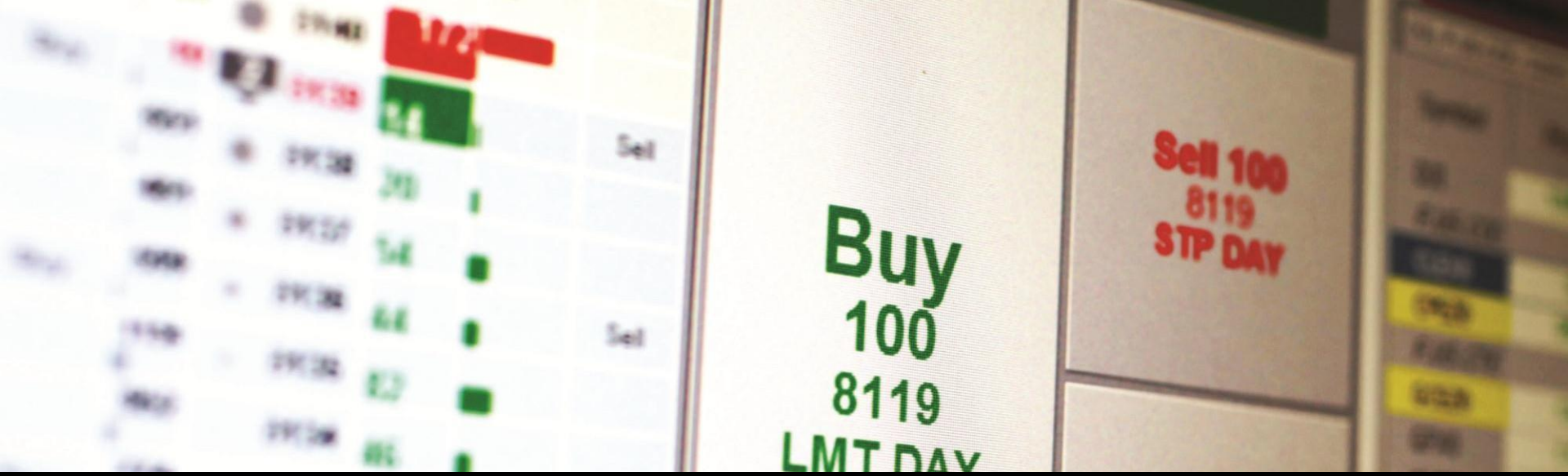


Проектирование ПО

Дрожжин Алексей

CQG

Структура лекции

Основные разделы и подразделы.

- Проектирование
- Архитектура
  - Слоистая,
  - Модульная,
  - Микросервисная,
- Шаблоны
  - Шаблоны проектирования,
  - Антипаттерны,
  - MVC-MVP-MVVM,
- Принципы программирования
  - SOLID
- Повторение
- Q&A
- Литература

Что такое проектирование

Проектирование (дизайн) — процесс создания проекта программного обеспечения, а так же дисциплина, изучающая методы проектирования.

Цели проектирования

Целью проектирования является определение внутренних свойств системы, детализация ее требований и их анализ

Цели проектирования

Затраченное время

Сложность проекта

Важное vs второстепенное

- Размер помещений важен
- Удобство важно

Важное vs второстепенное

- Размер помещений важен
- Удобство важно
- Материалы второстепенны
- Убранство второстепенно

Важное vs второстепенное

- Предметная область важна
- Сценарии использования важны

Важное vs второстепенное

- Предметная область важна
- Сценарии использования важны
- Представление второстепенно
- Хранение второстепенно

CQG

Что проектируем?

Проектированию обычно подлежат:

- Архитектура ПО
- Компоненты
- Интерфейсы

Проектирование

Проектирование должно:

- Быть инструментом
- Упрощать жизнь
- Решать проблемы

Проектирование не должно:

- Быть целью
- Усложнять жизнь
- Вносить проблемы

Архитектура

Архитектура ПО – совокупность решений об организации программной системы. Архитектура включает

- Выбор структурных элементов и интерфейсов
- Соединение элементов в более крупные системы
- Архитектурный стиль, соединение элементов

Архитектура

Критерии хорошей архитектуры

- Масштабируемость
- Расширяемость
- Заменимость модулей
- Возможность переиспользования
- Тестируемость
- Сопровождаемость

