Mobilizing the Enterprise: *A Better Way*

Verivo

INTRODUCTION

The tremendous proliferation of smart mobile devices in the workforce has brought substantial gains in productivity, flexibility and employee satisfaction. Simply the ability to coordinate through telephony and messaging, and the exchange of information through e-mail, has improved the ability of those on the road to be more productive and satisfied with their jobs. But these gains are just a small taste of the potential gains this technology has to offer. Mobile devices are essentially the easiest-to-use computers, and employee keep these devices with them nearly at all times, allowing for a deeper, more comprehensive integration of the person into your business.

To realize the full potential of these devices in the hands of employees, partners and customers, businesses need to embrace the use of mobile apps. There are already many apps that allow for individual productivity gains, but the biggest impact comes from enterprise apps – apps that are integrated with the corporate systems that the business has already invested years of work in, containing the information needed to truly make these mobile participants effective. Today there are many off-the-shelf enterprise apps that can be used, but as businesses look to create the apps they need for their business, they are driven to create custom enterprise apps, much as they have created custom web-based apps and websites.

Building these apps is much harder than is sounds. Enterprise apps, with their integration, security and operational requirements, can be challenging to build, and even harder to support once they have been built. Add to that the fragmented world of different mobile devices and operating systems used by customers, and the "Bring Your Own Device" (or "BYOD") trend exacerbating that, and many businesses find themselves overwhelmed and unprepared to take on these challenges, making the benefits elusive.

This whitepaper examines the challenges associated with enterprise mobility, and presents a comprehensive approach to overcome them.

MOBILE APP DEVELOPMENT FOR THE ENTERPRISE: A WHOLE NEW BALLGAME

It is no surprise that enterprise mobile app development brings with it numerous challenges. Some are challenges that IT departments have been solving for years while others are new, specific to mobile, and evolving over time. For example, exposing secure, on-premise corporate data for consumption in the outside world is a challenge that lies deep within the DNA of enterprise mobile apps. And a popular bring-your-own-device ("BYOD") trend is driving companies to let employees access sensitive corporate data on a piece of hardware that is not owned, controlled or operated by the company. Although security and data access are topics that have long been associated with enterprise technology, the mobile phenomenon brings with it a unique spin on these challenges (and, of course, a new set of potential problems).

The key challenges associated with enterprise mobility fall into four top-level categories.

1. How Do You Keep Transactional Apps Fully Functional When They're Offline?

For the enterprise, the process of building and operating offline apps brings with it a number of challenges:

a. **Offline Implies Transactional** – Offline apps must do more than just provide the ability to read data while out of coverage. They must allow users to initiate transactional functionality, like updating inventory, placing new orders, and removing items from a task list, while still enforcing important business logic.

b. **Back-End Data Access** – Transactional apps that run offline, by nature, imply that access to back-end data sources will be required. Building this integration can be tedious, especially for developers whose expertise lies with mobile development frameworks and not the ins and outs of complex back-end systems like SAP and Siebel.

c. **Sync Infrastructure** – Offline apps require data synchronization infrastructure, which can be difficult to build. This includes mechanisms to retrieve data from a central data source, massage the results, optimize data for mobile delivery, store data on the device (and merge with existing content), handle data changes while offline, enforce business logic and handle conflict resolution. Needless to say, these steps can be resource-intensive and may also introduce risk that data could be lost if not processed correctly.

d. **Offline Security** – Enterprise apps that run offline carry with them an overwhelming security burden. Mobile developers will need to find ways to ensure data is safe, users see only the data to which they're entitled, and corporate policies remain enforced. The challenge is that even with some basic built-in security capabilities on each mobile OS, there are still gaps. For example, Apple's Data Protection APIs only encrypt data when a device is locked.

e. **Operations** – Offline apps may require specific types of controls in order to keep them running smoothly. For example, an incremental sync engine may be preferred to reduce the amount of data transferred over the air. Or, there may be a requirement only to allow a user to perform a sync when he's connected via Wi-Fi, to reduce the impact on the device's data plan. Detailed logging and audit trails may also be required to help with the troubleshooting process.

How Do You Keep Transactional Apps Fully Functional When They're Offline?

2. How Do You Ensure Your Apps, And Your Data, Are Secure?

It may go without saying that security plays a critical role in the success of mobile within the enterprise. Apps must meet rigorous security standards, they must comply with existing corporate policies, and they must be implemented in an environment that is safe from hackers and malicious attacks. Each of these requirements represents a slew of challenges for companies to overcome.

A few key security challenges for enterprise mobility include:

- Data must be protected, both at rest and in transit, despite differences in each mobile platform's ability to encrypt data
- The infrastructure must be safe from attack
- Different types of users will need different levels of access and permissions in each app
- Apps may need to authenticate against multiple back-end systems, each with different types of credentials required
- Corporate security policies will need to be enforced, and those policies may change over time
- Audit trails are required to ensure various levels of compliance and accountability
- Apps need to integrate with existing identity management systems

3. How Do You Manage and Control an App Once it's Built?

When companies first set out to build mobile apps a few years ago, they sometimes did so with a knee-jerk reaction. In that early phase of enterprise mobility, there was often a "wild west" mentality that played out as follows: The business unit identified a need for an app, whether for productivity gain, competitive advantage or purely for marketing sizzle. In the rush to get the app to market, Corporate IT tended not to be part of the solution and did not always have the opportunity to offer guidance on strategy and best practices. Instead, a developer associated with the line of business scrambled to build something that met the requirements as quickly as possible. The business need was urgent and the developer was encouraged to get the app out the door quickly.

