Key Approaches for Mobile Success

Xamarin
Table of contents

01  Why mobile matters
02  Users demand native experiences
04  The diverse device landscape
06  Approaches for delivering mobile apps
06  Development approaches
07  Xamarin Platform
14  Mobile quality approaches
16  Xamarin Test Cloud
22  App monitoring approaches
23  Xamarin Insights
26  Accelerators for going mobile
27  Xamarin University
26  Enterprise class support and services
29  Case study: Mobile banking
Why mobile matters

Mobile devices are the fastest-growing and most impactful innovation of our time. By 2018, the majority of Internet users will access the Internet via tablet or smartphone, using traditional PCs for complex tasks only. Users will move even further from their PCs as “voice, gesture and other modalities grow in popularity.”

A new generation of context-aware mobile apps is transforming business processes; ubiquitous devices access a wealth of personal and corporate information, and, as demonstrated by wearables and the Internet of Things (IoT), have the power to redefine and enhance our environment. Successful businesses are creating uniquely mobile experiences to enhance productivity and engage with customers in new ways—from redefining retail, to making the full power of enterprise data available to employees at any place and time.

As of June 2014, mobile apps dominate the proportion of time spent online.

85% of Facebook’s users access the site on mobile devices, with more than one third exclusively using mobile. The money is following suit: in 2014, mobile provided 69% of Facebook’s $3.6 billion advertising revenue.

Mobile is projected to account for 46.6% of global e-commerce by 2018.

By 2016, 70% of the mobile workforce will have a smartphone. With BYOD employees purchasing half of them; 90% of enterprises will have two or more platforms to support.

Companies that embrace mobile today will expand their business’ value and reach. Those that don’t will lose relevance as more agile competitors jump ahead.
Users demand native experiences

Whether accessing order history during a sales call or checking a flight status, users expect information to be instantly accessible and presented in a way that makes the most of a device’s capabilities and form factor. Apps that fail to meet these demands are abandoned.

Performance, reliability, design, and usability are the main factors that determine user engagement. With respect to these factors, a fully native app has conclusive advantages over non-native or partially native alternatives.

Essential characteristics of native apps

- **Native user interface**
  Native apps are built with standard, native user interface controls according to each platform’s specific design guidelines.

- **Native API access**
  Native apps have direct access to the full spectrum of functionality exposed by the underlying platform and device, such as fingerprint authentication, payments, health sensors, and other speciality hardware.

- **Native performance**
  Native apps offer optimal performance, including smooth scrolling, stutter-free interaction, and gorgeous transitions, leveraging platform-level hardware acceleration to deliver unmatched responsiveness.
“The results from our new field sales app are phenomenal. Our sales people love the app and are able to engage customers and close sales more effectively. Key to the app’s success is the beautiful, fast user experience made possible by Xamarin.”

Kim MacDougall
SeniorCapability Manager, Kimberly-Clark.
The diverse device landscape

The mobile computing landscape is far more fluid and fragmented than it was during the PC era, and is changing much more rapidly as well. From 2009 to 2014, Android vaulted from a 4% smartphone market share to a whopping 76.6%. While Android dominates device volume, Apple generates more sales and higher revenue.8

![Graph showing the share of global unit sales from 2009 to 2014 for Android, iOS, Windows Phone, BlackBerry, and Others. Source: IDC, Strategy Analytics, BI Intelligence Estimates.](image)
With the increase in wearables and IoT devices, cross-platform development is an essential element of a successful mobile strategy in this highly diverse ecosystem.
Approaches for delivering mobile apps

Companies need a cross-platform mobile strategy to be resilient in the face of unprecedented innovation. They need a sustainable way to build apps that keep up with the proliferation of new devices and capabilities. They need efficient ways to support and maintain apps. And they need to be more agile and deliver business value faster. Consumers and employees have a tremendous level of choice when it comes to which devices and apps they use—and which they abandon—so developers must ensure that apps are engaging and work as expected, every time and on any device.