Архитектура

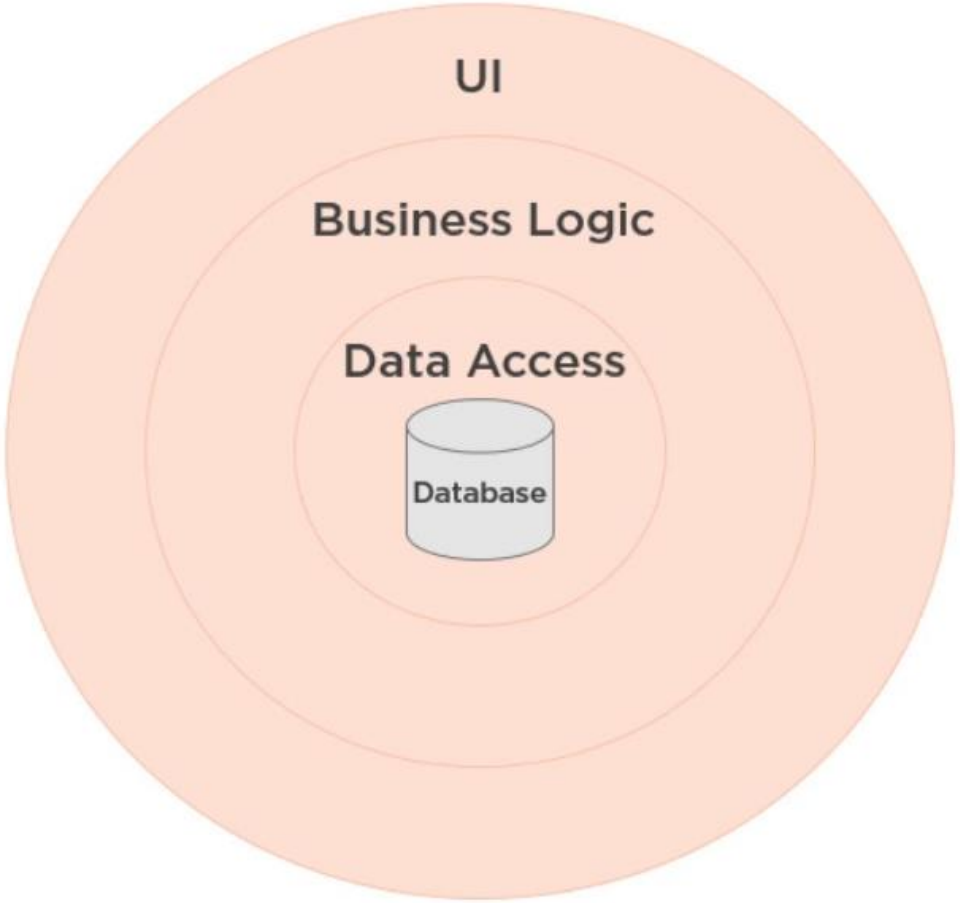
Критерии плохой архитектуры

- Жесткость
- Хрупкость
- Неподвижность

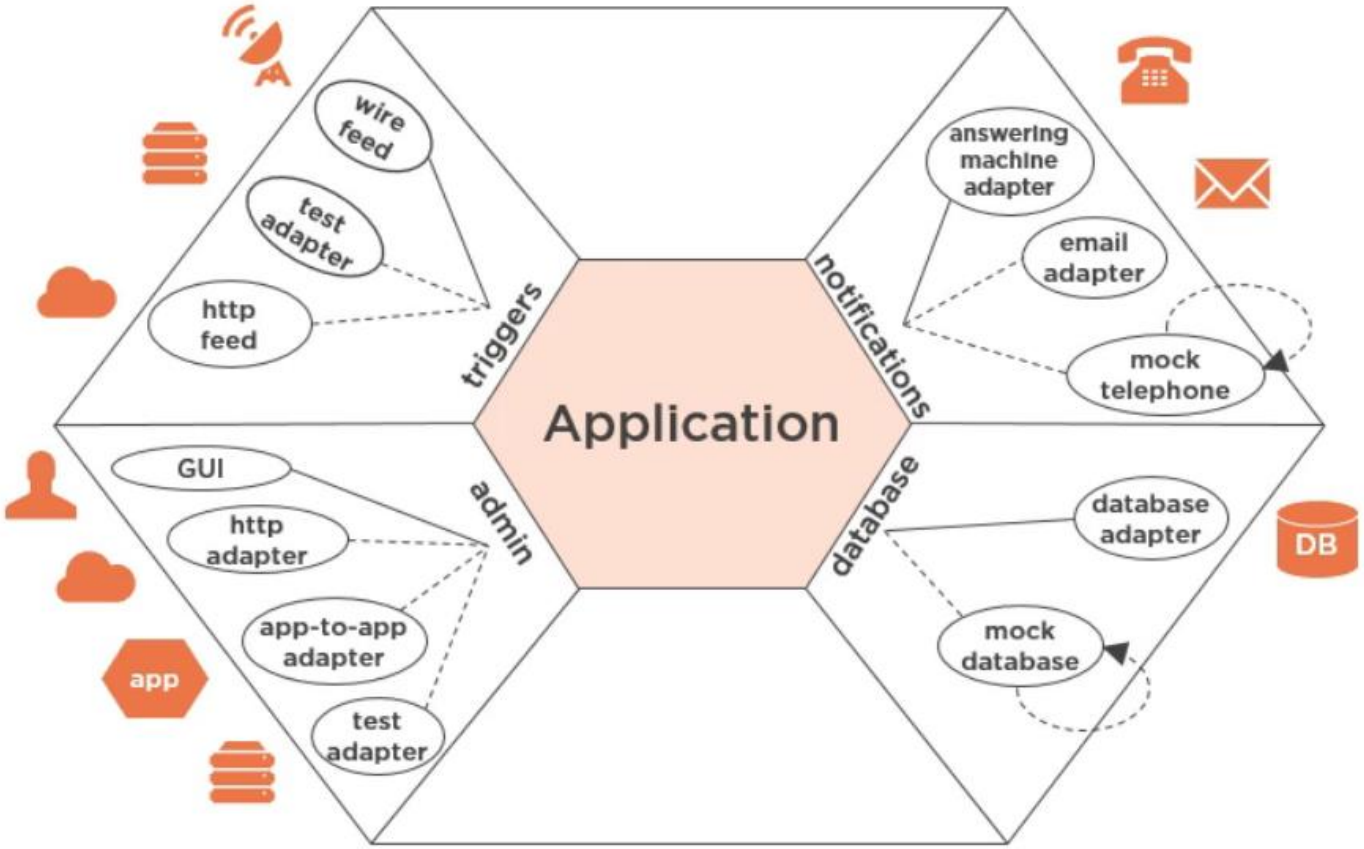
Типы архитектуры

- Монолитная
- Модульная
- Микросервисная

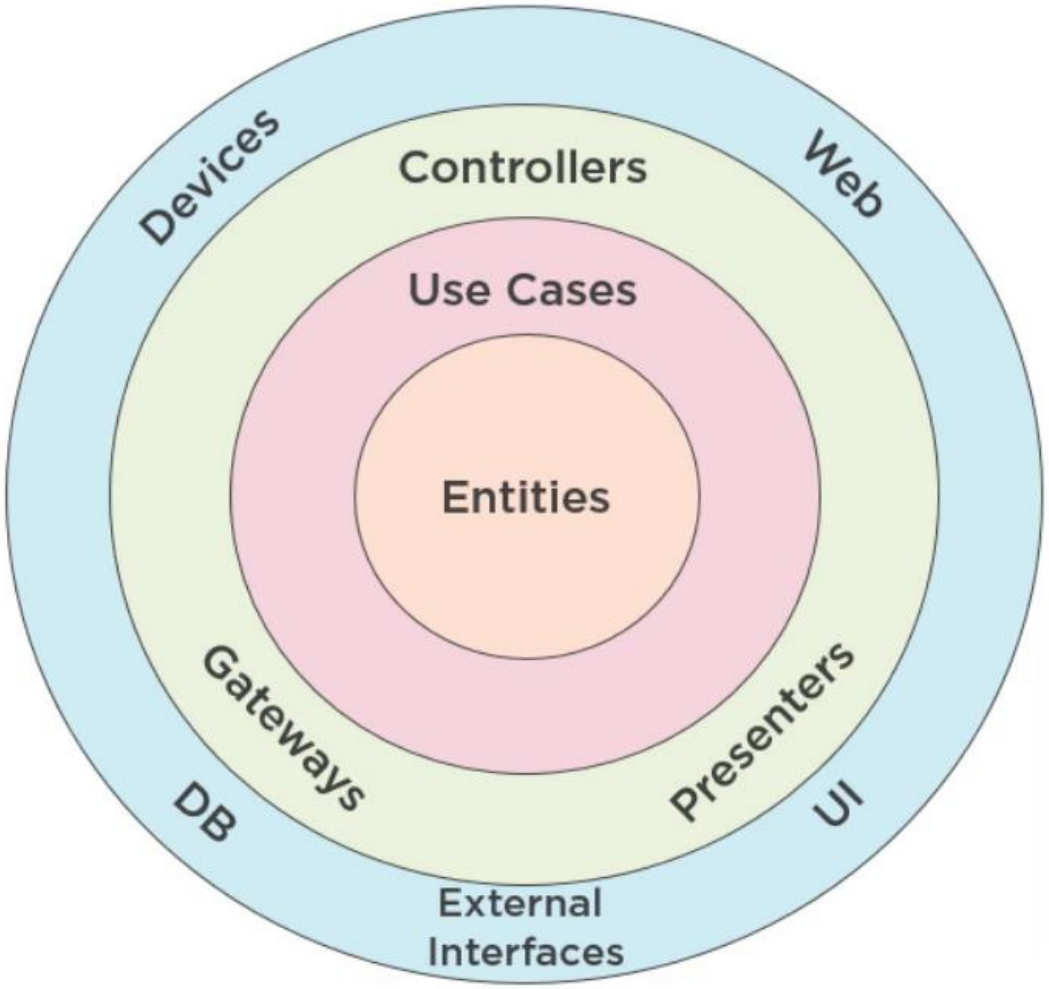
Многоуровневая архитектура