This phenomenon, which is reminiscent of the early web server days of the late 1990s, is fraught with challenge and risk for IT. Here are just a few reasons:

- Mobile developers may be encouraged by stakeholders to prioritize functionality over scalability, security, operational support costs and standardization. This may result in a faster time-to-market but can introduce tremendous risk and overhead down the road.
- The Help Desk needs to be able to understand what is happening within the app and take specific action to fix problems (and not just tell the user to restart his iPad). Without this, confidence in the app is low and adoption is damaged.
- Each app developed will be a "one-off" that does not share infrastructure or resources with other apps. As a result, developers will be forced to reinvent the wheel for every app they build, spending time and effort on mundane but crucial areas like data sync, offline authentication, and data access. Each must be debugged separately, providing no leverage or reusability to the developer.
- Each app may be developed using a different tool or technology based on business needs. This is not necessarily a problem – in fact, many organizations have successfully adopted a combined native/HTML5/hybrid strategy based on the specific requirements of each app – however when it comes to the maintenance, management and troubleshooting of these apps, IT is left with a disparate array of technologies and products.

- As good as a mobile developer's intentions may be, best practices around security and data management can be difficult to implement well. Developers may inadvertently introduce security flaws, performance problems, and erroneous logic that can result in serious risk and data loss.
- When pressed for time, developers may be tempted to hardcode app-specific settings, environment variables and system-level connection parameters within the source code of an app. These types of shortcuts can have an adverse effect on deployments and environment migrations as well as troubleshooting and support, since all future changes will require recoding, recompilation and redistribution of the client app. It is certainly possible to separate an app's code from environment-specific details like server names and URLs, but those approaches typically require substantial time and foresight.
- Apps built without any common infrastructure make it incredibly difficult for the organization to support and enhance each app. This creates risk any time an app needs to be modified, and as a result IT Management may be reluctant to fix bugs, add new features, or support the app on new platforms. Business perceives this as a lack of responsiveness from IT, and adoption of the app may drop over time as a result.

How Do You Manage and Control an App Once it's Built?

4. How Do You Build and Support Apps that Run on a Variety of Device Types?

Somewhere around 2010, as mobile web technologies including HTML5, JavaScript and CSS3 demonstrated enough material improvement to offer a respectable alternative to native development, an enthusiastic debate began over whether the future of mobile apps would lie with native or mobile web. Jump ahead to today and there is little doubt that the mobile ecosystem is large enough to accommodate both approaches. In fact, many corporations have decided to make that decision on a per-app basis, depending on the needs and circumstances of a particular app. For example, an internal field services app with users that spend significant amounts of time traveling to remote locations or hospital basements will likely have a heavy offline requirement (and, based on the platforms available today, be best suited for a native app), whereas a consumer-facing product catalogue app will likely need to be accessible from as many device types as possible (and therefore be a good candidate for HTML5 delivery).

The good news is that with increased options for native and mobile web development has come a proliferation of tools that help companies build the UI components of their apps. As a result, developers can often get started quickly and can produce a great UI in a matter of days or hours.

The bad news is that companies who build their apps with varied development tools and technologies will find it difficult to support and maintain those apps over time. Additionally, each app will likely require key foundational functionality to be written separately (e.g., back-end data access, data sync, enforcement of user roles, data encryption, etc.), requiring a non-trivial amount of time to be allocated to building app "plumbing" in each project. Imagine the challenge of maintaining five mobile apps, each built with a different tool or framework, all handling security, connectivity to back-end data and logging differently. Without centralized management of the apps, IT Operations is faced with limited visibility for troubleshooting, upgrades and general control.

VISION FOR ENTERPRISE MOBILITY

Requirements

What if there were a way to build, secure and control mobile apps with the perfect balance of freedom and manageability? With the right enterprise mobile app platform, companies would have the flexibility to select the best possible development tools while still implementing the control and oversight needed to keep apps running smoothly and securely.

The benefits of such a platform would be dramatic across the entire IT organization, including:

1. **Developers** could build apps, using their mobile technologies of choice, on top of a common infrastructure. That means developers could focus on adding business value and building compelling UI instead of rewriting sync engines, figuring out how to cache credentials securely, handling entitlements or navigating the complexities around offline vs. online data handling.

2. **IT Security** could rest assured knowing that apps built on the platform automatically inherit company-provisioned security policies and will follow corporate guidelines on audit trails, data wipe, identity management and data encryption.