Development approaches

Whether building consumer apps or apps for employees and partners, early choices about mobile technology and methodology can make or break a mobile strategy. These choices have huge implications in the creation of a sustainable mobile strategy, including the ability to staff teams, reuse existing tools and code, and support future use cases and devices.
Key approaches for mobile success

Triple implementation

One approach to supporting multiple device platforms is staffing individual iOS, Android, and Windows development teams to create independent Objective-C, Java, and C# apps for each mobile platform. This approach has many drawbacks:

- **Siloed development environments (multiple languages, tools, and teams)**
- **Triple the effort to add or change features**
- **Slow release cycles**

Companies attempting this siloed approach must manage separate languages, tools, and teams and endure slow development cycles caused by implementing each feature three times. Over time, maintaining feature parity becomes ever more difficult, slowing innovation to a snail’s pace.

Write once, run anywhere

Many cross-platform framework vendors advocate a magical-sounding approach known as Write Once, Run Anywhere (WORA): write an app once using a single codebase, then use an app generator to output platform-specific apps from that codebase. Apache Cordova and Ionic are examples of this approach. WORA pitfalls include:

- **Poor user adoption due to poor user experience**
- **Poor app performance due to the use of intermediate languages**
- **Limitations on the device APIs developers can use**

WORA promises to eliminate redundancy in development, but it comes at the expense of performance, user experience, and up-to-date support for new devices and capabilities. By creating a platform abstraction layer that developers code against, WORA apps elide many platform-specific design details, and the abstraction layer is slow or unable to accommodate new device categories such as Apple Watch and Android Wear. WORA apps disappoint users, look and feel dated, and put mobile projects at risk.

The approach’s overhead slows innovation, which hurts both the company and its end users.

A framework that abstracts away platform-specific details can’t express the full range of the underlying platform’s design language.
Xamarin Platform

The Xamarin approach to building cross-platform native apps combines the essential characteristics of native apps—native UI, native performance, and native device access—with the efficiency and time-to-market advantages of code sharing. In addition, Xamarin’s approach makes it possible for businesses to use their existing teams, tools, and code to go mobile.

Xamarin’s model for cross-platform development

- **Powerful, native experiences.** Build rich apps that deliver optimal performance and access all functionality of the underlying platform and device.
- **Start now with existing teams.** C# and Visual Studio developers have no need to learn new languages or tools, so they can be productive with Xamarin from day one.
- **Integrate with existing enterprise architecture.** Leverage the wealth of existing .NET libraries for interacting with the web and other data services and share C# business logic across client and server.
- **Bring existing code mobile.** Bring C# codebases developed over years to mobile overnight, avoiding the pitfalls of porting code to niche mobile languages like Objective-C or Swift.

Up-to-date native capabilities

Xamarin apps look native because they are native. Anything developers can do in Objective-C, Swift, or Java, they can do in C# with Xamarin. Our unique binding technology allows C# to call the same APIs and use the same UI controls as apps built in platform-specific languages. All of the underlying platform features are exposed, including fingerprint authentication, payments, health sensors, Bluetooth, NFC, and other hardware and OS services.

Xamarin’s binding technology enables same day support for new platform and device features. From iOS 5 through iOS 8.1, Xamarin provided updates within 12 hours of Apple’s public releases, bringing same-day support for new platform capabilities like WatchKit and larger iPhone screen sizes. With Xamarin, apps can always access the latest features that users expect from day one.
Reach 2.6 billion devices with existing teams and code

Xamarin provides a straightforward path for bringing existing C# skills, teams, tools, and code to the world’s most popular mobile platforms. Companies that introduce Xamarin to existing .NET teams become productive on mobile within days, foregoing the arduous process of staffing new platform-specific teams.

