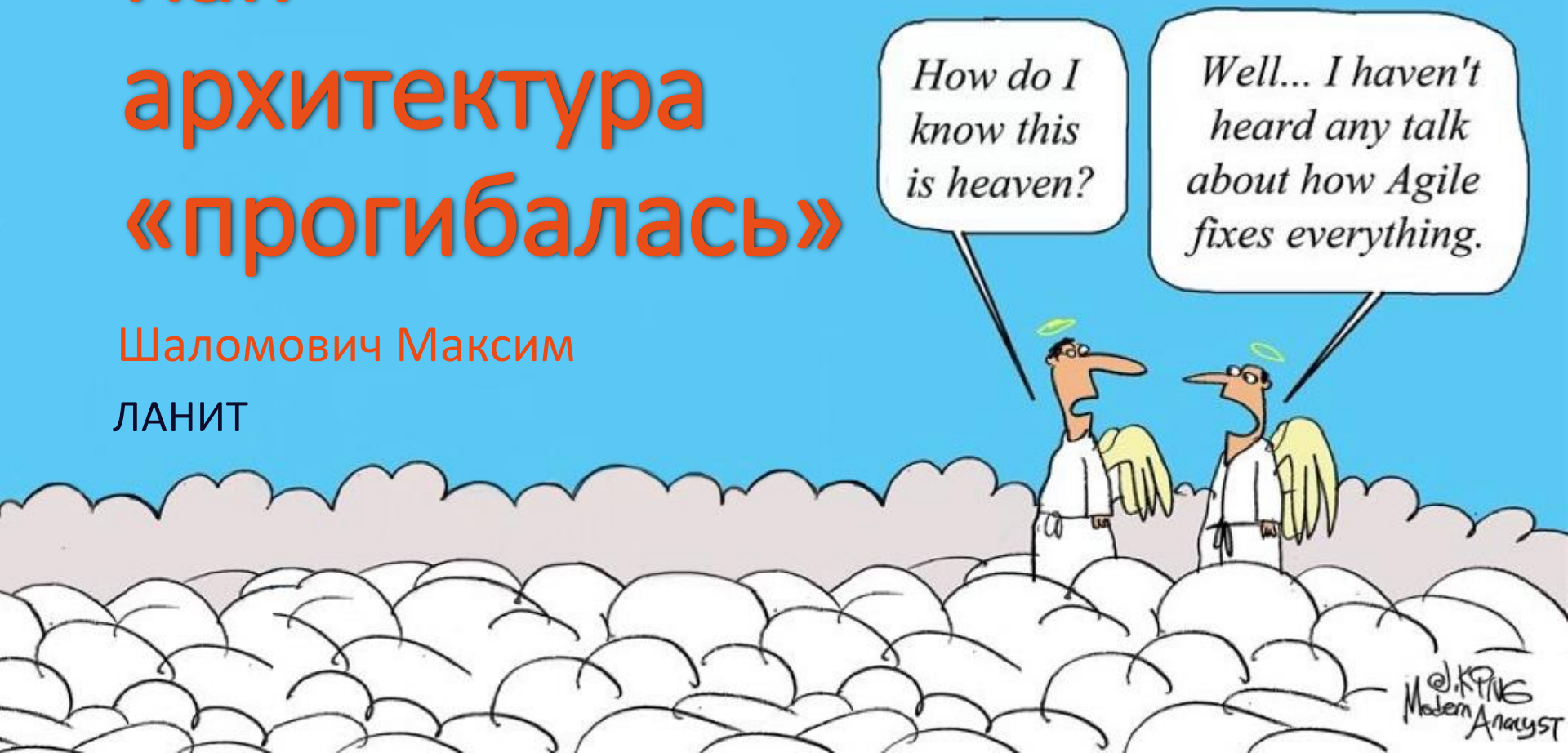
Как архитектура «прогибалась»

Шаломович Максим

ЛАНИТ



Вступление

Тут пока не очень много полезной информации

Докладчик

Максим Шаломович

- ✓ИТ-архитектор
- ✓Проектирую системы и решения
- ✓Интересуюсь ИТ-проектами и ИТ-процессами
- ✓Выступаю на конференциях, делюсь опытом
- ~~✓Пою, пляшу, крестиком вышиваю~~

Доклад

Архитектурная практика в Agile-проектах

- ✓«Не до архитектуры нам, кодить надо!»
- ✓«Лучше один раз вовремя, чем семь раз хорошо!»