3. **IT Operations**, tasked with operating and deploying mobile apps, could feel confident that go-live events will run smoothly thanks to a clean separation of server code and environment-specific settings. They would gain the ability to troubleshoot problems in real-time, across a variety of device typesThis type of platform could be used, not just to build new apps, but also to reign in existing "rogue" apps that were previously built as one-offs by individual business units. This means companies would have a way to take control of their mobile initiatives, both for future development and for legacy apps.

This type of platform could be used not just to build new apps, but also to rein in existing "rogue" apps that were previously built as one-offs by individual business units. This means companies would have a way to take control of their mobile initiatives, both for future development and for legacy apps.

The Capabilities

To accomplish this, the platform would need to exhibit many of the following characteristics:

- **Provide Client Libraries and a Centralized Server to Facilitate Cross-Platform Development** – As the mobile landscape continues to mature, so will companies' in-house development capabilities. Although a monolithic, code-free MEAP approach may have been perfect for the market in years past, today more and more companies are capable of building their own mobile apps with sophisticated UI using native, web and hybrid technologies. A platform that offers a company the freedom to choose its own mobile tools and IDEs, but also provides central infrastructure or "plumbing" for common functionality, will be key.

- **Open and Highly Extensible** – Companies will always need to implement custom business logic and will often want to add in their own components, extensions and configurations. A good platform must provide a mechanism for this or risks being a closed (a.k.a. unusable) solution.

- **Scalability and Light IT Footprint** – A mobile platform must support a variety of configurations and deployment models, from smaller-scale deployments to global, high-availability models. Server-side components must be compatible with companies' existing infrastructure including virtualization, failover, load-balanced web farms and high availability scenarios. Ultimately, the platform's IT footprint must "play well" with others.

- **The Developer Community** – Proprietary platforms with limited documentation send a mixed message. They may advertise a do-it-yourself mantra but actually require heavy professional services in order to get you started with the tool. A true platform is open to developers to try and is backed by strong documentation, API references, a free trial and an open developer community.

- **Future-Proof** – A good mobile strategy is intended to cover today's needs as well as the challenges of tomorrow. A good platform should mirror that behavior and should be able to adapt as a company's needs change. For example, a company's data may reside 100% on premise today but might migrate to the cloud over the next 5 years. Or, a company may be heavily invested in native development today with plans to migrate to HTML5 over the next 2 years as that technology matures. The right mobile app platform will accommodate these future scenarios as well as the ones from today.

- **Based on Standard, Open Technologies** – The value of standards-based components cannot be overstated. Proprietary tools require training and highly-specialized skillsets. A platform should allow a company to leverage well-known, industry-standard technologies and tools. If not, the costs to operate and update resulting apps will be astronomical.

- **Help You Prepare for Change** – By now, most companies recognize that the first version of their apps will not be the last one. Mobile solutions that are cobbled together risk falling apart once they are revisited to make fixes and enhancements. A platform should allow apps to be built in a controlled, consistent manner that gives IT the freedom to respond to change quickly and confidently.

- **Good Citizen in the Mobile Ecosystem** – A platform that cannot integrate with other products, or a platform that is overly invested in its own set of products and services, will prevent companies from getting the true value of the product. The mobile ecosystem is ever-growing with products relating to MDM, MAM, API Management Platforms, Analytics Tools and Identity Management Systems. No one platform will be able to solve every problem for every company. As a result, a good platform must be equipped to work side-by-side with a variety of other solutions.

The Old Solutions

It is no secret that previous approaches to enterprise mobility have not stacked up well against this vision. Most solutions have fallen into one of three main categories:

1. Traditional MEAP

A broad category of software systems known as Mobile Enterprise Application Platforms ("MEAPs") showed initial promise addressing these challenges. Although traditional MEAPs were designed to meet the challenges of enterprise mobile app development, they typically did so in a monolithic fashion. In fact, MEAPs were often known to have a couple of major flaws.

- MEAPs were an all-or-nothing deal, wherein the enterprise needed to use the MEAP's proprietary development tools to build the app in order to realize the platform's benefits. Access to the server was hidden and embedded within this development environment. The world of mobile app development has since matured rapidly and therefore the imposition of such proprietary environments is no longer acceptable to the market. Mobile app development has come into its own, with outstanding IDEs from Apple, Adobe and Google among others. Standardized IDEs have emerged as the clear winner in mobile app development and any supposition that development should occur in a closed proprietary platform is a non-starter.

- Traditional MEAPs typically offer a "black box" approach to solving the enterprise mobility challenge. They offer a core set of features that tackle the largest problems. However, if a company wants to implement functionality that is not included in the out-of-the-box product, it is either very cumbersome or impossible to get outside of the MEAP's walled garden. Some MEAPs are starting to offer API access into their proprietary server, but this is nothing more than an API into a closed system.

In short, while the MEAPs provided an interesting way to begin enterprise mobile app development, the rapidly maturing market has shifted so that they no longer meet the requirements of most companies today.

