Microsoft platform and tools for mobile application development

White paper

Microsoft Corporation

Microsoft

PUBLISHED BY
Microsoft Corporation
One Microsoft Way
Redmond, Washington 98052-6399

Microsoft

Contents

Summary

Understanding and creating a mobile app strategy is an important process for today's enterprise decision-makers who aim to open up new business opportunities or to empower their employees to be more productive with new capabilities.

Microsoft defines an end-to-end (E2E) strategy for the agile creation of mobile apps that can target any platform (iOS, Android, or Windows), spans the requirements for consumer or employee scenarios, offers development teams tools to improve quality and to achieve faster time to market, and allows for integration with existing enterprise systems.

This strategy enables:

- **No-code development** for line-of-business (LoB) specialists and analysts ("citizen developers," as coined by Gartner), as well as IT staff and system integrator partners who help them solve problems, to create secure, single sign-on–based (SSO-based) business apps for mobile devices that connect to back-end enterprise systems, using visual tools that make building and iterating faster than traditional custom software development.
- **Code-focused development** of apps, which allows developers to leverage their skills in JavaScript/C#/C++ for a broad range of app solutions. These cover browser, containerized web, hybrid, or native app solutions, using highly productive integrated development environments (IDEs) and editors, such as *Visual Studio*, *Visual Studio Code*, or *Xamarin Studio*, with cross-platform capabilities.
- **Back-end services development** (Cloud, hybrid, or on-premises) that enables mobile scenarios with mobile app services capabilities (mobile back end as a service [MBaaS]) and allows developers to build their back-end APIs that can scale as required by leveraging *Microsoft Azure App Service Mobile Apps* (MBaaS), microservices architectures with *Azure Service Fabric* (platform as a service [PaaS]), or simply *Azure* Virtual Machines (infrastructure as a service [IaaS]).
- **Agile team development** for bug tracking, scrum boards, and task management using *Visual Studio Team Services* (VSTS) or *Visual Studio Team Foundation Server* (TFS) on-premises.
- **Mobile-focused developer operations** (DevOps) for continuous integration, continuous delivery and release management (to public stores, enterprise stores, or beta release management vehicles) with a range of build tasks from first-party to third-party vendors in the ecosystem, using VSTS or TFS. Developers and IT can enable a wide range of scenarios, such as iOS and Mac builds, Android builds, Windows builds, unit testing, and real-device testing in the cloud.
- **Beta-release management and feedback** with automated deployment to mobile devices by using *Microsoft HockeyApp* integrated to VSTS/TFS builds and release pipes.
- **App health monitoring and analytics**, such as crash reporting behavioral usage, using Microsoft HockeyApp SDK and portal for mobile apps, and *Application Insights* for back-end services telemetry and analytics.

- **Analyze end-user behavior and drive targeted marketing campaigns** by segmenting end users and leveraging campaigns based on mobile push notifications using *Azure Mobile Engagement*.
- **Manage and secure production apps (mobile application management [MAM]) and devices (mobile device management [MDM])** using *Microsoft Enterprise Mobility Suite* (EMS) and *Microsoft Intune / Intune MAM SDK* as part of EMS.

Purpose

This paper outlines the end-to-end platform from Microsoft that form the critical capabilities for organizations selecting technologies and tools for a Mobile Application Development Platform (**MADP**) and Rapid Mobile Application Development (**RMAD**).

This paper also describes the core decision factors that organizations should consider, and it outlines the technical capabilities available with Microsoft solutions and products.

Who should use this guide

Technical decision-makers who require a high-level overview of Microsoft solutions and technologies for building mobile apps can benefit from this paper.

Key benefits

Readers obtain an understanding of the key components and decisions that are necessary for their organizations to establish a comprehensive, mobile app development strategy—one which enables flexibility as target devices, scenarios, and business requirements evolve.

Implementing the right strategy and development technology decisions in your enterprise can help you to effectively build and manage mobile apps and to enable new opportunities for business-to-consumer (B2C) scenarios. These decisions can also empower productive employees for business-to-employee (B2E) solutions, with the business goals of agility, quality, fast time to market, and lower costs.

CHAPTER 1

Introduction to the Microsoft platform for mobile app development

Vision

Create an adaptable, enterprise-grade, mobile app strategy that spans your development, IT operations, and production management.

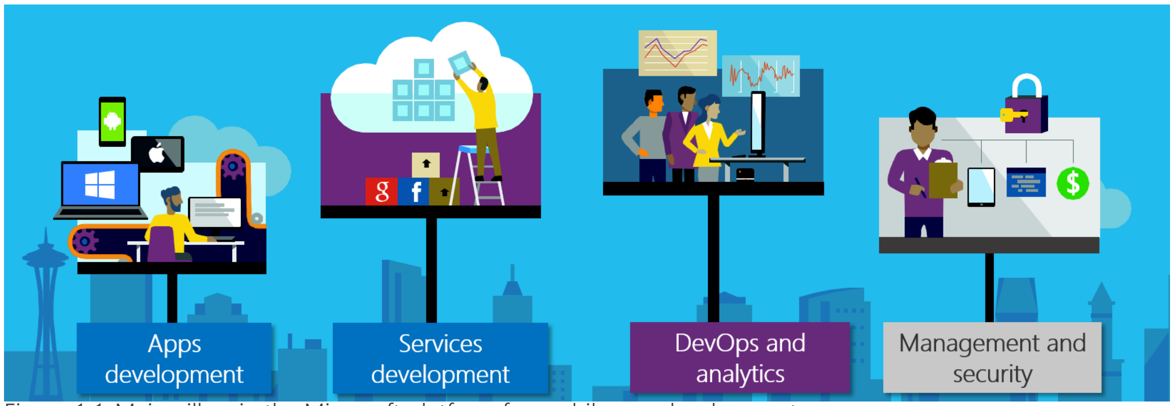


Figure 1-1. Main pillars in the Microsoft platform for mobile app development

Building a mobile app development strategy to drive new business opportunities or to empower productive employees means many decisions. More than simply selecting a programming language, it means deciding whether to invest for and build apps for iOS, Android, or Windows with single-platform languages or to take a cross-platform approach. It means building a strategy that enables the organization to adapt as the platforms and devices evolve and a strategy that delivers mobile back-end services that can scale. And it means that apps are secure if the device is lost or if users or capabilities of the app need to be restricted. It even means integrating seamlessly with back-end data, whether online or offline.

Decision-makers want a flexible, secure, and enterprise-grade strategy that can evolve with their business. Enterprise developers want to leverage and expand on their skills—whether web/JavaScript, .NET/C#, or C++ skills and existing code base. And they want to be able to connect to back-end systems, either in the cloud or on-premises, and to deliver continuously and quickly—as required by the business. IT wants to have confidence in its secure management of apps and devices, and marketing wants to drive effective, mobile, targeted campaigns.

Clearly, it is not just about developing client mobile apps, which is only the tip of the iceberg. There are many other areas to take into account, like back-end services and a full mobile app lifecycle. These are all enabled through Microsoft tools and services.

Introduction

An organization may see its mobile client app strategy fulfilled by investments in websites, simple client apps made up of web content or in fully enabled client mobile apps that make use of many of the capabilities of the device. Indeed, many organizations take a multichannel approach to their business needs, by investing in multiple approaches in a complementary manner.

A mobile web presence provides a broad approach that has a simple update process across all form factors. It is, however, limited when thinking about device capabilities (such as the ability to interact with a barcode scanner) and therefore less capable of engaging or of employee productivity scenarios. Mobile client apps, on the other hand, which are distributed via stores, have full device capabilities and have engaged experiences but require compilation and packaging for each target platform (iOS, Android, or Windows).

Selecting to go web or mobile client app (or to choose some combination of them) means decisions that range from an architectural point of view to considerations of cost, quality, and time to market (and can vary from project to project). In today's fast-moving world, it is not only about the construction of apps but also about how productive the development teams are, how well they can adjust to feedback and fix issues, how IT manages apps and devices in the modern "bring your own device" (BYOD) environment, and how well marketing can engage with its audiences. Decision factors may therefore include:

- A developer's or team's existing skills or ability, along with costs to retool and retrain.
- The types of apps to be built and their business objective, such as B2E, Business to Business (B2B), or B2C.
- Technological requirements, such as device capabilities, security, existing enterprise systems, new capabilities (like push notifications, beacons, or analytics to drive app health), user telemetry, or marketing and engagement campaigns.
- Community and product support capabilities, including integration with existing tools and processes.
- The time to market, quality, and adherence to the look and feel or user experience (UX) of the desired target platforms.
- Costs, skills, time, and ability to drive an agile, quality-focused development process from staging to beta to production—the software development lifecycle (SDLC).
- Secure mobile client app delivery, management, or device management requirements.

Microsoft tools and solutions span the multichannel approach and offer solutions that can leverage existing skills, connect to back-end systems on-premises or in the cloud and effectively enable both IT and marketing teams.

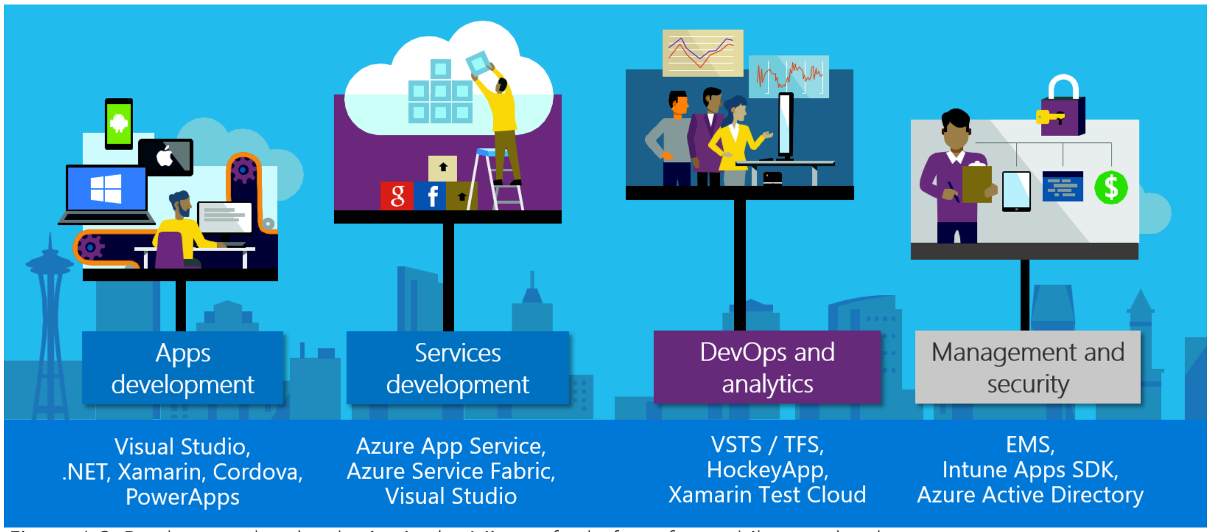

Apps development
Services development
DevOps and analytics
Management and security
Visual Studio, .NET, Xamarin, Cordova, PowerApps
Azure App Service, Azure Service Fabric, Visual Studio
VSTS / TFS, HockeyApp, Xamarin Test Cloud
EMS, Intune Apps SDK, Azure Active Directory

Figure 1-2. Products and technologies in the Microsoft platform for mobile app development

The Microsoft platform for mobile app development, as defined in Figure 1-2, has the following components:

- **Apps development.** Developers can use client-side technologies to build client apps themselves, using specific frameworks and patterns for a cross-platform approach. With Microsoft technologies, developers can build native (**native-single-platform** using languages like Objective-C and Java with *Azure SDKs*, **native** and **cross-platform** apps using *Xamarin, .NET* and *C#*), **hybrid** (using *Cordova*) and its variants, or **websites** (*ASP.NET*), depending upon their decision factors.

  Professional developers building client front ends can make use of IDEs and code editors, such as Visual Studio, Xamarin Studio, or Visual Studio Code on PCs and Macs, to construct their client apps. These tools offer a RAD approach to building client mobile apps with designers, IntelliSense, and other productivity features, such as cross-platform debugging. In addition, a no-code, visual RAD tool, called *Microsoft PowerApps*, allows users to build codeless mobile enterprise apps quickly and easily with connections to their back-end enterprise systems.

- **Back-end services development.** *Azure App Service, Azure Service Fabric*, and Azure IaaS VMs provide the foundation from MBaaS to PaaS and IaaS, depending upon levels of customization, scale, and coding. Developers can create APIs backed by connectors to enterprise systems, SaaS, and ERP/CRM systems. Developers leverage **mobile services**, such as push notification, data sync online/offline, and authentication, and a **data platform**, which together form a comprehensive and expansive mobile back end as a service that supports a variety of app construction scenarios. Developers wanting to leverage a microservices architecture can use *Azure Service Fabric* (PaaS) to produce compose-able APIs for scale and performance. This all runs on *Azure* cloud and on-premises with *Azure Stack*.

- **DevOps and app analytics.** Developers creating mobile back ends and client front ends can leverage *Visual Studio Team Services* (online) or *Team Foundation Server* (on-premises) to build out a comprehensive application lifecycle management (**ALM**) or agile team environment—from source-code control and bug tracking to scrum workload tracking.

Developers can create mobile and cloud-focused **DevOps** for a fast, iterative process that covers continuous integration (**CI**) delivery, continuous delivery **CD**, and release management (**RM**). Developers can produce native builds for all platforms and can run unit tests and UI automation—including against emulators or real devices in the cloud—using *Xamarin Test Cloud* or partners, such as *Perfecto Mobile* or *Sauce Labs*. Developers can also automate their back-end code releases in Azure staging slots, from development to staging and production.

Developers creating comprehensive build tasks can automate their releases through to beta test channels, such as HockeyApp, or deploy directly to enterprise stores or app stores or to enterprise management products, such as Intune.

HockeyApp allows the developer to easily distribute and gather exception telemetry from their apps. As part of the beta process, HockeyApp allows developers to easily distribute their beta versions to an internal or external audience. Additionally, by making use of HockeyApp SDKs, developers can monitor and respond to issues in their apps through crash reports.

- **End-user insights and business analytics.** *Azure Mobile Engagement* provides insights on users and how they are using the app, along with a way to segment users and to create targeted marketing campaigns based on push notifications sent to mobile devices.
- **IT management and security.** Developers using *Active Directory* (AD) can produce apps that support **Single Sign-On** (SSO), but integration with Azure App Service mobile apps back end (MBaaS) simplifies the process to allow developers to authenticate users using Azure AD or social authorization, such as Twitter, Facebook, or Google. Use of Azure App Service SDK also provides developers the ability to create secure online/offline data protection over the air and at rest, through local encryption.

  Developers and IT can further manage access to back-end APIs using *Azure API Management*.

  IT can manage apps and devices in several ways:

  - *Microsoft PowerApps* leverages Azure Active Directory to ensure access control across apps and connected services.
  - Apps deployed and managed through *Intune* can set policies to restrict features of the app or to restrict the user or enforce encryption or VPN transport.
  - Apps deployed to devices through *Intune* can be securely managed (for example, in the event of loss of the device).

Microsoft therefore offers a complete foundation for a mobile app strategy. At the same time, it is **a collection of technologies which allow you to optionally select and integrate with existing tools and processes**. The flexibility in a broad approach and the strength in the depth of capabilities place Microsoft in a strong position for enterprise mobile apps development.

CHAPTER 2

Developing client mobile apps

Microsoft cross-platform mobile app development tools and platforms provide a comprehensive solution for B2C or B2B apps, allowing you to share code across all target platforms (iOS, Android, Windows, and web) and helping you to lower your total cost of ownership (TCO).

Microsoft offers a range of developer tools for mobile client app construction that enable **cross-platform** or **code-sharing** solutions. (Note that developers who prefer to build pure native applications using native tools, such as Xcode or Android Studio, can still continue to take advantage of other tools and components from Microsoft for their back-end services or for DevOps, release management, beta testing, and analytics).