Xamarin also gives developers a practical path for extending the reach of their existing .NET skills and code to modern mobile environments. Companies that have existing desktop and web apps built with C# can use Xamarin to bring much of the underlying app logic to Android and iOS. With Xamarin, organizations can leverage their existing C# investment instead of starting from scratch.

Companies that initially built apps in Objective-C and Java are turning to Xamarin as a scalable solution for reaching new and evolving platforms without sacrificing user experience or performance. No other mobile development solution maintains these standards while allowing code sharing across platforms.

Code sharing advantages

When building software with Xamarin, developers select one of two architectures, depending on the app being developed:

**Xamarin.iOS / Xamarin.Android**

With Xamarin.iOS and Xamarin.Android, developers implement separate user interfaces for each platform in C#, while sharing app logic (e.g., models, validation, web services, persistence, enterprise integrations), also written in C#. Developers can expect to share around 75% of their code with this approach, making apps quick to develop and easier to test, while retaining the ability to express any platform-specific UI.

**Xamarin.Forms**

Xamarin.Forms enables developers to implement their app’s UI in shared C# code, further increasing code sharing to 90% or more while still delivering a native experience. Developers write each screen of their app once in C# or XAML. At runtime, these screens and their controls are mapped to native UI elements, creating a native user experience on each platform that adheres to the design principles and user expectations of that platform. Developers can still access native UI views or functionality on each mobile platform if necessary, making Xamarin.Forms the industry’s most versatile cross-platform mobile UI framework.
“With Xamarin, we were able to mobilize ten million lines of existing .NET code, greatly accelerating our timeline. We were glad to take the code we’ve built over the years and move it to a mobile environment in such a short amount of time.”

Nick Skyes
Director of Mobile Projects, Xactware

Comprehensive IDE support

Superior tools make developers more efficient, allowing them to focus on quality and innovation. Xamarin provides market-leading IDE integration with support for Visual Studio and Xamarin Studio, Xamarin’s own IDE. These intuitive, integrated tools give developers everything needed to design, develop, debug, and deploy great mobile apps.

Xamarin’s integration with Visual Studio makes building iOS and Android apps in Microsoft’s preeminent development environment possible. Visual Studio users finally have freedom to write code for all major platforms under one roof using the programming language they know and love, with convenient access to Microsoft’s ecosystem of extensions and tools like Resharper and Team Foundation Server.

Xamarin also comes with Xamarin Studio, a free, powerful, and highly usable cross-platform IDE, available for OS X and Windows. Xamarin Studio combines a strong foundation of general-purpose C# development with specialized mobile-development features and tightly-integrated support for Xamarin’s other tools and services. The result is an IDE tailored for building cross-platform apps in C#, offering a balance of power, extensibility, performance, and ease of use.
Xamarin Studio and Xamarin’s Visual Studio extensions include visual designers for iOS and Android. These design tools offer drag-and-drop UI construction, are tightly integrated with the Component Store and the rest of the Xamarin platform, and operate on the same file formats as the design tools that Apple and Google provide.

Xamarin Sketches provides a live coding environment that executes code as the developer types, providing a live display of the data being produced. Sketches is an ideal way to learn new APIs, prototype ideas, and quickly iterate on designs. It’s also a great place to study snippets from an open solution in an isolated live coding environment and an incredibly powerful tool for debugging, since a developer can see the results of code as it’s written.

“Mobile is an increasingly strategic channel for us to engage with our readers. Xamarin made it possible for our team to build an app much more quickly than we otherwise could have.”

Ian Rosen
VP & General Manager, MarketWatch & Smartmoney
The **Xamarin Android Player** lets developers simulate apps significantly more quickly and easily than the stock Android emulator provided by Google. Hardware virtualization and accelerated graphics provide fast startup times and fantastic performance. In addition to standard controls such as volume, navigation, and rotation, Xamarin Android Player can simulate battery level and GPS location. It uses ADB, so existing Android tools can easily communicate with this high-performance simulator. Installing apps or Google Services is accomplished by dragging and dropping APK or Play Services files onto the Xamarin Android Player window.
Xamarin Component Store

The Xamarin Component Store is a catalog of free and paid components that add beautiful UI controls, popular .NET libraries, and third-party web services to apps with a click and a few lines of code. If an app requires complex features like barcode scanning or a signature pad, developers can simply add a pre-built component to their app rather than implementing from scratch.