2. Build From Scratch

Companies may find themselves tempted to build mobile apps from the ground up. Although there are many great tools in the market to help developers build a rich front-end for their apps, a home-grown approach will not address the many challenges and risks associated with building app infrastructure. Companies will spend valuable time and effort defining complex, foundational components instead of focusing their resources on features that add real business value. Here are just a few examples of app infrastructure that companies will need to address:

- Data connectivity to multiple back-end systems, including relational databases, web services, commercial systems like SAP, Siebel and Salesforce, and proprietary data structures.
- Enforcement of various security policies including authentication, authorization roles, data protection, audit logs, session management and remote administration of user access.
- Performance and scalability considerations
- Troubleshooting mechanisms

Ultimately, the amount of time and risk associated with these foundational elements can make it very difficult for companies to focus on those aspects of their apps that add the most business value.

3. Ignore and Do Nothing

Some companies may choose to ignore those aspects of mobile development that are most challenging. They may be quick to invest resources in a mobile project without considering the real risks of a security breach, unpredictable performance, or the inability to scale. The long-term effects of this approach are expensive and can be quite damaging. Needless to say, companies often come to regret this decision.

INTRODUCING VERIVO AKULA

Verivo Akula is an enterprise mobile app platform that helps businesses tackle the key challenges associated with building, securing and controlling enterprise mobile apps. Akula is the first platform to offer centralized control and management of custom mobile apps balanced with a flexibility and openness that gives companies and their developers a choice of tools, frameworks and technologies. With Verivo Akula, there is a better way.

WHAT DOES AKULA DO?

Akula provides a set of core capabilities that address the four main challenges associated with enterprise mobility.

1. Building transactional apps that operate offline

Offline apps can be difficult to build and challenging to operate. Akula helps companies by providing services and infrastructure related to the retrieval, storage, and bi-directional synchronization of data. To accomplish this, the product includes a number of important capabilities:

a. Akula includes a data module SDK that facilitates connectivity to virtually any on-premise or cloud back-end system, including relational databases, SOAP and REST web services, 3rd-party commercial systems, and proprietary APIs and data structures.

b. Akula provides an incremental data sync engine that handles the complex plumbing associated with local storage of data, changes to data, working with offline and online use cases, parsing result sets and merging new data with existing content.

c. For developers, Akula offers a clean separation between client SDKs and the specifics of back-end data access, so mobile developers can interact with all data through common server endpoints. That means developers can focus on the best ways to display and represent data instead of spending time on the specifics of how to access that data.

d. Akula includes an intuitive set of data APIs that abstract the low-level details of local data structures, offering a simple interface for developers to interact with their data regardless of network state or user type.

e. With Akula, data structures and schema are defined through a configurable XML interface, so data definitions can reside within client code or centrally on a server.

2. Security

Apps that do not enforce stringent security policies put organizations at risk on many levels. From data protection to hardened infrastructure to protection of customer records, Akula provides security on a number of levels.

a. Akula helps companies ensure that data is protected from end to end. Libraries built into the Akula client SDKs encrypt data on-device while SSL keeps data protected in transit.

b. Akula provides a realm SDK to facilitate integration with a variety of identity management solutions including Active Directory, LDAP and Siteminder.

c. Akula's integration with entitlements systems ensure that user definitions, roles and permissions trickle down directly from the corporate directory, so companies can maintain user access and role-based settings centrally in existing systems.

d. Akula provides a secure, server-side credential store to facilitate single-sign-on across a variety of back-end systems. And for companies who utilize IAM systems like Tivoli and Siteminder, Akula's token-based session management capabilities fit right in without duplication of effort.

e. Akula provides a hardened server against top security threats including the OWASP Top 10 list. To ensure it remains safe, the server is regularly verified against industry-standard testing tools.

3. Managing the runtime operations associated with mobile apps

Companies that have tried to build mobile apps understand that a successful mobile strategy must span further than just the building phase. To that end, Akula provides tools to help companies manage the on-going runtime operations associated with their mobile apps. From helping to reduce operational costs to keeping deployments running smoothly, Akula was built with operations management in mind.

a. Akula provides a feature called Managed Properties, which allows developers to keep content dynamic and avoid hardcoding static values within the compiled mobile app. That means IT Operations can update key attributes at runtime, from a centralized console, that affect the behavior of the various mobile apps that have already been deployed. In many cases, no redeployment of the client app is necessary.

b. Akula allows the entire server to be bundled into a single, deployable file to help deployments run smoothly. This process, which can be fully scripted to run in an automated fashion, keeps product code separate from environment-specific settings.

c. Akula's lightweight IT footprint allows companies to deploy its server in any number of configurations, from highly-scalable, load-balanced on-premise web farms to lightweight cloud deployments.

d. Akula provides a comprehensive set of management APIs, so companies have the ability to control their apps through a provided console or through preferred management tools.

4. Supporting apps that run on multiple device types and OS versions

Given the high rate of change within the mobility space, it is clear that IT organizations need to be ready to adapt to new technologies, new expectations from the end user, and new requirements from the business. Akula offers freedom to select the right technologies with the structure and discipline to ensure apps will be built with sustainability in mind.

a. Akula provides idiomatic client SDKs for iOS, Android and JavaScript that provide much of the common plumbing needed for enterprise apps. Akula's SDKs can be used with standard development tools like Xcode, Eclipse, PhoneGap and Appcelerator, which means developers do not need to learn proprietary tools or languages to build apps.

