White Paper

Enterprise Mobility:
A Competitive Comparison of Leading Mobile Application Development Platforms
Table of Contents

Executive Summary ........................................................................................................................................... 1
Introduction ..................................................................................................................................................... 2
Objective ......................................................................................................................................................... 2
A Historical Example of the Need for a Single Platform Solution ................................................................. 2
How to Become Mobile? ................................................................................................................................... 2
Audience ........................................................................................................................................................... 4
Mobile Considerations for the Enterprise ......................................................................................................... 5
Mobile Application Development Platform ....................................................................................................... 5
App Security and Device Management ............................................................................................................. 6
Integration ........................................................................................................................................................ 7
Testing and Quality Improvement ...................................................................................................................... 8
Comparing the Leading MADP Platforms ......................................................................................................... 9
Use Case Scenario: Rail Engineer to Repair Malfunctioning Junction Box at Remote Location ................. 12
Development Environment and Approach ....................................................................................................... 13
Ability to Integrate Native and Web Components ........................................................................................... 14
Strong Support for CLI ...................................................................................................................................... 15
Access to Open Source Libraries ..................................................................................................................... 15
Rich Developer Ecosystem ................................................................................................................................ 15
Strict MVC Adherence ..................................................................................................................................... 16
Integration with Back-end Systems and User Authentication .......................................................................... 17
Authentication .................................................................................................................................................. 17
Authorization ................................................................................................................................................... 18
Transaction Management ................................................................................................................................... 19
Advanced Security Methods ............................................................................................................................. 20
Application Quality Improvement, Operations, and Control ............................................................................. 21
Conclusions and Recommendations ................................................................................................................ 23
Executive Summary

The market for enterprise application development is expected to reach $61B by 2018, according to Strategy Analytics. Gartner predicts that by the end of 2015, 50 percent of business processes that involve human tasks will require near-real-time responses that are mobile-enabled. Mobility has quickly become an expected norm as opposed to a new trend. The fact remains, however, that the market for Mobile Application Development Platforms (MADP) is crowded and confused. It is highly fragmented and is bracketed by small niche players that are providing lightweight and focused solutions on the low end and global firms providing “do everything” platforms, from design and development through to test and deploy. Gartner lists 22 platforms alone in their Magic Quadrant, and those companies needed to exceed Gartner’s stringent financial guidelines to be considered.

The purpose of this paper is to provide the considerations related to the top three vendors in the enterprise app development space: IBM, SAP, and Kony. A single use case will be employed, which will demonstrate many of the key areas where a single platform for design, development, test, and deployment of mobile apps can benefit an organization. Products from SAP, Kony and IBM, will be examined, and capabilities of each company’s solution will be compared.

The use case will focus on a field worker repairing a failed component in an area with limited connectivity. It will illustrate the capabilities for developing the app, use of native OS features, integrating with back-end data, creating secured transactions from identified sources, continuous integration, consistent deployment, on-going quality improvement, and rapid improvement of app usability and feature-set.

In summary, this study is intended to provide an overview of mobile delivery features and capabilities for the enterprise, and to compare solutions from SAP, Kony and IBM using a real-world use case that enables an informed decision when choosing an enterprise mobility platform.

Upon evaluation, Edison has concluded that the IBM MobileFirst Platform provides a superior software product and a complete mobile platform with distinct advantages to SAP and Kony in aspects of development, security, transaction creation, and quality improvement.
Introduction

Objective

There is a great deal expected from mobile apps. Given that the use of handheld devices has become such a part of daily lives, it seems unthinkable that the business world in general is so far behind in many respects. But that is, in fact, the case. The effort for IT departments to move as quickly as the mobile world is a struggle. This is due, in large part, to the old IT paradigm of executing projects, where development cannot begin until funding is received, but funding cannot happen until the IT team does enough analysis to determine a detailed cost. It is also in part because IT departments have spent a lot of money building complex systems that require extensive support and funding to maintain and enhance. With this in mind, some aspects of mobility as it relates to IT and the initiative to get mobile will be explored.

A Historical Example of the Need for a Single Platform Solution

Much of how an organization addresses mobility is based on its size, funding and history, just as was the case for previous IT transformations. Recall that during the web arena of the late 1990s, there was a period of time when organizations took a “solve it now” approach for building web sites. Web sites were departmentally funded. They were disconnected sites, with different development platforms (such as PERL) and were often not built to talk to each other. Web content was managed by webmasters who often updated HTML files. Eventually, Web Content Management systems were created, and soon enough, “portals” became prevalent. Organizations felt they needed a web portal, perhaps thinking of AOL or Yahoo. A single, unified platform was desired - IT wanted to support one platform, not 25 separate web sites. Of course, there were also the browser wars - building sites targeting a specific browser or different sites for different browsers.

How to Become Mobile?

Replace Netscape and Internet Explorer with iOS and Android, and there is a sinking feeling that little has been learned from the recent past. Many of the same approaches that were used for web sites in the late 1990s are being applied to mobility today. The good news is that unlike the situation then, there are already alternatives and answers to the conundrum felt by IT leaders struggling to quickly gain some manner of reuse and control over what is happening in the mobile development space.
In looking at the mobile maturity steps that many large enterprises go through, the first step is to select a particular device, with a majority in the last five years selecting an iOS device. After deploying productivity apps, such as email, these organizations will build several native apps that are not connected directly to back-end data. According to the Gartner “Rule of Three”\(^1\) when an enterprise needs to support more than three apps, three device OSes, or integrate three back-end systems then a development platform will bring savings and strategic advantages. Since 2012 the platform landscape has grown exponentially, with a large variety of complexity and capability, and cost. Over that three-year period the Gartner Magic Quadrant has seen significant change and consolidation through acquisition, with SAP, Syclo, and Antenna being the only platforms to crack the upper right. Interestingly, while SAP appeared in that report with a platform, they subsequently bought Syclo to solidify their offering.

