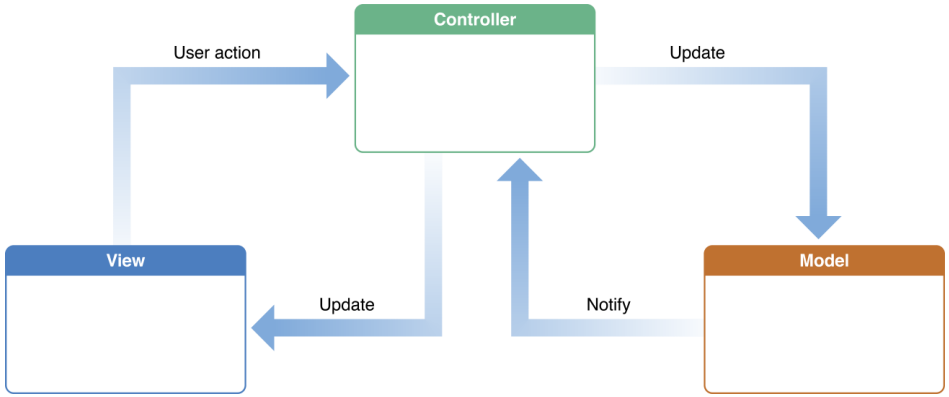
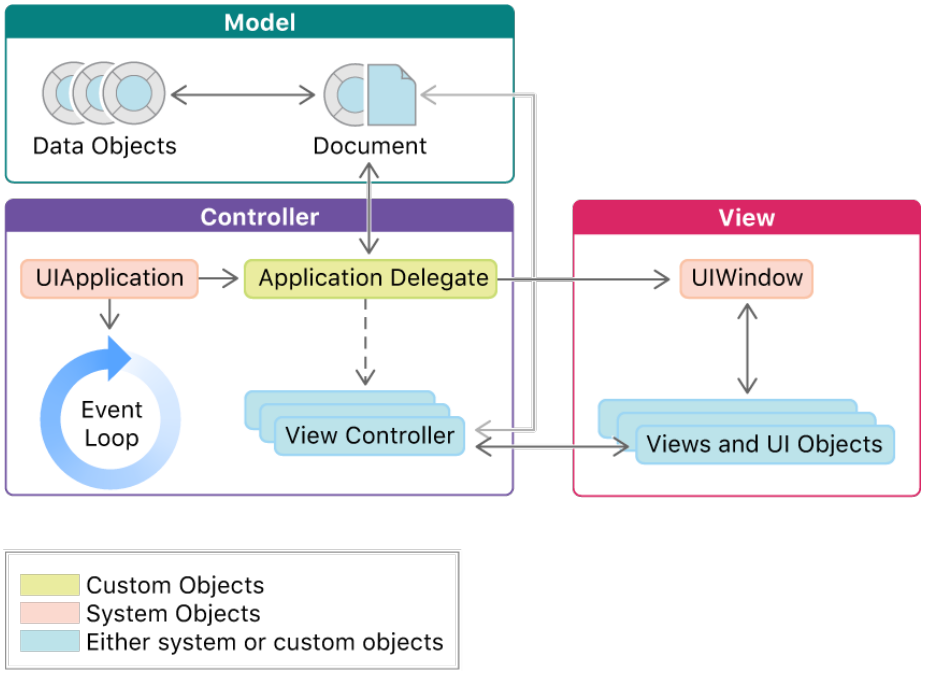
1. MVC

Xerox 1978(1979)