Factors to take into account:

- Choice of technologies and tools (*Visual Studio Tools for Apache Cordova* or *Visual Studio with Xamarin/C#*, Xamarin Studio, or Microsoft PowerApps) can vary depending upon several factors, as previously outlined, including developer skills, coding requirements, time to market, UX demands, app performance, and business objective (B2E or B2C, for example).
- The development platform and technologies integrate with Azure App Service capabilities (PaaS and MBaaS) through client SDK and tool extensions, but this does not preclude a developer from choosing to integrate with third-party services or other MBaaS products.
- You can reduce TCO by sharing the same code base and by consolidating development teams and skills through a cross-platform mobile development strategy when building apps across two or more platforms (iOS, Android, and Windows), and you can achieve a faster time to market and meet higher quality, performance, and UX requirements and standards.

Choices for cross-platform mobile development

Although it's possible to develop a native app for each platform individually (such as developing with *Objective-C/Swift* for iOS and/or *Java* for Android) and to deliver a great user experience, when targeting several platforms, the costs of such an approach can be prohibitive, both in terms of time to market and TCO across the app's lifetime. To help control and lower these costs, different cross-platform development technologies have evolved to produce platform-specific app packages from a shared code base.

For cross-platform or code-sharing solutions, Visual Studio provides a best-in-class IDE for developers on PCs. However, developers can also leverage cross-platform editors, such as Visual Studio Code or Xamarin Studio on a Mac. These tools focus on making the developer as productive as possible, by tightening the iterative workflow from editing with IntelliSense to debugging with cross-platform tools and emulators.

Developers have a range of potential solutions for their cross-platform mobile apps:

- **Mobile web.** Apps are built with HTML, CSS, and JavaScript to run in a mobile browser, and apps are deployed to a web host rather than appearing in platform app stores. Access to platform APIs is limited to those that are exposed through HTML5. Since this approach uses exactly the same skills and technologies as when developing regular web apps with single-page application (SPA) approaches with a responsive design, this paper is not focusing on it. There are many other available resources focusing on web development.

- **Hybrid** using **Visual Studio Tools for Apache Cordova** or **Visual Studio Code** with extensions for **Cordova.** Build hybrid HTML/JavaScript apps (leveraging web development skills) by taking advantage of a single shared JavaScript API which provides full code reuse across any device and which offers access to native device capabilities through Cordova plugins. Hybrid technologies fit especially well when creating B2E and B2B apps, although B2C is also a possibility, depending on the UI requirements.

  With the hybrid approach, you can share components with websites and reuse web server–based apps with "hosted web apps" approaches based on Cordova.

- **Native** using **C# and .NET with Xamarin in Visual Studio and Xamarin Studio.** Build stunning native apps that share nearly 100 percent of code across iOS, Android, and Windows with Xamarin Platform. To increase speed of development and efficiency, developers can use the Xamarin.Forms API to quickly build common platform-specific UI elements, or they can write directly to platform-specific APIs in C# for maximum control. With Xamarin Platform, C# developers can do anything Java, Objective-C, and Swift developers can do—with a single code base. Native Xamarin apps are a great fit for B2C apps or for enterprise apps requiring the highest levels of performance, security, and device access.

- **Native**, shareable components using **C++** or **C#** (Portable Class Libraries [PCLs]). Build native components and libraries for mobile apps for any device with C++ in Visual Studio. Create impressive 3D graphics with OpenGL in C++ and embed components into Xamarin apps, or reuse legacy C++ logic/code by creating cross-platform libraries that can be reused from mobile apps.

- **Native** games with **C#/Unity.** Build games with C# and Unity for all platform targets, with Visual Studio and the power that the IDE brings to bear.

- **Visual Studio Code** with extensions for **React Native** to create native mobile client apps.

- **Data-driven enterprise apps.** Citizen developers, including line-of-business analysts and their counterparts in IT, can produce business apps for all platforms, with **Microsoft PowerApps**, using a no-code authoring tool with support for easy connections to back-end enterprise systems and data.

Visual Studio professional and enterprise SKUs also integrate with ALM tools in VSTS or TFS.

More details on these options are highlighted in the following sections.

Building hybrid mobile apps based on HTML/JavaScript

Using Visual Studio Tools for Apache Cordova

Using HTML, CSS, and JavaScript, developers leverage their skills from building websites and apps to build mobile apps for iOS, Android, and Windows with Apache Cordova. Most developers achieve nearly 100 percent code reuse while leveraging the Cordova shared JavaScript API to access native device options, like the camera, calendar, and other hardware capabilities.

A Cordova app is composed of the same HTML/JavaScript/TypeScript code that can be compiled for each platform (iOS, Android, and Windows).

Developers have the freedom to select their favorite JavaScript and controls frameworks which can integrate with Visual Studio. The range of advantages and tooling enhancements with particular patterns and frameworks includes:

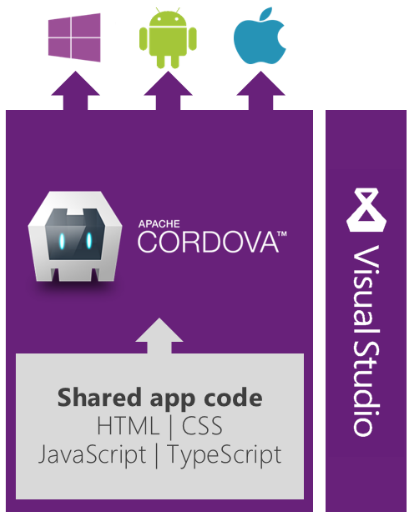


Figure 2-1. VS Tools for Apache Cordova

- UI controls, like the ones provided by **Ionic**. Ionic 2 is built on AngularJS 2 (from Google), which leverages *TypeScript* (from Microsoft). Developers get full IntelliSense and build support for TypeScript and JavaScript-based apps.
- Visual Studio provides powerful, cross-platform development features integrated in the IDE, such as emulators, device deployment, and debugging against emulators or tethered or remote devices.
- Scale to complex enterprise apps through optional TypeScript support.
- Visual Studio based on NPM acquires and installs all the necessary components (plugins and frameworks) required to build and maintain up-to-date Cordova-based mobile client apps.
- Visual Studio contributes heavily to the Cordova platforms and plugins, raising its quality and robustness.
- Visual Studio includes references to core Cordova plugins that are matched to the Cordova platform version for the enterprise.
- For iOS apps development, Visual Studio integrates with a remote build agent to provide builds, deployment, and debug.
- Leverage Azure App Service Plugin and JavaScript libraries to connect to back-end systems and connectors and to leverage mobile services.
- Interoperability and flexibility—as Visual Studio leverages regular Apache Cordova and web technologies, your Cordova apps developed with Visual Studio can interoperate with any third-party service or technology compatible with JavaScript consumption.

Building native and cross-platform apps

Using C#, .NET, and Xamarin in Visual Studio and Xamarin Studio

Developers can create stunning native apps using C#, Xamarin, and .NET across iOS, Android, and Windows, with a single, shared C# code base. With this approach, developers can:

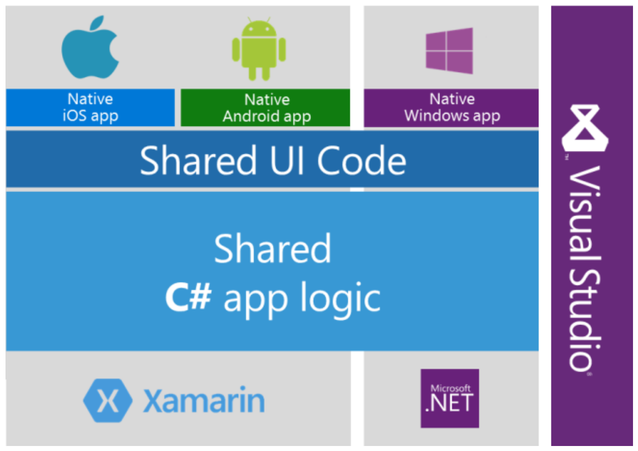


Figure 2-2. C# cross-platform with Xamarin and .NET

- Leverage existing C# skills.
- Create 100 percent native user interfaces customized for each platform, using full-featured Android, iOS Designers, and Windows 10 Universal Windows Platform (UWP) Designers. Xamarin and UWP apps are native apps, so they look and feel like users expect. Developers can design their app's interfaces through the IDE's designers.
- Maximize code shared with PCLs and Shared Projects.
- Build a custom native user interface for each platform using Xamarin.iOS, Xamarin.Android, and Windows 10 UWP, or use Xamarin.Forms to write single, shared user interface (based on XAML) across those platforms.
- You can install the latest version of Xamarin from Visual Studio. Choose a Xamarin project to start building your app, and use standard C# IntelliSense, debugging, and other powerful features of the Visual Studio IDE.
- Leverage Azure App Service SDKs to connect to back-end systems and connectors and to leverage mobile services.
- Benefit from interoperability and flexibility. Xamarin apps can use 100 percent of the platforms' APIs (iOS and Android) that are exposed through Xamarin/C#, which means that anything you can do in Objective-C, Swift, or Java can be done in C# and Visual Studio with Xamarin.

  Xamarin promises the industry's fastest support for new iOS and Android APIs as they are released to the public.

  In addition, Xamarin and .NET apps can consume any third-party service (like standard HTTP services based on JSON and OAuth).

Building native components using Visual Studio C++ cross-platform

Develop C++ cross-platform mobile code for Android, Windows, and iOS. With Visual Studio and C++, developers can:

- Develop shareable libraries and components, using C++, that are compiled to native.
- Share and reuse existing C++ code.
- Embed C++ components within Xamarin cross-platform apps.
- Migrate existing C++ libraries to target Android and Windows platforms, or use these C++ libraries to build complete Xamarin Android Native or Java Native Interface applications. You can also develop complete Android Native-Activity applications—all within Visual Studio.

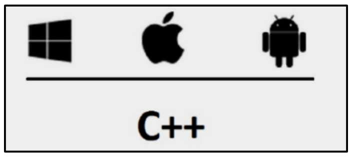


Figure 2-3. C++ cross-platform

Building native games using Visual Studio Tools for Unity

Developers can build games using the Unity engine and C#:

- Build multi-platform games.
- Debug from Visual Studio.
- Create Unity scripts in Visual Studio.
- Enhanced productivity with Visual Studio.
- *Visual Studio Tools for Unity* is natively supported in Unity.
- Get Unity and Visual Studio tools all in the Unity installer.

Figure 2-4. Unity cross-platform

The Unity engine integrates into one unparalleled platform to create 2D and 3D games and interactive content. Create once, and publish to 21 platforms, including all mobile platforms, WebGL, Mac, PC and Linux desktop, web, or consoles.

Creating custom LoB apps quickly using Microsoft PowerApps

Microsoft PowerApps is a tool that allows you, without writing code, to rapidly develop web and mobile apps which are connected to existing enterprise data sources. The product is built for those closest to the business app development process today: analysts and specialists in lines of business, as well as the IT developers and system integrator partners they rely on for custom software needs. For many teams today, the process of developing business apps is slow and costly, and the demand for innovation far outstrips the capacity of developers and resources.

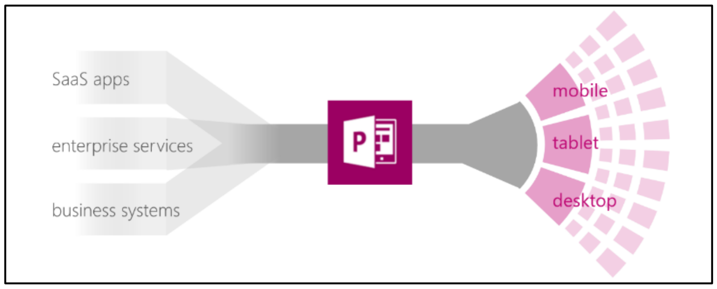


Figure 2-5. PowerApps

Microsoft PowerApps changes the economics of custom software development. PowerApps Studio is an *Office*-like visual design environment that connects to a range of data sources that businesses already rely on, from *SharePoint Online*, Salesforce, and Dropbox, to *SQL Server* and even custom APIs. Users can quickly build web- and mobile-optimized business apps that create, read, update, and delete records in *Excel* spreadsheets, CRM systems, databases, and more. PowerApps also features a workflow automation tool that gives app creators the ability to trigger business processes and actions across their connected services, such as approval workflows or syncing records between systems.

PowerApps is ideal for a range of business app development projects. App creators can quickly modernize legacy systems by connecting to those data sources and giving them a fresh web and mobile UI. Other frequent projects include the delivery of mobile-optimized workflow forms, such as site inspection apps or quote calculators, which are specific to a business process and can take advantage of mobile features, like cameras, pen inputs, and GPS location.

With PowerApps, IT developers can move much faster than they can in traditional development environments. Likewise, business specialists can directly participate in the app development process by applying their existing skills to manage screen layouts, colors, themes, and fonts, as well as creating dynamic interactions through Excel-like functions.

In summary, the PowerApps service:

- Is an enterprise service for rapidly developing cross-platform business apps that run on the web, iOS, Android, and Windows.
- Integrates with enterprise systems and data connectors and leverages mobile services, such as authentication and push notifications.
- Contains a workflow automation feature enabling app creators to trigger business processes across their connected services.

CHAPTER 3

Developing back-end services

Build intelligent back-end services for mobile apps with corporate sign-on and secure connection to on-premises resources and SaaS solutions. Create robust apps that remain useful when there are network issues, so users can create and modify data even when they are offline. Broadcast personalized push notifications to millions in minutes.

Microsoft Azure (cloud) and Microsoft Azure Stack (on-premises) provide the perfect home for enterprise mobile application services in the cloud where teams can concentrate on the code that matters without worrying about the scalability and security of the platform. The platform supports the back-end services required to rapidly build engaging mobile apps for enterprise needs.

Every mobile app with a cloud-hosted back end enjoys the benefits of connectivity and scale, regardless of whether that back end utilizes infrastructure as a service or platform as a service.

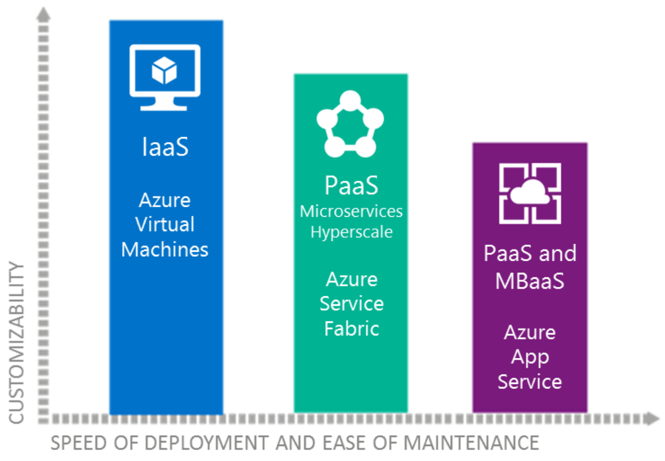


Figure 3-1. From IaaS to PaaS and MBaaS

The tradeoff between IaaS and PaaS comes down to customizability and control versus speed of deployment and ease of maintenance. IaaS virtual machines (VMs) are highly customizable—users are responsible for everything from OS patching to middleware and runtime to your application and its data.

An important subset of PaaS optimized for mobile app development is commonly referred to as *back end as a service* or *mobile back end as a service*. An MBaaS not only provides the connectivity and scalability that comes with all cloud-based services but also supplies turnkey solutions for common mobile development challenges, like push notifications, user authentication, and offline scenarios.

Note: Developers are not restricted to using the abstractions in an MBaaS but are free to use *API App* in Azure App Service and microservices to build out custom mobile back ends.

A typical MBaaS solution (like *Azure Mobile Apps*) provides a turnkey way to add data storage, user authentication, push notifications, social media integration, offline sync, analytics, and more. The value of an MBaaS allows developers to forget about the infrastructure and to focus on delivering a differentiated user experience. Therefore, Azure Mobile Apps (part of Azure App Service) provides everything a mobile developer needs in a mobile back end for most requirements.

For more complex or customized scenarios, PaaS microservice clusters, like Azure Service Fabric, can do a lot more of the heavy lifting for developers, automatically deploy OS patches and middleware especially made for microservices, and hyperscale even with stateful services with no latency between data and logic.