At the lower end of the solutions on the quadrant are more inexpensive tools that appear to be easy to deploy. Some of these solutions are open source, while others with a one-time cost or low monthly fee provide the ability to create forms, integrate with other business solutions such as Microsoft Office or Google Docs, or provide a way to interact with a website - if the device is connected. GoCanvas, doForms, Capriza, and Webalo are examples of companies that offer the latter type of solution. This path usually requires more development effort than anticipated and generally does not solve the whole problem. For example, these solutions may allow a mobile worker to manually type in part numbers, but do not provide the ability to scan a bar code; or the solutions do not connect easily to the organizations ERP system for prepopulating forms with relevant data, which is most efficient in terms of both lowering error rates and speed of entry. Those that follow the path utilizing these solutions face the challenge of potentially needing a more mature option that meets their needs more completely, or abandoning a platform altogether.

The purpose of this paper is to provide an analysis of the three most popular MADPs for the purpose of fast-tracking the adoption of a mobile platform, while still addressing the needs of enterprise mobility. The intent is to provide a foundation for choosing the right platform, so that one is not left looking for features and extensibility a few months after getting started. The example use case will be evaluated across the three platforms - SAP, Kony and IBM MobileFirst Platform to illustrate clearly the pros and cons of each. The result should be a blueprint of next steps for an audience that needs to make a real-world decision on a platform, not find themselves in a quagmire reminiscent of the late 1990s: having dozens of apps to support, all developed for two or three platforms; using

---

\(^{1}\) “Magic Quadrant for Mobile Enterprise Application”, Gartner Note G00162969, 18 December 2008
a myriad of open source libraries, both supported and unsupported; and so forth. The path to mobile maturity is not long or difficult if informed choices are made up front.

**Audience**

The audience for this white paper will be IT decision makers in organizations and business executives who are looking to educate themselves about the MADP marketplace and their purchasing options.
Mobile Considerations for the Enterprise

On the surface, choosing a development platform may appear to be the most impactful decision regarding the evolution of a mobile enterprise. While the MADP is certainly a major component of a mobile enterprise, there are other considerations that are closely related to the MADP of choice, and it could be argued that these other considerations are equally important. In fact, in Edison’s view, these aspects of mobility are tightly linked, and it is recommended that a foundational understanding of their importance and relationship be established.

Mobile Application Development Platform

The development platform for mobile applications is the primary topic of this paper and critical to the success of a mobile strategy. The MADP market has become fragmented and it is becoming unclear what the term MADP really means. Since the introduction of jQuery Mobile, and more recently Ionic/Angular.js and other JavaScript frameworks, new tools are being introduced quite often. Some are being built by a handful of developers using Github, while others are backed by larger entities or venture capital. For the purposes of this paper, we have classified the development platforms into four (4) categories:

- Full-fledged MADPs
- Open Source
- “Mini”-MADPs
- Point Solutions

A full-fledged MADP integrates design, development, testing, and deployment within a single integrated solution.

---

2 An HTML5 and mobile web development framework, based upon Jquery, and optimized for mobile apps. [jQuerymobile.com](http://jQuerymobile.com).
3 A front-end framework for developing native-feeling hybrid mobile apps in HTML5 using Sass and AngularJS. [ionicframework.com](http://ionicframework.com). AngularJS is an open-source web application framework maintained by Google primarily geared towards the development of single-page applications (SPA). SPAs have become the standard for app user experience, as the user is presented with a main page and all features are extensions of that page. [angularjs.org](http://angularjs.org).
4 Web-based Git repository hosting service, which offers all of the distributed revision control and source code management (SCM) functionality of Git. Github also incorporates tools that support the open source community, including wikis, task management, bug and feature request tracking. It is the largest code host in the world. [Github.com](http://Github.com) and [git-scm.com](http://git-scm.com).
Open source refers to all of the tools that exist that are free from license fees for anyone to use under the GNA or other “open” licensing agreements. They often include a request for donation or a fee for support. The Apache Cordova⁵ project is a good example of this, and is not to be confused with PhoneGap, which has been purchased by Adobe. Ionic framework is an example of open source technology that can simply be downloaded and used to build mobile apps.

What we are calling “Mini-MADPs” are the tools that do some of what a true MADP does, but is not a full, enterprise ready solution. Tools such as GoCanvas, Webalo or doForms fall into this category. These tools have some proprietary software component that allows developers to create mobile applications to some extent, but lack a full set of capabilities to truly be considered Mobile Application Development Platforms. For example, GoCanvas provides a jQuery-like user interface to build the front end of a mobile app. It provides a way to create a form and submit a PDF via email, or potentially to connect to an API for more subtle functionality. While this may meet the needs of some users, it does not provide any further capability. In fact, many of these tools do not provide the full set of device features and require more development time than a traditional MADP.

Point solutions are simply mobile apps that do something very specific. MobileFrame has been a major player in this category, providing Proof of Delivery applications. This category is more aligned with the traditional “full solution” mentality, in that it is usually more than just the mobile app, but a full-fledged software solution that has a mobile component.