Компания ЛАНИТ

- Разработка ИС «под ключ»
- Проектирование комплексных решений
- Бизнес-проектирование

«Типичный заказчик»

- Государство
- Крупные коммерческие организации

«Типичный заказчик»

- Государство
- Крупные коммерческие организации

То есть…

- Снаружи – регуляторы, нормативы
- Внутри – тяжелые процессы бюджетирования, планирования, оценки рисков

Что не так?

Первая мысль, которая может прийти вам в голову

«У нас будет Agile-проект!»

«У нас будет Agile-проект!»

- …но с ограниченным бюджетом, четкими сроками сдачи и невозможностью отступать от контракта

Проблемы

Проблемы

- Нет MVP, слишком много деталей

Проблемы

- Нет MVP, слишком много деталей
- Гибкость = «делай, что хочешь, все равно переделывать»

Проблемы

- Нет MVP, слишком много деталей
- Гибкость = «делай, что хочешь, все равно переделывать»
- Непредсказуемые затраты

Проблемы

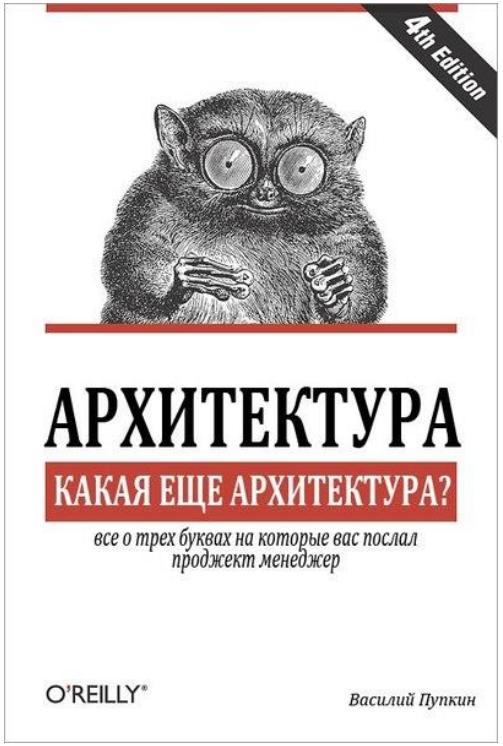
- Нет MVP, слишком много деталей
- Гибкость = «делай, что хочешь, все равно переделывать»
- Непредсказуемые затраты
- Бюджет все еще ограничен


AGILE SOFTWARE DEVELOPMENT
ON A FIXED BUDGET PROJECT

«Решения»

«Решения»

- Сокращаем «ненужные» стадии ЖЦ проекта



«Решения»

- Сокращаем «ненужные» стадии ЖЦ проекта
- Обкладываемся ограничениями ТЗ

«Решения»

- Сокращаем «ненужные» стадии ЖЦ проекта
- Обкладываемся ограничениями ТЗ
- Любое изменение проводим через ЗНИ

Результат

Результат

- «Гибкость» есть только на словах

Результат

- «Гибкость» есть только на словах
- Каждое изменение может привести к переработке большой части ИС

Результат

- «Гибкость» есть только на словах
- Каждое изменение может привести к переработке большой части ИС
- Каждое изменение стоит дорого

«У нас должна быть гибкая архитектура!»