Mobile back ends using Azure App Service

Any mobile app with a cloud-hosted back end (whether on infrastructure or platform services) enjoys the benefits of connectivity and scalability. However, choosing an MBaaS for mobile app development results in additional benefits. MBaaS solutions are optimized to streamline connected app development (connected, because in addition to supporting popular mobile platforms, like iOS and Android, most BaaS solutions also work with Windows Store and Mac OS X apps). MBaaS solutions handle the glue code associated with the most common mobile development tasks, such as storing app data in the cloud, authenticating users, sending push notifications, and more.

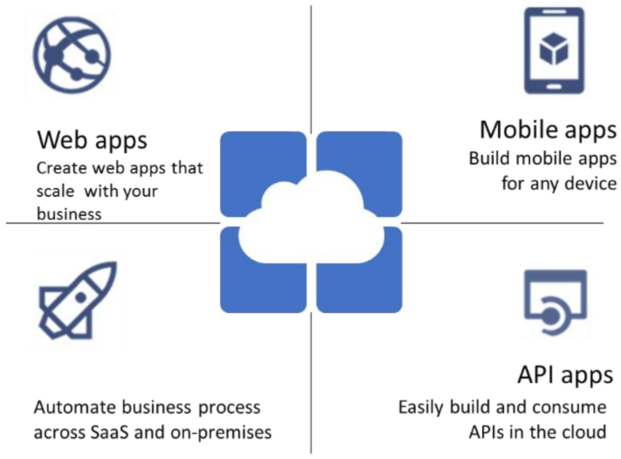

Web apps
Create web apps that scale with your business
Mobile apps
Build mobile apps for any device
Automate business process across SaaS and on-premises
API apps
Easily build and consume APIs in the cloud

Figure 3-2. Azure App Service

Azure App Service is a cloud platform for building powerful web and mobile apps that connect to data anywhere, in the cloud or on-premises. It includes Web Apps, API Apps, Mobile Apps, and Logic Apps.

Mobile back-end services with Azure Mobile Apps

Azure App Service Mobile Apps lets you:

- Easily add corporate sign-on to mobile apps and securely connect to on-premises resources (**SSO**).
- Create robust apps that remain useful when there are network issues, so users can create and modify data even when they are **offline**.
- Broadcast personalized **push notifications** to millions in minutes, by enabling a push notification service (APNS for iOS, GCM for Android, MPNS for Windows Phone, and WNS for Windows Store) in the back end.
- Easy configuration for authentication using **SSO and AD** or social authentication providers, like Facebook or Twitter.
- Many enterprises have existing **enterprise systems** that can be easily connected to support mobile apps using the Azure App Service **SaaS and other data connectors**, such as Salesforce, Office, or Dropbox, in their API or mobile apps. Similarly, since enterprises may still want to connect directly to their enterprise systems, they have options to use Azure Stack for an on-premises solution. This provides all the capability of Azure in the cloud, but on-premises.

Developers can create their back ends through the Azure portal or Microsoft tools, including quick-starts for mobile client apps, using templates for Visual Studio or other IDEs. Visual Studio also includes workflows to connect your mobile client app, with an MBaaS back end.

Azure App Service enables you to easily scale applications on demand with high availability and easy management and monitoring of assets and infrastructure though the Azure portal.

Figure 3-3. Azure portal with Azure Mobile Apps dashboard

Workflow and business processes with Azure App Service Logic Apps

Azure App Service Logic Apps allow any technical user or developer to automate business process execution and workflow using an easy-to-use visual designer. It offers the ability to:

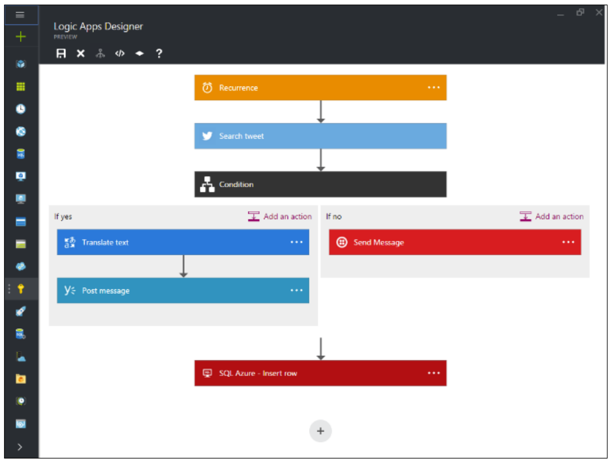


Figure 3-4. Azure portal with Azure Logic Apps Designer

- Create business processes and workflows visually, from the web.
- Deliver integration capabilities in web, mobile, and API apps.
- Integrate with your SaaS and enterprise applications through connectors like Office 365, Salesforce, Google, and more.
- Automate enterprise application integration (EAI), B2B, and business processes.
- Connect to on-premises data and LoB services.

Building microservices with Azure Service Fabric

Microservices enable long-term maintainability in large, complex, and highly scalable systems by designing applications based on many independently deployable services that allow for granular release planning. Some examples of microservices include protocol gateways, user profiles, shopping carts, inventory processing, purchase subsystems, payment processing, and queues and caches.

The microservices architecture is an approach to building a server or cloud application as a set of small services, in which each runs in its own process and communicates via protocols, such as HTTP and web sockets. Each microservice implements specific, end-to-end domain/business capabilities within a certain "Bounded Context" and is developed autonomously and deployed independently by automated mechanisms. Finally, each service owns its related domain data model and domain logic sovereignty and decentralized data management and can employ different data storage technologies (SQL, NoSQL) and different programming languages per microservice.

Microservices can scale out independently, as compared to giant monolithic application blocks that all scale together. This means that just the specific functional area that needs more processing power or network bandwidth to support demand can be scaled, rather than unnecessarily scaling out other areas of the application.

Architecting fine-grained microservice applications enables continuous integration and development practices and can accelerate delivery of new functions into the application. Fine-grained decomposition of applications also means running and testing in isolation. As long as you don't break the contracts or interfaces, you can change any microservice implementation under the hood and add new functionality without breaking the other microservices that depend on it.

Azure Service Fabric

Distributed computing and complex microservices deployments can be hard to manage if you do them by yourself. Azure Service Fabric provides the plumbing required to create, deploy, run, and manage microservices in an effective and efficient way.

Azure Service Fabric is a distributed systems platform used to build hyperscalable, reliable, and easily managed applications for the cloud. It addresses the significant challenges in developing and managing cloud applications, and developers and administrators can avoid having to solve complex infrastructure problems and focus instead on implementing mission-critical, demanding workloads, with the confidence of knowing that these are scalable, reliable, and manageable. Azure Service Fabric represents the next-generation middleware platform from Microsoft for building and managing these enterprise-class, Tier-1 cloud scale services.

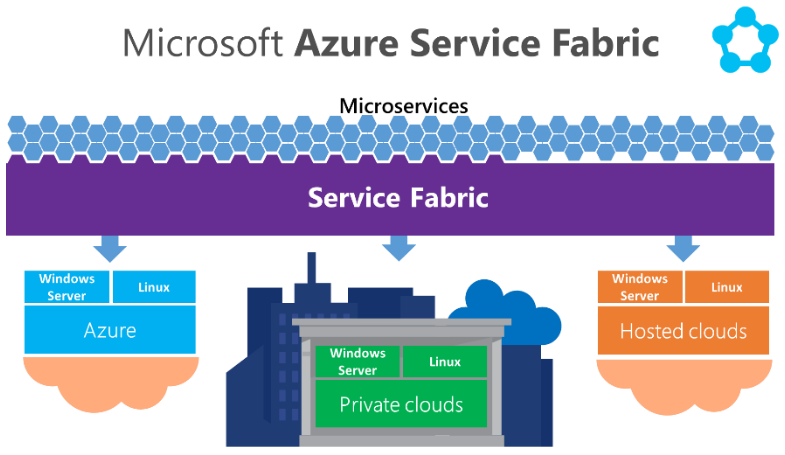


Figure 3-5. Azure Service Fabric

Azure Service Fabric is a universal deployment environment, which means that you are able to deploy any executable based on any language (including .NET, Node.js, Java, or C++) or even database runtimes (like MongoDB). Therefore, Azure Service Fabric is not limited to microservices-oriented applications. You can also use it to host and deploy traditional applications (web apps or services) and enjoy many benefits related to scalability, load-balancing, and fast deployment.

Implementing data platform services with Azure

Azure is an open ecosystem and offers a large variety of data platforms, including relational SQL databases, NoSQL databases, blob/files storage, cache as a service, and analytics and big data.

SQL databases using Azure SQL Database

Developers building SaaS applications can use *Azure SQL Database* to provide flexibility to support both explosive growth and profitable business models, based on the ability to scale, have high performance, high availability, and the peace of mind that comes with security.

- **Elastic database models and tools.** An elastic database gives developers the ability to pool resources to use among a group of databases for scaling, which can then be administratively managed by submitting a script as a job, and the SQL Database performs the script across the databases.
- **High performance.** High-throughput applications can take advantage of the latest version which delivers 25 percent more premium database power.
- **Backups, replication, and high availability.** Built-in replication and a Microsoft-backed SLA at the database level provide app continuity and protection against catastrophic events, especially when combined with, active georeplication, an ability to control when and where to failover, and self-service restore, which provides full control over "oops recovery" (data restoration from available data backups of up to 35 days).
- **Near-zero maintenance.** Automatic software is part of the service, and built-in system replicas help to deliver inherent data protection, database uptime, and system stability. System replicas are automatically moved to new computers, which are provisioned on the fly as old ones fail.
- **Security.** SQL Database offers a portfolio of security features to meet organizational or industry-mandated compliance policies. Auditing provides developers the ability to perform compliance-related tasks and to gain knowledge about activities. Developers and IT can implement policies at the database level to help limit access to sensitive data with row-level security, dynamic data masking, and transparent data encryption for Azure SQL Database. (SQL Database is verified by key cloud auditors as part of the scope of key Azure compliance certifications and approvals, such as HIPAA BAA, ISO/IEC 27001:2005, FedRAMP, and EU Model Clauses).

NoSQL databases using Azure DocumentDB

Azure DocumentDB is a NoSQL document database service designed from the ground up to natively support JSON and JavaScript directly inside the database engine. Coding against DocumentDB is simple, approachable, and open, and it does not require custom encoding or extensions to JSON or JavaScript. These benefits allow developers to:

- **Build modern, scalable, mobile, and web applications** with a unique combination of robust querying and transactional data processing. Developers can extend the power of DocumentDB with JavaScript-based custom query operators or user-defined functions.
- **Rapidly develop** by accessing databases through CRUD, query, and JavaScript processing over a simple RESTful HTTP interface. Developers can leverage a library of SDKs for JavaScript, Java, Node.js, Python, and .NET.

Storage using Azure Storage

Azure Storage is the cloud storage solution for modern applications that rely on durability, availability, and scalability to meet the needs of their customers. It provides the flexibility and hyperscale needed to store and retrieve large amounts of data so that, as storage demands increase (for example, petabytes of storage), developers can leverage 500 TB of total storage per account, and a single subscription supports up to 50 storage accounts. Developers can make use of REST-based APIs to access storage.

Deliver high-performance, low-latency disk support for I/O-intensive workloads running in Azure Virtual Machines, which is extremely durable and highly available (replication).

- ***Azure Blob Storage* (Object Storage).** Store unstructured data, such as documents and media files.
- ***Azure Table Storage*** for structured NoSQL data.
- ***Azure Queue Storage*** to reliably store messages.
- SMB-based ***Azure File Storage*** for existing or new applications.

Caching using Azure Redis Cache

Azure Redis Cache (based on the open-source Redis cache) gives developers access to a secure, dedicated Redis cache, managed by Microsoft and accessible from any application within Azure. It enables applications to become more responsive, even as user load increases, by leveraging the low-latency, high-throughput capabilities of the Redis engine. This separate, distributed cache layer allows the data tier to scale independently for more efficient use of compute resources in applications.

Azure Redis Cache can be easily managed (for example, monitoring its health and performance) through the Azure portal.

NOTE: Microsoft Azure provides additional data sources related to big data and analytics, but this paper focuses on mobile app development. For more information on big data and analytics in Azure, explore Azure HDInsight, Azure Data Lake, or Azure SQL Data Warehouse.

On-premises back-end systems with Azure Stack

Microsoft Azure Stack is a new hybrid cloud platform product that enables your organization to deliver Azure services from your own datacenter to help you achieve more. Get the power of cloud services, yet maintain control of your datacenter for true hybrid cloud agility. You decide where to keep your data and applications—in your own datacenter or with a hosting service provider. Easily access public cloud resources to scale at busy times of the year, for dev-test, or whenever you need them. Microsoft builds and runs its own hyperscale datacenters and delivers that proven innovation to your datacenter.

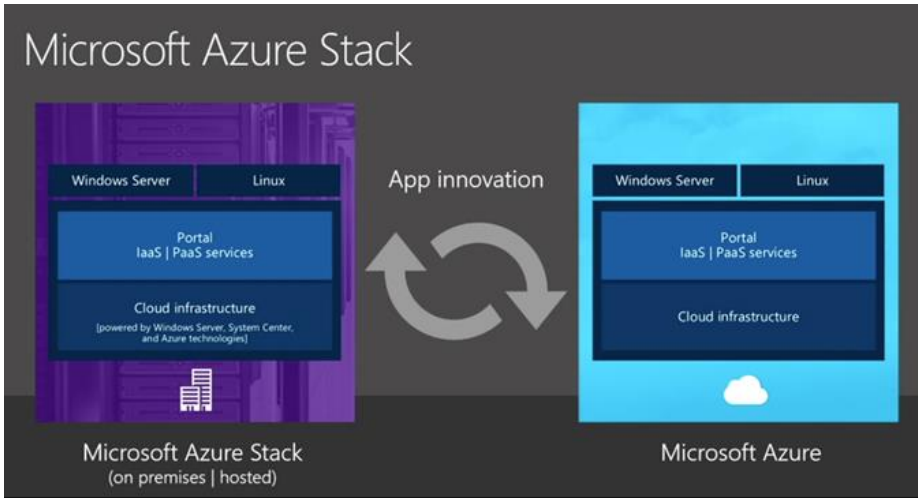


Figure 3-6. Azure Stack

Azure Stack extends the Azure vision by bringing the cloud model of computing to every datacenter. Azure Stack is a new hybrid cloud platform product that enables organizations to deliver Azure services from their own datacenters in a way that is consistent with Azure.

Organizations can create these Azure services from datacenter resources, enabling developers and IT professionals to quickly provision and scale services using the same self-service experience found in Azure. This all adds up to an environment in which application developers can maximize their productivity using a "write once, deploy to Azure or Azure Stack" approach, because the Azure APIs are consistent, regardless of where the resources are provisioned—Azure Stack is simply an extension of Azure.

On-premises services for mobile apps: Azure Stack App Service

The *Azure Stack App Service* is the Azure App Service brought to on-premises installations. It includes the web, mobile, and API services. Organizations can create content in Azure Stack App Service using *Azure Resource Manager* (ARM) templates or from the Marketplace, just as they can in Azure.

With Azure Stack developers use APIs that are identical to the ones deployed to Azure App Service in the cloud and create services based on .NET (like ASP.NET Web API) or non-Microsoft technologies (like Node.js) that can easily run on-premises or in the public cloud.

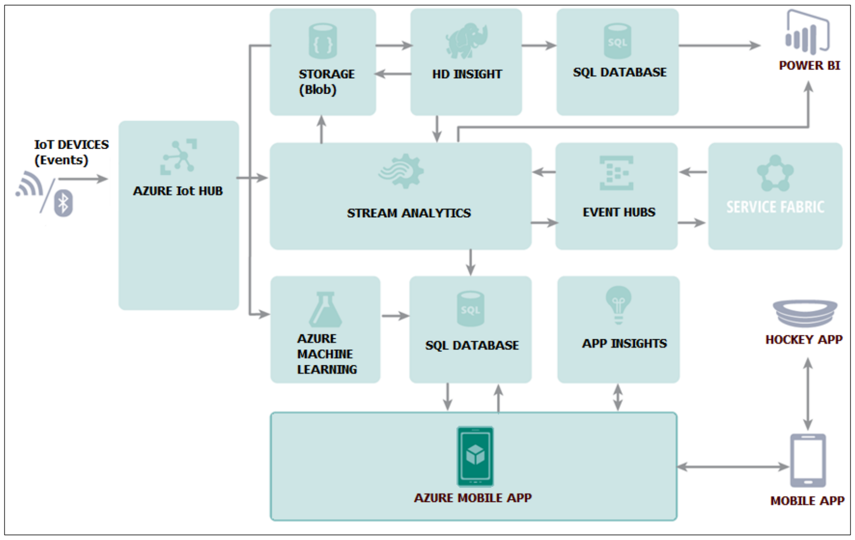
Back-end platform for IoT with Microsoft Azure

Azure IoT Suite is an enterprise-grade solution that enables developers to get started quickly through a set of extensible preconfigured solutions, which address common IoT scenarios, such as remote monitoring and predictive maintenance. These are complete, working end-to-end solutions, including simulated devices that make use of Azure services.