For the purposes of this paper, we are focused on the full-fledged MADP category, and specifically, SAP SMP 3.0, Kony and the IBM MobileFirst Platform.

The following sections describe key aspects of an MADP: App security and device management, integration, and testing and quality improvement.

**App Security and Device Management**

Validating the authenticity of the app, the transactions it creates/consumes, and the device on which an app is running are key pieces of the overall mobility story. While app/device management was once an operational task focused on locking down a device - usually a laptop - it is now something that needs to be considered as part of an overall

---

⁵ Set of device APIs that allow a mobile app developer to access native device function such as the camera or accelerometer from JavaScript. The original open source project was purchased by Adobe and is now licensed, for a fee, as PhoneGap. The Apache project has adopted its original name – Cordova.

cordova.apache.org
app build and deployment strategy. Leading device management software often has a mechanism to deploy an app, and can even provide an API for an app developer to leverage for security. Some key considerations related to device management are listed below.

- Provisioning of devices across the enterprise
- Ability to wipe a device, either selectively or completely
- Lock and unlock devices based on rules or manual interaction
- Management of VPN
- Compliance and usage reporting
- Jailbreak detection
- Whitelist/blacklist apps
- Performance monitoring for the device, network and apps
- App deployment

**Integration**

The integration aspect of a mobile app is often the biggest concern for mobile development, and biggest cost related to mobile development. Generally, only 25 percent of the overall cost of the app is for the client side development. As an example, to bring mobility to employees in the field related to supply chain or ERP system data, what is often found is that the mobility aspect of the project is the more direct and simplest piece of the solution. The front-end design and mobile code, security and deployment will have to be executed properly and with a mobile mentality. But that said, the underlying data may not be mobile-ready. The data sets may be too large, the transactions take too long, and the current APIs may be set up more for a web interface than a mobile interface.

For example, a large dataset being transferred over internal networks in an XML format to a web client on a secure desktop has a certain expectation on bandwidth and processing power on the client. That same transaction may perform poorly over a cellular network on a device with far less processing power and no single sign-on method. What this means is that even if the mobile aspect of the project is architectured correctly, it is likely that the existing services will need to be altered, especially in the case where server-side rendering of HTML is tightly woven into the solution. Decisions

---

6 Forrester 2014 mobile trends report indicated that 25-50% of the cost of an app is related to front-end development and the remaining cost is creative, design, backend development and testing.
related to using JSON instead of XML, device security, and image optimization for mobile, among others, are considerations that are not specifically MADP related, but should be addressed. The ability to alter middle-tier services and integration components is key to a successful mobile development strategy.

**Testing and Quality Improvement**

The distributed ecosystem within which the enterprise app operates is considerably more complex than even that for web apps. The mobile device is a fully-capable computing device that can operate both on and off-line and is generally connected to back-end data through a complex series of communication methods/protocols, firewalls and protection layers, and an intricate layer of data access methods. Testing the end-to-end solution from the touch capabilities of the device all the way through to the remote database transaction management is quite complex, requiring a vast array of tool types and orchestrations. Additionally, testing the user experience and visual elements of the app can predict its success or failure to provide the expected business value. Organizations are looking for tools and frameworks that can help standardize and consistently deliver improved quality levels, accounting for the myriad of changes and upgrades to each component of the mobile ecosystem.
Comparing the Leading MADP Platforms

Evaluating and contrasting products via a use case allows the reader to understand how basic development activities are addressed by each Enterprise Mobility Platform in a real world scenario. While this use case is not exhaustive, or intended to be so, it does enable a conversation and evaluation of the major MADP platforms, SAP SMP 3.0, Kony, and the IBM MobileFirst Platform. The present use case was developed based on a combination of hands-on review, subject matter expertise, and documentation-based research. The resulting evaluation is summarized in Tables 1 and 2.

<table>
<thead>
<tr>
<th>Company</th>
<th>Product/Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAP</td>
<td>UI5/Kapsel</td>
<td>UI5 is the UI framework for designing HTML5 hybrid mobile apps and Kapsel is the container within which the hybrid app will run</td>
</tr>
<tr>
<td></td>
<td>Agentry</td>
<td>UI components for creating SAP-connected native apps</td>
</tr>
<tr>
<td></td>
<td>Afaria</td>
<td>Mobile device management</td>
</tr>
<tr>
<td></td>
<td>HANA Mobile Platform</td>
<td>End point for client apps providing security, back-end data aggregation, analytics, and app lifecycle management</td>
</tr>
<tr>
<td></td>
<td>Netweaver Gateway</td>
<td>Back-end data integration manager providing access to SAP Business Suite for the SAP Mobile Platform</td>
</tr>
<tr>
<td></td>
<td>OData</td>
<td>REST-based protocol for CRUD data operations</td>
</tr>
<tr>
<td></td>
<td>Lean Java Server</td>
<td>Integration server for supporting non-SAP backends</td>
</tr>
<tr>
<td>Company</td>
<td>Product/Component</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>------------------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| IBM     | MobileFirst Platform Foundation | IBM MobileFirst Studio - IDE and CLI tools for mobile app development  
|         |                   | IBM MobileFirst Server - middleware that serves as a gateway between applications, back-end systems and cloud-based services  
|         |                   | IBM MobileFirst Device Runtime Components (formerly the Worklight container) - runtime client application program interfaces (API)  
|         |                   | IBM MobileFirst Application Center - Enterprise App Store  
|         |                   | IBM MobileFirst Console - administrative GUI providing real-time operational analytics for the server, adapters, and applications and push services to help you manage, monitor and instrument mobile apps  
|         | MobileFirst Application Scanning | Code scanning capabilities that enables you to identify source code vulnerabilities early in the software development lifecycle and addressed before deployment  
|         | MobileFirst Quality Assurance | A set of tools and features to help provide quality assurance to your mobile applications. It includes automated crash analytics, user feedback and sentiment analysis, in-app bug reporting, over-the-air build distribution to testers, and test/bug prioritization  
| Kony    | Studio           | Eclipse plug-in for coding mobile apps for all mobile OSes  
|         | Visualizer       | App design tool with drag and drop development capabilities, intended to be used by non-developers to define the look and feel and functional flow of an app  
|         | MobileFabric     | Cloud-based deployment platform and MBaaS  
|         | Management       | Cloud-based device and app management, with analytics and reporting capability  