Следующая мысль, которая может прийти к вам в голову

Типовой ЖЦ «гибкого» ИТ-проекта

- Предварительное планирование
- Разработка по итерациям
  - Анализ (спецификации)
  - Разработка (код)
  - Тестирование (тесты)
  - «Приемка» (обратная связь)

Точки принятия архитектурных решений

- Предварительное планирование с проектированием
- Разработка по итерациям
  - Анализ (спецификации)
  - Проектирование
  - Разработка (код)
  - Тестирование (тесты)
  - Архитектурный надзор
  - «Приемка» (обратная связь)

Предварительное планирование с проектированием

Предварительное планирование с проектированием

- Технологический стек

Предварительное планирование с проектированием

- Технологический стек
- Переиспользуемые компоненты системного уровня

Предварительное планирование с проектированием

- Технологический стек
- Переиспользуемые компоненты системного уровня
- Верхнеуровневое описание архитектуры

Предварительное планирование с проектированием

- Технологический стек
- Переиспользуемые компоненты системного уровня
- Верхнеуровневое описание архитектуры
- Показатели качества

Предварительное планирование…

…без проектирования	…с проектированием

Предварительное планирование…

…без проектирования	…с проектированием
Формальный набор документов (ТП, ТЗ, ЁПРСТ)	

Предварительное планирование…

…без проектирования	…с проектированием
Формальный набор документов (ТП, ТЗ, ЁПРСТ)	Документы позволяют зафиксировать важные моменты

Предварительное планирование…

…без проектирования	…с проектированием
Формальный набор документов (ТП, ТЗ, ЁПРСТ)	Документы позволяют зафиксировать важные моменты
Разработка не использует результаты этого этапа	

Предварительное планирование…

…без проектирования	…с проектированием
Формальный набор документов (ТП, ТЗ, ЁПРСТ)	Документы позволяют зафиксировать важные моменты
Разработка не использует результаты этого этапа	Документы максимально практические

Предварительное планирование…

…без проектирования	…с проектированием
Формальный набор документов (ТП, ТЗ, ЁПРСТ)	Документы позволяют зафиксировать важные моменты
Разработка не использует результаты этого этапа	Документы максимально практические
Основные потребности выявляются по ходу, no-DUF	

Предварительное планирование…

…без проектирования	…с проектированием
Формальный набор документов (ТП, ТЗ, ЁПРСТ)	Документы позволяют зафиксировать важные моменты
Разработка не использует результаты этого этапа	Документы максимально практические
Основные потребности выявляются по ходу, no-DUF	Just-in-time DUF с набором опций

Анализ и разработка итераций с проектированием

Анализ и разработка итераций с проектированием

- Определение порядка реализации сценариев с учетом архитектурных приоритетов

Анализ и разработка итерации с проектированием

- Определение порядка реализации сценариев с учетом архитектурных приоритетов
- Детальное проектирование критичных сценариев (+ технические практики)

Анализ и разработка итераций с проектированием

- Определение порядка реализации сценариев с учетом архитектурных приоритетов
- Детальное проектирование критичных сценариев (+ технические практики)
- Контроль актуальности предварительной архитектуры

Анализ и разработка в итерации…

…без проектирования	…с проектированием

Анализ и разработка в итерации…

…без проектирования	…с проектированием
Архитектура устаревает и перестает отвечать на вопросы разработки	

Анализ и разработка в итерации…

…без проектирования	…с проектированием
Архитектура устаревает и перестает отвечать на вопросы разработки	Архитектура обновляется по результатам изменения/ добавления требований

Анализ и разработка в итерации…

…без проектирования	…с проектированием
Архитектура устаревает и перестает отвечать на вопросы разработки	Архитектура обновляется по результатам изменения/ добавления требований
Критичные сценарии могут быть некачественно реализованы	

Анализ и разработка в итерации…

…без проектирования	…с проектированием
Архитектура устаревает и перестает отвечать на вопросы разработки	Архитектура обновляется по результатам изменения/ добавления требований
Критичные сценарии могут быть некачественно реализованы	Критичные сценарии детализируются перед разработкой

Архитектурный надзор…

…не выполняется	…выполняется

Архитектурный надзор…

…не выполняется	…выполняется
Нет результатов реализации архитектурных решений	

Архитектурный надзор…

…не выполняется	…выполняется
Нет результатов реализации архитектурных решений	Проверка архитектурных опций и гипотез через практику

