Análisis de tecnologías para aplicaciones en dispositivos móviles

1.- Introducción

En esta unidad introductoria, vamos a realizar un breve repaso a las principales tecnologías que disponemos actualmente a la hora de abordar la programación de una aplicación para un dispositivo móvil.

Imaginémonos que queremos desarrollar una aplicación móvil. ¿Qué es lo primero que pensaríamos? ¿Qué lenguaje utilizar? ¿Qué entorno? ¿Para qué dispositivos? ¿Móvil? ¿Tablet? ¿Reloj inteligente? ¿Televisión?

Se trata de una pregunta bastante más compleja que cuando nos planteamos realizar una aplicación de escritorio para el pc.

Como sabemos existen distintas plataformas y sistemas operativos para dispositivos móviles. Vamos a ver los principales.

1.1.- Android

Android es un sistema operativo desarrollado por Google y basado en el Kernel de Linux, fabricado específicamente para dispositivos móviles con pantalla táctil: teléfonos móviles, tabletas, relojes inteligentes, televisores o incluso, algunos coches.

La estructura de Android se compone de aplicaciones que se ejecutan en un framework Java de aplicaciones orientadas a objetos sobre el núcleo de las bibliotecas de Java. La máquina virtual de Java sobre la que se ejecutan estas aplicaciones era Dalvik hasta la versión 5.0, para cambiar en versiones posteriores al entorno Android Runtime (ARTE). La principal diferencia entre estas máquinas virtual era que Dakvik realizaba la compilación en tiempo de ejecución, mientras que ARTE compila el Java bytecode durante la instalación de la aplicación.

Las librerías que se utilizan están escritas en lenguaje C, e incluyen un administrador de interfaz gráfica (surface manager), un framework OpenCore, una base de datos relacional SQLite, una interfaz de programación gráfica OpenGL SE 2.0 3D, un motor de renderizado WebKit, un motor gráfico SGL, SSL y una librería estándar de C.

Con todo esto, el lenguaje para el desarrollo en Android ha sido tradicionalmente Java. Recientemente, Google ha adoptado el lenguaje Kotlin como lenguaje oficial de programación en Android, que es un lenguaje más potente y que genera código ejecutable directamente a la JVM.

1.1.- iOS

Se trata del sistema operativo más vendido por detrás de Android, y es propiedad de la multinacional Apple. Fue creado para el iPhone, y posteriormente adoptado en el iPod touch y el iPad. IOS no permite la instalación en hardware de otras compañías.

iOS aporta varios elementos de control novedosos, dando una respuesta a las órdenes del usuario inmediata y una interfaz fluida. El usuario puede interactuar con el sistema operativo mediante gestos, deslizamientos, toques, pellizcos…

iOS deriva de macOS, y este de Darwin BSD, que es un sistema operativo de tipo Unix.

La arquitectura de iOS cuenta con cuatro capas de abstracción: la capa del núcleo del sistema operativo, la capa de servicios principales, la capa de medios y la capa de Cocoa Touch.

El desarrollo de aplicaciones de forma nativa para iOS pasa por lenguajes como Objective-C y Swift.

2.- Desarrollo de aplicaciones móviles

El desarrollo de aplicaciones móviles requiere de las mismas etapas para el desarrollo del software que el resto de aplicaciones, no se puede pensar que al ser una aplicación pequeña el proyecto será pequeño. Es necesario la planificación y el uso de las tecnologías para el desarrollo de software complejo que son las aplicaciones móviles.

2.1.- Limitaciones que plantea la ejecución de aplicaciones en dispositivos móviles.

Los dispositivos móviles plantean serias limitaciones a la hora de ejecutar aplicaciones que debemos tener en consideración durante el desarrollo de la aplicación y que no tenemos en otros proyectos:

- **Desconexión**:

  Al tratarse de dispositivos que podemos llevar con nosotros en cualquier momento, lugar y permanentemente conectados, estos pueden sufrir desconexiones tanto de forma total como parcial, lo que, en el caso de las aplicaciones que utilicen datos de un servidor, provocará que se vean afectadas por la variabilidad de la conexión.

- **Seguridad**:

  En el ámbito de la seguridad de la información, este tipo de dispositivos son más vulnerables, al permitir conectarse a redes poco seguras o la posibilidad de instalar aplicaciones de dudosa procedencia, unido a la gran cantidad de sensores, como cámaras, GPS o micrófonos, de que disponen. En el desarrollo de aplicaciones debemos considerar usar los permisos mínimos necesarios para el funcionamiento de la aplicación.