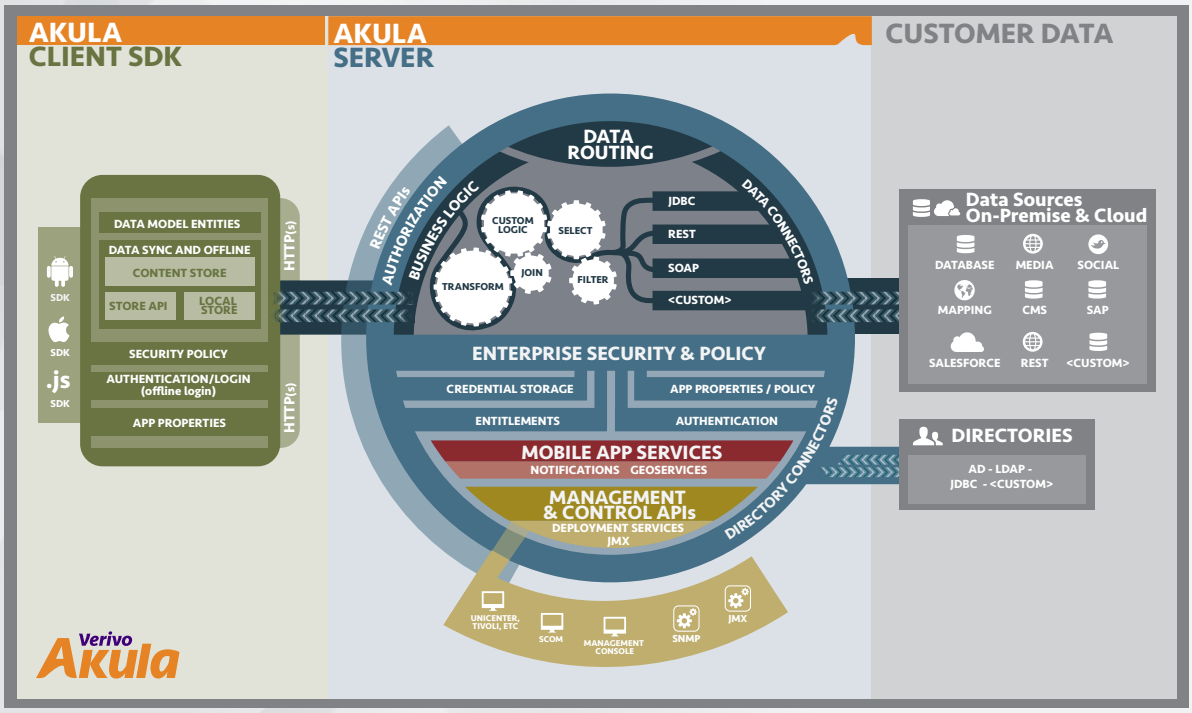
b. Akula's client SDKs allow developers to build whichever type of mobile app is most appropriate for their needs, from 100% native apps on iOS and Android, to mobile web apps using pure JavaScript, HTML5 and CSS3, to hybrid apps that wrap web content with a native wrapper, to a combination of all three.

c. Akula's client SDKs and server provide the common infrastructure needed to bring centralized control and management to apps built with a variety of mobile technologies. As a result, administrators can enforce corporate policy in a centralized way, even though apps might exist in a wide variety of form factors (native, web, smartphones, tablets, etc).

AKULA ARCHITECTURE

Akula consists of 4 main components:

- A Java-based server
- Idiomatic client SDKs
- Server SDK
- Management APIs



Details of each component are provided below.

Akula Server

The Akula server provides a mechanism for companies to incorporate data access, authentication, role-based authorization, monitoring and usage information into their apps. Companies define REST endpoints that expose data from various back-end systems like relational databases, web services, 3rd-party products like Siebel and Salesforce, and even proprietary data structures. With the server, companies can embed custom business logic and handle data routing.

Although apps built with the Akula client SDKs are the most likely consumers of the data and servers exposed by the server's REST endpoints, Akula endpoints are open and can be generally be consumed by any client. That can include a variety of browsers and custom-built clients.

The Akula server leverages several open and well-adopted frameworks including Apache Camel for data routing and Apache Shiro for security. Akula was built on Java EE 7 and runs in a variety of servlet containers like Tomcat and JBoss. As such, Akula can be deployed on any Microsoft Windows, Linux, UNIX, or OS X server

that supports Java EE 7. The Akula server can be installed on-premise or hosted, and it can be deployed virtually and in various high-availability configurations.

The Akula server makes it possible to connect to a variety of data sources via popular transport protocols including HTTP(S), REST, JMS, LDAP, JDBC and various web services. Akula provides data modules for popular back-end systems like Oracle, SQL Server, MySQL and others. However Akula's extensibility makes it possible for developers to leverage the Akula server SDK to connect to virtually any data source.

