**УДК 004.4**

# ИСПОЛЬЗОВАНИЕ ЧИСТОЙ АРХИТЕКТУРЫ ПРИ РАЗРАБОТКЕ ANDROID ПРИЛОЖЕНИЙ

***Р.Р. КРАСЬКО***
(***Представлено: канд. техн. наук, доц. А.Ф.ОСЬКИН***)

*В статье описываются особенности разработки мобильного Android приложения с использованием чистой архитектуры. Предлагается набор инструментов для программной реализации данного подхода.*

**Введение.** Архитектура приложения является важнейшим аспектом при разработке программного обеспечения - она должна быть надежной, стабильной, гибкой в тестировании, легко расширяться и изменяться. В то же время архитектура должна быть понятной для разработчиков с разным уровнем подготовки и опыта для обеспечения удобства сопровождения.

**Чистая архитектура.** При использовании чистой архитектуры весь код приложения разделен на уровни, которые придерживаются одного правила: внутренний уровень не должен ничего знать про внешний. При этом внутренний уровень содержит бизнес логику, а внешний ее реализацию в зависимости от платформы. Данный подход должен отвечать и другим требованиям:

- независимость от инструментов. Архитектура не должна полагаться на существование какой-либо библиотеки. Это позволяет использовать с минимальными затратами изменять реализацию бизнес-логики [1];
- тестируемость, бизнес-логика должна быть тестируемой без любых внешних элементов [1];
- независимость от базы данных, бизнес-логика не должна быть привязана к конкретным базам данных [1];
- независимость от любого внешнего агента, бизнес-логика не должна знать ничего о взаимодействии в внешними источниками данных [1].

На рисунке предоставлены различные уровни и компоненты архитектуры и их взаимодействие.

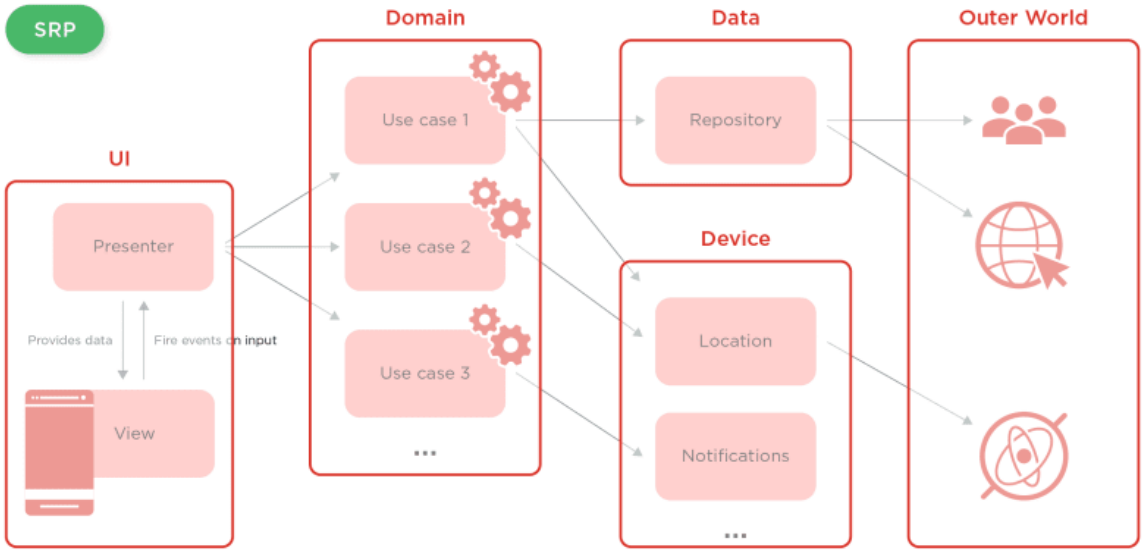


**Рисунок. – Уровни и компоненты чистой архитектуры**

**Уровень представления (Presentation/UI):**

- для данного слоя выбран паттерн MVP(Model VIew Presenter), также возможно использование MVVM;
- «View» представляет собой реализацию паттерна «Passive View» и предоставляет собой набор интерфейсов, которые должны быть реализованы компонентами из Android SDK (Fragment, Activity, Adapter, Custom View) [2];
- «Presenter» является посредником между пользовательским интерфейсом и бизнес логикой. Должен быть платформонезависимым для улучшения тестируемости кода [2];

− при тестировании используются «JUnit» и «Mockito». Для интеграционных тестов также используется «Esspresso».

**Уровень бизнес-логики (Domain):**

− Состоит из вариантов использования (Use Cases), которые представляют собой самые примитивные части бизнес логики приложения для максимальной простоты и возможности их дальнейшего переиспользования;

− При тестировании используются «JUnit» и «Mockito».

**Уровень данных (Data):**

− паттерн репозиторий отвечает за интерфейс, по которому варианты изпользования из доменного уровня могут получать необходимые данные [3];

− на этом уровне возможна реализация кеширования;

− при тестировании используются «JUnit», «Mockito», «Robolectrick».

**Платформозависимый слой (Device):**

− содержит реализации платформозависимой функциональности: уведомления, сенсоры, различные менеджеры и так далее.

**Взаимодействие между слоями, навигация:**

− внедрение зависимостей с использованием Dagger 2;

− поток данных между слоями осуществляется с помощью RxJava;

− для навигации используется Cicerone;

**Заключение.** Описаны особенности использования чистой архитектуры при разработке приложений для мобильных Android приложений. Предложен набор инструментов для реализации данного подхода. Данная архитектура является гибкой в поддержке, простой в тестировании и очень удобной при работе в больших командах с различным уровнем специалистов.

ЛИТЕРАТУРА

1. Чистая архитектура на Android и iOS [Электронный ресурс] / Apptractor © 2019 – Режим доступа: https://apptractor.ru/develop/chistaya-arhitektura-na-android-i-ios.html – Дата доступа: 26.09.2019.
2. Model–view–presenter [Электронный ресурс] / Wikipedia © 2019. – Режим доступа: https://en.wikipedia.org/wiki/Model–view–presenter – Дата доступа: 26.09.2019.
3. Repository [Электронный ресурс] / Martinfowle © 2019. – Режим доступа: https://martinfowler.com/eaaCatalog/repository.html – Дата доступа: 26.09.2019.