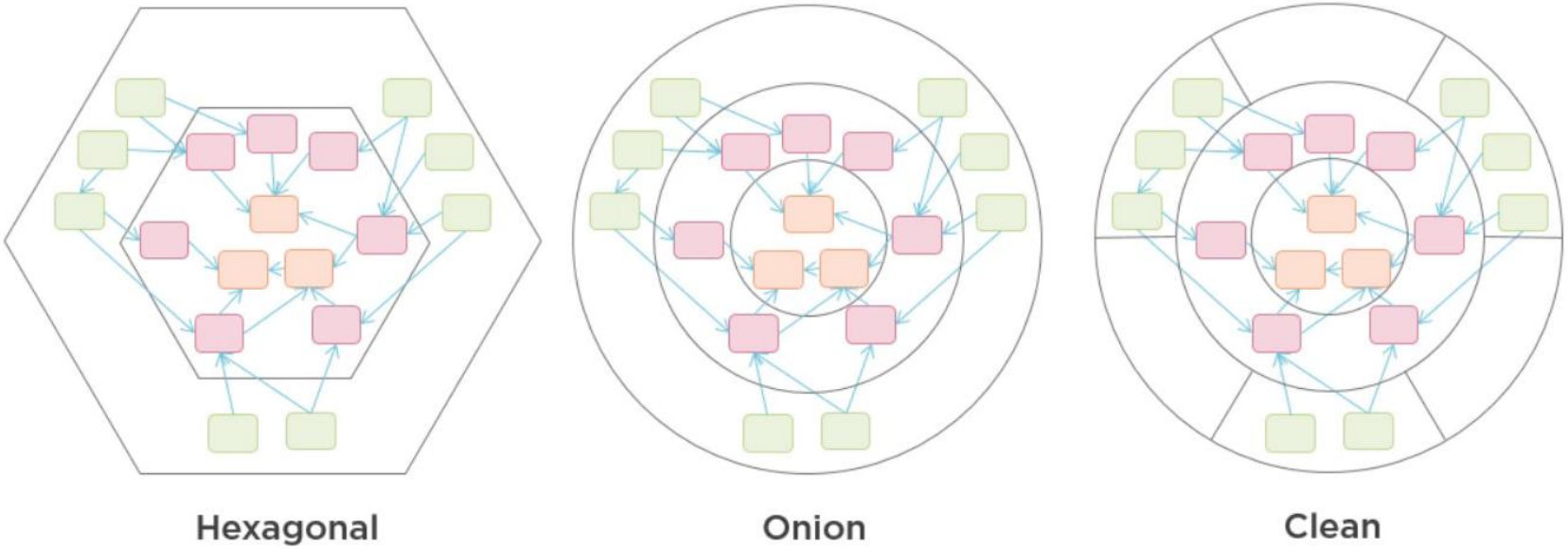
Многоуровневая архитектура



Многоуровневая архитектура



Сходство подходов



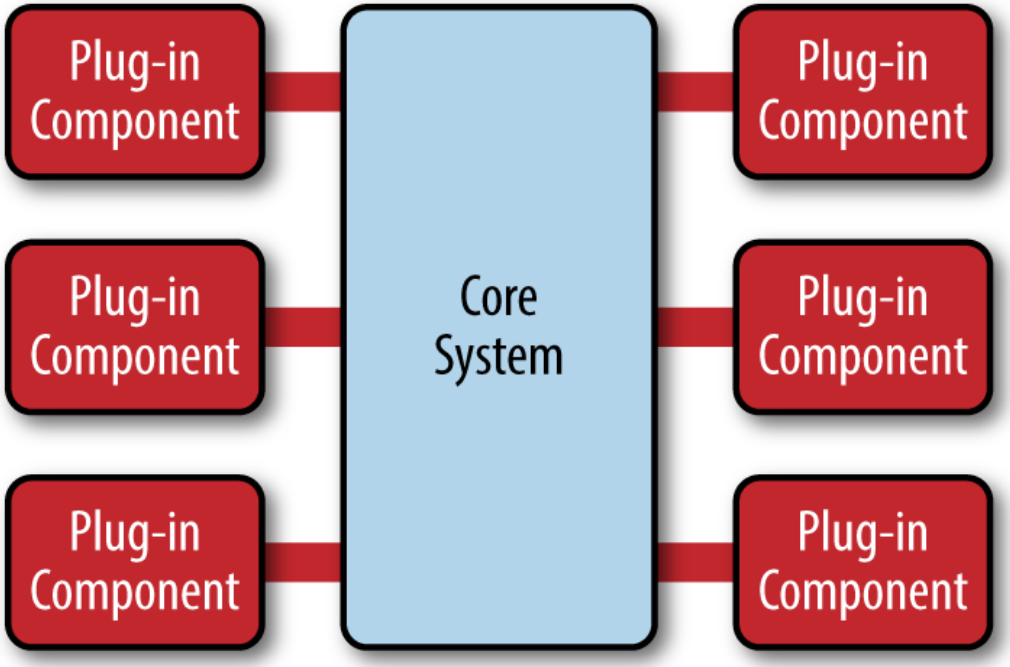
Hexagonal

Onion

Clean

CQG

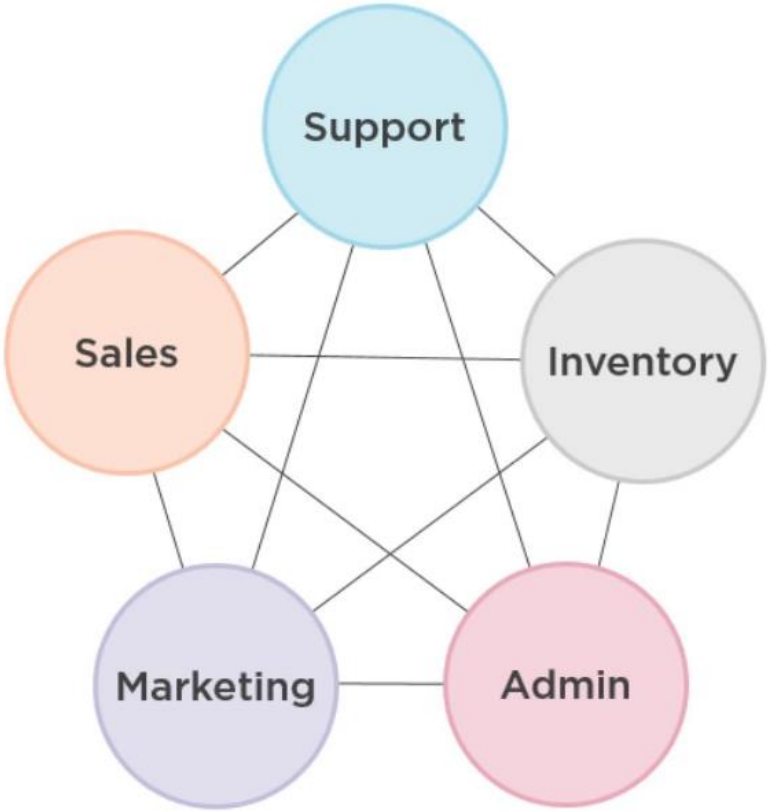
Модульная архитектура



CQG

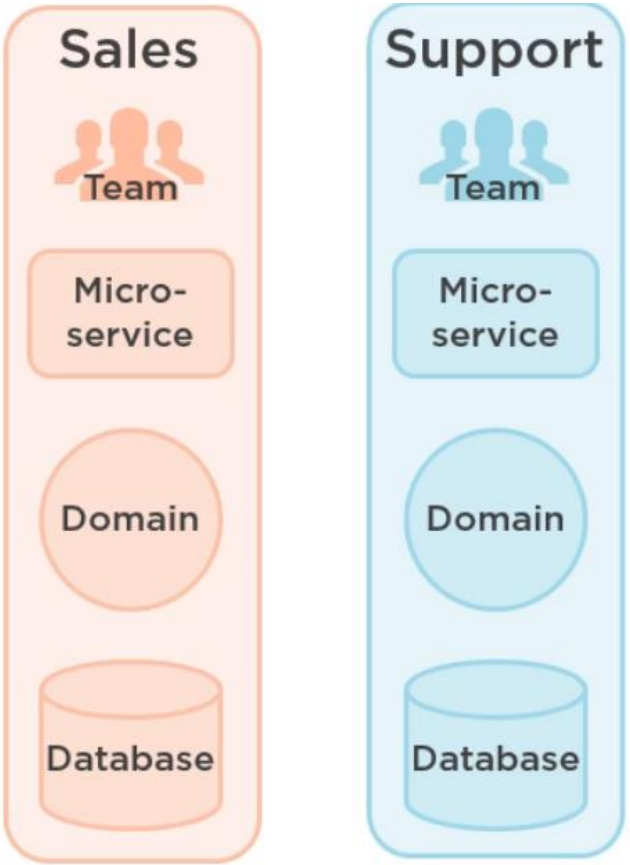
Микросервисная архитектура

- Разделение на под-системы
- Понятные интерфейсы
- Небольшие команды
- Независимость разработки и развертывания



Микросервисная архитектура

- Разделение контекстов
- Высокая сопряжённость и слабая связанность
- Фокус на одном контексте
- Независимость подходов и инструментов



Микросервисная архитектура

Преимущества

- Простота расширения системы
- Высокая сопряжённость и слабая связанность
- Независимость

Недостатки

- Издержки в начале разработки
- Закон Конвея
- Издержки распределенных систем

Микросервисная архитектура

Плохая

Слоистая

Микросервисная/модульная

Шаблоны (паттерны) проектирования

Шаблон (паттерн) – повторяемая архитектурная конструкция, решение часто возникающей проблемы проектирования

Шаблоны (паттерны) проектирования

Примеры шаблонов:

- Синглтон
- Фабрика
- Наблюдатель (observer)
- Посетитель
- …

Шаблоны (паттерны) проектирования

Плюсы:

- Снижение сложности
- Облегчение коммуникации
- Унификация решений

Шаблоны (паттерны) проектирования

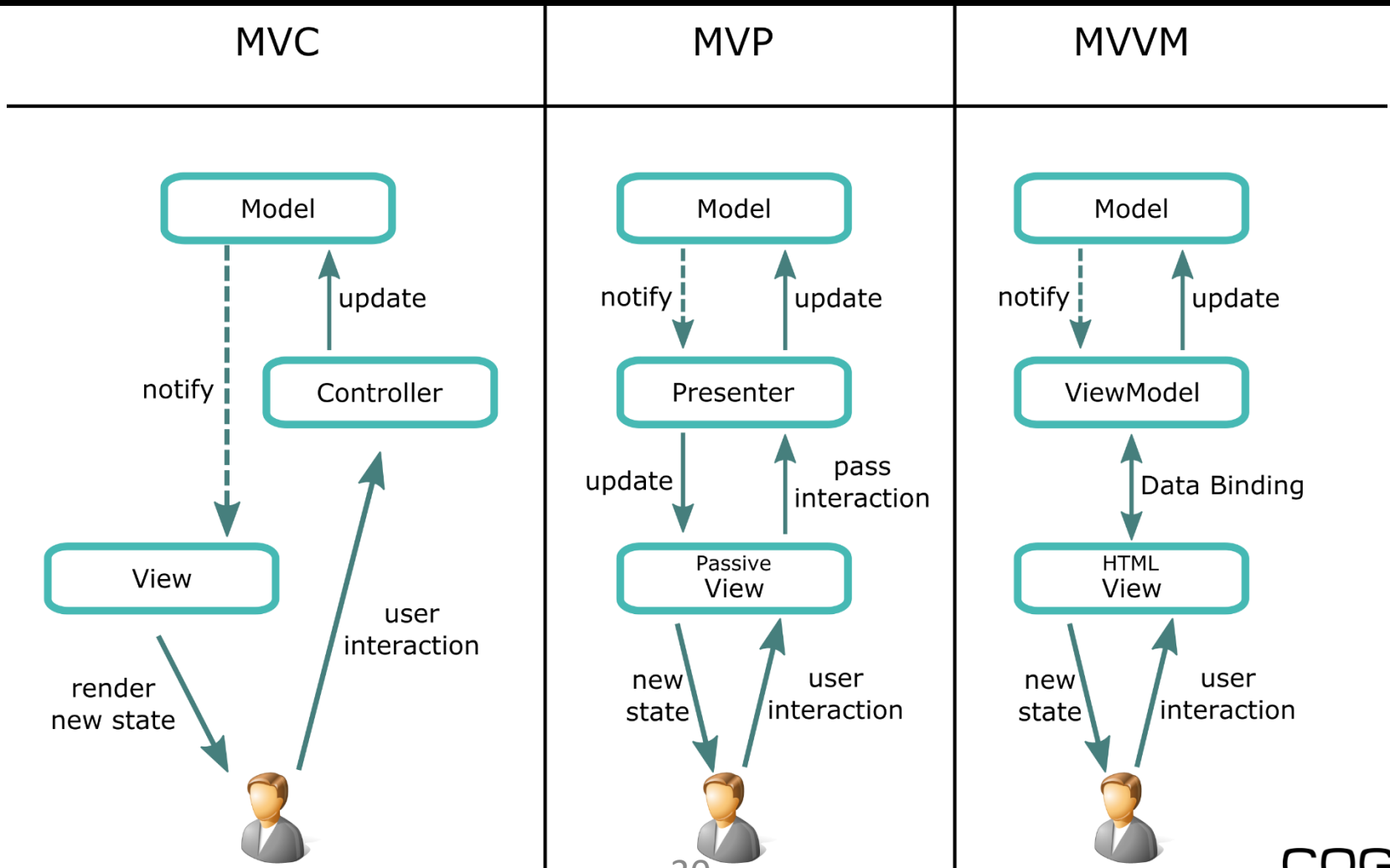
Минусы:

- Слепое следование приводит к усложнению
- Паттерны ради паттернов
- Иногда вызывает религиозные войны

Антипаттерны

- Магические константы
- Copy-paste programming
- Божественный объект
- Hard Code / Soft Code
- Изобретение велосипеда
- Паблик Морозов