➔ **Consumo de batería**:

La batería suele ser uno de los elementos más comprometidos, y el diseño de las aplicaciones debe tener en consideración qué recursos utiliza en cada momento.

➔ **Memoria, almacenamiento y procesador**:

Las aplicaciones móviles deben optimizar el uso de los recursos, ya que también debe compartirlos con otras aplicaciones y aunque cada vez es menos limitante, un usuario podría prescindir de nuestra app por exceso de consumo de recursos no justificados.

2.2.- Tecnologías disponibles

En el desarrollo de aplicaciones móviles, a lo largo de los últimos años han aparecido tecnologías que nos ayudan a la creación de aplicaciones según las necesidades del proyecto que debamos afrontar. Normalmente, los creadores de cada sistema operativo móvil han designado el lenguaje y el entorno de desarrollo que se debe utilizar para la creación de aplicaciones en su plataforma (Android, iOS) pero han aparecido framework que permiten el desarrollo multiplataforma.

El siguiente vídeo en los primeros 10 minutos te explica breve y claro las opciones actuales para el desarrollo para dispositivos móviles: https://www.youtube.com/watch?v=-pWSQYpkkjk

3.- Tipos de aplicaciones móviles

Teniendo en cuenta la proximidad al código nativo de cada plataforma, podemos distinguir diferentes tipos de aplicaciones (de menos próxima a más):

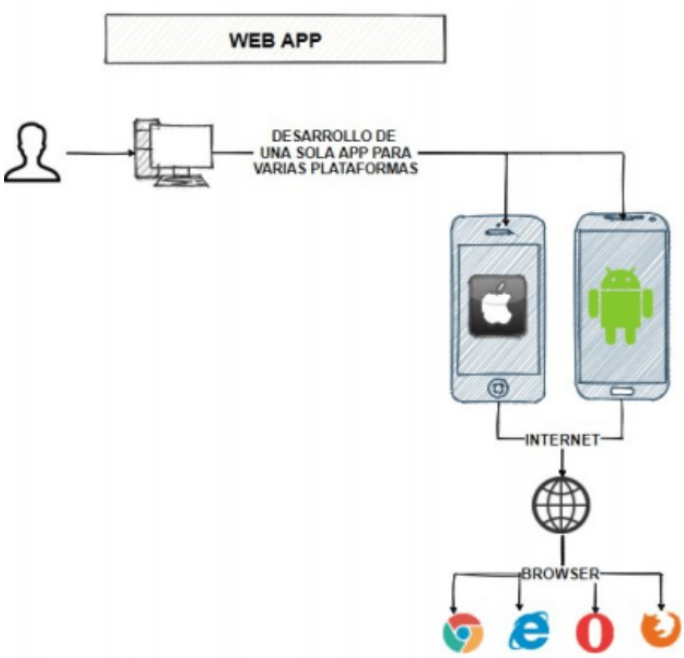
- Aplicaciones web responsive (Webapps)
- Aplicaciones híbridas
- Aplicaciones web progresivas (PWAs)
- Aplicaciones compiladas
- Aplicaciones nativas

3.1.- Aplicaciones web responsive (Webapps)

Se trata de aplicaciones basadas en tecnología web: HTML+CSS+Javascript, y que, para ejecutarse, necesitan únicamente un navegador web. El hecho de ser responsive implica que su interfaz se adapte a cualquier dispositivo.

Para este tipo de aplicaciones, no es necesario desarrollar nada de código nativo, y son totalmente multiplataforma, ya que se ejecutan sobre el propio navegador web del sistema operativo en el que estemos (Android, iOS, Windows, Linux, MacOS, etc...).

Así, disponemos de un único código para ejecutarse en todas las plataformas, sin embargo no ofrecen una experiencia de navegación al usuario tan buena como las aplicaciones nativas, sobretodo si se trata de aplicaciones complejas.



3.2.- Aplicaciones híbridas

Aplicaciones que utilizan tecnología web para construir un sitio web en HTML+CSS+JS y cargan este sitio en un *webview*. Un webview no es más que un navegador web sin la barra de navegación y otras opciones, de manera que aparentemente, es como si fuera una aplicación nativa del dispositivo.

