INTRODUCCIÓN AL DESARROLLO DE APLICACIONES MÓVILES

PROGRAMACIÓN DE DISPOSITIVOS MÓVILES

Carlos Rojas Sánchez

Licenciatura en Informática

Universidad del Mar

Contenido

1. Principales Sistemas Operativos de los Dispositivos Móviles
2. Principales Enfoques para el Desarrollo Móvil.
3. Proceso de Diseño y Desarrollo de una Aplicación Móvil

Principales Sistemas Operativos de los Dispositivos Móviles

Android

- Desarrollado por Google.
- Basado en Linux.
- Sistema abierto y personalizable.
- Tienda de aplicaciones: Google Play Store.
- Utilizado por múltiples fabricantes (Samsung, Xiaomi, Motorola, etc.).

iOS

- Desarrollado por Apple.
- Exclusivo para dispositivos de Apple (iPhone, iPad).
- Sistema cerrado, optimizado para su hardware.
- Tienda de aplicaciones: App Store.
- Enfocado en seguridad y experiencia de usuario.

HarmonyOS

- Desarrollado por Huawei.
- Compatible con múltiples dispositivos (teléfonos, tabletas, IoT).
- Basado en microkernel y diseñado para interconectividad.
- Tienda de aplicaciones: AppGallery.

KaiOS

- Derivado de Firefox OS.
- Diseñado para teléfonos básicos con funciones inteligentes.
- Compatible con aplicaciones web ligeras.
- Usado en dispositivos de bajo costo en mercados emergentes.

EMUI(Emotion UI)

Es una capa de personalización de Android desarrollada por Huawei para sus dispositivos móviles. No es un sistema operativo independiente, sino una interfaz modificada que se ejecuta sobre Android.

EMUI(Emotion UI)

Diseño personalizable Temas, íconos y opciones de personalización avanzadas.

Modo oscuro Reduce el consumo de energía y protege la vista.

Gestos de navegación Alternativa a los botones tradicionales.

Rendimiento optimizado Funciones como GPU Turbo mejoran la experiencia en juegos.

HMS (Huawei Mobile Services Sustituye los servicios de Google en los dispositivos sin GMS.

Años 90: Los primeros sistemas operativos móviles

1993 - IBM Simon OS Primer smartphone con pantalla táctil y correo electrónico.

1996 - Palm OS Utilizado en las PDA de Palm, pionero en pantallas táctiles con stylus.

1998 - Symbian OS Desarrollado por Psion y luego adoptado por Nokia, Sony Ericsson y Motorola.

Años 2000: Expansión de los sistemas operativos móviles

2000 - Windows Mobile Sistema de Microsoft para PDAs y teléfonos inteligentes.

2007 - iOS (iPhone OS) Apple revoluciona la industria con la App Store y la interfaz táctil.

2008 - Android Google lanza Android basado en Linux, con un modelo de código abierto.

Años 2010: Dominio de Android e iOS

2010 - Windows Phone Microsoft intenta competir con una interfaz moderna (descontinuado en 2019).

2011 - BlackBerry OS 10 Último intento de BlackBerry antes de su declive.

2013 - Firefox OS Sistema de Mozilla basado en la web (descontinuado en 2016).

2017 - KaiOS Basado en Firefox OS, diseñado para teléfonos básicos con apps ligeras.

Años 2020 - Actualidad: Alternativas y nuevos enfoques

2019 - HarmonyOS Huawei lanza su propio sistema operativo tras restricciones de EE.UU.

2021 - Android 12 / iOS 15 Siguen evolucionando con mejoras en privacidad y rendimiento.

2023 - HarmonyOS 4 Huawei avanza con su ecosistema sin depender de Android.

Principales Enfoques para el Desarrollo Móvil.

Desarrollo Nativo

Descripción Se crean aplicaciones específicas para un sistema operativo (Android o iOS) utilizando sus lenguajes y herramientas oficiales.

Tecnologías

- **Android** Kotlin, Java (Android Studio).
- **iOS** Swift, Objective-C (Xcode).

Desarrollo Nativo

Ventajas Alto rendimiento y mejor experiencia de usuario. Acceso total a las funcionalidades del dispositivo (cámara, GPS, etc.).

Desventajas Mayor costo y tiempo de desarrollo (se necesita una app diferente para cada SO).

Desarrollo Multiplataforma

Descripción Se usan herramientas que permiten escribir un solo código y ejecutarlo en Android e iOS.

Tecnologías

- **Flutter** Dart.
- **React Native** JavaScript/TypeScript.
- **Xamarin** C#.

Desarrollo Multiplataforma

Ventajas Código único para ambas plataformas. Reducción de costos y tiempo de desarrollo.

Desventajas Rendimiento menor comparado con el desarrollo nativo. Acceso limitado a algunas funcionalidades específicas del hardware.

Desarrollo Web Móvil (PWA - Progressive Web Apps)

Descripción Son aplicaciones web que funcionan en móviles sin necesidad de instalación desde una tienda de apps.

Tecnologías **HTML, CSS, JavaScript** Frameworks como Vue.js, React.js o Angular.

Desarrollo Web Móvil (PWA - Progressive Web Apps)

Ventajas No requiere instalación, accesible desde un navegador. Actualizaciones automáticas sin pasar por tiendas de apps.