Popular components include controls like login forms and progress overlays, fundamental libraries like Json.NET and RestSharp, platform-as-a-service providers including Parse and Azure, official SDKs from Facebook and Dropbox, and enterprise connectors for IBM MobileFirst, Salesforce, and SAP.

The Component Store is built into Xamarin Studio and Xamarin for Visual Studio, so developers can find and manage components from their IDE of choice. Xamarin curates components to ensure reliability and that documentation and sample projects are included.

The Component Store, coupled with our Nuget support, brings more than 20,000 libraries to mobile development with Xamarin.
Popular apps like Instagram, Dropbox, and Facebook Messenger have shaped user expectations for all apps: they must be beautiful, fast, and frequently updated. User sessions often last just seconds, so a high-performance app that delivers information quickly can mark the difference between a highly-rated app that attracts new users, and a poorly rated app that drives away audiences and generates negative word of mouth. Employee-facing apps are judged by these same experience standards; employees will abandon ugly, slow apps, breaking the productivity promise of a mobile workforce.

The huge variety of form factors, OS versions, and OEM configurations—known as “fragmentation”—makes guaranteeing app quality and consistency across a wide range of devices extremely difficult with standard methods for monitoring mobile quality.
Simulator-only testing

By testing an app on a simulator, developers can execute their apps in a runtime environment without leaving their development environment. This approach is convenient and reduces costs and setup time because it doesn’t require actual mobile hardware. This is the most basic way to spot check functionality. However, basing a production mobile quality strategy just on using simulators has significant downsides, whether using them in a manual or automated testing environment:

- Unrealistic CPU, memory, and performance caused by virtualization
- WiFi, GPS, camera, sensors, etc. are simulated so error conditions are often not exercised
- OEM and carrier customizations missing from simulators affect app behavior in unforeseen ways

Simulators may appear to lower expenses for testing apps, but they provide a false sense of security because they fail to take into account how apps perform and behave on physical devices, making simulator-only testing a dead-end for guaranteeing real-world quality.

Manual testing

Many mobile teams rely on manual testing, usually on a handful of devices with test coverage focused on new features, not on identifying regressions. Manual testing runs the gamut from developers testing features as they build them to a dedicated QA team following testing scripts and logging the results. Teams often assume users have the same devices they’re testing, but apps are downloaded on hundreds of different models and OS versions, so even rigorous manual testing has pitfalls:

- Poor feature and device coverage, as tests are biased toward new features on popular devices
- Slow result turnaround (often weeks)
- Security vulnerabilities because apps and data are in uncontrolled environments

Manual testing requires significant investment and ongoing expense, with a TCO upwards of $1,400 per device. Human testers often make mistakes or skip steps. Logging each step and reporting bugs is a painstaking process that can limit the number of features and devices tested and results often take weeks to report. Ultimately, manual testing adds significant expense and overhead along with a loss in time, agility, and innovation.
In contrast to manual testing and simulator-only testing, Xamarin Test Cloud lets teams test every feature on more than a thousand devices and on every commit. Catching bugs before released shortens development cycles and allows more time for innovation.
Xamarin’s model for mobile quality

- **Complete test coverage.**
  Using automation, developers can test every new feature and perform comprehensive regression tests from the UI down to ensure high-quality releases every time.

- **Comprehensive device testing.**
  Run an app on the largest device cloud in the industry with more than 1,200 real devices. Select devices based on form factor, manufacturer, operating system, or even target market popularity.