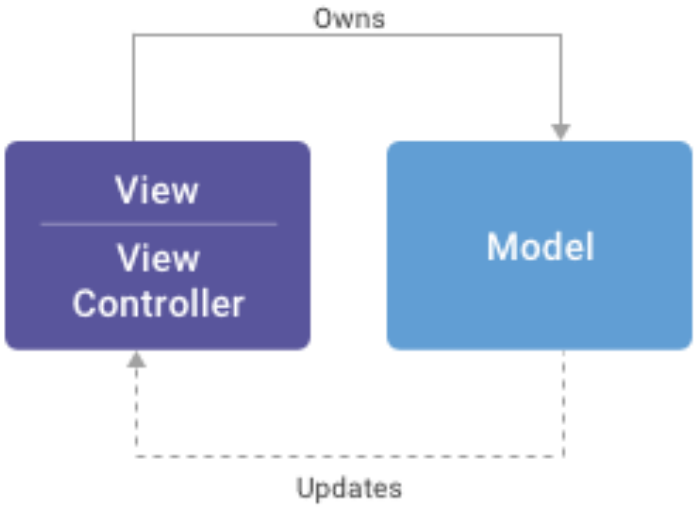


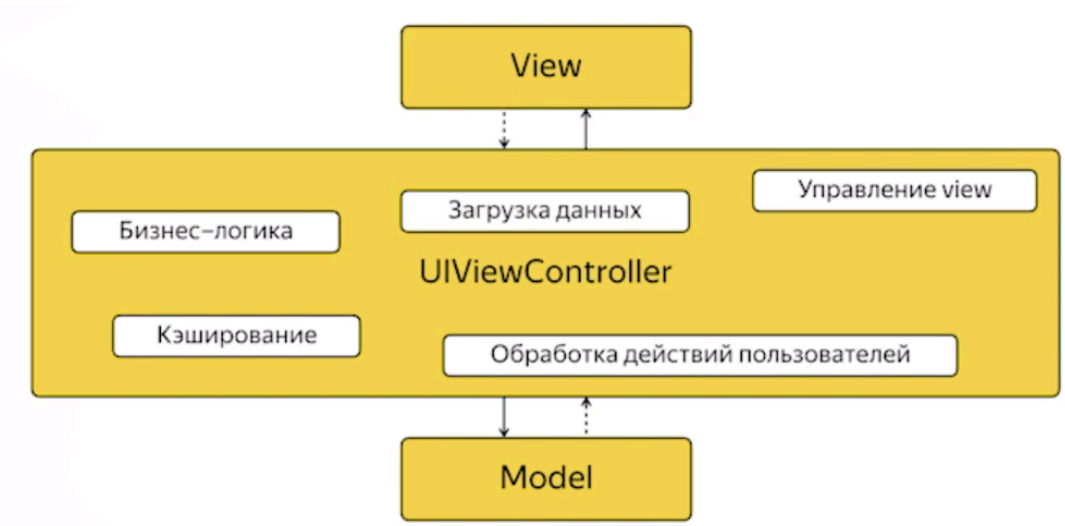


+ Меньше кода
+ Простота
+ Единственный архитектурный паттерн в документации от Apple

- Сложности в Unit-тестировании
- Сложная инкапсуляция
- Слабое разделение ответственности
- Сложно дебажить
- Плохо масштабируется
- Massive View Controller