Desventajas Menor integración con hardware y funcionalidades avanzadas del dispositivo. Puede no ofrecer la misma experiencia fluida que una app nativa.

Desarrollo Híbrido

Descripción Combina tecnologías web con acceso a funcionalidades nativas mediante frameworks.

Tecnologías

- **Ionic** Angular o React.
- **Apache Cordova**

Desarrollo Híbrido

Ventajas Código único para múltiples plataformas. Más fácil de desarrollar con conocimientos web.

Desventajas Rendimiento inferior a apps nativas. Puede haber problemas con la interfaz de usuario en algunos dispositivos.

Proceso de Diseño y Desarrollo de una Aplicación Móvil

Fases

- Planeación y Estrategia
- Diseño y Experiencia de Usuario
- Desarrollo
- Pruebas y Depuración
- Publicación y Lanzamiento
- Mantenimiento y Actualizaciones

Planeación y Estrategia

Definir el objetivo ¿Para qué servirá la app? ¿Qué problema soluciona?

Estudio de mercado Analizar competidores y validar la idea.

Definir público objetivo ¿Quiénes la usarán? (edad, intereses, necesidades).

Elegir el modelo de negocio ¿Será gratuita, de pago, con publicidad, suscripción?

Diseño y Experiencia de Usuario

Wireframes Bocetos de la estructura y flujo de la app.

Prototipo Versión interactiva para visualizar la interfaz antes de desarrollar.

Diseño UI (Interfaz de Usuario) Colores, tipografía, iconos y componentes visuales.

Pruebas de usabilidad Evaluar la facilidad de uso con usuarios reales.

Herramientas Figma, Adobe XD, Sketch.

Desarrollo

Elección del enfoque Nativo, Multiplataforma, Híbrido o PWA.

Desarrollo del Backend Bases de datos, servidores y API.

Desarrollo del Frontend Pantallas, interactividad y funciones de la app.

Integración de servicios externos Notificaciones, pagos, mapas, etc.

Tecnologías Android: Kotlin, Java (Android Studio). iOS: Swift (Xcode). Multiplataforma: Flutter, React Native. Backend: Node.js, Firebase, Python (Django)

Pruebas y Depuración

Pruebas funcionales Verificar que todo funciona como se espera.

Pruebas de compatibilidad Diferentes dispositivos, tamaños de pantalla y sistemas operativos.

Pruebas de seguridad Protección de datos y seguridad de usuarios.

Pruebas de rendimiento Medir velocidad y consumo de recursos.

Herramientas TestFlight, Firebase Test Lab, Appium.

Publicación y Lanzamiento

Google Play Store Cumplir con las políticas de Android.

Apple App Store Seguir las reglas de Apple.

Optimización ASO Palabras clave, descripción y capturas atractivas.

Estrategia de marketing Anuncios, redes sociales, influencers.

Herramientas Google Play Console, App Store Connect.

Mantenimiento y Actualizaciones

Monitoreo de errores Analizar crash reports y feedback de usuarios.

Actualización de funciones Mejorar la app con base en el uso y tendencias.

Soporte al usuario Resolver dudas y problemas técnicos.

Herramientas Firebase Analytics, Google Analytics, Zendesk.

Políticas y Requisitos en Google Play Store

1. Crear una Cuenta de Desarrollador.
2. Cumplir con las Políticas de Contenido de Google Play
3. Cumplir con las Normas de Diseño y Experiencia de Usuario
4. Preparar los Archivos y la Información para la Publicación
5. Pruebas y Lanzamiento

Crear una Cuenta de Desarrollador

- Registrarte en Google Play Console.
- Pagar una tarifa única de 25 USD.
- Completar la información de desarrollador (nombre, correo, contacto).

Cumplir con las Políticas de Contenido de Google Play

Privacidad y seguridad

- Debes tener una Política de Privacidad clara.
- Informar qué datos recopila la app y cómo se usan.
- Cumplir con el Google Play Data Safety.

Cumplir con las Políticas de Contenido de Google Play

Contenido prohibido

- No debe incluir contenido ilegal, violento, de odio, sexual explícito o engañoso.
- No permitir spam, malware o publicidad intrusiva.
- No infringir derechos de autor o marcas registradas.

Cumplir con las Políticas de Contenido de Google Play

Permisos y funciones de la app

- Solo solicitar permisos esenciales para el funcionamiento de la app.
- Explicar por qué se requieren ciertos permisos (ej. acceso a la cámara, ubicación).

Cumplir con las Normas de Diseño y Experiencia de Usuario

Usabilidad La app debe ser funcional, estable y sin errores críticos.

Publicidad transparente No mostrar anuncios engañosos o inesperados.

Compras dentro de la app Si hay pagos, deben gestionarse a través de Google Play Billing.

Preparar los Archivos y la Información para la Publicación

Archivo APK o AAB Subir el paquete de la aplicación (AAB recomendado).

Ficha de la app en Google Play Store Nombre, descripción, capturas de pantalla, icono y video promocional. Clasificación de contenido según la edad recomendada.

Compatibilidad Indicar los dispositivos y versiones de Android compatibles.

Pruebas y Lanzamiento

Pruebas internas y beta Se recomienda probar la app antes del lanzamiento público.

Revisión por parte de Google Puede tardar varias horas o días en ser aprobada.

Lanzamiento progresivo Google permite lanzar la app a un porcentaje de usuarios y expandir gradualmente.