- **Fast troubleshooting.**
  See screenshots and video for every step of every test, then receive performance data and compare reports against previous runs to find regressions and bottlenecks.

- **Accelerated cycles with continuous integration.**
  Xamarin Test Cloud integrates with TFS, Jenkins, TeamCity, or any Continuous Integration (CI) systems with custom post-build commands, enabling collection of test results in a CI system automatically.

- **Comprehensive support for all native and hybrid apps.**
  Xamarin Test Cloud tests all apps, including apps built in C# with Xamarin as well those built in Objective-C, Swift, Java, and other cross-platform frameworks.

“Xamarin Test Cloud is our path to a low-maintenance, high-quality-regression-free future. We’re excited about where this can take our team and product.”

Sean Beausoleil
Mailbox Engineering Lead at Dropbox
Troubleshoot quickly with screenshots and graphical reports

The results dashboard compares the most recent test run to historical data, showing trends in app size and peak memory usage over time. Drilling into an individual test run shows overall patterns such as problems with a particular operating system, form factor, or manufacturer.

Drilling down further, Xamarin Test Cloud shows full-resolution screenshots and recordings of every interaction on every page of an app, providing visual comparisons among screen sizes, form factors, and operating system versions.

For functional failures, Xamarin Test Cloud provides precise device specifications, console logs, and CPU and memory metrics for every step leading up to the failure.
Flexible automation

Xamarin’s UITest and Calabash frameworks enable powerful automated UI testing. Developers write tests that behave as users do, performing taps, wipes, rotations, and waiting for UI elements to appear. Xamarin Test Cloud results are available within minutes, and include new feature testing as well as comprehensive regression tests for complete coverage and peace-of-mind that apps will work once released.

C# Example

[Test]
public void ShouldBeAbleToPlacelimitOrders ()
{
    app.EnterText(x => x.Id("Username"), "cathy@business.com");
    app.EnterText(x => x.Id("Password"), "xka13482733");
    app.Tap(x => x.Text("Log in"));
    app.WaitForElement(x => x.Text("Place Order"));
    Assert.IsNotNull(app.Query(x => x.Class("Button")).Text("Limit"));
    app.Screenshot("Place limit order");
}

Calabash Example

Feature: Trades
Scenario: Selectable Trading Order Types
Given I am logged in as “Cathy”
When I go to the Trading Orders Screen
Then it should display the Trading Orders order by Id

The world’s largest collection of iOS and Android devices

Xamarin Test Cloud provides automated mobile testing on hundreds of real devices in a secure device cloud.

The only way to know the user experience is to test on the devices consumers use, not on simulators or emulators, which is why Xamarin Test Cloud features over 1,200 real devices that can be selected by market share, manufacturer, form factor, or OS version.

Scripts are object-based and flexible to UI changes that may happen in design and development, unlike brittle optical character recognition (OCR) solutions that can break with just small changes to the app UI, or with rotation or form-factor size changes.

Tests are written in C# in Xamarin Studio or Visual Studio, or in Ruby using Calabash, the industry’s most powerful framework for automated testing of native and hybrid apps.
Instead of running a test from start to finish on a device, Xamarin Test Cloud’s parallelization feature separates test runs and executes them across multiple identical devices simultaneously, significantly increasing testing speed and frequency.

Xamarin Test Cloud is designed to integrate with CI tools like Jenkins, Team City, Team Foundation Server, and Visual Studio Online, so a test run can kick off with each commit and test results and failures can be included in nightly reports. Issues that are found and fixed early in the development cycle significantly reduce risk, so teams can develop cohesive, high-quality software much faster.

"Xamarin Test Cloud is fundamental in our effort to maintain the highest possible quality of our products and also securing a short time-to-market."