Client SDKs

The Akula client SDKs offer iOS, Android and JavaScript developers common infrastructure on which to build their apps. The SDKs automatically handle some of the most complex aspects of development, like handling data storage, incremental sync, data encryption, log in, eventing, and more. The client SDKs follow the conventions and standards of their respective platforms, yet provide a consistent cross-platform programming model, so developers will be productive immediately. And, because Akula is based on open standards, companies have the freedom to choose their development framework. You might choose to build one app natively on iOS, a second app in Appcelerator, and a 3rd app as a hybrid using PhoneGap.

A few components of the Akula client SDKs include:

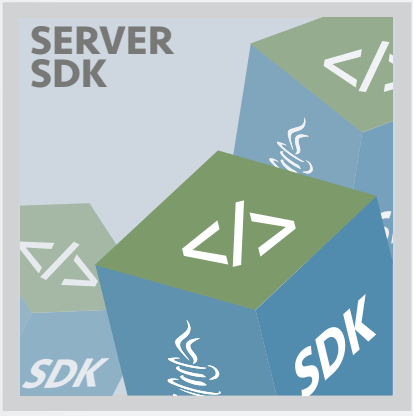
- Reliable, reusable infrastructure needed for most enterprise apps
- Secure communications
- Secure data storage
- Encryption of local data
- Offline login/logoff
- Cross-Platform Collection/Model APIs
- Incremental data sync for secure offline data usage
- Logging APIs
- Eventing
- Managed properties

Client apps communicate with the Akula server over HTTP(S). While any type of client can make direct requests to the endpoints on the Akula server, most developers will leverage the Akula client SDKs, which bundle many high-value pieces of functionality and structure into iOS, Android and JavaScript libraries.

Server SDK

The Akula server SDK provides tremendous extensibility to the platform. With the SDK, which is Java-based, companies can create Akula services, custom modules for data connectivity, realms for authentication, and implement any type of custom business logic, transformations or routing.

The Akula server SDK allows enterprises to use standard Enterprise Integration Patterns (EIP) to build modules that extend the capabilities of the server. Using Apache Camel, a well-known and tested open-source Java framework, developers can incorporate their own integration logic into their apps. Acting as an execution engine, the server will process each modular custom block of code while managing the passing of data between processes.



Management APIs

The Akula management APIs provide the mechanism to define roles, permissions, set values for managed properties and control administrator access to an app. Control is available through a management console or through APIs, which means companies could choose to build their own management interfaces or even leverage existing products like SCOM, Tivoli and others.

Akula's management APIs include control over:

- Role-based authorization of mobile users
- Role-based authorization of administrators
- Delegation of certain management to business owners
- Deployment and updates of apps scopes on the server
- Centrally management of app properties (e.g. connection parameters, CORS whitelist, etc.)
- Session management
- License management

AKULA SYSTEM REQUIREMENTS

Type	Requirement
Akula Server	
Operating System	• Windows Server 2008 R2 64-bit • Windows 7 SP1 • Mac OSX 10.8.x • RedHat Enterprise Linux 6.3* * Additional Operating Systems and versions should work as long as the appropriate Java JRE or JDK is available. For example, most recent variations of Linux including RedHat Enterprise Linux 6.3, though not tested, should run Akula properly.
Java SDK	Java JDK 7 http://www.oracle.com/technetwork/java/javase/downloads/java-se-jdk-7-download-432154.html
Java SDK JEE Server or Servlet Container	• JBoss EAP Version 6.0.1 from http://www.jboss.org/overview/ • Apache Tomcat 7 from http://tomcat.apache.org/download-70.cgi
RAM	4 GB
Disk Space	500 MB
Akula Client SDK	
Android SDK	Android OS v2.3.3 or later
iOS SDK	iOS v5.x – 6.1

HOW DOES AKULA IMPACT AND BENEFIT THE ORGANIZATION?

Akula helps reduce the complexities associated with enterprise mobile app development, and that's an important benefit that can be felt across the entire organization, from Development to IT Operations to the various Lines of Business. Akula helps companies build, secure and control their apps, which is particularly attractive to IT Management, who strives to deliver end-to-end technology solutions that "just work." IT Management looks at the big picture for a solution that:

- boosts development efficiency
- doesn't cost a fortune to operate
- integrates well with existing systems and policies
- helps ensure that projects are delivered on time and on budget
- reduces security risks
- keeps deployments running smoothly

When choosing a technology solution, IT Management will often consider the impact to three important divisions of the IT organization: IT Applications Development, IT Security and IT Operations. Let us take a moment to explore Akula's impact and benefits to each of these groups.

What Challenges Does Akula Help Solve?

1. Mobile apps must operate and remain transactional even when offline.
2. Apps must meet rigorous security standards.
3. IT must control and manage app deployment and usage.
4. Apps must run in multi-device, multi-OS environments.

IT Applications Development

Akula decouples the tasks performed by mobile and back-end developers. That allows each development group to focus on their areas of expertise, keeping them efficient and productive. The services defined on the Akula server are shared across each of the client SDKs, so companies have the control to enforce policies and business logic universally. Mobile developers using the Akula client SDKs will not only find the APIs intuitive, well-defined and idiomatic, but they will also find that Akula's mobile infrastructure (i.e., data models, sync engine and data access) can be implemented with fewer lines of code than if they were to build that infrastructure from scratch. So mobile developers benefit by leveraging pre-built app infrastructure while IT Operations and IT Security benefit from centralized control over their apps. That, combined with Akula's flexibility to let companies choose which mobile technologies are right for them, makes Akula a powerful tool for IT Applications Development.