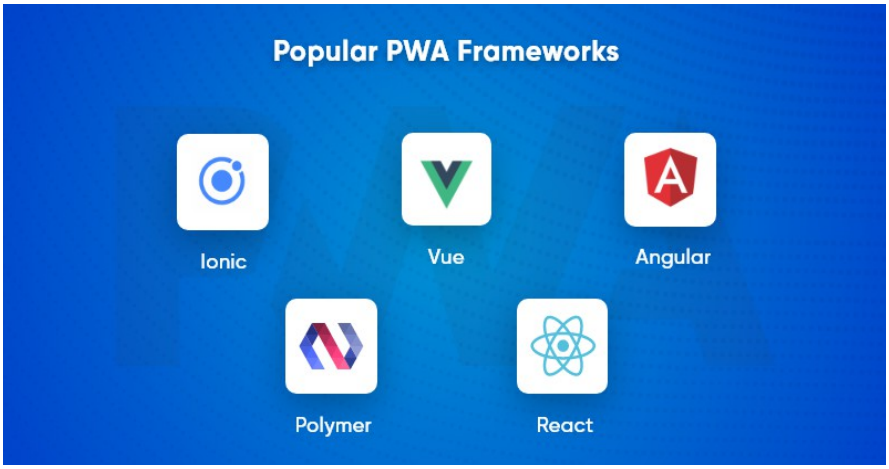
Además, las aplicaciones híbridas pueden acceder a través del navegador a algunas características del dispositivo, como la ubicación, acelerometro, etc.

Los frameworks más utilizados para este tipo de aplicaciones son **Ionic i Phonegap**. Se trata de dos frameworks muy utilizados para desarrollar aplicaciones web para realizar aplicaciones para dispositivos móviles. La idea de estos frameworks es sencilla: empaquetar la aplicación web con código HTLM+CSS+JS junto el webview. Este componente si que estará con código nativo de cada sistema pero la aplicación web será la misma en todos ellos. Estas aplicaciones reaccionan de una forma más lenta, puesto que han de comunicarse con el sitio web para obtener gran parte de los contenidos.

3.3.- Aplicaciones web progresivas (PWAs)

Algo más cerca de las aplicaciones nativas están las aplicaciones web progresivas. Aportan al usuario gran parte de las ventajas de las aplicaciones nativas pero con el desarrollo en base a tecnologías web. Además, a diferencia de las aplicaciones híbridas, permiten el funcionamiento sin conexión o con mala conexión en el servidor.

Existen muchos frameworks para el desarrollo de PWAs, entre los que se encuentran los principales frameworks para el desarrollo web: React PWA Library, Angular PWA Library, Vue PWA Framework, Ionic PWA Framework, Svelte, PWA Builder o Polymer.



3.4.- Aplicaciones compiladas

Se trata de tecnologías que pretenden utilizar solo un lenguaje de programación para generar aplicaciones móviles nativas. La idea general es trabajar con un única tecnología y lenguaje de programación y que el código compilado que se genera siga nativo en las diferentes plataformas.

Algunas de estas tecnologías más utilizadas en este tipo de aplicaciones son:

- **React Native**: React es un framework creado por Facebook, que utiliza el lenguaje Javascript y la librería React, que nos permite crear interfaces basadas en sus componentes. En estas aplicaciones, el código Javascript se ejecuta en hilo de ejecución aparte, mientras que los elementos de la interfaz de usuario son compilados a lenguaje máquina.
- **NativeScript**: Nos permite crear aplicaciones nativas mediante Javascript puro o bien utilizando otras librerías como Angular o Vue. También trae diversos componentes preconstruidos para utilizar a las interfaces de usuario. Al igual que React Native, no trabajamos con HTML
- **Flutter**: Framework desarrollado y mantenido por Google, que utiliza el lenguaje de programación Dart. La idea es escribir todo el código en Dart, y compilar a código nativo que se ejecute completamente al dispositivo. Además, Flutter compila a ARM y genera

librerías C/C++ por lo que está más cercano al código nativo y se ejecuta más rápido. Cuenta con una serie de widgets predeterminados que nos ayudan a crear la interfaz gráfica.