Niles Frydenholm
Software Architect, eBay Classifieds Group

Continuous quality
In a world of instant gratification, app crashes are a major driver for bad reviews and abandonment. 43% of users say that after a poor first experience they won’t use an app again, whereas referrals from happy users can boost adoption up to 45%.[^9]

Teams that are new to mobile often focus on getting the first version of an app to users as quickly as possible, without constructing a strategy for maintaining the app’s lifecycle. A well-formed plan for monitoring and fixing an app is the key to happier users, faster fixes, and more time for innovation.

Rather than integrating with a crash-reporting platform, many developers rely on app store ratings or bug reports from users to triage issues. Waiting for user feedback to uncover app crashes comes with several problems:

- Imperfect data because only a small percentage of users report issues
- Little-to-no visibility into why the crash is occurring
- No real-time coverage of crashes (reports may come days or weeks later)

A reactionary approach to app maintenance can result in decreased adoption because even when critical issues are resolved, reports of legacy issues will deter potential new users. This approach puts months of development at risk.

---

[^9]: Source: Various app store feedback and research on user behavior and app abandonment.
Xamarin Insights is an easy-to-use analytics and crash reporting solution that enables fast discovery, troubleshooting, and resolution of issues for live users. It’s a proactive method for monitoring and maintaining apps that significantly increases user satisfaction since problems can be resolved before they impact users.

Xamarin’s model for app monitoring

- **Comprehensive monitoring.** Developers are instantly alerted to crashes, exceptions and tracked events with detailed information for fast resolution
- **Optimized for cross-platform C# apps.** Xamarin Insights collects caught and uncaught .NET exceptions as well as native crashes.
- **A holistic view.** Xamarin Insights uses a simple API for crash reporting, user analytics and events, and market share data rather than utilizing multiple sites, libraries, and APIs.
- **Plugged into the workflow.** Xamarin Insights connects to a variety of popular external services to notify developers of new issues or changes to existing ones.
See crashes and user behavior

Xamarin Insights collects rich analytics, including durations for user actions, users affected by crashes and events, and the actions that lead up to each crash. These analytics help developers optimize performance and refine user experience.

Built for cross-platform C# apps

Xamarin Insights monitors both native crashes and managed exceptions for full-stack coverage. Since other crash reporting solutions aren’t created for apps built with C# and .NET, they only report native crashes, omitting important information about the C# code used to build the app. In addition, other monitoring tools report identical stacktraces in shared code separately by platform, so the same issue is reported multiple times. Xamarin Insights unifies stacktraces with the same root cause, even when crashes occur across platforms.

A unified view of apps and users

Xamarin Insights unifies crash reporting and user analytics into a single API, so developers can generate and view results more easily. Without Xamarin Insights, developers must pull multiple libraries and APIs into their apps and use multiple websites to monitor crashes, analytics, and events. Combined data sets also make troubleshooting easier; Xamarin Insights shows the events leading up to a crash, the exact device state, and any significant OS-level events (such as loss of internet connection) that occurred before the crash.
Key approaches for mobile success
Accelerators for going mobile

Xamarin offers the ideal platform for building, testing, and monitoring apps, as well as training and services to ensure success. Xamarin helps businesses at every step of the mobile journey—from training that transforms existing C# teams into mobile experts, to mobile strategy and architecture guidance, to world-class support for mission-critical apps.

—

Enterprise-class support and services

Enterprise support

Our enterprise support offerings include one day response SLAs, access to the latest hotfixes, technical training for getting started, and access to resources for troubleshooting.

Consulting partners

Customers can also tap into our Consulting Partners program that includes a worldwide network of over 300 Premier and Authorized Xamarin Consulting Partners who have first-hand experience helping clients ship great apps with Xamarin.

