Programación de Aplicaciones Móviles Nativas

Elección de una Arquitectura

Laura González Suárez

Introducción

En el mundo de la ingeniería del software, la elección de la arquitectura adecuada es esencial. Una buena arquitectura no solo simplifica el desarrollo y el mantenimiento de la aplicación, sino que también influye en aspectos críticos como el rendimiento, la seguridad y la experiencia del usuario (UX).

En este trabajo, abordaremos diversos escenarios prácticos relacionados con el desarrollo de aplicaciones móviles. Se nos desafiará a seleccionar la arquitectura más apropiada en función de las circunstancias y restricciones específicas de cada caso.

Supuesto 1: Aplicación de E-commerce para una PYME

Contexto: Una pequeña empresa quiere lanzar su tienda online a través de una aplicación móvil nativa.

Presupuesto: Limitado.

Tiempos de entrega: 4 meses.

Recursos humanos: Un desarrollador principal y un diseñador.

Rendimiento: Se espera un tráfico moderado, pero es esencial que la aplicación sea rápida y eficiente.

Evaluación del Supuesto 1:

Este supuesto debe tener muy en cuenta las restricciones de presupuesto, los plazos apretados, las limitaciones de recursos humanos y la necesidad de un rendimiento óptimo. Por ello, se ha seleccionado la arquitectura **MVP (Model-View-Presenter)** como la más adecuada para superar estos desafíos de desarrollo.

Consideraciones para la Elección de la Arquitectura MVP:

- **Presupuesto Limitado:**

  Dado al limitado presupuesto con el que se cuenta para el proyecto, es esencial maximizar la eficiencia y minimizar los costos de desarrollo. MVP se destaca en este aspecto debido a su simplicidad de implementación y mantenimiento. Al separar claramente la lógica de presentación (Presenter) de la interfaz de usuario (View), se facilita la colaboración entre el desarrollador y el diseñador, lo que puede reducir la necesidad de recursos adicionales y con ello, los costos.

- **Tiempos de Entrega Ajustados (4 meses):**

  El tiempo es un recurso crítico en este proyecto, y cumplir con el plazo de 4 meses es esencial. MVP ofrece una curva de aprendizaje más suave y una implementación más rápida en comparación con otras arquitecturas más complejas, como MVVM o Clean Architecture. Esto permite un desarrollo más rápido y una entrega oportuna del producto.

- **Recursos Humanos Limitados (un desarrollador principal y un diseñador):**

  Con un equipo de desarrollo limitado, es importante utilizar una arquitectura que facilite la colaboración y permita que los roles se centren en sus tareas específicas de forma simultánea. MVP ofrece una separación clara de las responsabilidades, lo que significa que el desarrollador principal puede centrarse en la lógica de negocio (Presenter) mientras que el diseñador se ocupa de la interfaz de usuario (View).

- **Necesidad de Rendimiento Eficiente:**

  Aunque MVP no se destaca específicamente por su rendimiento, ofrece un equilibrio adecuado entre separación de preocupaciones y simplicidad. Además, el rendimiento eficiente puede lograrse mediante la implementación adecuada de la capa Model, que maneja la lógica de negocio y la interacción con la base de datos.

En conclusión, debido la combinación de un presupuesto limitado, tiempos de entrega ajustados y recursos humanos limitados, la arquitectura MVP se presenta como la elección más adecuada para el desarrollo de la aplicación de E-commerce para una PYME. Su simplicidad de implementación, separación de preocupaciones clara y eficiencia en la colaboración entre el desarrollador y el diseñador hacen que MVP sea una elección sólida para cumplir con los objetivos del proyecto de manera efectiva y dentro del marco de tiempo y recursos disponibles.

Supuesto 2: Aplicación Social Interactiva para una Startup

Contexto: Una startup quiere crear una aplicación social con características interactivas, como chats en tiempo real y transmisiones en vivo.

Presupuesto: Moderado.

