Архитектура

Что такое архитектура?

«Уровни» решения задачи

Уровни бизнеса, продукта, проекта:

- Что решаем
- Почему решаем
- Для чего решаем
- Для кого решаем (кто клиент/заказчик/бенефициар)
- В каких условиях решаем

Уровень архитектуры

- КАК РЕШАЕМ

Архитектура

Упрощение разработки, развертывания и сопровождения

(с учетом *всех* проектных условий)

Естественная стратегия*:*

*как можно **дольше** иметь как можно **больше вариантов***

Задачи

- Поддержание жизненного цикла ПО
- Легкость освоения
- Простота разработки, сопровождения и развертывания

- Минимизация затрат на проект
- Максимизация продуктивности программистов

Жизнь

- Небольшая команда – монолит
- Несколько небольших команд – по компоненту на команду

Разработка – развертывание – сопровождение: разные процессы

- Что хорошо для разработки – может быть плохо для развертывания и наоборот

Зависимость разных аспектов проекта от архитектуры

Сверхсильная зависимость

- Сопровождение

Сильная зависимость

- Разработка, развертывание

Слабая/неоднозначная зависимость

- Эффективность

Почти нулевая зависимость

- Функциональность

Разнообразие вариантов

- «Политика»

Бизнес-правила и логика

- «Детали»

Базы данных, GUI, Протоколы, ввод-вывод, фреймворки, библиотеки

Думай о главном(с)

Откладывай вопрос о деталях

Думай о главном

Хороший архитектор **максимизирует** количество непринятых решений

Баланс

«Неизвестные» и «переменные» факторы

- Все варианты использования
- Эксплуатационные ограничения
- Структура команды разработчиков
- Требования к развертыванию
- Специфика применения
- Внешняя ситуация

Горизонтальные уровни

Режем по причинам изменений

Уровень – удаленность от ввода и вывода

Уровень	Сущности
1	Бизнес правила, связанные с предметной областью
2	Бизнес-правила, связанные с приложением
3	UI, База данных, драйвера внешних устройств

Вертикальные узкие срезы

Режем по меняющимся и появляющимся вариантам использования

Срез «варианта использования»

- часть UI
- часть бизнес логики приложения
- часть бизнес логики предметной области
- часть базы данных

Тонкости дублирования

- Устранение истинного дублирования
- Похожие сущности и алгоритмы в разных уровнях и срезах – это нормально

Режимы разделения

- Уровень исходного кода (монолит)
- Уровень развертывания
- Уровень локального независимого выполнения
- Уровень служб

Границы

Разработка архитектуры – искусство проведения разделяющих линий – границ.

Архитектура плагинов:

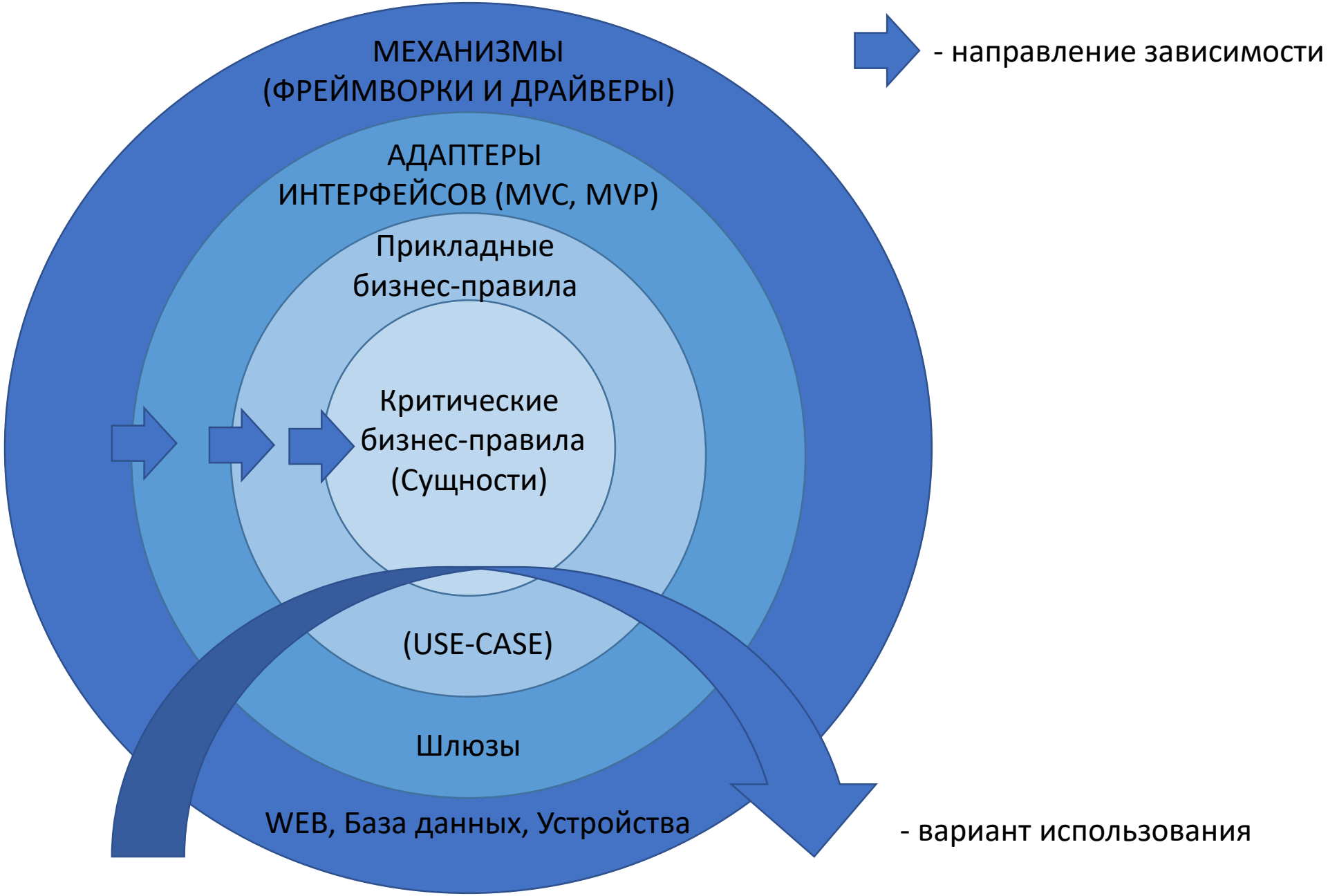
- Независимые высоукоровневые компоненты
- **Граница**
- Зависимые низкоуровневые компоненты

Жесткость границ – вопрос выбора архитектора

Как построить чистую архитектуру?

- Независимость от фреймворков
- Простота тестирования
- Независимость от UI
- Независимость от базы данных
- Независимость от любых внешних агентов

- **Явная зависимость от назначения**


МЕХАНИЗМЫ
(ФРЕЙМВОРКИ И ДРАЙВЕРЫ)
АДАПТЕРЫ
ИНТЕРФЕЙСОВ (MVC, MVP)
Прикладные
бизнес-правила
Критические
бизнес-правила
(Сущности)
(USE-CASE)
Шлюзы
WEB, База данных, Устройства
- направление зависимости
- вариант использования

Где MAIN ?

Микросервисная и SOA архитектура – спасение?

- Это не архитектура
- Это не спасение

Почему?

Потому что сервисы и микросервисы –

это **те же самые** компоненты