Mobile Client Development:

Mobile developers build apps on top of the Akula client SDKs, which means developers avoid the heavy lifting often associated with data connectivity, synchronization, and policy compliance. Instead, they're free to focus on delivering powerful apps, using the tools of their choice, to transform business processes. Verivo has eliminated the need for developers to learn new, proprietary languages or IDEs for developing apps. That means that using standard development languages like Objective-C, Java for Android, JavaScript and even IDEs like Appcelerator, developers leverage their existing skillsets to deliver rich, enterprise-grade, limitless apps.

Back-End Development:

The Akula server allows companies to create reusable endpoints that provide access to back-end data sources and identity management systems. Some back-end data modules exist out-of-the-box while others can be custom-built using the Akula server SDK to fulfill each company's specific needs. The server provides a secure mechanism to handle some of the most complex areas of mobile development, like data integration, authentication, authorization, credential storage, enforcement of app policies, data routing and business logic.

Akula, by design, is an open and extensible platform. That means that developers are able to leverage both packaged features and functionality that ship with the product as well as extend the platform through the creation of custom modules. As a result, there are very few limitations imposed on the enterprise app developer.

Akula helps developers to:

- Connect to any data necessary for the running of the app without the need for specific knowledge.
- Work with "offline data" as easily as if designing for connected data.
- Leverage an easy-to-use, smart and powerful sync engine.
- Allow app authentication against most widely-used corporate identification systems.
- Build highly scalable apps without the need for additional, specialty code.
- Debug backend data connection issues with speed and ease.
- Migrate apps from test data to production data without backend connection code changes.
- Comply with centralized user and security policies automatically, across all apps built on Akula.
- Ensure that apps and data are secure without the need for special code.
- Develop a common backend infrastructure and leverage it across all supported clients.
- Use tools that are well known and comfortable, such as Xcode, Eclipse, Dreamweaver and PhoneGap.

IT Security

Akula offers a variety of enterprise-grade security features to help companies build secure apps, including the ability to encrypt data both at rest on-device and in transit, enforce user entitlements through single-sign-on and authorization roles, audit transactional history through extensive logging, and enforce security policies centrally. The Akula server is hardened against common security threats including the OWASP Top 10 list and it has been validated with standard security testing tools. Akula facilitates the creation of secure apps on a few fronts:

1. **Access Control –**

For an enterprise mobile app, it is critical to limit access to the app and its data to ensure only the right people can access the appropriate data. To that end, Akula offers features in the following areas:

a. **Authentication** – Akula can integrate with a variety of identity management systems, including Active Directory, LDAP, Siteminder and others. The platform includes connectors ("realms") for some systems out of the box, but companies can always create customized behavior with Akula's Realm SDK. Akula's authentication mechanism adds instant value to the developer by supporting a number of features via the Akula client SDK, including offline login, session timeouts and the enforcement of related policies. For example, using the Akula client SDK, a mobile developer can make a call to authenticate a user with a single line of code, and both the online and offline login scenarios will be handled automatically. Akula also supports authentication via SAML tokens, OAuth and other common protocols.

b. **Single-Sign On** – Akula includes a server-side credential store that allows companies to store (and encrypt) user credentials for multiple back-end systems. Or, if a company has already invested in Siteminder, Tivoli Access Manager or another SSO tool, Akula can integrate with those via its built-in realm mechanism. Such credential storage mechanisms are critical for apps that must connect to multiple data sources, each authenticating with the end-users' credentials.

c. **Authorization** – Akula server endpoints can be configured to allow access only to certain roles. These roles are inherited directly from the system of record for identity management (e.g., Active Directory), so changes made on the back-end will trickle down to Akula in real-time. Also, user definitions only need to exist in the back-end system and do not need to be re-defined or managed separately on the Akula server.

d. **Session Management** – Akula handles session management for both online and offline scenarios and allows companies to configure timeouts and other related settings. The platform's server-side persistent storage manager allows session data to persist across multiple requests and even across multiple servers, allowing for full session management even in high-availability and load-balanced sever configurations.

2. **Data Protection –**

It cannot be overstated that protection of data is critical to the success of any mobile solution. Akula's client SDK provides data encryption mechanisms for locally-stored data on iOS and Android devices and protects each app's encryption keys. For data in transit, the Akula server supports SSL so that data connections between client and server can flow over HTTPS.

3. **Protection from Attack –**

Although the Akula server may often reside on-premise and behind a company's firewall, it is critical that the server be protected from malware and common types of attacks. To that end, the Akula server has been built and tested to withstand top security threats including injection, cross-site scripting, cross-site request forgery and others. The security of the Akula server is regularly tested with standard security testing tools including IBM AppScan and OWASP Zed Attack Proxy (ZAP).

Why Akula?

