



Mobile “systems of interactions” driving business innovation

Key technology trends motivate engaging applications

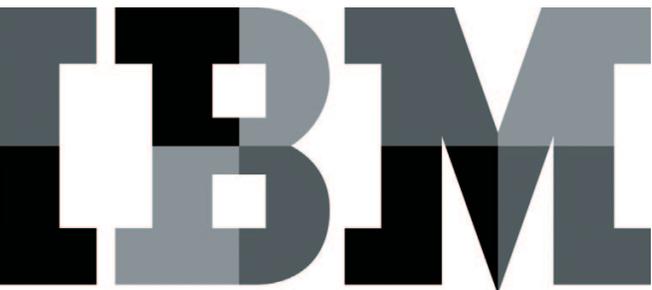
Contents

- 2 A systems of interaction example
 - 2 A formula for designing engaging systems
 - 4 Unique challenges for development of engaging applications
 - 7 Why IBM?
 - 8 For more information
-

Across the globe, more people are using mobile devices, which are increasingly user-friendly and intuitive, as their primary means of obtaining information and requesting services over the internet. And most enterprises realize that the users of their business applications have shifted from traditional personal computers (desktops and laptops) to mobile devices (smart phones and tablets) as a means to access web-based information. This applies whether the intended user for the application is a direct customer, employee, or business partner.

This crucial shift in user behavior has motivated enterprises to develop mobile channels for their existing business applications, and to plan for new kinds of applications that can exploit the unique characteristics of mobile devices. While there certainly is value in producing a mobile app user interface for an existing business application, the users of mobile applications have come to expect more from their mobile experience.

Compelling and successful mobile apps create an experience that fully engages the end user. So-called “systems of interaction,” these apps anticipate the desires of the user and take full advantage of the rich collection of data that the new devices offer. Plus, they motivate changes in the business processes used to support application users. Systems of interaction encompass both the engagement with the end user as well as the context that existing systems-of-record IT investments enable.



In short, they open up huge new avenues for innovation in business and propel new ways for businesses to interact with their various stakeholders.

The avenue for innovative user engagement isn't limited to mobile phones and tablet devices. The term "omni-channel" has been coined to refer to applications that offer end user engagement across a spectrum of devices, from phones to tablets to PCs to kiosks to automobiles, and many more forms of human-technology interaction. Each of these different application end points ("channels") offers unique characteristics that enable valuable interaction with the user under circumstances best suited to that specific channel.

This paper describes IBM's vision regarding mobile systems, known as "systems of interaction," including business and technical challenges, as well as implications for the application lifecycle these systems comprise.

A systems of interaction example

For example, consider the experience of an airline customer, who uses the airline web site two days before the flight, confirms the seat assignment, and requests an upgrade. Less than one day before the flight, this customer uses the airline's mobile app on their iPad Mini to check on their seat upgrade and "check in" for the flight (providing greater assurance to the airline that the customer will actually take the flight). Then, within a few hours of the flight, this same customer uses the airline app on their smart phone from the airport to check the gate assignment and double check the departure time for the flight (and check again the status of their upgrade request). They might also check if there is an airline lounge at the departure airport and also check on their current frequent flyer mileage totals.

The relationship established with this customer can continue in several ways. During the flight, this customer uses one of the airline's devices to buy WiFi access to the internet. They may check the updated arrival time for this flight and email a colleague to confirm their plans after arrival. The interaction doesn't stop there. At the destination airport, the customer can use the airline's system-of-interaction to track luggage, check on ground transportation, and navigate to a meeting. The connection with the customer can go on and on for days. They may need to confirm (or change) the return flight, or confirm frequent traveler mileage status, or plan new trips. These activities can be conducted on a wide variety of devices to connect to the airline IT systems and services at any time, day or night.

This is a system of interaction. It is ongoing, supporting far more than a single transaction. It requires context to be effective. It is intelligent, and it learns via feedback from the user. It is the "killer app" of the 21st century.