Tiempos de entrega: 6-8 meses.

Recursos humanos: Un equipo de tres desarrolladores, un diseñador y un programador backend.

Rendimiento: Se espera un alto tráfico y es crucial que la aplicación maneje interacciones en tiempo real.

Evaluación del Supuesto 2:

Este supuesto debe tener en cuenta las restricciones de presupuesto, recursos humanos y sobre todo la necesidad de manejar un alto tráfico y proporcionar interacciones en tiempo real. Por ello, a continuación, se justificará la elección de la arquitectura **MVVM (Model-View-ViewModel)** para abordar estos desafíos de desarrollo.

- **Separación de Responsabilidades:**

  MVVM promueve una clara separación de responsabilidades entre el Modelo (Model), la Interfaz de Usuario (View) y la Lógica de la Interfaz de Usuario (ViewModel). Esto permite que los desarrolladores se centren en áreas específicas sin interferir con las responsabilidades de otros, lo que es beneficioso en un equipo multidisciplinario. El diseñador puede trabajar en la capa de vista (View), mientras que los desarrolladores frontend y backend pueden colaborar en la implementación del ViewModel y la lógica de negocio.

- **Moderado Presupuesto y Tiempos de Entrega Razonables:**

  MVVM ofrece un equilibrio entre la capacidad de manejar complejidad y la eficiencia en el desarrollo. Aunque puede requerir más tiempo de desarrollo inicial que algunas arquitecturas más simples, se espera que esta inversión inicial se traduzca en un mantenimiento más sencillo y rápido a medida que evoluciona el proyecto.

- **Alto tráfico e Interacciones en Tiempo Real:**

  La elección de MVVM para una aplicación social con alto tráfico y requerimientos de interacciones en tiempo real se justifica por su capacidad de manejar eficientemente flujos de datos en tiempo real y eventos de usuario. Esto se debe a su enfoque en separar las responsabilidades y utilizar observables o patrones de binding en el ViewModel, lo que permite que la Vista se actualice de manera inmediata ante cambios en los datos en tiempo real. MVVM es especialmente adecuada para características como chats en tiempo real y transmisiones en vivo. Además, esta arquitectura escala de manera efectiva para manejar altos niveles de tráfico, asegurando una experiencia de usuario fluida y receptiva incluso en situaciones de alta demanda.

En conclusión, en el contexto de desarrollar una aplicación social interactiva para una startup con características de alto tráfico, la arquitectura MVVM se destaca como una elección sólida debido a su capacidad para manejar interacciones en tiempo real, su flexibilidad para equipos multidisciplinarios y su equilibrio entre complejidad y eficiencia en el desarrollo. Esta elección respalda los objetivos de rendimiento y entrega del proyecto.

Supuesto 3: Aplicación Financiera para una Gran Empresa.

Contexto: Una gran empresa financiera quiere desarrollar una aplicación para que sus clientes gestionen sus finanzas, con características como visualización de transacciones, transferencias y análisis financiero.

Presupuesto: Alto.

Tiempos de entrega: 10-12 meses.

Recursos humanos: Un equipo grande con múltiples desarrolladores, diseñadores, especialistas en seguridad y analistas.

Rendimiento: Se espera un tráfico muy alto y es esencial que la aplicación sea segura y eficiente.

Evaluación del Supuesto 3:

La aplicación cuenta con un presupuesto alto, tiempos de entrega razonables, un equipo grande con especialistas en diferentes campos y la necesidad de un rendimiento excepcional. Por ello, se justificará la elección de la arquitectura **Clean Architecture** para abordar estos desafíos de desarrollo.

- **Presupuesto Alto:**

  Con un presupuesto alto disponible, se pueden invertir recursos adicionales en la planificación y el diseño de una arquitectura sólida. Clean Architecture es una opción excelente para proyectos financieros donde la robustez y la seguridad son fundamentales.

