be required.

There are many ways to make this work again.

An expensive way is setting up VPN’s, which makes the devices actually part of the internal network again. This might not always be desirable, as with the limited control of the devices in most cases, this might introduce unsecure devices on the internal network.

Other ways to build connectivity is through the Front-end/Routing infrastructure, that device can connect to. In this case you also need to ensure the connectivity is secured (e.g. through SLL, TLS), but also ensure the setup of your secure connections is handled securely as Open Wi-Fi Access Points, allow easy eaves dropping and monitor any traffic from mobile device to your infrastructure.
A good Mobile solution address connectivity and ensures the device and company can safely communicate with each other.

**Access Control**
Access control should be a single integrated and integral part of any mobile solution. Access Control regulates, what kind of access users have to your data and processes and should be very carefully implemented within a mobile solution. As mobile devices have no direct access to your network by default and not integrated by default with your standard Access Control methods within your Enterprise it by default doesn’t take part of your standard processes for Access Control and is therefore broken by default on Mobile.

In many cases this leads to delegate User credentials, Identity, Authentication and Authorization to take place on the actual device. This is a very risky step to take, as in many cases you don’t have any control over the device itself to ensure the integrity of these processes on the device.

When implementing Access Control for mobile, try to leverage the same processes that are already in place and keep these processes close to your core Access Control mechanism and internal processes.

**User**
The user holds the aspects to properly identify himself to the systems and data he wants access to.

In many cases today this consist of a user specific name/email address and password, but many other means of identity aspects can be used (from biometric to Private key, etc..)

Users are responsible for safeguarding his/her identification/credentials.

A user should not store his/her credentials in an unprotected place, whether this is electronically or on paper.

Additionally be wary when ‘caching’ the credential for the users in Applications. Saving credentials of a user in an application is a potential security risk, as the method how the credentials are stored might not be safe.

A serious weakness of passwords and Human users, is the more complex the passwords, the more users will try to use the one and the same password for multiple systems. This is a large risk, as losing the credentials to one service, means compromising the other services too. This in combination with social engineering and password recovery tools currently can pose a great risk to companies.

When creating a mobile solution be aware of the possibilities for the User Credentials and choose one that fits the risk profile of your company and information that you store.

**Identity**
Identity is created and maintained within the corporate Identity repository and is linked to specific users. After properly identifying themselves by providing their credentials, internal systems of the company assume this User is now the same as the electronic identity.

The identity repository must be giving the highest priority to be protected against potential fraudulent abuse of the repository as compromised of the repository compromises the whole chain of any solution.
Additionally the breach of one identity system can lead to a cascade of other breaches as many of the user credentials might be reused for other solutions. (Recent major breaches with LinkedIn, Twitter, PSN, Facebook, etc.)

Also beware of ‘caching’ Identity on the devices for off-line authentication. Identity should be an integral part of your solution, creating ‘cached’ identities, breaks this standard process and can give rise to false authentications.

**Authentication**

Authentication is the process of matching a user credential with the electronic identity. Because once a identity is established it is used to get access to the Authorized information and processes, it’s critical a proper secure Authentication mechanism is used to match the user credentials with the electronic identity. It must not allow for easy guessing or be open to possible eaves dropping during the process and capture the user credentials.

As with identity and user credentials, beware of local Authentication. Any local device performed authentication is not tight into the corporate Authentication mechanism and process and therefore is not congruent. It can be abused to gain unauthorized access to your internal systems.

**Authorization**

Authorization is the process where a digital identity is given access to information and processes based on his assigned role(s).

It’s a separate process, but should build on the cornerstones of User, Identity and Authentication. Again, performing local device based Authorization on access, is separating the process from the integral part of your internal systems and is therefore prone to abuse.

**ENTERPRISE**

Enterprise is all the parts of the Mobile solution that are hosted behind the firewall of the company.

**Front-end/Routing**

The Front End/Routing part of the solution consists of multiple components. Relevant to Mobility compared to normal Enterprise clients, is the frequency and number of requests that are made. These are typically much higher as with normal Enterprise Clients. The actual data traffic might be much lower, but normally mobility puts a pressure towards standard middle-ware systems like Web-Sphere, #Net Servers, SAP, Oracle, they are unable to handle properly.

It’s therefore recommended to look at implementing routing based front-end that can accept a very high volume of requests and route it to the proper middleware systems and allows better spread of the traffic and load to the various components. Also to guarantee proper response times to the
mobile end-clients. Response times of over >10 seconds are perceived as unacceptable by mobile users.

**Middleware**

Most of the Enterprise Middleware is currently either targeted at thick clients or server side Web pages. Both of which mobile devices are quite bad at (unless using VDI solutions)

To be able to properly implement a mobile solution, the middle-ware will require either a rebuild or additional components (Which might also be more at home in the front-end, though here I like to keep them as middle-ware as they can also be targeted to internal systems).

As a mobile device has limited network bandwidth and processing capabilities compared to a desktop., loading large Web pages or creating heavy clients will not work so well. This is why many companies are moving to ‘lightweight’ Web Services and concentrating to expose their data and processes through these Web Services.

So instead of creating a form based (web-site/thick client) application, the core of the functionality of the Application/Data/Processes is exposed through Web Services in the form of API’s. These API’s are light-weight Web Services in the sense they very much limited the amount of raw data that needs to be send over the network, but in the system processing the Web-services a lot is happening, Making database queries, collecting and colligating data and processing incoming requests that also bring data to the middle-ware through the Web-Services.

As the middleware servers don’t have to render the data anymore a lot of load is taken of the servers, but the number of requests and things to process do have increased, due to the large amount of mobile devices connecting to them.

Using the web services API model has many advantages above former Architectures, mainly also because it removes the UX/UI from the logic of the applications and allows for a large variety of end client UI’s to be build against the Web Services API and maximizes the reusability of the core of the business processes to all end-user-clients.

It’s also build on top of existing open standards and much open source, which makes consuming these Web Services relatively easy on all platforms.

As transforming existing middleware to this new model is hard, many opt for introducing specific MEAP (Mobile Enterprise Application Platform) platforms that create lightweight Web Services from Database connection.

**Identity Repository**

Stores the digital identity of users.

Ensure this data is inaccessible for unauthorized access. Preferably only exposing it through the Authentication process, never exposed directly on the internet.
**Access Control Repository**

Stores the Authorizations of users.
The same applies to this as for the Identity Repository.

**Backend Data**

Holds the Enterprise data and information in Database and other data structures.
Should only be accessible by the Middle-Ware/Front-end systems that expose the stored data through their Authenticated and Authorized Web Services API’s.
With the increasing number of request to backend data for Big Data processing and Mobile applications, many turn to flatter data storage and less Relational Databases, to be able to increase speed and also volume of data. (video/audio).

**MALICIOUS CODE PROTECTION**

Malicious Code protection should take place on all levels of mobility.
Even when taking all precautions, Software and people can be hacked and any flaws in software can be abused to leverage non-wanted code to be executed. Not having any malicious code protection tracking this behavior will leave you prone and vulnerable and at the mercy of the vendors that provide you with the components affected. (Whether this is mobile device or middle-ware platform.)