Table 1 - MADP Components

7 Cloud-based model for providing web and mobile app developers with a way to link their applications to backend cloud storage and APIs exposed by back end applications, while also providing features such as user management, push notifications, and integration with social networking services.
<table>
<thead>
<tr>
<th>Category</th>
<th>Sub-Category</th>
<th>IBM MobileFirst Platform</th>
<th>SAP Platform 3.0</th>
<th>Kony</th>
</tr>
</thead>
<tbody>
<tr>
<td>Development Environment and Approach</td>
<td>Ability to integrate native and web components</td>
<td>9</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>Strong support for CLI</td>
<td>7</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>Access to open-source libraries</td>
<td>9</td>
<td>7</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>Richness of developer ecosystem</td>
<td>7</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>Strict MVC adherence</td>
<td>7</td>
<td>7</td>
<td>5</td>
</tr>
<tr>
<td>Integration with Back-end Systems and User Authentication</td>
<td>Authentication</td>
<td>9</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>Authorization</td>
<td>7</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>Transaction Management</td>
<td>8</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>Advanced Security Methods</td>
<td></td>
<td>9</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>Application Quality Improvement, Operations, and Control</td>
<td></td>
<td>9</td>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>Totals</td>
<td></td>
<td>81</td>
<td>58</td>
<td>59</td>
</tr>
</tbody>
</table>

Table 2 - MADP Rating Scorecard (10=High, 1=Low)
Use Case Scenario: Rail Engineer to Repair Malfunctioning Junction Box at Remote Location

In this use case, a sub-contracted field worker will log in remotely, via an app, and retrieve a work order electronically with the location and nature of the repair. He will load the truck with the anticipated replacement parts and tools. The app will provide directions and GPS tracking to the repair location identified in the work order. The repairperson will “clock in” at the site, via the app, and go to the malfunctioning junction box. Using the app, the worker will open repair schematics. The worker will make annotations to the schematics and take a photo of the junction box internals for future reference. The failing component is not the one identified in the work order, so the worker submits a change, which requires approval from his manager. The worker’s manager responds to the alert and approves the updated work order. The technician completes the repair, updates the work order and signs it, the parts register, and asset status, and “clocks out”.

This use case will require a developer to build an app with the following capabilities:

1. Validate the device, user and app as part of start-up
2. Authenticate against corporate Active Directory (AD) via LDAP
3. Authorize against corporate work order system
4. Integrate with the work order system
5. Use native mapping function, with work specific overlays, to locate address identified in work order
6. Allow for offline use at work site, and sync when connection is available
7. Update work order with start time
8. Background update of documents related to items for repair
9. Capture annotations to documents
10. Capture photo and associate with required document
11. Upload document updates and photos for internal review and store photo in a NoSQL⁸ database that complements the back-end database without having to change it

---

⁸ NoSQL databases were designed for modern web-scale applications, where large numbers of users and the dynamic nature of usage patterns made traditional SQL engines less performant. The simplest NoSQL DBses key value pairs, an attribute and its value. Popular examples of NoSQL databases are Hadoop and MongoDB.
12. Integrate with corporate BPM (Business Process Management) services to send change order approval request

13. Track approval request

14. Capture signature on work order

15. Create and submit transactions for four (4) different corporate systems, including authorization:
   a. Work Order Management, via Ventyx Ellipse
   b. Physical Asset Register, via Bentley InspectTech
   c. Parts Register, via SAP SPM\(^9\), not to be confused with SAP Mobile Platform (SMP)
   d. Time Tracking, via Click Software

Because the field worker is sub-contracted, the mobile device the worker will use is unknown, but has been limited to iPad mini (iOS) and Google Nexus7 (Android). This will require the developer to create an app that combines both native, to allow for Near-Field Communications (NFC) check-in and check-out, and web so that it works on two different operating systems without having to re-write everything.

As identified earlier, the key areas of focus for evaluating SMP 3.0, Kony, and the IBM MobileFirst Platform are:

- Development environment and approach
- Integration with back-end systems and user authentication
- Advanced security mechanisms
- Application quality improvement, operations, and control

These will be reviewed in relation to the defined use case.

**Development Environment and Approach**

Developers of enterprise apps are looking for several features in a development platform that can facilitate the agile delivery model that mobile requires. These include the ability

---

\(^9\) Service Parts Management (SPM) is a comprehensive, cross-component solution for supporting and managing the extended service parts network. It encompasses the entire range of service parts management activities, including planning, execution, fulfillment, collaboration, and analytics. SPM helps you link your entire service value chain for better visibility, coordination, collaboration, and control of the service parts network.
to easily integrate native and web components, strong support for Command Line Interface (CLI), access to open source libraries, rich source of threads in the Q&A website, `stackoverflow`, and strict MVC (Model-View-Controller) pattern adherence. The use case combines complex processes, mask flows (background transactions), and a large number of data interfaces, so that we can compare each platform against each agile requirement.

