УДК 004.415.23

МОДУЛЬНАЯ АРХЕТИКТУРА В МОБИЛЬНОЙ РАЗРАБОТКЕ НА ПРИМЕРЕ ПРИЛОЖЕНИЯ ДЛЯ КУРАЦИИ ПАЦИЕНТОВ

Матафонов Д.С.

ФГАОУ ВО «УрФУ имени первого Президента России Б.Н. Ельцина»,

Екатеринбург, Россия

dmatafonov2000@mail.ru

Аннотация. Описаны причины применения многомодульной архитектуры в приложениях под ОС Android. Модули классифицированы по своему назначению, на основании чего приведена универсальная структура их связей. Сформированы рекомендации, позволяющие учесть преимущества и ограничения подхода. Примеры приведены для проекта по сопровождению ВИЧ-положительных детей.

Ключевые слова: мобильная разработка, архитектура, android, модульность, структура проекта.

MULTI-MODULE ARCHITECTURE IN MOBILE DEVELOPMENT USING THE EXAMPLE OF A PATIENT SUPERVISION APPLICATION

Matafonov D.S.

Ural Federal University, Ekaterinburg, Russia

Abstract. The article describes the reasons for using a multi-module architecture in Android applications. The classification of modules according to their purpose is derived, based on which the universal structure of their connections is given. Recommendations have been formed to consider the advantages and limitations of multimodularity. Examples are provided for a project on the support of HIV-positive children.

Keywords: mobile development, architecture, android, modularity, project structure.

1. Введение

Одной из целей курации пациентов, в том числе и с ВИЧ-инфекцией, является выработка у них приверженности к лечению. В случае пациентов детского возраста в этом поможет геймификация [1]. Идея заключается в том, чтобы создать мобильное приложение-помощник, которое будет поощрять прием медикаментов и даст возможность онлайн-консультации с врачом.

Основой архитектуры такого приложения стала модульность, решающая проблемы скорости сборки, повторного использования кода и позволяющая построить строгую архитектуру. Идея разделить проект исходит от самой специфики приложения. Оно состоит из нескольких потенциально независимых частей (напоминания, игры, личные кабинеты пациента, врача, родителя), разработка которых ведется разными разработчиками, а изменения одной части не должны затрагивать все остальные. Чтобы правильно сформировать структуру, модули были классифицированы.

2. Основная часть

Главными недостатками монолитного подхода, когда используется один модуль, являются высокая связность, невозможность изолированности кода и, в случае Android и Gradle, низкая скорость сборки, структурной единицей которой и является модуль. Когда модуль один, каждое новое изменение требует полной сборки [2]. Но у «монолита» есть ключевое преимущество – простота реализации. Чтобы модульная архитектура не стала препятствием в условиях ограниченных ресурсов проекта, в команде разработки были приняты правила, делающие модульный подход обоснованным.

Главный принцип – стремление к «горизонтальной» структуре модулей. Структура тем ближе к идеальной, чем меньше зависимостей одного модуля от другого. Принцип разбиения сложился из специфики приложения – созданы модули функциональных возможностей, таких как чат, напоминания и т. д. Они могут быть модифицированы независимо друг от друга, и

именно эта гибкость является главным фактором отказа от монолитного подхода.

Уровнем выше от функциональных модулей реализован модуль приложения – app. App координирует функциональные модули, управляет запуском приложения, инициализацией внедрения зависимостей и навигации. App зависит от всех модулей, но никакой модуль – от App.

Различные функциональные модули требуют использования общих решений. Каждый класс экрана использует общую схему работы с жизненным циклом, навигацией, обработкой состояний и так далее – для этого создан модуль базовых классов. Во всем приложении используется единый стиль и компоненты UI, помещенные в модуль uikit. В некоторых случаях для сбора данных о пользовательском опыте был необходим модуль аналитики. Такие утилитные модули образуют отдельный слой – ядро. Часть получившейся структуры модулей показана на рисунке 1.

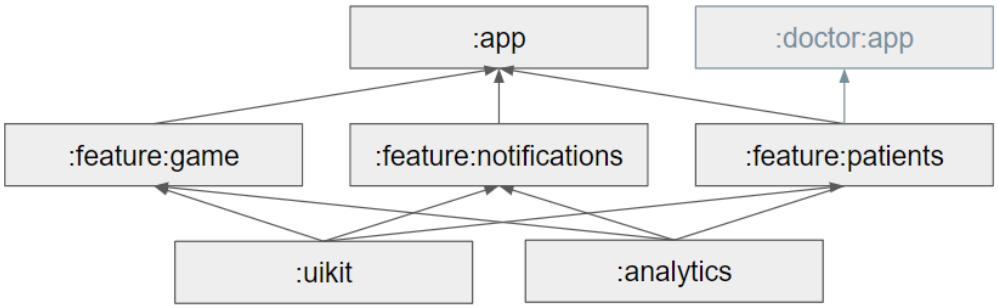

Рисунок 1 – Структура модулей приложения

Использование общего ядра позволяет не только использовать общий код, но и создавать несколько приложений на одной базе. Так, например, личный кабинет врача реализован в том же проекте.

Отметим принципиальный момент: внутри модулей одного уровня нет взаимозависимостей. Их появление лишает преимуществ модульного подхода и, в случае циклической зависимости, делает сборку невозможной. Если сущности из одного модуля нужны в другом, модуль делится на две части: общее и частное, интерфейс (внешний контракт) и реализацию [3].

Случай с функционалом чата и пациентов рассмотрен на рисунке 2, где common обозначены общие части, а реализации – impl (implementation).

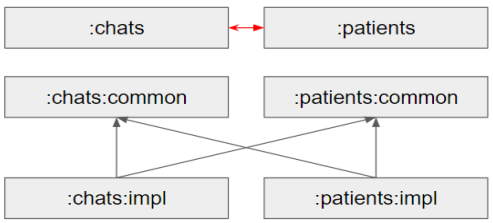

Рисунок 2 – Случай двух связанных модулей

В этом случае классы из чатов нужны в модуле для работы с пациентами, и наоборот – сборка становится невозможной. Выделение общих частей решает эту проблему. Даже если взаимной зависимости нет, выделение общей части улучшает архитектуру, изолируя реализации двух модулей. Расходы на создание модулей были минимизированы за счет использования шаблонов.

3. Заключение

Модули были классифицированы, что позволило выстроить связи между ними: функциональные модули, минимально связанные между собой, объединены модулем приложения и используют утилитные модули в случае необходимости. Минимизировать связность функциональных модулей помогает их разделение на общие части и реализацию. В рамках работы над приложением для пациентов модульность способствовала удобному разделению проекта и его масштабированию, что является определяющим фактором в выборе архитектуры.

Библиографический список

1. Н. В. Савченко. Разработка мобильного сервиса в сфере здравоохранения для детей, живущих с ВИЧ / Н. В. Савченко, К. И. Николаева, М. А. Уфимцева [и др.] // Уральский медицинский журнал. – 2020. – № 12(195). – С. 135-139.

2. Santiago-Salazar, J. Clean Architecture: Impact on Performance and Maintainability of Native Android Projects / J. Santiago-Salazar, D. Rico-Bautista. – Advances in Computing, 2023 – С. 82-90.

3. V. Gordienko. Модульная архитектура: преимущества – URL: https://proglib.io/p/modulnaya-arhitektura-chto-kak-i-pochemu-2023-04-04 (дата обращения: 24.03.2024) – Текст: электронный.