A formula for designing engaging systems

IBM has devised a formula for creating compelling new systems of interaction that drive business process innovation. Here's the equation:

$$\textit{mobile apps} + \textit{interaction characteristics} = \textit{business process innovation}$$

where:

- "Mobile apps" are smartphones, desktops, vehicles, and other devices
- "Interaction characteristics" are features applied to the mobile app
- "Business process innovation" refers to new capabilities for detecting, enriching, perceiving, and acting

The formula starts with mobile first. The **mobile app** is what interacts with the user and can be deployed to more than smartphones. Modern “mobile apps” have deployment target devices as diverse as cars, TVs, desktops, body monitors, and many more.

Interaction characteristics are applied to the mobile app. We define the compelling interaction characteristics as:

- **Omni-channel:** available across many different classes of end point device
- **Context and social aware:** to automatically tune the interaction to the place and relationships involved
- **Connected to systems of record:** such as existing business services and data sources, as well as cloud-hosted third party services
- **Experience driven:** able to learn and adapt to specific end user responses
- **Highly instrumented:** so that data for analysis can be obtained
- **Rapidly revised:** so that app developers can continuously improve interaction

These characteristics are applied by the system of interaction to yield **business process innovation**. The methodology that drives business process innovation must:

- **Detect** opportunities to engage customers and employees
- **Enrich** interaction context with historical data and trends
- **Perceive** via “in-the-now” dynamic interaction context from location, time, social media, and other events, and...
- **Act** on the insights gained to enable positive business outcomes.

Hence, the design characteristics for using mobile technologies to drive business innovation are simply: *Detect, Enrich, Perceive, and Act*. That’s it. All of these characteristics are evident in the previous airline passenger example.

Let’s consider another quick example to illustrate our formula for systems of interaction: in this case, an app for hailing taxis

As a consumer, you download the app and set up a personal profile including preferences and payment information. When you need a cab, you press “Hail Cab” within the app. Your current location is captured, and in a few seconds you are informed that a driver named Mike is heading in your direction. This cab can be available if you confirm within the next minute. Perhaps some social factors can help influence your action here. Have your friends had a good experience using Mike for taxi service? When you confirm, you see Mike’s actual location as he approaches, along with an estimated time of arrival (ETA). And Mike’s cab location is updated even if Mike takes a wrong turn along the way. When he arrives at your location, you get in the taxi cab and payment is established to Mike upon conclusion of the ride.

Let’s see how our methodology might apply to this example scenario.

- **Detect** – customer loads the app on their smart phone
- **Enrich** – get customer profile, favorite cab, and favorite destination
- **Perceive** – location of the customer, location of the closest cab, other nearby friends heading to the same destination
- **Act** – connect the cab driver and the customer through notification and establish secure channel to pay through the app

This app meets all of the criteria for a good system of interaction, providing all of the dimensions to deliver compelling value to the user.

Unique challenges for development of engaging applications

The creation of systems of interaction involves some unique requirements and challenges.

A recent paper from IBM titled “A mobile application development primer”¹ provides a broad overview of IBM’s recommendations for planning, developing, testing, and deploying mobile applications. Some of those recommendations are echoed in the following paragraphs, and the reader can refer to the earlier work for more details.

Form factors and user input technology

The first and most obvious aspect of mobile applications is that the form factor for display and user interaction is significantly different from what is used by other forms of software.

A smaller form factor means that the amount of data displayed to the end user, and layout of that data, needs to be tuned to the “real estate” available on the device. Significantly less data may be displayed on some devices and therefore it must be exactly the “right” data (most relevant to what the user needs at that point in the application). This variety in form factors motivates the “responsive design”² approach for application presentation, where the same application takes advantage of the display resources available on the device where it is running.