Архитектурный надзор…

…не выполняется	…выполняется
Нет результатов реализации архитектурных решений	Проверка архитектурных опций и гипотез через практику
Причина (история) реализации неочевидна	

Архитектурный надзор…

…не выполняется	…выполняется
Нет результатов реализации архитектурных решений	Проверка архитектурных опций и гипотез через практику
Причина (история) реализации неочевидна	Появляется обоснование принятых решений и реализации (ADR)

Итог

Краткое резюме всего, что я тут вам наговорил

Архитектура в Agile-проектах позволяет

Архитектура в Agile-проектах позволяет

- Удерживать целостность постоянно изменяемого и развиваемого решения

Архитектура в Agile-проектах позволяет

- Удерживать целостность постоянно изменяемого и развиваемого решения
- Оптимально развивать решение

Выводы

Выводы

- Архитектурные задачи надо решать явно

Выводы

- Архитектурные задачи надо решать явно
- Архитектор – не Ivory Tower эксперт

Выводы

- Архитектурные задачи надо решать явно
- Архитектор – не Ivory Tower эксперт
- Архитектор не должен быть узким звеном цепи производства

Выводы

- Архитектурные задачи надо решать явно
- Архитектор – не Ivory Tower эксперт
- Архитектор не должен быть узким звеном цепи производства
- Перечень точек принятия решений, решений и комментариев и ссылки – в приложенных материалах

Шаломович Максим, архитектор, ЛАНИТ

Мои контакты

Полезные материалы

Жду ваших вопросов!

«Типичный заказчик»

- Государство, около-государство, крупный энтерпрайз

«Типичный заказчик»

- Государство, около-государство, крупный энтерпрайз
- Снаружи – регуляторы, нормативы

Далее – рассказ о том, как архитектура существует в таких условиях

Далее – рассказ о том, как архитектура существует в таких условиях

- Мой опыт и опыт моих коллег

Далее – рассказ о том, как архитектура существует в таких условиях

- Мой опыт и опыт моих коллег
- Универсального рецепта нет

Далее – рассказ о том, как архитектура существует в таких условиях

- Мой опыт и опыт моих коллег
- Универсального рецепта нет
- Есть несколько советов Капитана Очевидность

Очень полезное определение

- Набор практик и взаимодействий, которые применяют архитекторы ПО/ системы/ предприятия в гибкой разработке программного обеспечения

Очень полезное определение

- Набор практик и взаимодействий, которые применяют архитекторы ПО/ системы/ предприятия в гибкой разработке программного обеспечения
- (…иногда – неосознанно, неудачно и с непредсказуемым результатом)

Agile Architecture Interactions

- Обзорная статья в IEEE Software magazine от 2011 года

Agile Architecture Interactions

- Обзорная статья в IEEE Software magazine от 2011 года
- Ключевые мысли:
  - Точки взаимодействия (Interaction point)
  - Функции архитектора (Architecture function)
  - Навыки архитектора (Critical skills)

Agile Architecture Interactions

- Обзорная статья в IEEE Software magazine от 2011 года
- Ключевые мысли:
  - **Точки взаимодействия (Interaction point)**
  - Функции архитектора (Architecture function)
  - Навыки архитектора (Critical skills)

Agile Architecture Interactions

- Обзорная статья в IEEE Software magazine от 2011 года
- Ключевые мысли:
  - **Точки взаимодействия (Interaction point)**
  - Функции архитектора (Architecture function) – могут сильно отличаться от «общепринятых»
  - Навыки архитектора (Critical skills)

Agile Architecture Interactions

- Обзорная статья в IEEE Software magazine от 2011 года
- Ключевые мысли:
  - **Точки взаимодействия (Interaction point)**
  - Функции архитектора (Architecture function) – могут сильно отличаться от «общепринятых»
  - Навыки архитектора (Critical skills) - также могут отличаться, плюс – выходят за пределы доклада