**Ability to Integrate Native and Web Components**

Given the complexity of the app requirements, including mapping with overlays, photo and document annotation, and signature capture, it would be logical to create a native app in iOS and Android and then add the web capabilities for the back-end integration and transaction management. To provide the hybrid (web) capabilities, all three products use a container to run the web components, similar to Cordova in functionality. SAP and IBM inherited their container through acquisition, while Kony has built theirs from the ground up. As is the case with Cordova, each container exposes a set of APIs to the app for accessing native features and security.

With the IBM MobileFirst Platform, at a high level, one would replace the AndroidManifest with AndroidManifest-WL (for Android) or update the Xcode project with the worklight.plist and WorklightSDK. At that point one can easily mix and match native and web components through a familiar native interface.

With the Kony platform, a developer is required to stay within the Kony Studio and either use the Kony SDK framework for that OS or wrap native components to run within their container. For app developers with native experience this will slow down the team velocity and restrict access to functions that have not yet been exposed by the Kony container (e.g. biometric authentication).

The same applies to SAPUI5, due to the dependence on the remaining components of the Syclo engine. Even using Kapsel\(^\text{10}\) would require wrapping native functions and using the exposed API as opposed to direct access to native functions, storage, and security. In the use case this would inhibit the creation of the mapping/overlay data, geo-fencing for start/stop time capture, and background update of documents (an issue when a hybrid app loses focus). Each of these can be done more efficiently in the IBM MobileFirst platform due to IBM’s approach of generating native code and not operating

---

\(^{10}\) Kapsel is a set of SAP® plugins for Apache Cordova, allowing the container to better integrate with the SAP Mobile Platform server.
within a web container. It also reduces the dependence upon the Kony or SAP-specific framework and SDK.

**Strong Support for CLI**

Most long-time app developers utilize CLI. If one has experience in a platform then the CLI will be a much faster way of setting up and managing the development environment. The IBM MobileFirst Platform has both a full CLI along and an Eclipse plug-in (Studio IDE), which allows developers to pick and choose what interaction method they want whether individually or as part of a team. Kony restricts access to a full CLI to protect less experienced developers from negatively impacting the streamlined functioning of the Kony Studio. The “guardrails” become frustrating once more experience is gained and developer speed gets slowed down. The SAPUI5 interface is part CLI and part menu-driven, but is cumbersome as even the most experienced developers have to spend more time tweaking their environment to adapt to changing requirements and platform upgrades. The direction for SAP, in this area, is not clear, but provides the slowest development environment of the three evaluées.

**Access to Open Source Libraries**

With a developer community of over 200K developers these libraries are rich, tested, and always improving. They provide a growing list of functions to solve both industry-specific and global app requirements. Over the last two years integration with most work-order, parts order, and time management systems has been simplified by the creation of open source APIs written specifically for the specific products. This can speed integration and delivery of the app. The IBM MobileFirst Platform has embraced open-source and allows for the easy implementation of most functions. The platform is architected to be extended by the open source community. Kony Studio is a very vanilla implementation of an Eclipse plug-in, which allows limited access to open-source. This is likely because the basic platform is written and maintained by the Kony development team. To move outside of the boundaries that Kony Studio provides requires additional products produced by systems integrators, such as QuinScape. The SAP Mobile Platform looks to embrace open source, especially with the adoption of OData, but it is still a very SAP-focused architecture and will require time to mature and integrate.

**Rich Developer Ecosystem**

A key factor because it is one of the best indicators of what experienced developers are using. It has become one of the key stopping points for resolving issues. The more issues represented, the more help that is available. Often both the original and current
developers of a product review this site, where they provide detailed code snippets to speed app delivery and re-usability. When reviewing stackoverflow.com, one finds a growing list of threads providing detailed solutions for the IBM MobileFirst Platform, including a number of positive responses. Kony, with its dependency on their India-based development team, has very few threads and many that remain unanswered for months. SAP has threads mostly geared towards integration with SAP and its various modules.

**Strict MVC Adherence**

An MVC pattern has three (3) components: Model, View, and Controller. The model directly manages the data, logic and rules of the application. A view can be any output representation of information, such as a chart or a diagram; multiple views of the same information are possible, such as a bar chart for management and a tabular view for accountants. The third part, the controller, accepts input and converts it to commands for the model or view. Strict adherence to this pattern allows a developer to quickly understand the functioning of the code and how to support and change it:

- To simplify the implementation of an MVC pattern, without having to code a full integration framework, developers rely on several open-source tools. Currently the most popular UI and architectural frameworks are Sencha, backbone.js, angular.js, and more recently Ionic, which is optimized for angular.js. These tools allow for the creation of a more native look and feel, while preserving the cross-platform capability of HTML5. Kony provides its own UI framework, which limits the capabilities to those provided, without allowing for the easy inclusion of other libraries. SAP SMP 3.0 and the IBM Platform both allow for use of open-source frameworks, with both having strong roots in jQuery Mobile. One note, the division between SAPUI5 and OpenUI5 may cause issues in the same way that Phonegap and Cordova are starting to drift apart, and a developer in one cannot easily switch to the other, reducing the available pool.