Another obvious physical difference for mobile applications is that the mechanisms for user input are different. Mobile devices have pioneered the use of non-keyboard “gestures”—e.g., touch, swipe, and pinch—as an effective and popular method of user input. Gestures must be planned for and supported for a satisfying mobile application user experience. In addition to tactile user input, mobile devices are a natural target for voice based user input. In fact, the traditional keyboard typing form of user input is probably the least effective and least popular mechanism for input to the new systems of interaction.

Besides gathering input directly from the end user, new systems of interaction have the capability to receive input from other sources, such as geo-location from the GPS component of the device and image information from the camera typically built into the device. These forms of input make mobile apps more powerful and useful than applications with a more limited array of input possibilities, and they must be considered during mobile application design and development.

Usability and user interaction design

There are several reasons why usability and user interaction needs greater attention in designing mobile applications.

The difference in form factors and user input methods is one. It is much more difficult and time consuming to plan how to display only the data that is precisely necessary than it is to simply display all possible data and let the user sift through it visually for what is needed. Displaying all possible data is rarely an option for the mobile app designer, who has to consider the screen’s limited real estate. The best applications designs often present a broader scope of data with multiple layers of detail, and use a progressive discovery approach that allows the user to drill down into incrementally greater levels of detail focused on specific items.

The rich variety of input methods available on mobile devices is another reason that early design work must identify and leverage more efficient ways for input data to be delivered than the simple “just type it in a form” design, which is a default for traditional web and PC applications. Designers must avoid extensive keyboard typing for mobile apps in order to reduce user frustration (with drastically smaller touch keyboards and lack of traditional typing feedback). Yet, identifying non-keyboard ways in which information can be gathered and delivered to the mobile app is a significant design challenge.

And there is still a more subtle reason for paying extra attention to the mobile app design effort. The way in which end users interact with mobile devices and the applications running on them is different from how they interact with stationary PCs, even laptops. Mobile-device users typically hand-hold the device while interacting with the immediate circumstances of their physical situation. Mobile users typically cannot concentrate on the mobile app for very long before switching attention to the physical surroundings. The interaction model for users of mobile apps is short, interrupted, and “bursty” (meaning that they need to very quickly complete the application task before switching attention).

All of these factors drive the need for applying user-centered design very early in the mobile app development project. Ideally these usability and design considerations should be codified in the requirements for the mobile application, and then linked to the later stage development deliverables, along with the tests that validate that the user interaction and “consume-ability” of the app is as satisfying as possible.

Choice of implementation technology

There is a spectrum of implementation choices for mobile applications on the market, and no one answer is perfect for all implementations—each choice has its advantages and disadvantages. So the challenge for mobile development teams is to understand the trade-offs between the technologies, and make a choice based on the specific application requirements. (The IBM mobile development primer paper referenced earlier includes a description of the implementation choices, along with a comparison table).

The choice of implementation technology for a mobile project impacts other decisions related to the application’s development, including:

- Limiting choices for development tools.
- Team roles and structure.
- How the application is tested and verified
- How the app is distributed and delivered to the end user.

So, the choice of implementation approach for a mobile application is a crucial and early stage decision needs to be made very carefully.

Native application implementation

A “native” implementation means you are writing the application using the programming language and programmatic interfaces exposed by the mobile operating system of a specific type of device. For instance, a native implementation for an iPhone will be written using the Objective-C language and the iOS operating system APIs that Apple supplies and supports.

Native application implementation has the advantage of offering the highest fidelity with the mobile device. Since the APIs used are at a low level and are specific to the device for which the application is dedicated, the application can take full advantage of every feature and service exposed by that device.

However, native mobile app implementations are completely non-portable to any other mobile operating system—e.g., a native Apple iOS app must be totally rewritten if it is to run on an Android device. That makes this native implementation a very costly way of implementing a mobile business application.