Xamarin professional services

Xamarin offers a range of consulting services designed to accelerate development at any stage in the app lifecycle and help businesses get the most from their mobile channels and Xamarin investment. Mobile experts provide architectural guidance, mobile strategy, best practices, code samples, and reviews. Xamarin’s services help new mobile teams build high-quality apps faster.
Xamarin University is live, online mobile development training. Offered six days a week, it features more than 50 classes, from introductions to iOS and Android development to advanced topics such as memory management, caching and synchronization, and backend integrations. Instructors lead small, interactive classes and offer one-on-one help during office hours. Recordings are available and the curriculum is always up-to-date with new platform and Xamarin releases.

Xamarin University transforms C# developers into skilled mobile developers with best practices that further accelerate development and mobile success.
Companies that adopt a Xamarin-based mobile strategy have the advantage.

Get started: xamarin.com

Case study: Mobile banking
Crédito Agrícola, one of the largest banks in Portugal, serves more than 1.2 million customers across 700 locations. It has prospered over the past 100 years through strong customer service and by keeping an eye on the future. Responding to the proliferation of smartphones and other mobile devices, Crédito Agrícola created consumer-banking apps for three mobile operating systems in the platform-specific languages. The costs of maintaining three separate code bases, however, drove the bank to evaluate cross-platform mobile development solutions for its next app, focused on its enterprise banking customers.

“We analyzed tools from the cross-platform development market, looking at factors including code reuse and compatibility of the final code, and ensuring the end-user experience was equal to that of native development without compromising performance. None of the solutions we looked at seemed to meet our needs except for Xamarin.”

Jorge Correia
Applications Development Director
Crédito Agrícola
Sérgio Viana, Associate Partner & Microsoft Solutions Lead at Xpand IT, had worked with Xamarin on previous projects and was certain it was the right solution.

“Native cross-platform development is something that the market has been asking for, but we were not confident in betting on any other solution before Xamarin,” Viana says. “Prior to Xamarin, developers had to choose between the expense of developing and maintaining separate codebases for each mobile operating system, or accept compromises in performance and the user experience inherent in other cross-platform solutions. We gain tremendous development and maintenance efficiencies with Xamarin, while the performance is indistinguishable from what we would get using the platform-specific languages.”

Using Xamarin with cloud-based notification mechanisms through Microsoft Azure Cloud Services, Xpand IT built the initial release of its app in just three months. Reusing code across the three platforms accelerated development by 60%.

“A secure banking app in three months

“The user experience is very important to us. One of our requirements was that the app for each platform offered the usability that the users expect. There can be no compromises and, with Xamarin, that’s what we have achieved.”

Jorge Correia
Applications Development Director
Crédito Agrícola

Reusing code across the three platforms accelerated development by 60%.
Higher quality with Xamarin Test Cloud & Insights

In addition to the development benefits of Xamarin, Crédito Agrícola took advantage of Xamarin Test Cloud and Insights to live up to the quality expectations of their enterprise banking customers.

The app had to work well on a variety of devices. On previous projects, manual testing had taken anywhere from five to ten days to cover just a few devices. Even then, users complained about many bugs that weren’t caught during testing. Xpand IT knew that they needed a different mobile quality approach.

Xamarin Test Cloud significantly expanded the device and feature test coverage on these new enterprise apps well beyond what Xpand IT had done with past projects, and test results were delivered within minutes instead of days or weeks.

The team improved quality further with Xamarin Insights, an intelligent analytics and crash reporting system to detect problems before they affect more users.

Viana reports that “Everyone was very impressed with how quick and easy it was to deploy Xamarin Insights, and Crédito Agrícola was happy to see the depth of information they could tap into just a few days after deploying it.”

References

1. www.gartner.com/newsroom/id/2939217
2. www.gartner.com/newsroom/id/2939217
3. www.slideshare.net/a16z/mobile-is-eating-the-world-40841467
5. www.internetretailer.com/2014/03/10/mobile-commerce-will-be-nearly-half-e-commerce-2018
6. www.netmarketshare.com/
7. www.idc.com/prodserv/smartphone-os-market-share.jsp
9. blog.xamarin.com/infographic-build-right-shift-left/