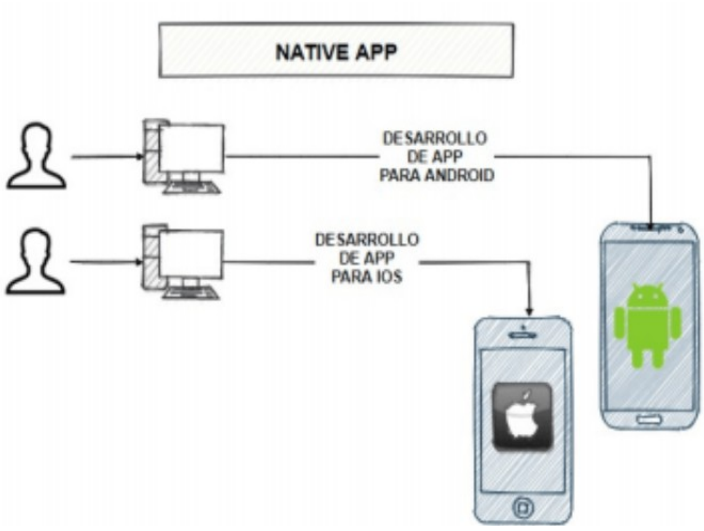
➔ **Xamarin**: Desarrollado por Microsoft. Usa el lenguaje de programación C# para desarrollar aplicaciones para Android, iOS y Windows.

3.5.- Aplicaciones nativas

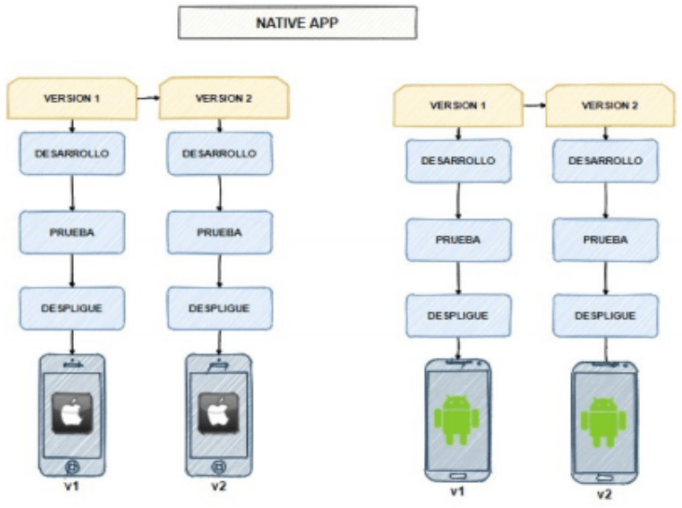
Las aplicaciones nativas son aquellas que se implementan en el lenguaje específico que facilita la plataforma/fabricante para el desarrollo. Permiten acceder a todas las funcionalidades de las plataformas y siempre tendremos disponible antes que nadie cualquier novedad o característica. Por ello, las aplicaciones resultantes son fluidas y ofrecen la mejor experiencia al usuario. Como punto negativo, tenemos el incremento en el coste de producción y mantenimiento. Actualmente el mercado está dominado por Android y iOS de Apple para iPhone.

Android: Tenemos Java y Kotlin sobre Android Studio.

iOS para iPhone: En la plataforma de Apple para iPhone nos encontramos con Objective-C en las aplicaciones antiguas y Swift en las nuevas, sobre el entorno Xcode.



En un proyecto de una aplicación, tendremos dos versiones diferentes, una por cada sistema operativo y el flujo de los proyectos en el control de versiones, utilizará diferentes repositorios.



3.6.- Comparativa de las tecnologías

Tipos de aplicaciones web/ Características	Aplicaciones Nativas	Aplicaciones Híbridas	Aplicaciones Web
Coste de desarrollo	Alto	Medio	Bajo
Tiempo de desarrollo	Alto	Medio	Bajo
Multiplataforma	No	Sí	Sí
Rendimiento	Alto	Medio	Bajo
Apps Stores	Sí	Sí	No
Acceso al dispositivo	Completo	Alto/ Completo	Parcial
Conexión a internet	No siempre necesario	No siempre necesario	Siempre
Espacio en el dispositivo	Sí	Sí	No

Algunos enlaces de interés:

- ➔ https://americavirtual.net/desarrollo-aplicaciones-multiplataforma/
- ➔ https://medium.com/@noebranagan/https-medium-com-noebranagan-hablemos-demultiplataforma-2ba36fb25265
- ➔ https://www.21twelveinteractive.com/flutter-vs-react-native-vs-ionic-vs-nativescript/
- ➔ https://www.ftxinfotech.com/blog/react-native-vs-ionic-vs-flutter-vs-phonegap/