Web applications

Newer smart phones and tablets come with advanced web browsers pre-installed, and it is relatively easy to implement a standard-web, mobile business application with special style sheets to accommodate the mobile form factor and approximate the mobile device “look and feel.” Mobile applications implemented using this approach support the widest variety of mobile devices, since web browser support for JavaScript and HTML5 is fairly consistent. There are several commercial and open source libraries of Web 2.0 widgets that help with this approach, and the web programming model for mobile application implementation is well-suited for most enterprises, since their developers are already trained in the languages and techniques for web application development.

The disadvantage of pure web application implementation is that such apps have no access to functions/features that run directly on the mobile device, such as the camera, contact list, and so forth.

Hybrid mobile application implementation

Hybrid mobile application implementation is a compromise between pure native implementation and pure web implementation. You write the mobile apps using industry standard web programming languages and techniques such as HTML5 and JavaScript, but you package the app into a natively installable format that is distributed via the app store mechanism.

Hybrid apps are linked with additional native libraries that allow the app to have access to native device features from the single application code base. Because the bulk of a hybrid application is implemented using device-agnostic technology, most of the code for the application is portable and reusable across many different mobile operating systems. However, small segments of native code can also be integrated with the hybrid app, which means that the developer can decide how much of the app implementation shares a common code base (using the web technology) and how much is device-specific customization (written in native code).

Mobile application build and delivery

Because businesses want to deliver mobile applications into the market quickly, mobile development projects typically have extremely aggressive time lines. Inception-to-delivery time frames of a few months are common. The pressure to deliver mobile apps quickly results in the adoption of agile development methods for most mobile projects.

An important element in agile development practices is continuous integration and builds. Application changes that are delivered by developers need to be processed immediately for all of the mobile operating systems on which the application is required to execute. If the mobile application is a hybrid or

native implementation, several different builds of the application need to be triggered each time a change set for the application is delivered by a developer. The build setup and configuration for each supported mobile environment will be different from the others, and it is most likely that a small “farm” of build servers will need to be provisioned and available to handle these builds of the mobile application for multiple operating systems.

Testing

Testing poses another huge challenge for mobile application development, because it represents a step-jump in complexity and cost over more traditional applications. Unlike traditional PC and Web applications, the range of potentially supported mobile devices and release levels is staggering. Test matrices for mobile projects commonly contain hundreds and even thousands of permutations of device, mobile OS level, network carrier, locale, and device orientation combinations.

There are more variables for mobile testing that aren’t relevant for other kinds of software. The same device model may function in a subtly different way when connected to a different carrier network. And the quality of the network connection can have profound impact on the behavior of a mobile application. Even the movement of the mobile device itself may be an important factor in the behavior of the application (some applications specifically exploit device movement).

The majority of mobile apps are based on multi-tier architecture, with the code running on the device itself serving as the “front-end” client to data, and the services supplied by more traditional middle-tier and data center representing the “back-end.” Effective and comprehensive testing of mobile apps requires that all tiers of the application be addressed, not only the code on the mobile device. The set-up and availability of test versions of the middle tier and back-end services can present very large cost and complexity challenges for the testing of mobile applications.

Many mobile projects start by using manual testing approaches, which is the quickest way to begin testing. But you have to buy all of the various mobile devices that you plan to support with the app, and pay someone (likely a team of people) to tediously go through a written script of instructions describing the tests on each of those devices for every build of the application. While manual testing serves an important purpose in providing crucial usability feedback for the app, it is extremely expensive and inefficient.

As an alternative, there are mobile app testing solutions that rely on running an agent program on the device for interaction within an automated execution. This approach has the flexibility of using either real physical devices or emulators for testing, with the added efficiency of automation. However, the test team bears the costs of setting up the devices to be tested and installing the test agent on them.

Why IBM?

The IBM MobileFirst approach for mobile enterprise application development combines years of experience in the field of general enterprise software development processes with new tools and techniques specific to mobile devices and their underlying software foundations.

With deep expertise in the design and deployment of enterprise software across a wide array of industries, the IBM approach offers a proven solution for the development needs of mobile application projects.