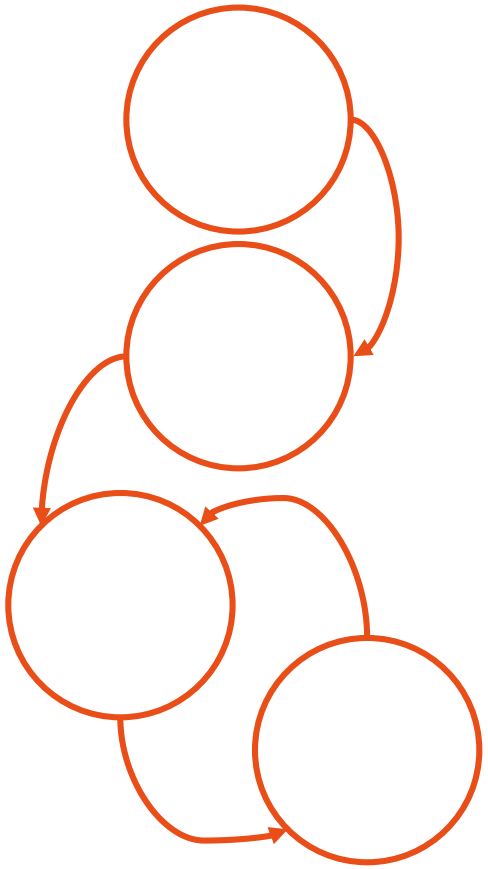
Точки взаимодействия (принятия решений)

Предварительное планирование (Up-front planning)

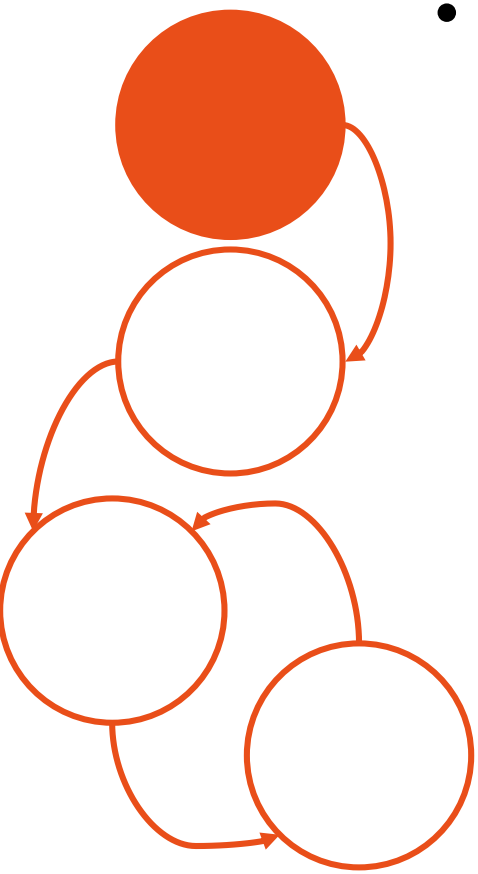
Наполнение «бэклога» (Storyboarding)