- JSON has become the standard within the mobile community for the transmission of data packages between a mobile device and the server, and the storage of that data on the device. The IBM MobileFirst Platform provides an ability to use an encrypted JSON Store within the app's MVC pattern. As a result, the developer always works internally versus the local store, and uses APIs to detect data conflicts and an adaptor to sync actions with the back-end when on line or queues it when offline. Kony prefers use of the native SQLite datastore, and SAP uses the REST specific open source OData store. The key advantage for IBM is that, with the introduction of Cloudant, it offers off-line data synch rules and capabilities out-of-the-box, where
Kony and SAP require one to construct off-line sync rules using pre-built methods. Kony does provide a Sync Server, which makes the implementation of the sync rules easier than SAP SMP. The IBM MobileFirst Platform removes the developer's concern about being offline and simplifies the code.

- Ability to authenticate users while the app is offline by trying to open the encrypted JSON Store with the user-provided password. If this succeeds, then the user provided the right password and is authenticated. This overcomes the issue of needing to be connected to authorize app access.

IBM’s ability to use open-source frameworks and strong embracing of JSON as a data store provide a distinct advantage for most developers.

**Integration with Back-end Systems and User Authentication**

A significant percentage of the cost of app development is in back-end integration, including user authentication/authorization. Most enterprises do not have a single back-end data source or a fully orchestrated and exposed set of APIs/Web Services. This requires the app to be able to provide a level of orchestration across a wide variety of back-end systems. Additionally, apps require easy access to reference data, without requiring a round-trip to a back-end system due to performance implications. For many apps, such as in the field worker use case, the requirement exists to allow for off-line transaction management. Not only does the platform have to allow for complex integration to multiple system types, but the transactions management must handle the impact of stored transactions, dirty cache, and the business processes that would surround transaction breaks.

**Authentication**

Step one in the user identification process is device validation. For Kony, this step is performed by a separate MDM product, such as Airwatch or MobileIron. What Kony provides is a separate solution for user and app validation. Kony has an integrated mobile application management (MAM) server, which stores policies such as the ability to cut and paste, and geo-fencing. The app can be set up so that it will only work when at an authorized location. Assuming the user is in a proper location, authentication with Active Directory will occur via an Identity Service, and the field worker app and any associated apps will be functional. SAP and IBM have added MDM and MAM capability through acquisition, so device, user, and app verification can

---

11 For provisioning and controlling access to internally developed and commercially available mobile apps used in business settings on both company-provided and “bring your own” devices.
happen directly within their mobile platform. The key difference is in the implementation of device and app authentication. When the field worker requests a work order using the IBM MobileFirst Platform, platform will check to see if the app has the appropriate authority. A new approach for providing a complete app security policy is the use of realms12. The IBM MobileFirst Platform is the only product of the three to have implemented realms, which provides a level of security that integrates physical and logical security policies, so they can be managed in one place. The IBM MobileFirst Platform uses a platform generated challenge request token as its primary transaction initiator. SAP does not have an integrated token management system. Additionally, for the IBM MobileFirst Platform the app processes the token via a native compiled code to prevent 3rd party attacks, which SAP SMP 3.0 does not currently have. The use of realms provides significant flexibility to match security policies and ensures acceptance or denial much closer to the device than either Kony or SAP. Often field workers have a scheduled time to complete their work and any delay in using the app limits that window. The IBM MobileFirst Platform’s generation of native code versus a web container allows for much faster processing of security checks, so the possibility of missing their “window” for completing the work is reduced.

Authorization

Each back-end system in the use case (Ellipse, InspectTec, SAP SPM, and Click) has its own authorization process. The app will need to manage five tokens to prevent the field worker from constantly having to re-enter their ID and password. The IBM MobileFirst Platform already has a token management system, which can manage these five tokens. The key is to create adapters for the back-end systems into a single realm, or have the client be a member of multiple realms. SSO would then be a property set within the login module. SAP uses a basic authentication method against non-SAP backends, which requires that the ID/PW used for SMP authentication also be used for authorization against Ellipse, InspectTec and Click. In most organizations without SSO across all platforms this would not be the case, and would require significant business change to accomplish. Kony uses the Authentication module within Sync services to create application and user access. Using the Identify Service provides a token per backend, which is stored at the Fabric layer. This limits the SSO token management required on the device but requires additional coding to create relationships between

12 A realm represents a fully configured security check that must be completed before it can allow access to a protected resource. The semantics of the checks are not limited to the authentication, but can implement any logic that can serve as protection for the server-side application resources (i.e. user authentication, device authentication and provisioning, application authenticity check, remote disable of the ability to connect to the backend gateway server, direct update, Anti-XSRF check (cross-site request forgery)).
tokens that will be required to update multiple systems at one time. The IBM MobileFirst Platform makes SSO authorization across a number of diverse backend much easier by leveraging the realm and Login.

**Transaction Management**

The use case will require the app to request all of the work orders for the field worker, along with the associated status of the assets to be repaired and the parts status for those assets. This requires access to three systems: Ellipse, InspectTec, and SAP SPM, none of which is fully REST API-enabled.

Each access will be composed of a series of coordinated web service and DB calls that support the individual systems update rules. In Ellipse, this will be done via a access to individual data table web services, which need to be exercised in a coordinated way, to compile the location and timing of the work, the tasks to be performed, sign-offs required, and any special requirements. Ellipse does not currently provide any orchestration of the web services, so that will have to be provided by the app (we assume that the enterprise does not have an orchestration layer). Additionally, in field operations, if the configuration of the work site differs from that on the work order, then the field worker submits a change to the work order, with approval from their manager, and often safety compliance.