With integration of Azure App Service and mobile apps, developers can connect their IoT devices to ingest data to Azure, perform operations over that data, and expose the data and other APIs to client mobile applications. In particular, the following Azure products are of interest in IoT scenarios:

- ***Azure IoT Hub.*** Developers can easily and securely connect new devices and can connect existing ones, using open source device SDKs for multiple platforms, including Linux and Windows, to reliably (intermittent connection) and securely send commands and notifications to connected devices and to track message delivery.
- ***Azure Event Hubs.*** A highly scalable publish-subscribe service that can ingest millions of events per second and stream them into multiple apps. This lets developers process and analyze the data produced by connected devices and apps and to transform and store it by using any real-time analytics provider or with batching/storage adapters.
- ***Azure Stream Analytics.*** Developers can rapidly develop and deploy low-cost solutions to gain real-time insights from devices, sensors, infrastructure, and applications, such as real-time remote management and monitoring, or gain insights from devices like mobile phones and connected cars.
- ***Azure Machine Learning.*** This powerful cloud-based predictive analytics service makes it possible to quickly create and deploy predictive models as analytics solutions. It provides tools to model predictive analytics but also provides a fully managed service to deploy predictive models as ready-to-consume web services. Azure Machine Learning provides tools for creating complete predictive analytics solutions in the cloud: quickly create, test, operationalize, and manage predictive models.

Figure 3-7. Sample IoT services architecture in Azure



CHAPTER 4

DevOps for mobile

Visual Studio Team Services, Team Foundation Server, Xamarin Test Cloud, and HockeyApp provide a comprehensive ecosystem for developer and IT operations that allow your team to manage projects and to rapidly build, test, and deploy mobile apps and back-end services.

With Visual Studio and Visual Studio Team Services in the cloud, along with Team Foundation Server on-premises, development teams can productively build, test, and release for all target platforms (iOS, Android, and Windows). Teams can manage their sources (via Git or TFS) and can manage their work through scrum and bug-tracking management.

Microsoft tools can automate the pipeline for iOS, Android, and Windows device apps from global builds with VSTS, to test with Xamarin Test Cloud, to deploy to mobile devices with HockeyApp, and to provide feedback and crash analytics about the app back to the developer. Every code commit can trigger a build and deploy the app to test users. Crash data and user feedback with screenshots are directly collected when running the app and fed back into work items for the next cycle.

The complexity of mobile app development increases steadily with new devices, new form factors, and

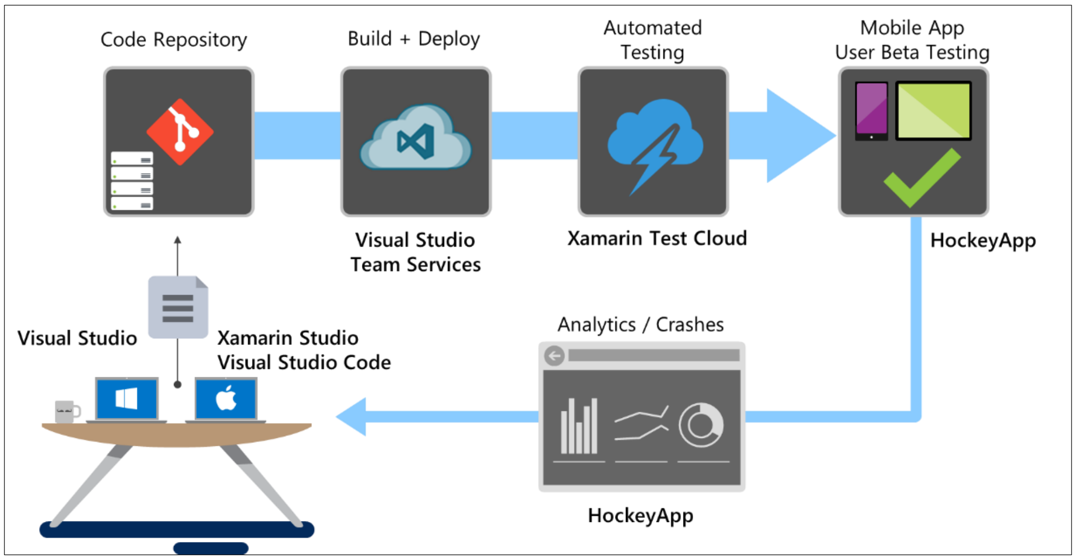


Figure 4-1. DevOps for mobile with VSTS, Xamarin Test Cloud, and HockeyApp

new operating system versions. Android runs on more than 10,000 device models, and even iOS developers now have to consider five different device types and 10 screen resolutions when developing universal apps for *iOS*, *tvOS*, and *watchOS*. To succeed in this environment, your project must automate the whole lifecycle—not only build and deployment but also management of versions and test users, along with the collection of feedback and telemetry. In summary, VSTS offers the following capabilities:

- VSTS/TFS source code management (based on Git or Team Foundation Version Control), agile planning (Agile, Scrum, and CMMI are supported), continuous integration, release management, and other tools for agile teams.
- VSTS/TFS include a powerful and growing ecosystem of first- and third-party extensions that allow you to easily construct a continuous integration, build, test, delivery, and release management pipeline for mobile client apps, (including options to leverage your local Mac or remote Macs for iOS build targets).
- VSTS/TFS builds can stream directly to HockeyApp, which deploys apps to your testers. After your app is installed on a test user's device, HockeyApp collects usage data, crash reports, and user feedback with screenshots and will show an alert when the next build is available. Closing the loop, HockeyApp can automatically create work items for a new crash group or feedback thread and keep the status in sync.
- They tighten the DevOps lifecycle with delivery to release solutions, such as HockeyApp (for testing, A/B experimentation, beta feedback management, and crash data) or to public stores.
- Azure App Service also supports DevOps for the back end, allowing you to configure and define slots for development, staging, and production, integrated from VSTS and allowing you to configure, deploy, and manage mobile services across those slots, for your mobile apps.

Managing teams and projects using VSTS/TFS

With the ability to create an unlimited number of private Git and/or Team Foundation Version Control (TFVC) repositories, VSTS and TFS provide the flexibility needed for teams all of sizes, regardless of whether they prefer distributed or centralized version control. Support for branching and pull requests allows for modern collaboration workflows, and gated builds and code reviews enable the enforcement of best practices.

Be agile, on your terms. Capture, prioritize, and track work with backlogs and customizable Kanban boards. Work items link directly to code to ensure transparency, and they can be used to build rich dashboards for easy reporting.

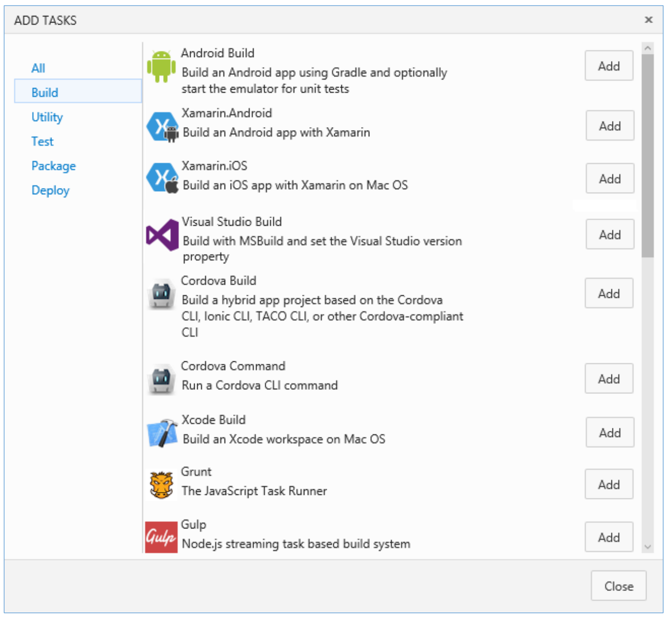
Key benefits include:

- **Flexible version control.** Use Git for distributed version control to maximize collaboration, or use TFVC for centralized version control.
- **Unlimited private repos.** No need to limit your development projects. VSTS/TFS provide teams with the ability to create as many Git or TFVC repos as are needed for any project.
- **Modern collaboration workflows.** Branches isolate risk in a development project. Pull requests provide tools that facilitate collaboration and code reviews for changes being merged back into the mainline.
- **Branch policies.** Enforce best practices by requiring that all code submissions have code reviews, and eliminate build breaks with gated build.
- **IDE integration.** Use your favorite language and development tool. Version control supports any language, as well as any Git client (including Xcode). Java teams can access code and work items through free plugins for Eclipse and IntelliJ and can run continuous integration builds based on config files from Ant or Maven.

- **Build integration.** Create and manage build processes that automatically compile and test your applications in the cloud, either on demand or as part of an automated continuous integration strategy.
- **Backlogs.** Quickly define, prioritize, and decompose the work for your project. Prioritization is easy with drag-and-drop reordering, which helps you keep the most important work at the top of your backlog.
- **Scrum planning.** Scrum teams will feel right at home. Plan sprints using team-based capacity planning, assign work by dragging-dropping, and monitor progress throughout the sprint with real-time burndown charts.
- **Task boards.** Run your sprint using a fit-for-purpose Taskboard, where you can watch the work as it happens. Pivot the board by team member or story, making daily standups quick and efficient.
- **Custom queries.** Queries let you track and organize your data to fit the needs of every project and situation. Create custom queries to look for stale work, impediments which are blocking progress, or backlog items that need attention.

Continuous integration, deployment, and release management using VSTS/TFS, Xamarin Test Cloud, and HockeyApp

Leveraging the comprehensive yet easy-to-use continuous integration support in VSTS and TFS, developers can set up and maintain an automated build and test server for any mobile platform, without needing to write hundreds of lines of custom script code. This lets you spend more of your time building high-quality mobile apps instead of creating the infrastructure to operationalize them. The completely revamped core of VSTS and TFS is fully cross-platform and adopts a lightweight task framework with a library of pre-build tasks and an entire new marketplace of extensions. For the first time, builds can run on Windows, Linux, or Mac machines, opening up mobile device and server-based scenarios like never before.



Android Build
Build an Android app using Gradle and optionally start the emulator for unit tests

Xamarin.Android
Build an Android app with Xamarin

Xamarin.iOS
Build an iOS app with Xamarin on Mac OS

Visual Studio Build
Build with MSBuild and set the Visual Studio version property

Cordova Build
Build a hybrid app project based on the Cordova CLI, Ionic CLI, TACO CLI, or other Cordova-compliant CLI

Cordova Command
Run a Cordova CLI command

Xcode Build
Build an Xcode workspace on Mac OS

Grunt
The JavaScript Task Runner

Gulp
Node.js streaming task based build system

Figure 4-2. Sample Team Build tasks available in VSTS

To enable better release agility, the new Release Management (RM) capability in VSTS/TFS allows you to set up a continuous delivery (CD) server that streamlines the process of publishing updates to beta and public stores, while giving you the control and visibility needed to match your organizational needs.

Key benefits include:

- **Continuous integration.** Simplified task-based experience for setting up a CI server for both native (Android, iOS, and Windows) and cross-platform (Xamarin, Cordova, and React Native) mobile apps, in addition to Microsoft and non-Microsoft (Node.js, Java) based server technologies.
- **Continuous testing.** Builds display integrated test results, which can be run using *Grunt*, *Gulp*, *xctool*, and *Gradle*, among others, and which allow for automatic work item creation when failures occur.
- **Test clouds.** Build/test pipeline integrations with *Xamarin Test Cloud* or with partners, like *Perfecto Mobile*, *Sauce Labs*, and *Keynote*, which allow you to run your integration and UI automation tests as part of your CI builds.
- **Build agents.** Leverage the cloud-hosted build machines provided by VSTS (Windows) and *MacinCloud* (OS X), or configure a self-hosted build agent running on your own infrastructure that integrates with on-premises TFS or cloud-hosted Team Services instances.
- **Continuous delivery.** Automate the deployment of your mobile apps, regardless of whether you're publishing to HockeyApp (for beta), *Google Play*, or the *App Store* (for production). Continuously deploy the service side of your mobile apps using Azure deployment tasks, along with Chef, Docker, and more.
- **Release management.** Configure multiple environments for your app (QA, staging, production), each of which can have pre- and post-approvers to help ensure that updates aren't unintentionally released.
- **Marketplace.** The rich ecosystem of first- and third-party services extensions allows further customization of your CI and CD experience.

Managing a beta release program using HockeyApp

HockeyApp provides a comprehensive solution for management of betas across your applications. All workflows for creating and managing betas are supported.

Upload of beta builds is simple through integration with workflows used by a development team to produce builds, from manual local dev machine builds to integration with CI servers. Easily manage builds by placing them in a central location—the HockeyApp service.

HockeyApp enables direct over-the-air delivery to testers and to internal or external beta customer devices. Through a centralized dashboard, accessible via the web or native mobile apps, testers can have access to beta apps and all available versions.

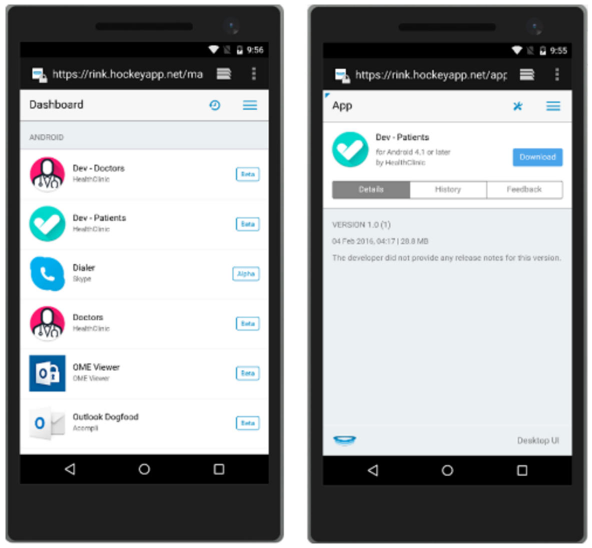


Figure 4-3. HockeyApp app for deployment to devices

Easily keep your users updated via the mobile app notification of new versions or control versions in the wild by requiring users to update to a specific version.

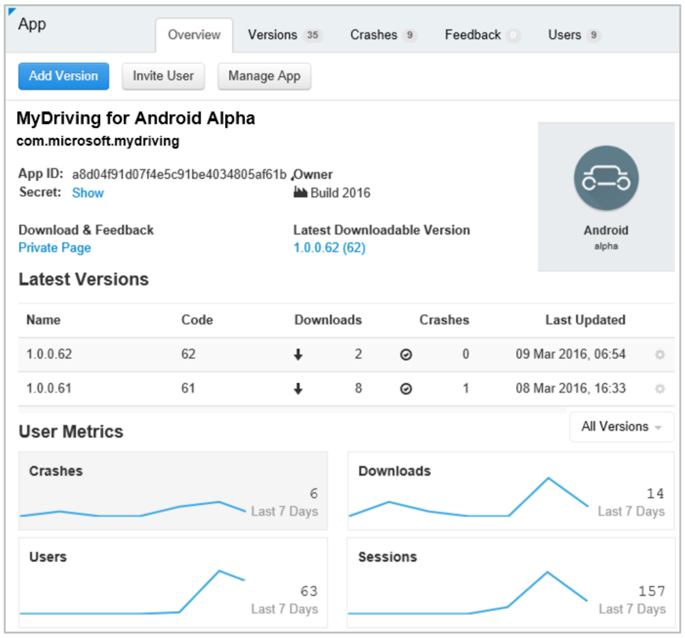
HockeyApp provides a rich set of management tools for users and devices. Users can log in using existing accounts in common SSO providers, such as Google, Facebook, Twitter, Microsoft, and Azure Active Directory. Gain new testers via automatic recruitment and invitation via URL links or email. Control access to apps and beta per user or via teams, with fine-grained controls—even down to specific versions. HockeyApp also collects user device information, helping to manage UDIDs for iOS and to understand the device test coverage.