1. Build apps faster with common infrastructure and idiomatic client SDKs
2. Reduce costs by leveraging existing investments
3. Centralized management of custom mobile apps with delegation of control
4. Secured access to enterprise data, optimized for mobile connectivity
5. Transform data from disparate back-end systems
6. Visibility through rich logging
7. Apps are created using preferred methodologies
8. Clean separation between application code and properties
9. Secure management of enterprise data locally on mobile devices

4. **Logging –**

Akula provides extensive logging for all calls to server endpoints including data requests, authentication attempts and management API requests. The logs are highly configurable and can be tailored to specific business and security needs. For example, logs can be configured to write varying levels of detail out to a log file depending on which module is being requesting or which user is requesting it. Or, a company may decide to direct output for a particular event to an Oracle database instead of a flat file in the file system. Akula's logging mechanism is based on the SLF4J implementation of Apache log4j, so it is highly configurable and extensible. Companies may take advantage of this flexibility to report on specific abnormal pattern requests or keep an audit trail of specific management roles changes.

IT Operations

Akula provides rich capabilities to help companies manage the runtime operations associated with enterprise mobile apps. Whether an app is built from the ground up using Akula, or whether a previously-existing app is retrofitted onto the platform, Akula facilitates the central control and management that IT Operations needs to keep its technology running smoothly.

A few of Akula's key operational capabilities include:

1. **Reducing Operational Costs** – Akula's agile nature allows it to fit well into existing IT infrastructure, whether that be on-premise or in the cloud. Akula's server can be run in a virtual environment, and processing power (CPU allocation) can scale up or down quickly as a company's needs change. On-going operational costs can have a big impact on the total cost of ownership of any product, and Akula was designed with a nimble and efficient architecture that allows it to have minimal impact to an organization's IT infrastructure.

2. **Deployment** – Most IT organizations will admit that go-live events are often associated with some amount of stress and risk. The act of moving software into a production environment, whether for an initial deployment or for an upgrade, receives high visibility and needs to go smoothly. Akula helps ease this process by separating server code from environmental settings, allowing deployments to be fully-scripted and well-tested. This approach also reduces risk, as deployments can now run without requiring changes to production connection parameters for databases, web services and other data sources.

3. **Troubleshooting** – Akula provides an extensive (and extensible) logging framework that allows developers and IT Operations to implement comprehensive logging strategies across all of their mobile apps. This accelerates troubleshooting and issue resolution, should a problem arise in an app. For example, if a user out in the field is having trouble syncing his data, the IT Operations group responsible for supporting that app needs quick, easy access to system logs to detect anomalies.

4. **Enforcement of Corporate Policies** – The Akula platform includes a number of out-of-the-box properties that help administrators control the behavior of both Akula services on the server and the behavior of mobile apps built using Akula's client SDKs. In addition to the properties already built into the platform, Akula includes a mechanism to allow companies to create custom properties at both the app and server level. For example, an app property might be set to control the maximum number of invalid offline login attempts allowed before an end user is required to return to network coverage before a successful login can occur. Or, IT Operations might use an app property to control the geographic area in which an end user is allowed to perform data sync. Needless to say, the ability to enforce policies across a variety of apps and devices, and the ability to make changes to the policies at runtime, provides IT Operations with tremendous control.

5. **Scalability & Performance** – The Akula server was built to scale. It handles mobile transactions efficiently and fits well into a variety of load-balanced and high-availability configurations. For fastest time to develop, Akula leverages embedded data storage mechanisms to persist credentials, policies, and session data. However, for larger-scale production environments with multiple instances of the Akula server that reside behind a load balancer, Akula leverages Memcached to extract those data objects to an external source like a SQL Server or Oracle database.

6. **Management Console and APIs** – Akula provides robust management APIs to allow outside tools access to the various properties, roles and permissions defined for each app. By default, Akula includes a Python-based command-line console to configure settings via the management APIs. Third-party or custom-built tools can access and modify these properties by making calls to the management APIs' REST endpoints. The management APIs can also be integrated with existing 3rd-party management tools like Tivoli, SCOM and others.

TAKE THE NEXT STEP

With the acceleration of mobility as a critical requirement across the enterprise, companies must take action immediately to implement a mobile strategy. Companies who delay risk falling behind their competition and lagging behind the market. There is a clear need for compelling enterprise mobile apps across a variety of device types that can be controlled and managed centrally. Although there is no shortage of options for mobile development today, very few of them are enterprise-ready and offer the right balance of flexibility and control and are critical to the long-term operation of each resulting app.

Visit Verivo's DevCenter to get started with Akula today. DevCenter provides access to product documentation, API references, customer support, a develop community and a link to begin a free trial of the software.

Whether you're looking to get started building new mobile apps, or gain control over existing apps circulating around your company, you can't afford to delay. Get started with Verivo Akula today.

ABOUT VERIVO SOFTWARE

A leading provider of enterprise mobility software, Verivo Software helps companies accelerate their business results. Its unique technology empowers teams to centrally build, secure, control and update their enterprise mobile apps — rapidly and across multiple devices. Hundreds of companies in numerous industries around the world rely on Verivo's platform to drive their mobile initiatives. To learn more, visit www.verivo.com.