- **Tiempos de Entrega Razonables:**

  Aunque se cuenta con un plazo de 10-12 meses, Clean Architecture proporciona una estructura que facilita el desarrollo paralelo y una evolución ordenada del código. Esto puede ayudar a cumplir con los plazos sin sacrificar la calidad del software.

- **Recursos Humanos Abundantes:**

  Con un equipo grande y multidisciplinario, los desarrolladores pueden trabajar en capas específicas, como la lógica de negocio y la interfaz de usuario, mientras que los especialistas en seguridad pueden garantizar que se cumplan los estándares de seguridad y los analistas pueden centrarse en el análisis financiero.

- **Rendimiento y Seguridad:**

  En una aplicación financiera, el rendimiento y la seguridad son fundamentales. Clean Architecture permite una separación adecuada de las capas, lo que facilita la implementación de medidas de seguridad sólidas y la optimización del rendimiento sin comprometer la estructura del código.

- **Escalabilidad y Mantenibilidad:**

  Clean Architecture promueve una separación clara de las preocupaciones y una independencia de las capas, lo que facilita la escalabilidad a medida que la empresa crece y la capacidad de mantener y mejorar la aplicación a lo largo del tiempo.

En conclusión, Clean Architecture es la elección recomendada para una aplicación financiera de gran envergadura. Ofrece la robustez, la seguridad y la flexibilidad necesarias para un proyecto financiero de alta importancia, con un presupuesto generoso, un amplio equipo de desarrollo y la necesidad de un rendimiento excepcional.

Supuesto 4: Plataforma de Salud y Bienestar para Hospitales

Contexto: Un hospital de renombre desea desarrollar una aplicación móvil nativa que permita a los pacientes acceder a sus historiales médicos, programar citas, chatear con especialistas y recibir recomendaciones personalizadas basadas en su historial.

Presupuesto: muy alto.

Tiempos de entrega: 12-15 meses.

Recursos humanos: un equipo multidisciplinario compuesto por varios desarrolladores móviles, desarrolladores backend, especialistas en seguridad de la información, diseñadores UX/UI y analistas de sistemas.

Rendimiento: se espera un tráfico constante y alto debido a la gran cantidad de pacientes. La seguridad y privacidad de los datos es primordial.

Evaluación del Supuesto 4:

En el contexto de desarrollar una plataforma de salud y bienestar para hospitales de renombre, con un presupuesto muy alto, tiempos de entrega de 12-15 meses, un equipo multidisciplinario y la necesidad de un rendimiento excepcional y una seguridad de datos primordial, la elección de la **Clean Architecture** se justifica por las siguientes consideraciones:

- **Presupuesto Muy Alto:**

  Con un presupuesto significativo disponible, es posible invertir en una arquitectura sólida que garantice la calidad del software y cumpla con los estándares de seguridad requeridos en el sector de la salud.

- **Tiempos de Entrega Razonables:**

  Aunque el proyecto tiene un plazo de 12-15 meses, la Clean Architecture facilita un desarrollo ordenado y escalable. Su estructura modular y su enfoque en la separación de capas permiten a los equipos trabajar en paralelo, lo que puede acelerar el desarrollo y asegurar la entrega oportuna.

- **Recursos Humanos Multidisciplinarios:**

  Con un equipo multidisciplinario que incluye desarrolladores, especialistas en seguridad, diseñadores y analistas, la Clean Architecture se adapta bien a este entorno. Cada grupo puede enfocarse en su área de especialización dentro de la arquitectura, lo que facilita la colaboración y la asignación de responsabilidades.

- **Rendimiento y Escalabilidad:**

  Dado que se espera un alto tráfico debido a la gran cantidad de pacientes, la Clean Architecture permite una gestión eficaz del rendimiento y la escalabilidad. La separación de capas y la modularidad facilitan la optimización y el escalado de componentes críticos para mantener un rendimiento óptimo.