Operational and behavioral analytics with HockeyApp

Track application health alongside usage metrics and application crash analytics. Get powerful dashboards, where you can filter for different views and levels of detail—from segmenting data to drilling down to an event.

Understanding real-world crash behavior is essential for mobile applications. HockeyApp integration of fully open-source SDKs enables highly reliable automatic crash collection of every application crash. HockeyApp crash collection is supported natively on most major mobile platforms, iOS, Android, Windows, Xamarin, Cordova, React Native, and others.

The HockeyApp dashboard provides the app owner with a summary of crashes and user feedback, including messages from users and automatic screenshots. It groups the crash reports on all platforms by similarities, as shown in Figure 4-4.



MyDriving for Android Alpha
com.microsoft.mydriving

App ID: a8d04f91d07f4e5c91be4034805af61b
Secret: Show

Owner
Build 2016

Download & Feedback
Private Page

Latest Downloadable Version
1.0.0.62 (62)

Latest Versions

Name	Code	Downloads	Crashes	Last Updated
1.0.0.62	62	2	0	09 Mar 2016, 06:54
1.0.0.61	61	8	1	08 Mar 2016, 16:33

User Metrics

Crashes 6 Last 7 Days
Downloads 14 Last 7 Days
Users 63 Last 7 Days
Sessions 157 Last 7 Days

Figure 4-4. HockeyApp portal for mobile app analytics

Enjoy powerful crash analytics tools. Full symbolization support for iOS and .NET native code, and de-obfuscation of ProGuard on Android provides actionable human readable stack traces for all crashes. Automatic crash grouping identifies crashes resulting from the same cause. Integration with most major bug tracking systems allows streamlined workflows for notification and status tracking of crash defects. Utilize the HockeyApp Mac app to view crash reports, and navigate directly to the crashing line of code in your IDE. Understand the device and OS distribution for each crash to gain insight into platform issues. Enable faster triage by getting the details on total impacted users for each crash.

Learn about basic user behavior automatically—without additional coding. Fully understand DAU, MAU, and user sessions for your application and each individual version. Dive deeper into user behavior by tracking any user action as a custom event. Visualize custom event statistics to learn about factors influencing end user behavior. Engage directly with end users via feedback, with bi-directional communication from end users in-app to the development team, including the ability to share screenshots of customer’s experience.

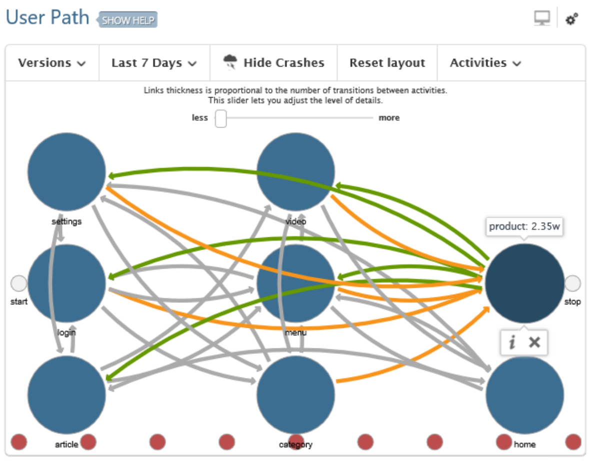
Get mobile customer insights, and leverage marketing campaigns using Azure Mobile Engagement

Azure Mobile Engagement is a SaaS user-engagement platform that provides data-driven insights into mobile app usage by end users, along with real-time user segmentation, and it enables contextually aware push notifications and in-app messaging.

With Azure Mobile Engagement, application publishers and marketing professionals can better understand and interact with the app users.

Real-time actionable analytics to maximize return on investment

Trigger engagement scenarios according to user behavior and demographics, by combining big data collection with real-time message processing. Mobile Engagement can answer nearly any question relevant to your particular business needs. For example, you can create custom dashboards to measure key performance indicators (KPIs), rapidly find and fix usage bottlenecks in a user funnel path, track retention and user stickiness, and then determine which campaigns are driving the highest return on investment. The Mobile Engagement 360-degree user path view allows you to easily and continually enhance and optimize the user experience, driving higher retention rates and improved app usage.



Links thickness is proportional to the number of transitions between activities.
This slider lets you adjust the level of details.

Figure 4-5. User path and app usage analytics

Value-added push and communications platform

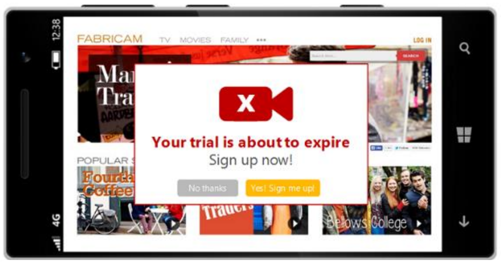


Figure 4-6. Example of marketing campaign based on push notifications created with Azure Mobile Engagement

Mobile Engagement provides in-app messaging capabilities and works seamlessly with native push notifications gateways, such as Google GCM, Apple APNS, and Microsoft MPNS. And it goes beyond, to give you the power to create targeted campaigns by analyzing user behaviors to identify unique customer segments.

This benefits app developers by reaching their customers in a highly effective and non-intrusive manner.

Open APIs and ease of integration

By providing open APIs and SDKs that ease integration, you can leverage data from your existing CRM, CMS, or other back-end systems. This allows you to further improve your audience targeting and protect your investments.

DevOps for back-end mobile services using VSTS, Azure, and Application Insights

Developers and testers can easily and quickly provision production-like dev and test environments using templates in Azure. This pipeline can start with Visual Studio or VSTS by triggering automated builds for every check-in and with automated deployments (CD).

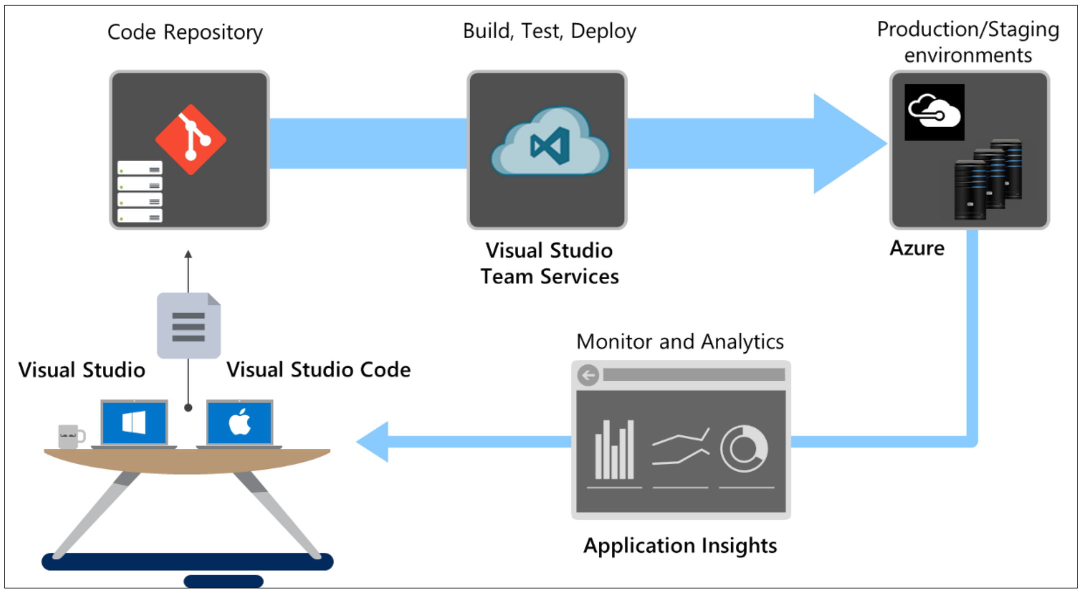


Figure 4-7. DevOps for services with VSTS, Azure, and Application Insights

Developers can create automated tests at every stage of deployment and can define necessary approvals before and after deployments.

- **Azure Resource Manager.** Developers can easily provision virtual machines and any Azure PaaS components using ARM templates saved in source control with tools they are already comfortable working with.
- ***Azure Marketplace* templates.** Developers can leverage the cloud as a test harness for a number of real world situations, such as testing a large number of requests and multiple clients, simulating realistic and distributed load, at the same time validating the scale and tolerance of your application. Azure Marketplace templates have a host of test frameworks and services that enable developers to do functional and performance testing. Customers can leverage the on-demand and scalable infrastructure of Visual Studio Team Services load testing service to generate unlimited virtual users for your application from various Azure datacenters across the globe.

Analyzing the health of back-end services using Visual Studio Application Insights

Application Insights is an extensible analytics service that monitors your live application. It helps you detect and diagnose performance issues and to understand what users actually do with your app. It's designed for developers, to help you continuously improve the performance and usability of your services or applications.

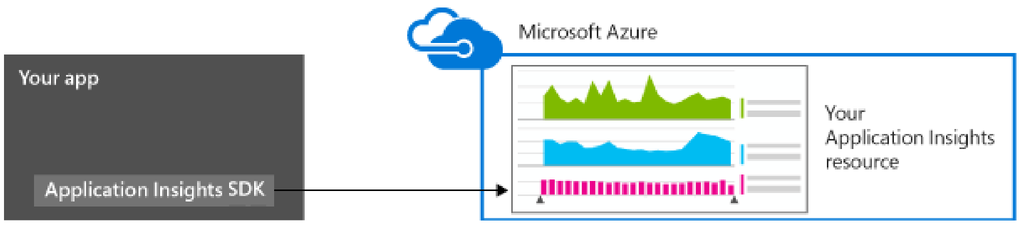


Figure 4-8. Application Insights

It works with both web/services and stand-alone apps on a wide variety of platforms: .NET or J2EE, hosted on-premises or in the cloud.

Application Insights is aimed at the development team. With it, you can:

- Analyze usage patterns to understand your users better and to continuously improve your app.
- Count page views, new and returning users, geolocation, platforms, and other core usage statistics.
- Trace usage paths to assess the success of each feature.
- Detect, triage, and diagnose performance issues, and fix them before most of your users are aware.
- Get alerts on performance changes or crashes.
- Explore metrics to help diagnose performance issues, such as response times, CPU usage, and dependency tracking.
- Get availability tests for web apps.
- Gain insight from exception reports and alerts.
- Access the powerful diagnostic log search (including log traces from your favorite logging frameworks).

CHAPTER 5

Securing and managing mobile apps

Developers can build secure mobile apps, and IT managers can feel confident by securely managing mobile apps and devices in an enterprise BYOD environment, along with their desktop infrastructure.

Securing mobile apps

With the range of possible mobile app technologies and the mobile back ends, Microsoft provides recommendations and solutions for developers creating their mobile app strategy. This ranges from components to create mobile apps that securely manage data (such as data encryption) or to provide authentication of the user (like SSO with Azure Active Directory), to the ability for IT managers to securely manage the deployment of apps and devices in a corporate environment, and it extends through MAM/MDM capabilities with Microsoft Intune and Microsoft Enterprise Mobility.

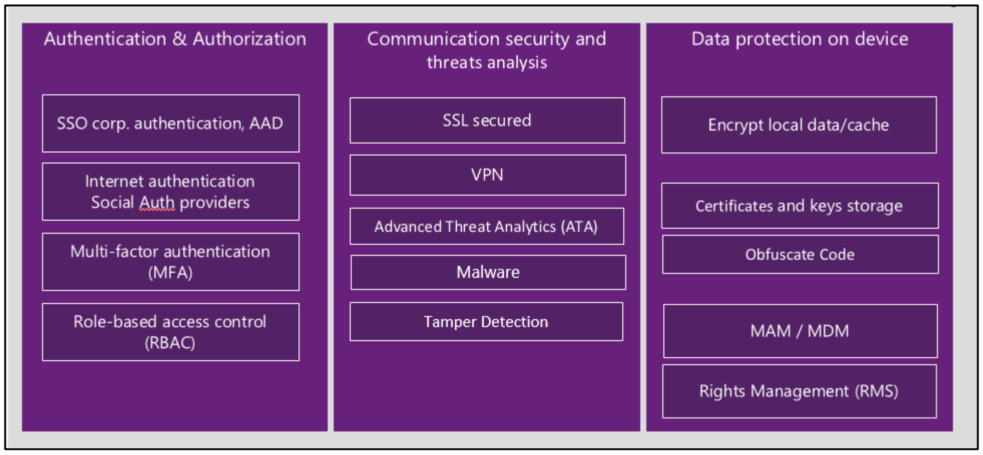

Authentication & Authorization
SSO corp. authentication, AAD
Internet authentication Social Auth providers
Multi-factor authentication (MFA)
Role-based access control (RBAC)
Communication security and threats analysis
SSL secured
VPN
Advanced Threat Analytics (ATA)
Malware
Tamper Detection
Data protection on device
Encrypt local data/cache
Certificates and keys storage
Obfuscate Code
MAM / MDM
Rights Management (RMS)

Figure 5-1. Main security pillars

There are many security considerations to take into account in a mobile app strategy that cover application scenarios, IT and developer scenarios, as shown in Figure 5-1 and explained afterwards.

- User **identity, SSO, authorization and authentication**. Azure App Service and the SDKs allow developers to authenticate users through a number of means, such as Active Directory, Azure Active Directory for SSO support, or other social login systems, such as Twitter, Facebook, and Google, and includes more fine-grained or higher levels of security through multi-factor authentication (MFA) and roles-based access to APIs. Azure App Service provides developers a simple authentication process to handle the management of tokens and to facilitate federated authentication across enterprise back-end systems and SaaS providers.

- **Back-end API management.** Use Azure API Management to drive API consumption among internal teams, partners, and developers, while benefiting from the business and operational insights available in the Azure portal. API Management gives you the tools you need for end-to-end API management—provisioning user roles; creating usage plans and quotas; applying policies for transforming payloads; and setting up throttling, analytics, monitoring, and alerts.
- **Data at rest** (local encryption) or **in motion** (communication). Azure Mobile App SDK provides the means to securely encrypt data stored locally on the device during offline/online scenarios, but developers can also make use of SDKs, Cordova plugins, and more, for specific app scenarios with and without Azure SDKs.
- **Tamper** and **malware detection**.
- **Keys, certificates, and secrets.** *Azure Key Vault*.
- **Application and device management** (MAM/MDM).

Identity, SSO, authentication, and authorization

- **SSO corp. authentication and authorization, Azure Active Directory, and Active Directory Federation Services.** Active Directory provides an industry-leading identity server, both in the cloud and on-premises, through Azure Active Directory and Active Directory Federation Services (ADFS*)*. Developers can securely authenticate, authorize, access information in AD and can take advantage of device-level SSO and MFA capabilities, along with storage through the powerful *Active Directory Authentication Library* (ADAL) available for all major native and cross-platform mobile and server-side technologies. In addition, Intune MAM features include the ability to force authentication against AD for any app, further enhancing an enterprise's ability to control access to sensitive apps and data. Additionally, products like *Azure Active Directory Identity Protection* help security-conscious organizations implement Microsoft identity as a service (IDaaS) solutions with confidence.
- **Internet authentication social authentication providers.** *Azure App Service Authorization* provides a unified, simplified mechanism for authenticating against Azure AD, Facebook, Twitter, Google, and Microsoft accounts—from not only services and web apps but also mobile apps, through the use of Azure Mobile Apps libraries, plugins, and SDKs for Android, iOS, Windows, Xamarin, and Cordova. The common interface means that developers are abstracted from provider interface changes and will be able to instantly take advantage of new auth providers as they come online in the service. Azure AD now also has a preview of B2C support, enabling you to manage logins, using credentials from Facebook, Google, LinkedIn, Amazon, and Microsoft accounts, and to take advantage of the same power of Azure AD available for enterprise accounts. It is currently in preview for Android, iOS, and Windows native apps, with other technologies coming soon.
- **Role-based access control (RBAC), Rights Management**. Azure AD and ADFS are essential directories for configuring RBACs for mobile apps. Organizations can establish specific roles that have access to an app through Active Directory groups that can then be validated using the AD Graph API. In addition to app level access controls, the broad ADAL support for server-based technologies, like .NET and Node.js, allow organizations to further enhance their

security by using these same capabilities to provide or restrict access to specific server-side data sources. In the cloud, Azure Mobile Apps can further streamline the process of getting up and running with RBACs through the use of features like Easy Tables that set up all the infrastructure needed while still enabling developers to implement customized authorization controls. Finally, Azure Rights Management is a comprehensive cloud service that enables tight role-based controls to Office, SharePoint, and *OneDrive* persisted data that extends to apps using Office APIs to access data.