Due to the complexity of these transactions and the possibility that the device may be offline for some parts of the transaction, a business process manager, sitting outside of the device, is needed to successfully deliver an update transaction. For all integration SAP SMP 3.0 now offers the Integration Gateway for accessing the data in Ellipse, although it is missing ODBC connectivity for any data not exposed by a web service.

OData is the SAP-preferred model for using Integration Gateway. A service implementation project, which is pre-defined Eclipse project structure, needs to be created to generate and store the artifacts needed for OData service provisioning. In addition, to manage the orchestration of the web services SMP 3.0 integrates with Fiori, but not SAP Universal Worklist (UWL). This limits functionality when delivering the work order to the device, as Fiori does not support ad-hoc workflows, substitution, personalization, and customization of task and work lists. A separate ABAP application will need to be written to handle the re-authorization of the work order.

With the Kony platform, web service orchestration would be handled in the MobileFabric by first creating the Identity Service for Ellipse, then a separate Integration Service for each web service, and finally creating a Composite Service, which will
include both Sequential and Concurrent services to manage the complexity of Ellipse access. With this model Kony can support the ad-hoc workflows, but because it sits outside of the back-end infrastructure, it will require significant custom coding to provide in-line substitution of the request recipient, based upon the values contained within the sign-off request. The substitution is needed to easily support the sign-off workflow when the transaction enters back in to the enterprise network and is linked to a desktop-based capability for management review.

When the field worker selects a work order, IBM BPM solves the issues highlighted above by providing an integrated layer of business services, within the backend network layer. The IBM MobileFirst Platform integrates with BPM to allow the BPM functions such as Fetch Task List (add, delete, update WorkOrder in Ellipse) and allows the user to essentially interact directly with the BPM processes. To complete the work order and update the parts register, the IBM MobileFirst Platform developer will need to do use Service Discovery for SAP access. By exposing and managing access to the various backends, BPM limits the amount of orchestration that needs to be done on the device, and when a backend web service changes, there will be no need for an app update, as BPM will provide the abstraction.

**Advanced Security Methods**

Unlike desktop applications, mobile apps operate on a device that can be easily lost or stolen, and often need to operate with limited or no connectivity to standard authentication/authorization mechanisms. Additionally, supporting BYOD or B2B policies prevents implementing standards where the device/user may be authorized but is not registered within the enterprise security management system. The platform needs to provide a proven mechanism for ensuring user/app/data security in unknown situations.

A fine-grained authorization framework is required to manage the diverse systems outlined in the use case. No two systems use the same model. In addition, there will need to be a differentiation of challenge/response and token management for internal field workers versus contracted resources. Kony’s model is to provide strong token-based access between the client and MobileFabric, and then the developer would create a set of SAML services and backend connectors to support the specific login details. In addition, the app is automatically validated, as registered, by MobileFabric. The challenge would be the disabling of specific functions without re-deploying the app. In the use case, if a sub-contractor’s role was changed, then they might not be allowed to update, but only read work orders. User Groups created under the Kony Management function do not operate at the transaction level. So, for the sub-contractor changing roles,
the change would be managed at the AD level and reflected in Kony Management. This would not solve the requirements of the use case as Kony Management cannot create variations at the transaction level.

The IBM MobileFirst Platform has solved this issue through their use of realms and adapter-based authenticators, which allow for transaction level authentication. To address the use case issue, identified above, a read-only Ellipse adaptor could be created and connect to the BPM transactions created for read-only access. Then the adaptor would return a read-only payload to the client and the client could then refine its UI to disable any Update button, giving the user a visual cue that they no longer had that feature available.

SAP’s reliance on Afaria for fine-grained management of devices and apps is compromised by the fact that it is not yet integrated effectively with Integration Gateway, nor is it tightly integrated with Kapsel or UI5. Changes would be required on Server, Gateway, Afaria and client, to move to a read-only Ellipse transaction.

**Application Quality Improvement, Operations, and Control**

The move to agile and FAST IT approaches for delivering an app from idea into production quickly means that rapid changes will need to be made in both business logic and technical functioning as the app evolves. These changes will be driven by feedback from the use of the app, both at a feature level and within the various app functions. Mobile platforms need to provide the ability to gain insight into how a user is using the app and how the app is behaving. Analytics could be used to solve the lone worker issue, where location tagging and pushing emergency instructions would solve a significant issue where the worker is remote. This information should be easily captured and correlated for use by the agile team and the business owner to increase the app’s effectiveness.

For the most part, all three products have strong testing capabilities during the initial development phase, but the differences occur when the app is moved to production. At that point app analytics become critical in determining app performance, usability, and stability. For all three products, the Server Platform becomes the key collection point for data and monitoring. The key differences lie in the richness of the embedded analytic functionality within each platform. Kony offers a closed system with monitoring capabilities primarily around the proper functioning of the Integration Services that have been set up. In the use case, to monitor the end-to-end transaction would require several monitoring products outside of Kony’s Console. In addition, Kony limits the ability to enhance the console with open-source add-ons. When the user states “my app
is not working”, the triage process will be slowed by the lack of a central consolidated console/logging facility. Another important function in a field worker app is the ability to integrate device and app specific operational analytics. The use case represents the replacement of several paper-based processes. If the app does not make the worker’s job easier, they will stop using it. Google Analytics can be used to provide the information for a Kony built app, but that data cannot easily be consolidated with other operational data. SMP 3.0 has integrated analytics on its roadmap, but does not currently provide it. This is a clear differentiator for the IBM MobileFirst Platform with the Platform Operational Analytics tool, which can monitor the following:

- Crash events of an application on iOS and Android devices (crash events for native code and JavaScript errors)
- Interactions of any application-to-server activity (anything that is supported by the IBM MobileFirst Platform client/server protocol, including push notification)
- Server-side logs that are captured in traditional the IBM MobileFirst Platform log files
- Custom charts in the operational analytics dashboard that get critical insight about your specific app such as “How many times did the user press the submit button?”
- Security related analytics in the operational analytics console such as monitoring failed logins, visualizing the number and reason for authentication failures, and visualization of other critical security information and configuration

This provides a single platform for end-to-end monitoring, with the exception of integration with BPM. The open-source capabilities of the IBM MobileFirst Platform would allow for coding a connector to bring in BPM data and augment the platform, but that is not available as of today.
Conclusions and Recommendations

While each solution has a unique place in the market, the overall conclusion related to these technologies is clear: when selecting a platform to support rapid, but supportable building and deployment of enterprise mobile apps, as outlined by an organization’s mobile strategy, the IBM MobileFirst Platform provides a significantly stronger option than the competition.

IBM’s overall vision for mobility in the enterprise is complete and relatively mature. The IBM MobileFirst Platform provides distinct advantages in:

- **Development** - with a stronger foundation in Open Source than Kony or SAP, IBM offers developers a much richer canvas for rapid development of truly business value-based apps. Reusability, of both internally and externally developed functions, directives, promises, plug-ins, etc. will shorten the overall development lifecycle and allow for enterprises to handle the “tsunami” of mobile app ideas by rapidly creating PoC’s and evaluating business value before hardening for service transition. The IBM MobileFirst Platform makes all of that available by limiting the “guardrails” employed by both Kony and SAP

- **Security** - with a sophisticated authentication and authorization built around IBM Trusteer, and fine-grained policy implementation, which is becoming increasingly important as more data is exposed to a growing variety of known and unknown users

- **Transaction Creation** - with greater out-of-the-box and easy to support connectors and orchestration simplifying the significant integration cost and delivery time, and making it much more supportable during the app’s lifecycle

- **Quality Improvement** - with strongly integrated app analytics and continuous integration testing capabilities, the IBM MobileFirst Platform outpaces Kony Integrated Solution and SAP SMP 3.0 in areas where IBM has traditionally been strong. The number of factors impacting both the use and performance of mobile apps requires a level of commitment to maintaining quality that the IBM MobileFirst Platform demonstrates

From device management, to analytics, to security and testing, to IBM’s already world-class integration platforms, IBM provides superior software and a complete mobile platform. IBM’s MobileFirst Platform provides a set of tools for repeatable processes, and components based on open-source technologies in all cases. As a result, finding developers to use the IBM MobileFirst Platform’s tools is an achievable goal. This is a
major advantage over both SAP and Kony, which have a limited pool of resources available and require substantial training for new or transitioning developers.

Despite SAP’s insistence that its platform can be productive independent of its SAP foundation, Edison has found that the SAP platform is best suited for existing SAP clients who are leveraging existing SAP business objects. SAP has made some strides to be a more complete platform, joining its Syclo and SUP platforms under one roof. Its industry solutions, the legacy Syclo offerings, are strong in their focus areas. They are, however, very much point solutions. Furthermore, the SAP mobile toolset continues to be overly complex, and when it comes to operational analytics and a complete platform for consumer facing apps, SAP falls short. Its strength remains in surfacing SAP data to a mobile device and offline capabilities for specific use cases. But as a complete platform, SAP leaves something to be desired.

Kony has an interesting value proposition, in that it supports almost every device that is on the market in a manner that is sustainable. This is very compelling for a consumer-facing app, and the Kony platform seems to have the most robust way to handle this. However, Kony has a “closed” user experience - in order to be successful with Kony, users are very reliant on Kony through all phases. Development support will likely be required. And while Kony is used in the enterprise, especially with its “industry solutions”, Edison does not see the platform as something that can stand on its own without consistent support from Kony. So even though the upfront adoption of Kony may seem like a short runway, sustaining Kony as a platform to is likely to require a great many resources. Kony has a significant amount of adapters for enterprise systems, but they are not of an “open-source” nature, as IBM has executed. The Kony platform is certainly robust, providing a development environment, app store and analytics, as well as security. Upon review, Edison identified the most problematic aspects of Kony as issues with ease of use and with its open capabilities, as well as Kony’s frequent requirement for users to adapt existing knowledge to the Kony platform, instead of vice-versa.

In conclusion, Edison has found that, while SAP and Kony both have valuable places in the market, they do not compare to the IBM MobileFirst Platform in sustainability or in adaptability. This conclusion is based on SAP’s specific target audience and Kony’s maturity in the space. Through the sample use case, Edison was able to quickly and easily build the basic functionality of an app in a manner that provides repeatability and robustness. The methodology Edison followed and its findings from this study could be applied to a wider area, including connectivity to enterprise systems, partner systems or any other approach based on the open architecture and adaptor framework. The IBM
platform provides the full set of tools needed for mobility, not just a handful of niche aspects that enable one to merely build a mobile app.

Whether you are building B2E, B2B, or B2C web, hybrid, or native apps, the IBM MobileFirst Platform provides the comprehensive and developer-friendly set of tools to do this in a mature, enterprise fashion, with a vision toward the future.