- **Seguridad y Privacidad de Datos:**

  En el ámbito de la salud, la seguridad y la privacidad de los datos son fundamentales. La Clean Architecture permite la implementación de medidas sólidas de seguridad en cada capa, lo que garantiza la integridad de los datos y el cumplimiento de las regulaciones de privacidad.

En conclusión, la elección de la Clean Architecture es apropiada para una plataforma de salud y bienestar en un hospital. Ofrece la estructura, la escalabilidad y la seguridad necesarias para abordar los requisitos de rendimiento y privacidad de datos en un proyecto de esta magnitud.

Supuesto 5: Aplicación Prototipo para un Hackathon

Contexto: Un grupo de estudiantes decide participar en un hackathon de 48 horas. Su objetivo es crear un prototipo funcional de una aplicación móvil que ayude a las personas a encontrar compañeros de viaje para compartir gastos en carreteras de peaje.

Presupuesto: Mínimo. Los estudiantes usarán herramientas y recursos gratuitos disponibles.

Tiempos de entrega: 48-72 horas.

Recursos humanos: Un equipo de tres estudiantes con habilidades mixtas: un desarrollador, un diseñador y alguien con habilidades de negocio**.**

Rendimiento: Como es un prototipo, no se espera un tráfico real. La aplicación debe ser lo suficientemente funcional para demostrar la idea.

Evaluación del Supuesto 5:

En el contexto de desarrollar un prototipo funcional de una aplicación móvil para un hackathon de 48 horas, con un presupuesto mínimo, un equipo de tres estudiantes y la necesidad de una funcionalidad básica para demostrar la idea, la elección de la arquitectura de software adecuada debe ser ágil y centrada en la entrega rápida. En este caso, se justifica la elección de la arquitectura **MVC (Model-View-Controller)** debido a las siguientes consideraciones:

- **Presupuesto Mínimo:**

  Dado que el presupuesto es limitado y se deben utilizar herramientas y recursos gratuitos, MVC es una opción simple y económica de implementar.

- **Tiempos de Entrega Muy Cortos:**

  El tiempo es un factor crítico en un hackathon de pocas horas, y MVC permite una implementación rápida y directa. La estructura de tres componentes (Modelo, Vista y Controlador) es fácil de comprender y trabajar de manera eficiente en un período de tiempo tan ajustado.

- **Recursos Humanos Limitados:**

  Con un equipo pequeño y mixto de estudiantes, MVC es una opción apropiada debido a su simplicidad. Cada miembro del equipo puede concentrarse en una de las tres áreas principales (Modelo, Vista o Controlador), lo que facilita la colaboración y la asignación de tareas.

- **Funcionalidad Básica para Demostrar la Idea:**

  Dado que se trata de un prototipo y no se espera un tráfico real, MVC permite desarrollar la funcionalidad básica necesaria para demostrar la viabilidad de la idea en un corto período de tiempo.

- **Entrega Ágil:**

  MVC se adapta bien a una metodología de desarrollo ágil, lo que permite realizar iteraciones rápidas y realizar cambios en función de la retroalimentación del hackathon.

En conclusión, la elección de la arquitectura MVC es adecuada para el desarrollo de un prototipo funcional en un entorno de hackathon con recursos limitados, plazos ajustados y un enfoque en la entrega rápida de una idea. Proporciona la simplicidad y la agilidad necesarias para cumplir con los objetivos del proyecto en un corto período de tiempo.

Bibliografía:

MVC, MVP y MVVM en Android. KeepCoding. https://keepcoding.io/blog/que-es-mvc-mvp-y-mvvm-en-android/ [en línea] 05/10/23

Arquitectura Hexagonal. Becas Santander. https://www.becas-santander.com/es/blog/arquitectura-hexagonal.html [en línea] 04/10/23

Introducción a las arquitecturas limpias. Universidad Nacional Autónoma de México. https://www.redisybd.unam.mx/redisybd/pluginfile.php/1565/mod_resource/content/1/ArquitecturasLimpias.pdf [en línea] 05/10/23