Communication security

- **Key management.** SSL provides an essential building block for all secure communications, and all Azure services support SSL-based communication. Azure Key Vault can help improve overall network-based communication security by centralizing sensitive certificates, keys, and secrets in a secure, audited hardware- or software-based central service and by removing the need for apps to persist these values directly. The centralization allows organizations to regularly change keys globally without having to redeploy apps or services, in addition to providing a more granular level of control over the keys themselves.
- **VPN and Wi-Fi access.** Microsoft Intune can provide additional peace of mind by enforcing mobile device resource access control policies. Intune allows you to require VPN or secure Wi-Fi access to connect to key services, helping you to manage device profiles.

Threats analysis

- **Identity protection.** Azure Active Directory Identity Protection takes secure identity and access management to the next level by detecting attacks in real time, informing you of risks and applying controls to help keep your enterprise safe. The service detects suspicious activities, based on signals like brute force attacks, leaked credentials, sign-ins from unfamiliar locations, infected devices, and more, and provides remediation recommendations to protect against these activities in real time. Based on these suspicious activities, a user risk severity is calculated, and risk-based policies can be configured and can automatically protect the identities of your organization from future threats. These **risk-based policies**, in addition to other conditional access controls provided by Azure AD and other EMS services, can either block or provide adaptive remediation actions that include password reset requests and MFA. The service is built on a decade of Microsoft experience in protecting consumer identities, and it has special features to reduce false positive rates and noise.
- **On-premises threat detection.** Microsoft Advanced Threat Analytics (ATA) is an on-premises product that can help customers protect their enterprise from advanced targeted attacks by automatically learning, analyzing, and identifying normal and abnormal entity (user, devices, and resources) behavior. ATA leverages deep packet inspection technology, as well as information from additional data sources (SIEM and Active Directory) to build an organizational security graph and to detect advanced attacks real time. The solution is agnostic to the device type and operating system version—ATA witnesses all authentication and authorization.
- **SaaS/Cloud Application Security.** Microsoft Cloud App Security, based on the Microsoft Adallom acquisition, is a comprehensive cloud service that provides deeper visibility, stronger controls, and increased security for the cloud applications.

- **Malware detection.** When building an internal-facing app, Intune mobile device management and mobile application management solutions detect malware on Android and report jailbroken or rooted devices for iOS and Android. Intune MAM capabilities can also be used on their own or to complement an existing MDM solution.

Data protection on device (data at rest)

Although base device encryption capabilities and cross-platform plugins and components can provide a certain degree of security for app developers on their own, the **Microsoft Intune** MAM features provide the ability to enforce policies at the **app level**, including encryption of all local data. It's therefore a low friction way to increase your security.

Intune provides two solutions for enabling its MAM features for Android and iOS devices: an app-wrapping tool and an app SDK. The app-wrapping tool can be run on internal LoB Android and iOS apps to enable certain capabilities, such as limiting cut-copy-paste while the app is running, requiring a PIN, or requiring app-level encryption. The *Intune App SDK* takes this a step further, allowing apps to target corporate data only, leaving personal data completely intact, and ensuring that users cannot store their data in personal app-connected services.

Managing client mobile apps and devices

As part of Enterprise Mobility Suite (EMS), Microsoft Intune provides enterprises with the ability to manage mobile devices, PCs, and apps so that employees are productive and a company's information remains protected.

Mobile device management. IT admins use mobile device management to manage, monitor, and secure employees' mobile devices. Intune supports MDM of iOS, Android, Mac, and Windows devices. IT admins can manage both personal and corporate devices, with users' ability to enroll their devices and install apps with Intune via company portal app. Using the Intune admin console, IT admins can manage a variety of scenarios, including policies, corporate e-mail, certificates, VPN, and many more device settings. They can also use Intune Conditional Access policies to control access to on-premises Microsoft Exchange e-mail from mobile devices.

Mobile app management. MAM is typically used for deploying and managing employees' apps. Intune MAM allows IT admin to deploy and manage both store and internal company apps, while ensuring that data remains protected in these apps through MAM policies configured by the admin.

Intune MAM also offers a variety of options for data protection. Apps can be managed **with and without MDM**, allowing companies with BYOD devices to manage apps with little friction for their employees. MAM can also be deployed alongside third-party MDM providers with a goal to meet customers' needs, regardless of where they are. In addition, IT admins can manage a large ecosystem of MAM-enabled productivity apps, including Office apps and third-party apps, such as Adobe Reader and Box.

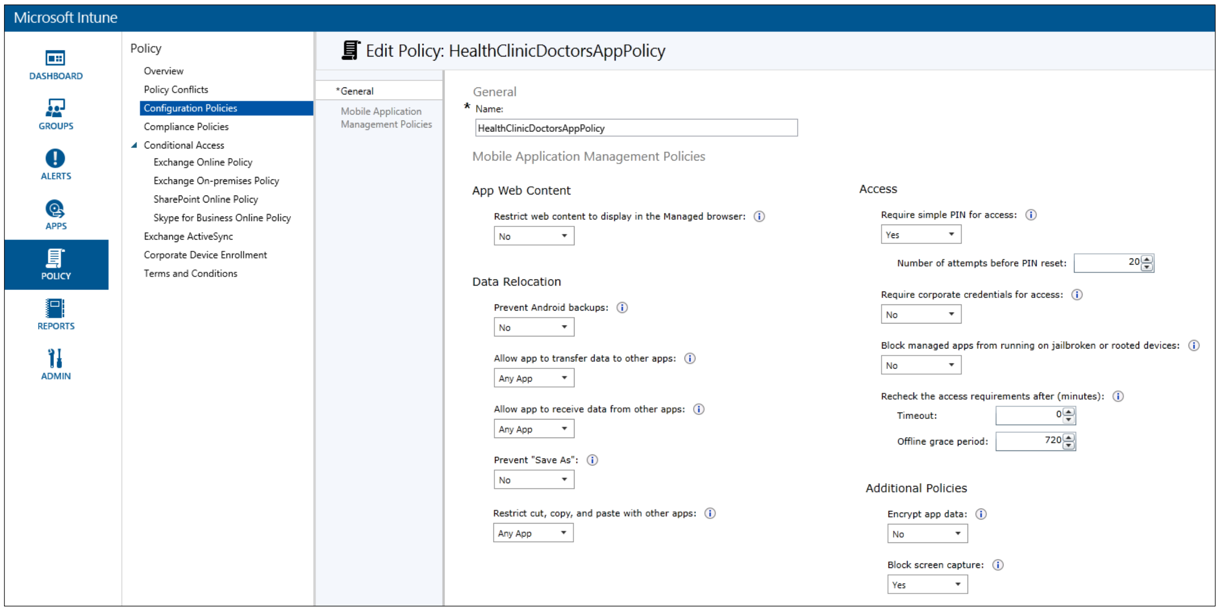
Developers can enable their mobile apps for MAM through Intune App SDK, which supports native iOS and Android, in addition to Cordova and Xamarin. IT admins can also utilize an app-wrapping tool to enable compiled LoB apps for management. After an app has been MAM-enabled, admins can deploy a variety of policies to protect corporate app data, leaving personal data intact, to help ensure an uncompromised end-user experience.

Securing mobile apps end to end

IT admins can deploy MAM policies that protect data and help ensure secure access. Intune App SDK typically handles the implementation of these policies, creating a consistent experience for admins across apps and allowing developers to quickly enable MAM.

The Intune App SDK (MAM) supports Xamarin apps, Cordova apps, and single-platform language apps, like *Objective-C* apps for iOS and Java apps for Android.

Figure 5-2 shows an example of an Intune policy being edited with available security options which can be applied in a decoupled way (no need to code) in mobile apps managed by Intune.



Edit Policy: HealthClinicDoctorsAppPolicy

General

* Name: HealthClinicDoctorsAppPolicy

Mobile Application Management Policies

App Web Content

Restrict web content to display in the Managed browser: No

Data Relocation

Prevent Android backups: No

Allow app to transfer data to other apps: Any App

Allow app to receive data from other apps: Any App

Prevent "Save As": No

Restrict cut, copy, and paste with other apps: Any App

Access

Require simple PIN for access: Yes

Number of attempts before PIN reset: 20

Require corporate credentials for access: No

Block managed apps from running on jailbroken or rooted devices: No

Recheck the access requirements after (minutes):

Timeout: 0

Offline grace period: 720

Additional Policies

Encrypt app data: No

Block screen capture: Yes

Figure 5-2. Sample policy for Intune App SDK in the Intune portal

After an app is MAM-enabled, an IT admin can:

- **Control users' ability to move corporate documents.** Admins can deploy a policy that disables file backup apps to prevent backing up corporate data to the cloud.
- **Configure clipboard restrictions.** They can deploy a policy so that end users are unable to use the clipboard to cut/copy from an Intune-managed app and to paste into a non-managed app.
- **Enforce corporate access requirements.** Admins can require an access challenge to the user, such as full authentication or app PIN, to launch the app. Authentication relies on users' AD identity and therefore can benefit from all AD identity features. Intune App SDK uses Azure AD to provide an SSO experience, in which the credentials, after they are entered, are reused for subsequent logins. Authentication of identity management solutions federated with Azure AD are also supported.
- **Enforce encryption on saved data.** Admins can enforce a policy which ensures that all data stored on the device by the app is encrypted.
- **Remotely wipe corporate data.** Corporate data can be remotely wiped from an Intune-managed app when the device is unenrolled from Microsoft Intune. This feature is identity-based and deletes only the files that relate to the corporate identity of the end user.
- **Enforce the use of a managed browser.** Using the Intune-managed browser helps to ensure that links which appear in emails (in an Intune-managed mail client) are kept within the domain of Intune-managed apps.
- **Check device health and compliance.** Admins can check the health of the device and its compliance with corporate policies before end users access Intune-managed apps. On the iOS platform, this policy checks whether the device has been jailbroken. On the Android platform, this policy checks whether the device has been rooted.

Conclusions

Enterprise mobile apps are constantly evolving, with high expectations from the business, either around improved quality or reduced time to market. To survive in this new era, mobile app development leaders must be able to establish an end-to-end development strategy based on platforms which cover every area in the app's lifecycle but which are also flexible and interoperable with other platforms.

Key takeaways

- **Cross-platform world.** Mobile devices and operating systems are fragmented across different vendors (iOS, Android, and Windows). As soon as an enterprise targets more than one platform, "siloed" development approaches (different skills and development teams per OS) provoke very high costs. A cross-platform mobile strategy can help lower your TCO by consolidating skills and development teams.
- **Interoperability and multichannel.** No single development platform or approach solves all mobile app use cases in the enterprise. Therefore, platforms and technologies (not only for developing client apps and back-end services but also for DevOps, security, and management) must be able to interoperate and integrate with different vendors and technologies. Interoperability must be driven by a multichannel strategy based on back-end services/API, consumable from mobile apps and web apps, and based on any platform/technology.
- **Flexibility.** Mobile app technologies (mobile devices, languages, and operating systems) evolve fast, compared to traditional development. This can lead to the instability of many tools (especially OSS platforms that are improved in a very dispersed way), resulting in the need for the enterprise to adopt a flexible architecture strategy with agile tactics across different vendors, rather than rigid end-to-end bounded platforms.
- **App lifecycle improvement.** The increasing demand for mobile apps means that you need to accelerate app development and delivery while still assuring quality. This can be achieved by improving productivity in the development processes, increasing the collaboration between development and IT operations (DevOps), and having an agile application lifecycle with fast but reliable continuous delivery. Agile frameworks supported by automated build/test/delivery tools, plus production management tools, are fundamental for achieving those goals.

As mentioned at the beginning of this paper, Microsoft offers a complete foundation for a mobile app development strategy—a collection of technologies to choose from, which integrate with existing tools and processes. This flexibility in a broad approach and this strength in the depth of capabilities provide the ability to adopt the E2E platform covering the whole lifecycle or to adopt only specific technologies complementing your existing environment, depending on your needs.

Appendix: Technology decision tables

Mobile app development technologies

Microsoft offers a variety of technologies and tools to support the approach you take for cross-platform (web, hybrid, or native cross-platform). No matter which development platform and technology you select, they are supported by Visual Studio—one of the most powerful integrated development environments available and one that provides one of the highest quality development experiences.

As far as frameworks, components, and libraries, Microsoft supports a large variety of them. These vary, depending on the specific approach (hybrid and cross-platform versus native and cross-platform).

Hybrid and cross-platform: Visual Studio Tools for Apache Cordova

Development platforms and technologies for hybrid/Cordova mobile apps (iOS, Android, Windows)

Technologies	When to use and why
Apache Cordova	• Leverage your HTML/JavaScript skills. • Appropriate when building mobile apps, like B2E or B2B apps. • It allows you to create mobile apps that are fully installed in the device so you can also have offline scenarios.
Cordova plugins	• It is part of the Apache Cordova platform. • A plugin is a bit of add-on code that provides a JavaScript interface to native components. • Plugins allow your app to use native device capabilities beyond what is available to pure web apps.

References	
Apache Cordova	https://cordova.apache.org/
Cordova plugins	https://cordova.apache.org/plugins/

Microsoft development tools for hybrid/Cordova

Technologies	When to use and why
Visual Studio Tools for Apache Cordova Visual Studio	• Appropriate when you want to take advantage of a full IDE like Visual Studio. • It allows you to edit code with IntelliSense and to debug JavaScript/TypeScript. • in apps running in emulators or real devices, configure Cordova projects and plugins easily in project settings. • Take advantage of your Visual Studio skills and insights.
Visual Studio Code + **Cordova Tools Extension** + **Cordova CLI (Command Line Interface)** Visual Studio Code	• Appropriate when you want to use a light editor, like Visual Studio Code, which is also cross-platform (Visual Studio Code runs on Windows, Mac, and Linux). • It allows you to debug your code, find commands in the Command Palette, and use IntelliSense.

References

Visual Studio Tools for Apache Cordova	https://www.visualstudio.com/en-us/features/cordova-vs.aspx
Visual Studio Code	https://code.visualstudio.com/
Cordova Tools Extension	https://marketplace.visualstudio.com/items?itemName=vsmobile.cordova-tools
Cordova CLI (Command Line Interface)	https://github.com/apache/cordova-cli

Languages for hybrid mobile apps (iOS, Android, Windows)

Languages	When to use and why
JavaScript JS	• Appropriate when building typical Cordova apps which are not very complex in the JavaScript layer. • Take advantage of your web development and JavaScript skills.
TypeScript TypeScript	• TypeScript is a programming language created by Microsoft that is a superset of JavaScript. You can develop a Cordova app with TypeScript that will compile into simple JavaScript to be deployed as part of your app. • Appropriate when building complex business applications with a heavy volume of client code. • TypeScript allows you to have a better structured code, thanks to certain object orientation, based on classes, modules, and interfaces.