Mobile development lifecycle management

IBM offers a comprehensive lifecycle management platform that supports collaborative tasks and helps link the various artifacts developed over the course of the product lifecycle. This solution also enacts delivery workflows and provides task management capabilities to effectively run the mobile application development project. It is augmented with integration of mobile-specific capabilities in the phases of the project lifecycle where such capabilities are needed.

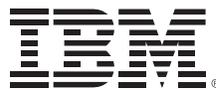
Agile team collaborative development tools allow the definition of multiple short iterations within the overall project. Work items can be easily moved from the project backlog to a particular iteration plan.

As developers edit files inside their mobile app code development tool integrated with the software version control system, a change set is automatically updated and maintained. Developers don't have to do anything special to produce the change set other than edit the files that they need to work on.

Change sets can be shared among team members before being fully integrated with the main code stream. This means a change set created by the web service developer that alters the format of data supplied by the web service can be shared with the UI developer working on the logic that displays the new data without the rest of the team being affected. When both UI code changes and web service code changes match and are deemed ready, they can be integrated in one synchronized task into the mainline code stream for the rest of the team to pick up and use.

IBM MobileFirst platform delivers a mobile application runtime

Collaborative lifecycle management capabilities integrated with development tools for mobile apps is important, but not all the challenges in developing enterprise class mobile apps can be addressed by tools and practices alone. In order to deliver the capability for a single common mobile application programming model, we offer the IBM MobileFirst platform. Our comprehensive mobile business application development solution combines powerful team development capabilities embodied in IBM Mobile Development Lifecycle Management with the mobile tools and runtime capabilities delivered in the IBM MobileFirst platform.



As more and more enterprises in all industries realize the need for compelling systems of interaction for their business applications, they need the enterprise-class approach to application development that IBM has established. Our solutions enable mobile-specific tools and technology to be used with the same efficiency and rigor as other kinds of enterprise application development.

For more information

To learn more about the IBM MobileFirst initiative, please contact your IBM representative or IBM Business Partner, or visit the following website: ibm.com/mobilefirst/us/en/

Additionally, IBM Global Financing can help you acquire the software capabilities that your business needs in the most cost-effective and strategic way possible. We'll partner with credit-qualified clients to customize a financing solution to suit your business and development goals, enable effective cash management, and improve your total cost of ownership. Fund your critical IT investment and propel your business forward with IBM Global Financing. For more information, visit: ibm.com/financing

About the author

Leigh Williamson is an IBM Distinguished Engineer who has been working in the Austin, Texas lab since 1988, contributing to IBM's major software projects including OS/2, DB2®, AIX®, Java, WebSphere® Application Server, and the Rational® portfolio of development solutions. He is currently a member of the Chief Technology Officer team, influencing the strategic direction for products that address the needs of software development teams, with a primary focus on tools and best practices for mobile application development. Follow Leigh on Twitter at [@leighbawillia](https://twitter.com/leighbawillia).

© Copyright IBM Corporation 2013

IBM Corporation
Software Group
Route 100
Somers, NY 10589

Produced in the United States of America
May 2013

IBM, the IBM logo, ibm.com, and Rational are trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at "Copyright and trademark information" at ibm.com/legal/copytrade.shtml

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

This document is current as of the initial date of publication and may be changed by IBM at any time. Not all offerings are available in every country in which IBM operates.

THE INFORMATION IN THIS DOCUMENT IS PROVIDED "AS IS" WITHOUT ANY WARRANTY, EXPRESS OR IMPLIED, INCLUDING WITHOUT ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND ANY WARRANTY OR CONDITION OF NON-INFRINGEMENT. IBM products are warranted according to the terms and conditions of the agreements under which they are provided.

¹ Find this paper at: https://www14.software.ibm.com/webapp/iwm/web/signup.do?source=swg-rtl-sd-wp&S_PKG=500028155

² For users curious about this phrase, the best starting point is probably the Wikipedia entry: http://en.wikipedia.org/wiki/Responsive_web_design



Please Recycle