Работа в итерации/ спринте

Анализ результатов

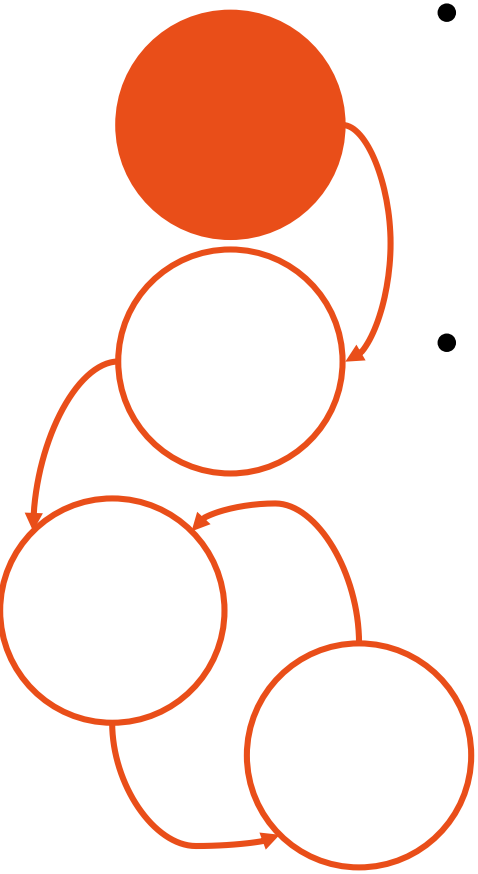


Общий подход



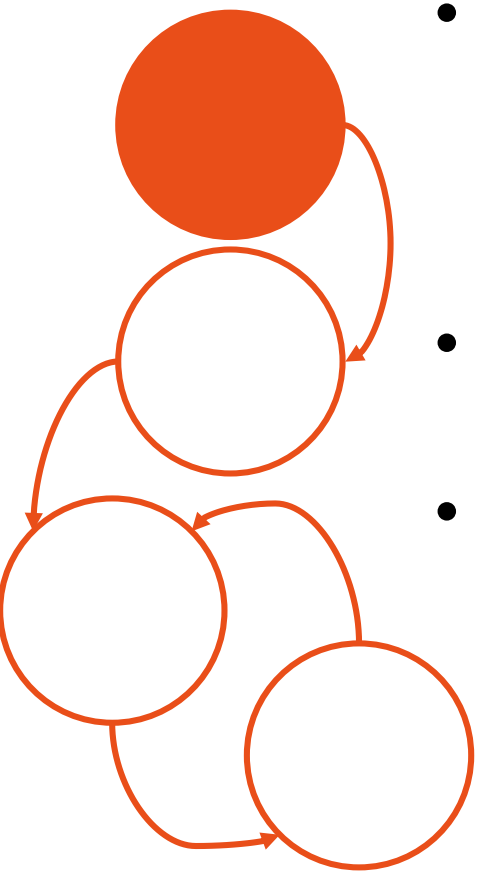
- Гибкая архитектура тоже требует предварительных решений (см. статью Брауна про эволюционную архитектуру)

Общий подход



- Гибкая архитектура тоже требует предварительных решений (см. статью Брауна про эволюционную архитектуру)
- Основной результат – набор опций

Общий подход



- Гибкая архитектура тоже требует предварительных решений (см. статью Брауна про эволюционную архитектуру)
- Основной результат – набор опций
- Журнал для всех решений (см. например ADR)

Технологический стек

- Соответствует корпоративным политикам
- Соответствует возможностям команды
- Не нужен «космос»

Переиспользуемые компоненты системного уровня

- Декомпозиция системы (см. статью Парнаса и мой доклад про декомпозицию)
- Определение применимых паттернов уровня системы

Верхнеуровневое описание архитектуры

- Оценка потребностей «сверху»
- Максимально простые средства коммуникации
- Максимально практическое описание (Model to Code gap, см. статью Брауна про архитектуру и код)

Показатели качества

- Выявление архитектурных требований (см. мой доклад про архитектурные требования)
- Привлечение всех возможных заинтересованных лиц и поиск компромиссов

Определение порядка реализации сценариев

- Критичные бизнес-сценарии (a-la MVP) - вперед
- Проверку архитектурных гипотез и опций – вперед
- Атрибуты качества - вперед

Программная архитектура отдельного компонента или сценария

- Выделенный архитектор компонента – «красиво», но долго. Применяем только для критичных решений

- Для всего остального – набор правил (фреймворк) и поддерживающие инструменты

Контроль актуальности архитектурных решений

- Постоянное взаимодействие с командами
- Явно принимаем технический долг (в идеале – через Заказчика)

Технические практики (правила и инструменты)

- «Monolith first», при условии:
  - Multi-module build
  - Структурированный монолит
  - Управление зависимостями через API
  - Управление общими зависимостями
- Использование абстракций
  - Замена хранилищ данных
  - Замена транспорта
  - Замена взаимодействий