References

JavaScript	https://www.javascript.com/
TypeScript	http://www.typescriptlang.org/

Main web frameworks for Apache Cordova

Frameworks	When to use and why
AngularJS	• AngularJS is a very popular JavaScript framework created by Google. It is a library written in JavaScript (although Angular 2 is written in TypeScript). • It allows you to have dynamic views in your app. AngularJS lets you extend HTML vocabulary for your application. • Appropriate when creating Cordova apps, either with JavaScript or TypeScript. • Take advantage of a resulting environment that is expressive, readable, and quick to develop.
Ionic	• Ionic is a front-end SDK for developing hybrid mobile apps. It offers a library of mobile-optimized HTML, CSS, and JavaScript CSS components, gestures, and tools for building highly interactive apps. • Built with Sass and optimized for AngularJS. • Most popular choice when developing Cordova apps with AngularJS and/or jQuery. • Take advantage of your web development skills.
Onsen UI	• Onsen UI framework is designed and implemented to deliver a positive user experience for your hybrid apps. • Appropriate when developing Cordova apps with AngularJS and/or jQuery.
Backbone	• Backbone.js gives structure to web applications by providing models with key-value binding and custom events, collections with a rich API of enumerable functions, and views with declarative event handling. • Appropriate when creating rich client-side applications. • It allows you to create a structured code, decoupling views from models.

References

Angular.js	https://angularjs.org/
Ionic	http://ionicframework.com/
Onsen UI	https://onsen.io/
Backbone	http://backbonejs.org/

Native and cross-platform: C#—Visual Studio with .NET and Xamarin

Development platforms and technologies for C# cross-platform mobile (iOS, Android, Windows)

Technologies	When to use and why
PCL (Portable Class Library)	• Leverage your .NET/C# skills, and share code across platforms. • Share key artifacts for C# cross-platform across Windows, iOS, and Android. • Appropriate when developing with traditional Xamarin, Xamarin.Forms, Windows 10 UWP, and .NET. • It allows you to share most of the C# logic across platforms, like Model-View-ViewModel (MVVM), and even the views/XAML, if using Xamarin.Forms.
Xamarin **Xamarin.Forms**	• Leverage your .NET/C# skills while building native apps. • **Traditional Xamarin** ○ It allows you to create apps for **iOS**, **Android**, and **Mac**, with specially tailored UI per platform (it is possible to have a different UI/views per platform), while sharing the same C# logic. ○ It should be used in conjunction to PCL to share approximately 80 percent of the code (C# logic) between Xamarin and UWP/.NET apps. ○ Appropriate when building mobile apps with the best possible UI and performance, like B2C apps. • **Xamarin.Forms** ○ It allows you to share most of the implementation (approximately 95 percent, including C# and XAML views) in a PCL, when creating apps for **iOS**, **Android**, and **Windows 10 UWP**. ○ Appropriate when building apps for iOS, Android, and Windows, which will have the same UI/views with almost no differences in layout across platforms. ○ Can be used in a mixed approach (Xamarin + Xamarin.Forms) so you can get the best of both worlds—using Xamarin.Forms for simpler views and traditional Xamarin with native views and UWP views, depending on the platform.
UWP with .NET	• Leverage your .NET/C# skills. • **Universal Windows Platform** (UWP) allows you to create the same app running on **Windows 10** (desktop/tablet) and **Windows 10 Mobile**. • It should be used in conjunction with PCL, to share approximately 80 percent of the code (C# logic) between Xamarin and UWP/.NET apps. • Appropriate when building mobile apps with the best possible UI and performance, like B2C apps.

References

Xamarin	https://xamarin.com
Xamarin.Forms	https://xamarin.com/forms
UWP with .NET	https://dev.windows.com/en-us/windows-apps
PCL and cross-platform	https://msdn.microsoft.com/en-us/library/gg597391(v=vs.110).aspx

Development tools for Xamarin (iOS, Android, Windows)

Technologies	When to use and why
Microsoft Visual Studio with Xamarin Extension	• Appropriate when you want to take advantage of a full IDE, like Visual Studio. • It allows you to visually design views with the iOS and Android, edit code with IntelliSense, and debug C# code in apps running in emulators or real devices. • Take advantage of advanced Visual Studio features and your C# skills.
Xamarin Studio	• Appropriate when you want to use an IDE that works cross-platform on Mac OS X and Windows. • It allows you to visually design views with the iOS and Android, edit code with IntelliSense, and debug C# code in apps running in emulators or real devices. • Take advantage of your C# skills.

References

Microsoft Visual Studio plus Xamarin Extension	https://www.visualstudio.com/en-us/features/xamarin-vs.aspx/
Xamarin Studio	https://xamarin.com/studio/
Apple Xcode	https://developer.apple.com/xcode/

Main frameworks and SDKs for C#/Xamarin/.NET

Frameworks	When to use and why
MVVM built-in in Xamarin.Forms	• Appropriate when using Xamarin.Forms and you want to implement a clean and simplified MVVM architecture for your client apps targeting iOS, Android, and Windows UWP. • It allows you to use an MVVM framework that comes out-of-the-box with Xamarin.Forms.
MvvmCross	• Appropriate when using traditional Xamarin and .NET for Windows UWP and you want to implement a MVVM architecture for your client apps targeting iOS, Android, and Windows UWP. • MvvmCross is a popular and open source MVVM framework that has been cross-platform since its creation and has been tested in many cross-platform apps.
MVVM light toolkit	• Appropriate when using traditional Xamarin and .NET for Windows UWP and you want to implement a MVVM architecture for your client apps targeting iOS, Android, and Windows UWP.

MVVM light toolkit	• MVVM light toolkit is a popular and open source MVVM framework that was widely used for WPF, Silverlight, Windows Store, and for Windows Phone. It added support for Xamarin.Android and Xamarin.iOS as a cross-platform framework in v5 in 2014.
Prism for Xamarin.Forms	• Appropriate when using Xamarin.Forms and you want to implement composite apps with MVVM architecture for your client apps targeting iOS, Android, and Windows UWP. • Prism is not just an open source MVVM framework—it also offers additional capabilities, like dependency injection, commands, EventAggregator, and other capabilities. • Prism was originally created by the Microsoft Patterns & Practices team, was open sourced in 2015, and has been driven by an independent team ever since.
Azure Mobile Apps .NET Client SDK	• Appropriate when consuming Azure Mobile App services from Xamarin or .NET apps. • It allows you to add a scalable back end to your connected client applications and to have structured storage, authentication, push notifications, and offline scenarios (automatic data-sync between local SQLite in the device and Azure SQL Database in the cloud) to your Xamarin- or .NET-based mobile apps using Microsoft Azure Mobile Apps.
Akavache Cache and local store	• Appropriate when implementing client cache/store for native apps, like Xamarin apps (iOS and Android) and .NET apps (Windows and UWP apps). • Akavache is an asynchronous, persistent (that is, writes to disk) key-value store created for writing apps in C#, based on SQLite. • Akavache is great for both storing important data (like user settings), as well as cached local data that expires.
SQLite.NET	• Appropriate when storing relational SQL data locally in the mobile device into a SQLite database. • SQLite.NET is an open source, minimal library to allow .NET and Xamarin apps to store data in SQLite databases. • It is a thin, fast, and efficient library, not a full SQLite driver. If you need that, use Mono.Data.SQLite or csharp-sqlite.
SQLCipher	• Appropriate for Xamarin and .NET apps when you need to secure/encrypt SQLite database in the device. • It allows for transparent and secure 256-bit AES encryption of SQLite database files. • SQLCipher has a small footprint and great performance. As such, it's ideal for protecting embedded application databases and is well-suited for mobile development.
FileDb NoSQL .NET database	• Appropriate when storing NoSQL data locally in the mobile device into local files. • FileDb is a simple database solution for .NET and Xamarin apps. • FileDb is a NoSQL database meant for use as a local data store for applications. • Take advantage of LINQ to join tables, and get all the relational and grouping power that LINQ offers.

References

MVVM built-in in Xamarin.Forms	https://developer.xamarin.com/guides/xamarin-forms/user-interface/xaml-basics/data_bindings_to_mvvm/

MvvmCross	http://mvvmcross.com/
MVVM Light	http://www.mvvmlight.net/
Prism for Xamarin.Forms	https://github.com/PrismLibrary/Prism
Azure Mobile Apps .NET Client SDK	https://components.xamarin.com/gettingstarted/azure-mobile-client/
Akavache Cache	https://github.com/akavache/Akavache/
SQLite.Net	https://github.com/praeclarum/sqlite-net/
SQLCipher	https://www.zetetic.net/sqlcipher/
FileDb – NoSQL db	http://www.eztools-software.com/tools/FileDb/default.asp

Rapid Mobile App Development (RMAD) with Microsoft PowerApps

PowerApps-related technologies

Technologies	When to use and why
PowerApps	• PowerApps is an enterprise service for LoB and IT analysts and developers to connect, create, and share business apps across an organization on any device in minutes. • The PowerApps tool allows you to create mobile apps that can run on iOS, Android, and Windows.
Swagger	• To consume HTTP services from PowerApps (like services in Azure App Service), those services need to expose Swagger metadata so PowerApps can discover what can be consumed. • Swagger is a very popular RESTful API description metadata so services can be discoverable from the outside. It is basically what Azure API Apps, Azure Logic Apps, and PowerApps use to understand how to use services/APIs and to connect to them. • Azure API Apps already provide Swagger metadata by default, but if you want to consume Azure Mobile App services (or any other HTTP service, like a regular ASP.NET Web API service) from PowerApps, you need to add functionality so they provide Swagger metadata.
Swashbuckle	• Swashbuckle is a convenient way to rapidly and automatically generate Swagger metadata from a Web API .NET project, like any ASP.NET Web API service or an Azure Mobile App service. • Swashbuckle is basically a NuGet component that you can add to your Web API service so it automatically generates the Swagger metadata related to your methods (no need to manually generate it).

References	
PowerApps	http://powerapps.microsoft.com/
Swagger	http://swagger.io/
Swashbuckle	https://github.com/domaindrivendev/Swashbuckle/

Back-end and cloud services technologies

Microsoft offers a large variety of technologies and tools to be used when creating back-end and cloud services. Microsoft covers the most important approaches for mobile back ends (MBaaS, PaaS, and microservices, among others) and for IoT back ends.

PaaS and MBaaS

Azure App Service / Azure Stack App Service–related technologies

The Azure App Service technologies can be used from the Azure public cloud (in Azure App Service) and from the on-premises implementation of Azure, called Azure Stack (in PREVIEW, as of H1 2016).

Technologies	When to use and why
Azure Web Apps	• A web or service deployed into Azure App Service. You can deploy ASP.NET MVC apps, plain HTML web apps, Node.js, Java, PHP, Python web/services, and more. • Appropriate when you just need to deploy a web/service into the cloud. • It allows you to easily manage your web app configuration from Azure portal and to deploy directly from Visual Studio when you are developing.
Azure Mobile Apps	• An Azure App Service Mobile App is internally similar to an Azure Web App, but it is specially made to be consumed by mobile apps. • It allows you to deploy services (based on ASP.NET Web API or Node.js) but Azure Mobile Apps provide additional capabilities, like offline scenario (sync local device database with database in the cloud), simplified push notification implementation based on Azure Push Notification hub, and an easy implementation of Internet authentication (Facebook, Google, Twitter, Microsoft) or enterprise Azure Active Directory authentication. • Appropriate when the services you want to deploy will be consumed by any mobile app (including native, cross-platform, or hybrid.).
Azure API Apps	• The API Apps support within Azure App Service enables you to easily create, consume, and call APIs. Azure API Apps provide metadata describing your services (based on Swagger) so other apps can easily discover what your services are offering. • It enables you to easily expose and integrate APIs across a wide variety of languages. • Discoverability and integration features integrate API Apps with Logic Apps. • Appropriate when consuming services (API Apps) from apps like Logic Apps or PowerApps.
Azure Logic Apps	• Logic Apps enable you to automate workflows and business processes. • It allows you to configure workflows that integrate and transform data between LoB systems (like Microsoft Dynamics or Oracle) with SaaS systems (like Office 365, Salesforce, or Twitter) or your custom applications/services. • Take advantage of already available SaaS/LoB connectors or custom API Apps so you can easily build integration systems.

References	
Azure App Service Web Apps	https://azure.microsoft.com/en-us/services/app-service/web/
Azure App Service Mobile Apps	https://azure.microsoft.com/en-us/services/app-service/mobile/
Azure App Service API Apps	https://azure.microsoft.com/en-us/services/app-service/api/
Azure App Service Logic Apps	https://azure.microsoft.com/en-us/services/app-service/logic/

PaaS: Microservices and hyperscale

Azure Service Fabric technologies

Azure Service Fabric is a platform especially made for hyperscale and microservices architecture–based applications. It offers several APIs and programming model options, depending on the needs.

Technologies	When to use and why
Stateless Reliable Services	• Your stateless services in Azure Service Fabric are implemented in a way that is similar to that of other types of cloud/servers (like any ASP.NET Web API service). • Appropriate when you just want to deploy regular stateless services into Azure Service Fabric and the data is stored in any external database (SQL or NoSQL). • Take advantage of Azure Service Fabric hyperscale capability based on its cluster of services and advanced management for deployment, updates, and monitoring.
Stateful Reliable Services	• A *stateful service* means that the data resides within the same microservice's process, in memory, persisted on the hard drive and replicated to other nodes in the cluster. • Use Stateful Reliable services when you need to maintain logic and queries across multiple entity types and components; you want to decide on, manage, and implement the communication protocols (for example, WebAPI, WebSockets, or WCF, among others); you use reliable collections (like .NET reliable Dictionary and Queue) to store and manage your state/entities; you want to control the granularity and concurrency of your state; you want to control the partitioning scheme of your stateful service.
Reliable Actors	• An actor programming model for Service Fabric, which provides an asynchronous, single-threaded actor model. An actor represents a unit of state and computation. • Use Stateful Reliable Actors when your scenario involves many small independent units/objects of state and logic (live IoT objects or gaming back-end scenarios are great examples); you work with a massive amount of single-threaded objects while still being able to scale and maintain consistency; you want the framework to manage the concurrency and granularity of state; you want Service Fabric to manage communication protocols for you; you want Service Fabric to manage the partitioning schema of Stateful Actor services so they are transparent for you.

References	
Service Fabric Reliable Services	https://azure.microsoft.com/en-us/documentation/articles/service-fabric-reliable-services-introduction/
Service Fabric Reliable Actors	https://azure.microsoft.com/en-us/documentation/articles/service-fabric-reliable-actors-introduction/

Service development technologies

Service development technologies with .NET

Technologies	When to use and why
ASP.NET Web API in .NET 4.6 ASP.NET	**HTTP-based, REST approach, resource-oriented** • It is the preferred technology for flexible service development with REST approaches, OData, or JSON requirements. Try to use Web API as your first choice when evaluating which technology to use. Use any of the other technologies if Web API does not meet your needs. • Especially made for REST services. • Embraces HTTP verbs (PUT, GET, POST, DELETE...) as the application drivers. • Resource-oriented. • High scalability, thanks to Internet caches (Akamai, Azure CDN, Level3, and others) based on HTTP verbs.
ASP.NET Core 1.0 Web API ASP.NET	**Cross-platform, light framework, and best performance.** **HTTP-based, REST approach, resource-oriented** • New generation of HTTP services in ASP.NET. Similar to ASP.NET Web API in .NET 4.6 but it adds new capabilities, like cross-platform (Windows, Linux and Mac OS X), a very light and internally decoupled framework with the best performance. • Current state is RC and will be RTM in 2016. It will be mainstream eventually, superseding/replacing the former ASP.NET Web API. • Still (as of 2016), ASP.NET Core and .NET core are just "the new beginning," and ASP.NET 4.6 continues on, released and fully supported with a more matured environment. • Use ASP.NET Core if you need cross-platform (Linux/Mac/Windows) or you are creating new services from scratch and want to be prepared for the new wave in ASP.NET technologies with great news and innovation.
ASP.NET SignalR ASP.NET	**Real-time communications** • Use for real-time functionality on the client side (web, mobile, or desktop clients). • This approach enables your server-side code to push content to connected clients in real time and at high scale, even to millions of users. • It is HTTP- and WebSockets-based. It can be consumed from JavaScript in browser clients, Xamarin native clients for iOS and Android, .NET native Windows clients and server side events, and long polling.
WCF (Windows Communication Foundation)	**Decouple and flexible approach but not "default approach" for HTTP services** • If implementing HTTP services, Microsoft recommends choosing ASP.NET Web API rather than WCF, which is an older technology. • Use WCF when you need SOAP interoperability or you want to use a non-HTTP transport. WCF can use any protocol (such as TCP, named

	pipes, or HTTP), data formats (like SOAP, binary, JSON, or others), and hosting processes. • As of 2015, WCF Client library is open sourced and compatible with .NET Core.
WCF Data Services	**Older technology but supported in .NET 4.x** • Used to create data/resource-oriented and mostly CRUD and data-driven services. • It only supports OData. It is straightforward to use, but offers less flexibility and control than ASP.NET Web API. • Shares the same OData core libraries with ASP.NET Web API. • If implementing new OData services, Microsoft recommends using ASP.NET Web API rather than WCF Data Services. • It is not supported in .NET Core.
Workflow Services	**Older technology but supported in .NET 4.x** • Used when your service logic is internally a Windows Workflow Foundation (WF) workflow. Externally, it is a WCF service. Workflow Services has all the benefits and characteristics of WCF and WF, but is coupled to WCF.