Шаблоны проектирования

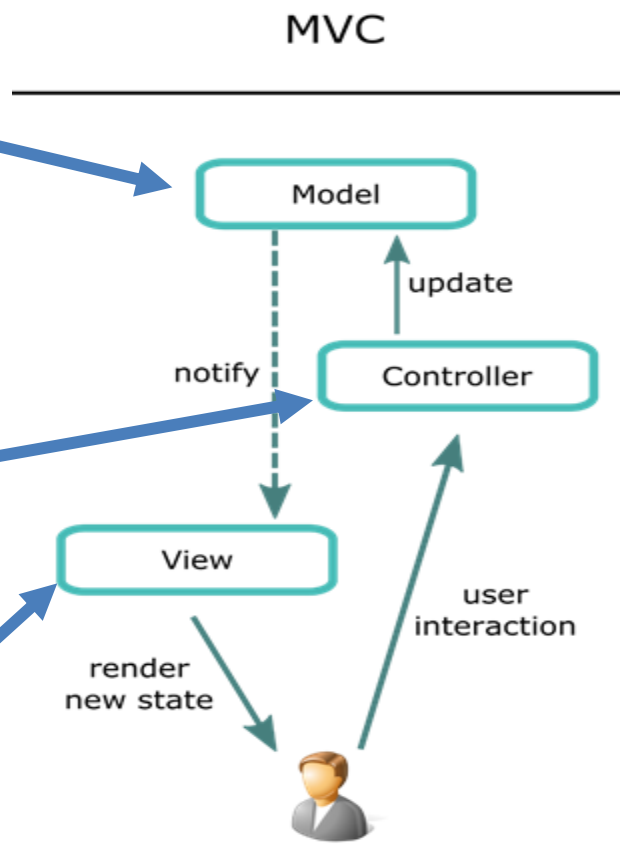


MVC

Предоставляет данные и реагирует на команды контроллера, меняя своё состояние

Интерпретирует действия пользователя, меняет модель

Отображает данные и реагирует на изменения модели

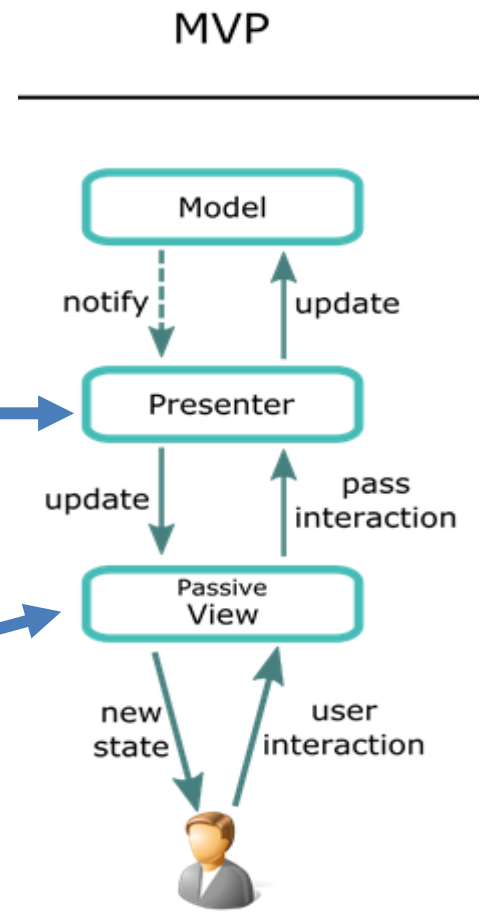


MVP

Данные для отображения

Содержит в себе всю логику, получает данные от модели и обновляет их

Отображает данные и реагирует на изменения модели

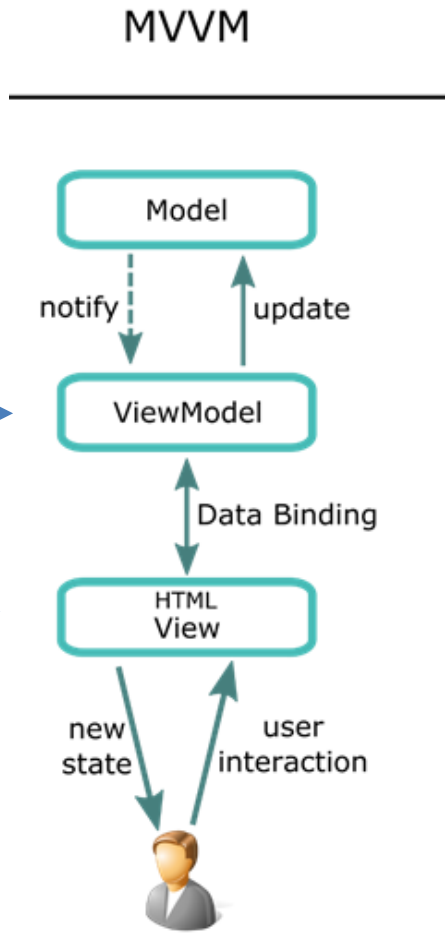


MVVM

Данные для отображения

Обертка над моделью + команды для ее изменения

Графическое отображение модели



Принципы

- KISS (Keep It Simple, Stupid)
- DRY (Don’t repeat yourself)
- UNIX way
- Less is more
- YAGNI (You aint gonna need it)
- SOLID (на следующем слайде)

SOLID

- Single responsibility
- Open-closed
- Liskov substitution
- Interface segregation
- Dependency inversion

Повторим

При проектировании, мы должны:

- Понять требования
- Определить важное и второстепенное
- Выбрать подход к проектированию
- Выбрать тип архитектуры

Повторим

При создании архитектуры:

- Разделить уровни ответственности
- Отделить данные от кода
- Использовать паттерны
- Следовать принципам программирования

Q&A

Очень интересная литература

- «Банда четырёх»: Приемы объектно-ориентированного проектирования. Паттерны проектирования
- С. Макконнелл: Совершенный код
- Р. Мартин: Чистая архитектура. Искусство разработки программного обеспечения

CQG