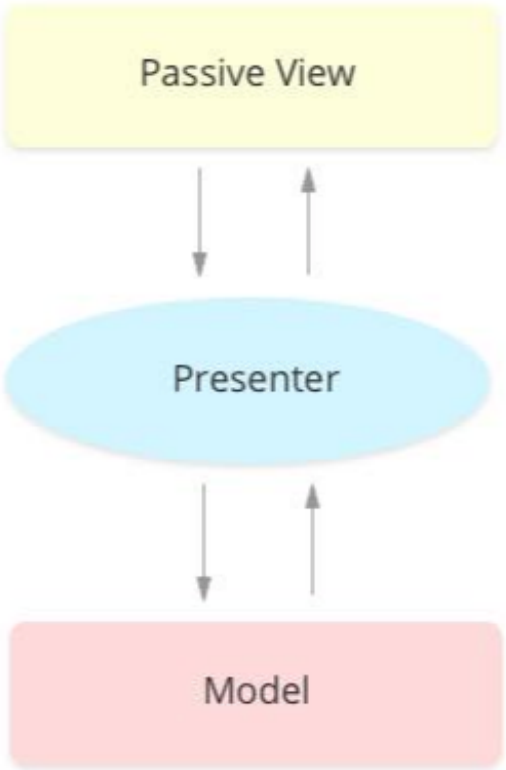
~~Проблемы~~ Нюансы MVC в iOS





2. MVP

IBM 1990

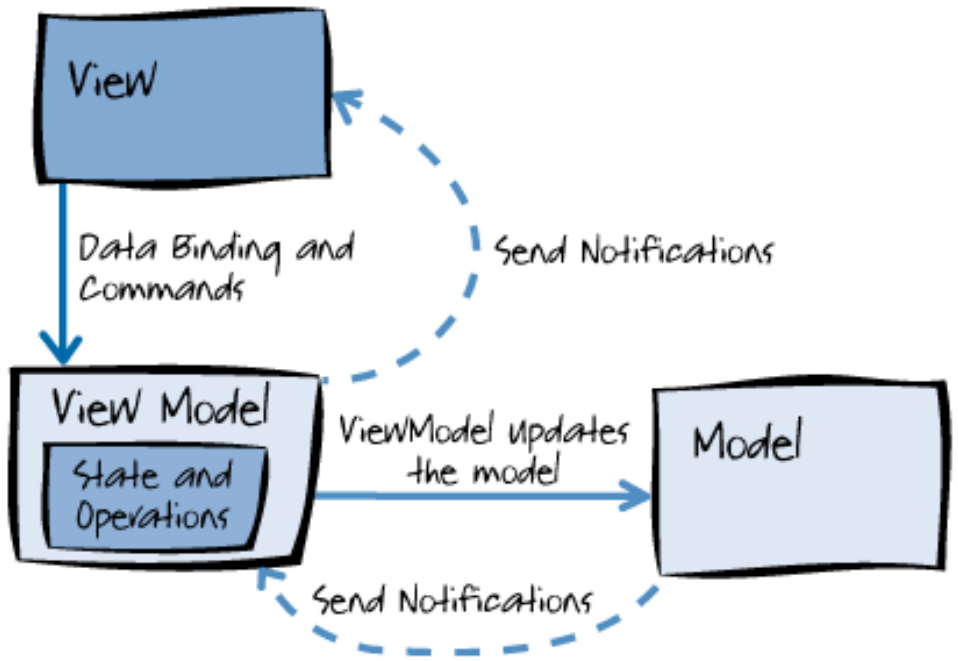


+ Возможность модульного тестирования
+ Упрощение понимания кода
+ Соответствие SOLID принципу единственной ответственности (The Single Responsibility Principle)
+ Одно представление используется разными презентерами, или наоборот - один презентер используется для разных представлений

- Лишний шаблонный код
- Уменьшается скорость разработки для небольших проектов

3. MVVM

Microsoft 2005



+ Более компактная логика

- Обязательное связывание данных (data binding)

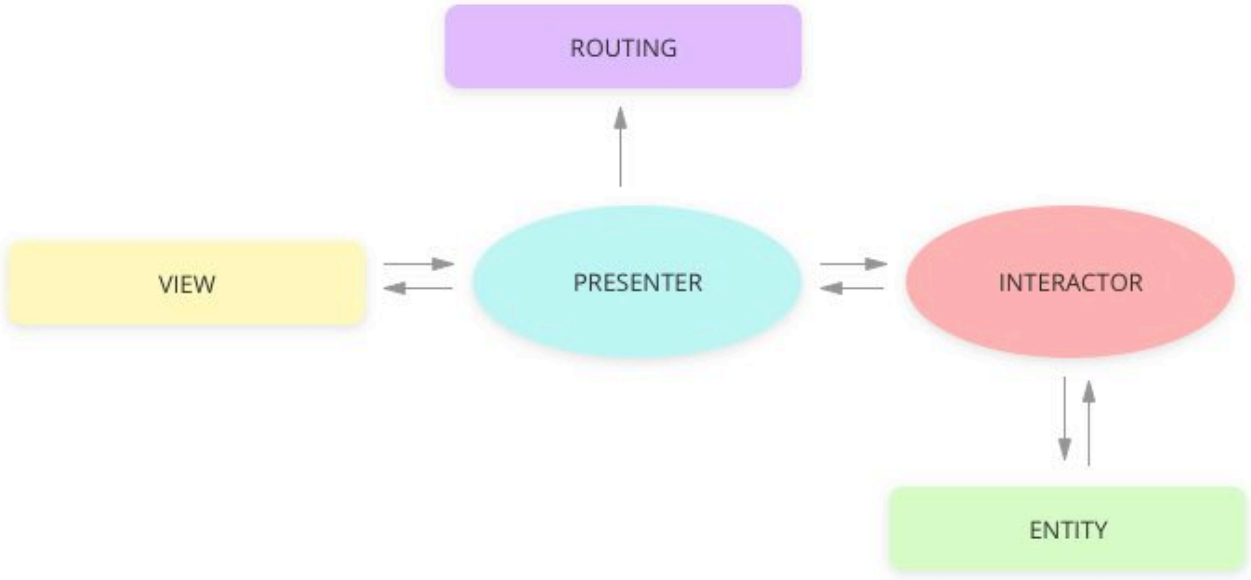
+- Иногда возникают абсурдные ситуации

ViewModel не имеет ссылки на интерфейс представления (View).
Изменение состояния View-модели автоматически изменяет представление и наоборот, поскольку используется механизм связывания данных (Bindings).

Часто используется с Functional Reactive Programming (напр. RxSwift).

4. VIPER

Статья [The Clean Architecture](#) от Роберта Мартина 2012
MutualMobile 2013



+ Повышение тестируемости
+ Полная независимость модулей друг от друга, что позволяет независимо их разрабатывать и переиспользовать как в одном приложении, так и в нескольких
+ Простота входа для опытного разработчика

- Увеличение количества классов в проекте
- Увеличение времени разработки
- Проблемы с документацией: отсутствие best practices и примеров для сложных или специфических задач
- Сложность входа для неопытного разработчика