References

ASP.NET Web API	http://www.asp.net/web-api
ASP.NET Core	https://www.asp.net/vnext
ASP.NET SignalR	http://www.asp.net/signalr
WCF	https://msdn.microsoft.com/en-us/library/ms731082(v=vs.110).aspx https://github.com/dotnet/wcf (WCF client only)

Data platform in Azure for mobile apps

Note that big data (HDInsights/Hadoop) and other data sources for analytics like Azure Data Lake and Azure Data Warehouse are excluded from the MADP scope. Big data is not specific to MADP but is worth further exploration in other contexts.

Data services in Azure

Technologies	When to use and why
Azure SQL Database	• It is the most popular data back end in Azure for mobile apps. • It allows you to leverage relational data from SQL Server databases that might be already available in the enterprise. • Take advantage of Azure Mobile App offline scenario with automatic data sync capabilities between a local SQLite database in the device and Azure SQL Database, by using the Azure Mobile App plugin for Cordova apps and the Azure Mobile Apps .NET Client SDK for Xamarin and .NET mobile apps.
Azure DocumentDB	• Azure DocumentDB is a NoSQL document database service designed from the ground up to natively support JSON and JavaScript directly inside the database engine. It's the right and modern solution for applications that run

	in the cloud when predictable throughput, low latency, and flexible query are key. • Appropriate when creating a data back end for mobile apps (for example, through ASP.NET Web API services) and the type of data to store is document-oriented, which means a schema-free JSON data stored, so even when the application schemas can be constantly evolving, you can fit it into DocumentDB because you don't need to specify the schema or secondary indices up front, like in relational databases. • Take advantage of the native JSON data model, which enables easy integration with web platforms and tools. • Additional "good fit" scenarios include using the "Aggregate" pattern and/or Event Sourcing and Command and Query Responsibility Segregation (CQRS) approaches.
Azure Storage	• Azure Storage provides the flexibility and hyperscale needed to store and retrieve large amounts of data. Use Azure Blob Storage (Object Storage) to store unstructured data, such as document files and media files. Use Azure Table Storage for structured NoSQL data. Use Azure Queue Storage to reliably store messages. And use SMB-based Azure File Storage for existing or new applications—no code changes are required. • When developing mobile apps, take advantage of Azure Storage, which is typically used when you need to store blob/files like photos, video files, and media in general.
Azure Redis Cache	• Azure Redis Cache is based on the popular open-source Redis cache. It gives you access to a secure, dedicated Redis cache, managed by Microsoft and accessible from any application within Azure. • Appropriate when you want to improve throughput, consistent low-latency data access to power fast, scalable Azure services for your mobile apps, thanks to this cache in the cloud, which is provided as a service and is very easy to use.

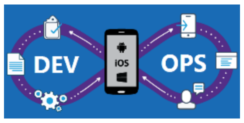
References

Azure SQL Database	https://azure.microsoft.com/en-us/services/sql-database/
Azure Document DB	https://azure.microsoft.com/en-us/services/documentdb/
Azure Storage	https://azure.microsoft.com/en-us/services/storage/
Azure Redis Cache	https://azure.microsoft.com/en-us/services/cache/

DevOps for mobile: Tools and SDKs

Microsoft offers a large variety of tools and SDKs to increase efficiency in your application lifecycle management and DevOps work and to improve collaboration between development teams and IT operations.

Development team collaboration services and tools



Services/servers for end-to-end ALM and DevOps

Microsoft offers ALM and DevOps capabilities in the cloud and on-premises.

Both approaches are valid, when managing the application lifecycle of mobile apps, and the Microsoft goal is to have the highest possible feature parity for both environments, although usually innovation comes first for the cloud approach. This is because the cloud allows for continuous deployment and innovation. Innovation is adopted in a slower cadence when it is released as software for on-premises installation.

Services/servers for ALM, code/build/test/deployment, and tracking

Tools/Services	When to use and why
Visual Studio Team Services (VSTS) Visual Studio	**ALM and DevOps in the cloud** • VSTS provides services in the cloud for teams to share code, track work, and ship software—for any language. • Use as the central pillar to manage all your ALM and DevOps tasks. • Allows continuous deployment of new service versions to staging and production environments, in the cloud or on-premises. • Take advantage of already available services—no infrastructure setup—the easiest way to get started. • Appropriate when you want to host all your code and operations in the cloud.
Team Foundation Server (TFS) Visual Studio	**ALM and DevOps on-premises** • TFS is an enterprise-grade server (on-premises) for teams to share code, track work, and ship software—for any language, all in a single package. • Use as the central pillar to manage all your ALM and DevOps tasks. • Allows continuous deployment of new service versions to staging and production environments in the cloud or on-premises. • You are responsible for the server installation and infrastructure management. • Appropriate when you need to host all your code and operations on-premises instead of in the cloud.
Xamarin Test Cloud	**Test mobile apps in real devices in the cloud** • Automate your app using these powerful testing frameworks or Test Recorder. • Upload your test suite, and run it on thousands of real devices in the cloud. • Analyze detailed reports with results, screenshots, and performance metrics.
Microsoft	**Continuous deployment to mobile devices**

HockeyApp Visual Studio	• One of the main goals of HockeyApp is to automatically distribute beta versions of mobile apps to beta testers' devices. Closing the loop, those versions can be provided automatically from a continuous integration process and builds from VSTS. • NOTE: HockeyApp is also used for analytics. See next table.
Microsoft CodePush CodePush	**App self-update for Cordova and React.native** • CodePush is a cloud service that enables Cordova and React Native developers to deploy mobile app updates directly to their users' devices. It works by acting as a central repository that developers can publish updates to (JavaScript, HTML, CSS, and images) and that apps can query for updates from (using provided client SDKs for Cordova and React Native). This allows you to address bugs and/or add small features that don't require you to rebuild a binary and to redistribute it through the respective app stores.

References

Visual Studio Team Services (VSTS)	https://www.visualstudio.com/products/visual-studio-team-services-vs/
Team Foundation Server (TFS)	https://www.visualstudio.com/products/tfs-overview-vs/
Xamarin Test Cloud	https://xamarin.com/test-cloud
Microsoft HockeyApp	http://hockeyapp.net/
CodePush	https://microsoft.github.io/code-push/

Analytics, monitoring, and learning

Tools/Services	**When to use and why**
Microsoft HockeyApp	**Analytics for mobile apps** • One of the main goals of HockeyApp is to provide analytics and monitoring information by collecting live crash reports from apps in mobile devices and by getting feedback from real users. • Appropriate when monitoring and analyzing technical behavior of mobile apps. • The main goal of HockeyApp is to focus on mobile apps. • NOTE: HockeyApp is also used for mobile app deployment. See previous table.
Application Insights Visual Studio	**Analytics for services/back end** • With Application Insights, you can find performance issues earlier, diagnose crashes faster, and get smarter about users with a 360-degree view of your apps and services. • Especially appropriate when monitoring and analyzing technical behavior of services (cloud or on-premises). • The main goal of Application Insights is to focus on services and web applications, although those services can be consumed from mobile apps.
Xamarin Insights	**Analytics for mobile apps**

	• Use it to get operational insights, like crash reports from the apps, plus behavioral insights, like traits, devices, sessions, and events for every user of your app, on an individual or global basis.
Azure Mobile Engagement	**Business/user analytics of mobile apps** • Azure Mobile Engagement is oriented to business user analytics by enabling collection of real-time analytics that highlight users' behavior. You can measure and act on analytics using a single dashboard. • The second important pillar from Azure Mobile Engagement is related to Marketing Campaigns, so you can create dynamic segments based on collected data and then create marketing campaigns using push notifications targeting specific segments.

References	
Microsoft HockeyApp	http://hockeyapp.net/
Application Insights	https://www.visualstudio.com/products/application-insights-vs
Xamarin Insights	https://xamarin.com/insights
Azure Mobile Engagement	https://azure.microsoft.com/en-us/services/mobile-engagement/

Technologies for securing mobile apps

Microsoft offers a large variety of technologies and SDKs to secure mobile apps. The following tables drill down on the specific technologies per approach/area, so you can get insights on when to use each technology.

Identity, SSO, and authentication

Infrastructure for identity providers

Infrastructure	When to use and why
Active Directory	• Appropriate for enterprise application authentication and SSO with infrastructure placed on-premises. Therefore, you are responsible for infrastructure setup, backup, and maintenance of domain controllers. • Can be federated/linked to Azure Active Directory in the cloud.
Azure Active Directory	• Appropriate for enterprise application authentication and SSO with infrastructure placed in the cloud. It is easy to get started, since the directory service is already available in the cloud—no need to invest in infrastructure setup/maintenance/backup. • Azure Active Directory can be federated/linked to Windows Active Directory on-premises so you can also have SSO across the cloud and on-premises.

Azure Active Directory B2C	• Appropriate for consumer identity and access management for apps in the cloud. • Usage and experience is similar to Azure Active Directory, but this version is extended and especially made for secured apps that provide user accounts to consumer users. • Consumers can sign up for your applications by using their existing social accounts (Facebook, Google, Amazon, or LinkedIn, for example) or by creating new local credentials (email address and password, or username and password) into the AD directory. • It scales to hundreds of millions of consumer users.
Social account providers	• Social account providers, like Facebook, Google, Twitter, and Microsoft. • Appropriate for consumer users' identity. • Can be used in conjunction with Azure AD B2C or with turnkey/custom development.
Azure Multi-Factor Authentication	• Stronger authentication extending to phone/text/notification validations, Azure Multi-Factor Authentication helps safeguard access to data and applications while meeting user demand for a simple sign-in process. • It delivers strong authentication with a range of easy verification options—phone call, text message, or mobile app notification—allowing users to choose the method they prefer.
Azure Active Directory Premium	• Azure Active Directory Premium (part of Microsoft EMS) provides single sign-on to thousands of cloud (SaaS) apps and offers access to web apps you run on-premises. • Includes MFA; access control based on device health, user location, and identity; and holistic security reports, audits, and alerts.

References

Active Directory	https://msdn.microsoft.com/en-us/library/bb727030.aspx#EFAA
Azure Active Directory	https://azure.microsoft.com/en-us/services/active-directory/
Azure Active Directory B2C	https://azure.microsoft.com/en-us/services/active-directory-b2c/
Multi-Factor Authentication	https://azure.microsoft.com/en-us/services/multi-factor-authentication
Azure Active Directory Premium	https://www.microsoft.com/en-us/server-cloud/products/azure-active-directory/default.aspx

Authentication client SDKs for mobile apps

SDKs for Azure Active Directory and Active Directory

SDKs	When to use and why
Cordova plugin for Azure Mobile Apps	• It is currently the "by default" choice for Cordova and Azure AD because of its easy-to-use and unified authentication interface capabilities. • The unified authentication interface currently supports authenticating against Azure Active Directory, Facebook, Google, Twitter, and Microsoft accounts. • This unified interface means that you're abstracted from downstream changes and can expect additional provider options and features in the future to streamline things even more. • It taps into Azure App Service Auth on the server side, which means you are only able to connect to authenticated, custom server App Service API Apps or other services that also use Azure App Service Auth. • You cannot get the security token in the client side (JavaScript). • It is limited to authentications in the cloud (for example, Azure AD). It cannot authenticate against Windows AD on-premises.
ADAL Cordova plugin	• The Active Directory Authentication Library plugin for Cordova allows you to securely authenticate, authorize, and access information in Azure AD or on-premises AD and to take advantage of device level SSO and MFA) capabilities. • Appropriate if you want to authenticate and get the security token in the client side (JavaScript). • You need the ADAL Cordova plugin if you want to authenticate against an on-premises Windows AD directory (ADFS v3 and up).
Azure Mobile Apps .NET Client SDK Microsoft .NET	• Appropriate when developing Xamarin or .NET apps that need to authenticate against Azure AD to consume services in Azure App Service (like an Azure Mobile App service). • Its unified authentication interface currently supports authenticating against Azure Active Directory, Facebook, Google, Twitter, and Microsoft accounts. • This unified interface means that you're abstracted from downstream changes and can expect additional provider options and features in the future to streamline things even more. • It taps into Azure App Service Auth on the server side, which means you are only able to connect to authenticated, custom server App Service API Apps or other services that also use Azure App Service Auth. • You cannot get the security token in the client side (C#). • It is limited to authentications in the cloud (such as Azure AD). It cannot authenticate against Windows AD on-premises.
Azure Active Directory Authentication Library (Azure ADAL) Microsoft .NET	• The Active Directory Authentication Library provides a Portable Class Library with Azure and Windows AD authentication functionality for your C# client on various platforms, including Xamarin.Forms, Xamarin.iOS, Xamarin.Android, and .NET client apps for Windows 10 UWP and Windows desktop. • Appropriate if you want to authenticate and get the security token on the client side (C#/Xamarin/.NET). • You need ADAL SDK/NuGet component if you want to authenticate against an on-premises Windows AD directory (ADFS v3 and up).

References	
Cordova plugin for Azure Mobile Apps	https://www.npmjs.com/package/cordova-plugin-ms-azure-mobile-apps
ADAL Cordova plugin	https://github.com/AzureAD/azure-activedirectory-library-for-cordova
Azure Mobile Apps .NET Client SDK	https://github.com/Azure/azure-mobile-apps-net-client https://components.xamarin.com/view/azure-mobile-client
Azure Active Directory Authentication Library (Azure ADAL)	https://github.com/AzureAD/azure-activedirectory-library-for-dotnet

Managing and securing mobile apps and devices

MAM/MDM

Products	When to use and why
Microsoft Intune	• Using Intune, organizations can provide their employees with access to corporate applications, data, and resources from virtually anywhere on almost any device, while helping to keep corporate information secure. • Microsoft Intune provides mobile device management, mobile application management (with its Intune Apps SDK), and PC management capabilities from the cloud. • Through MAM and MDM, organizations can secure and manage devices and apps based on policies administered by IT operations through the Intune portal.
Microsoft Intune App SDK	• Using Intune App SDK, developers can build data protection into their mobile iOS and Android apps, helping to ensure that corporate data stays protected without the overhead of building these features into the mobile app.

Other security products

Security

Products	When to use and why
Azure Rights Management Services (RMS)	Azure RMS provides: • Encryption policy at the file level, which follows the document within and outside of your organization. • Collaborate more securely by protecting virtually any file type on any device platform, using Azure Rights Management. • Safely share files in email or using your favorite cloud storage service, such as OneDrive or Dropbox.

	• Choose from flexible on-premises or cloud deployment options, based on your organizational needs. • Track your shared documents to learn about document use or abuse.
Microsoft Advanced Threat Analytics (ATA)	ATA provides: • Analysis for abnormal behavior: behavioral analytics uncovers suspicious activities and abnormal behavior, leveraging Machine Learning. • Malicious attack detection: ATA detects known malicious attacks almost instantly. • Alerts for known security issues and risks: detected using the work of world-class security researchers.

References

Microsoft Intune	https://www.microsoft.com/en-us/server-cloud/products/microsoft-intune/overview.aspx
Azure Rights Management Services (RMS)	https://www.microsoft.com/en-us/server-cloud/solutions/information-protection.aspx
Microsoft Advanced Threat Analytics (ATA)	https://www.microsoft.com/en-us/server-cloud/products/advanced-threat-analytics/