# РАСШИРЕНИЕ ПАТТЕРНА MVC ЗА СЧЕТ ДОБАВЛЕНИЯ ИЕРАРХИЧЕСКОЙ СТРУКТУРЫ И СЕРВИС-СЛОЯ ПРИ РЕАЛИЗАЦИИ ПРОГРАММНОЙ ЛОГИКИ МОБИЛЬНОГО ПРИЛОЖЕНИЯ НА ПРИМЕРЕ РАЗРАБОТКИ ПРОЕКТА «ЛИЧНЫЙ КАБИНЕТ СТУДЕНТА ИГУ»

*Д. Д. Закаулова, С. А. Карноухов, В. С. Кедрин*

Одним из важнейших этапов разработки программных продуктов является проектирование архитектуры, которое позволяет сформулировать описание связей между логическими компонентами этой системы, а также определяет принципы её проектирования и развития.

Архитектурный паттерн Model-View-Controller (MVC) представляет собой разделение прикладных программ на три части: 1) модель, содержащую данные; 2) представление, отображающее информацию пользователю; 3) контроллер, обрабатывающий действия пользователя и реализующий бизнес-логику приложения[1]. Такой подход позволяет реализовать логику изолирования модели данных от представления и избежать дублирование кода по принципу Don’t Repeat Yourself (DRY).

В описываемом паттерне принято придерживаться принципа целесообразности: слои приложения общаются между собой посредством модельных объектов – классов с «прозрачной» реализацией, состоящих исключительно из акцессоров и мутаторов. Подобные классы не несут с собой никакой логики и предназначены лишь для хранения и перемещения данных[2].

Использование шаблона MVC для одновременной разработки различных частей приложения можно реализовать с помощью организации иерархической структуры архитектуры приложения за счет расширения архитектуры MVC до Hierarchical Model View Controller (HMVC). Блоки MVC в данной концепции функционируют независимо и взаимодействуют через контроллеры. Такой подход позволяет расширить возможности для повторного использования кода и уменьшить зависимости между различными частями приложения. Это делает программный продукт доступным для масштабирования без потери легкости поддержки.

Стоит отметить, что иерархическая структура HMVC используется как для разделения серверной и клиентской частей, так и при их разбиении на несколько независимых друг от друга блоков, в каждом из которых используется шаблон MVC.

Паттерн MVC подразумевает, что вся логика приложения реализуется в контроллере, который, в случае разработки достаточно сложной клиентской части, описывает не только методы передачи данных, но и бизнес-процессы, заложенные в интерфейсе пользователя и логике доступа к модели данных. Это нарушает The Single Responsibility Principle (SRP) в классах контролера, а также значительно перегружает код в данной части приложения и может спровоцировать определенные трудности при масштабировании. В этой ситуации имеет смысл говорить о выделении отдельного блока – сервис-слоя, реализующего бизнес-логику приложения.

Такая интерпретация подразумевает разделение модуля на четыре основных блока: модель, сервис-слой, контроллер и представление (Model – Service – Controller – View). Добавление нового блока позволяет разделить функциональность контроллера на две части: взаимодействие с представлениями и вызов сервиса будет в ведении контроллера, а за взаимодействие с моделями и прочую бизнес-логику теперь будет

отвечать сервис. Внедрение нового слоя повлияет и на реализацию HMVC. Теперь взаимодействие между блоками MSVC подразумевает вызов стороннего контроллера сервисом.

Таким образом, возможно расширение паттерна MVC до HMSVC (Hierarchical Model Service View Controller), что позволит значительно повысить возможности для повторного использования кода и увеличит его способность к масштабированию.

На практике модель HMSVC наглядно представима на примере разрабатываемого мобильного приложения «Личный кабинет студента ИГУ». Иерархия реализуется за счет разделения разработки на серверную и клиентскую, причем последняя представлена в двух частях: для android и для iOS. В клиентской части view отвечает за UI, controller принимает и обрабатывает данные из view-слоя и вызывает service-слой, который работает с model, реализует остальную бизнес-логику и возвращает результат своих действий. Клиентский service-слой так же вызывает контроллер сервера, который передает данные своему service-слою, реализующему бизнес-логику сервера и работающему со своей моделью. View-слой на сервере явно не используется.

Таким образом, предложенная архитектура HMSCV позволяет значительно упростить командную разработку мобильного приложения, повышает способность кода к повторному использованию и решает проблемы, связанные с реализацией логической структуры как всего приложения, так и отдельных ее частей.

**Литература**


1. Reenskaug T. MVC. Xerox PARC 1978-79. // http://heim.ifi.uio.no/~trygver/themes/mvc/mvc-index.html (дата обращения: 20.10.2010)
2. Мейер Б. Объектно-ориентированное конструирование программных систем/ Б. Мейер. - Русская Редакция, 2005. - 1204с.