Skillbox

Архитектурные стили

Эволюция архитектуры

Павел Елисеев

Software Architect
Сбер

Skillbox

На этом занятии

- Рассмотрим, как развивалась архитектура ПО.
- Познакомимся с фундаментальными архитектурными стилями.

История

Кто забывает уроки истории, обречён на их повторение.

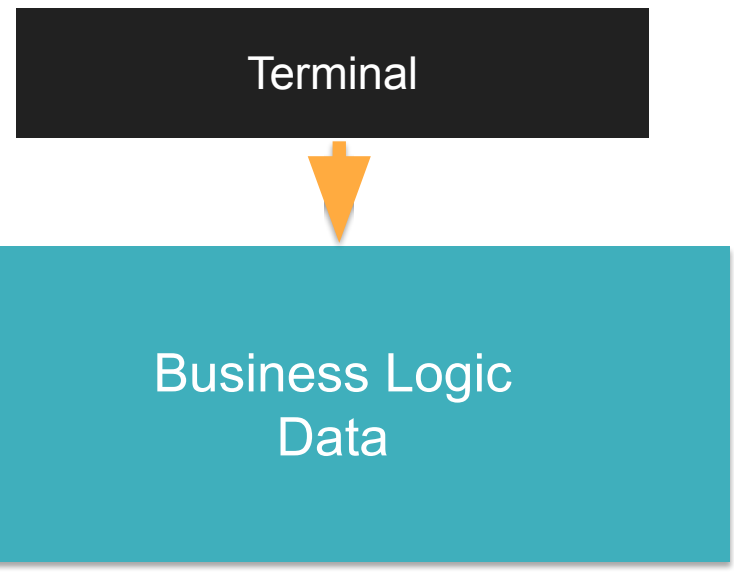
Джордж Сантаяна

Skillbox

История архитектурных стилей. Начало

И был вначале монолит...

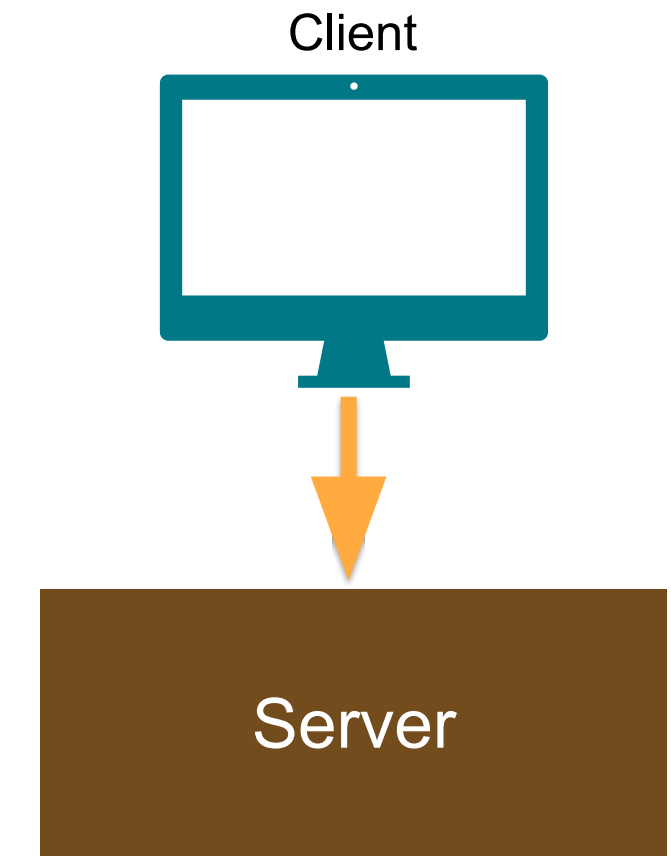
One-tier Architecture



- Отсутствие GUI (используется CLI)
- Очень простые приложения
- Немасштабируемые приложения

1980`s Client-Server

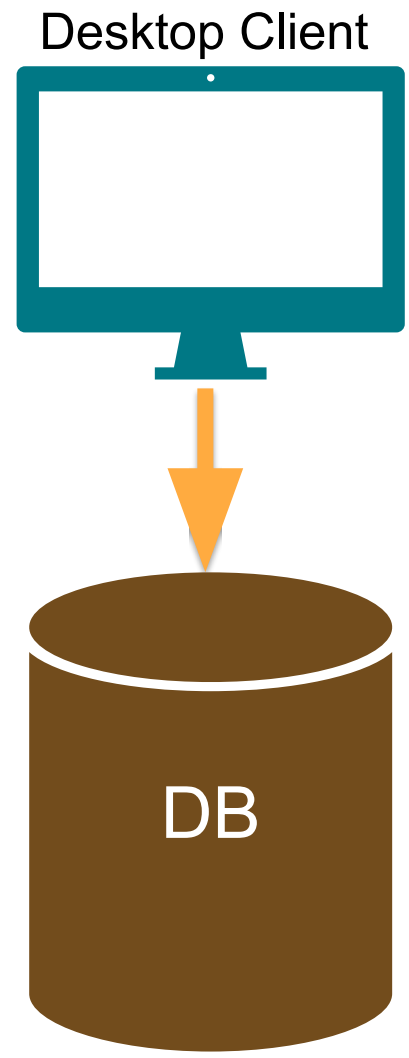
- ООП
- C++
- Erlang
- Perl



Skillbox

Client-Server

- Разделение ответственности
- Проще обновлять



N-tier Architecture

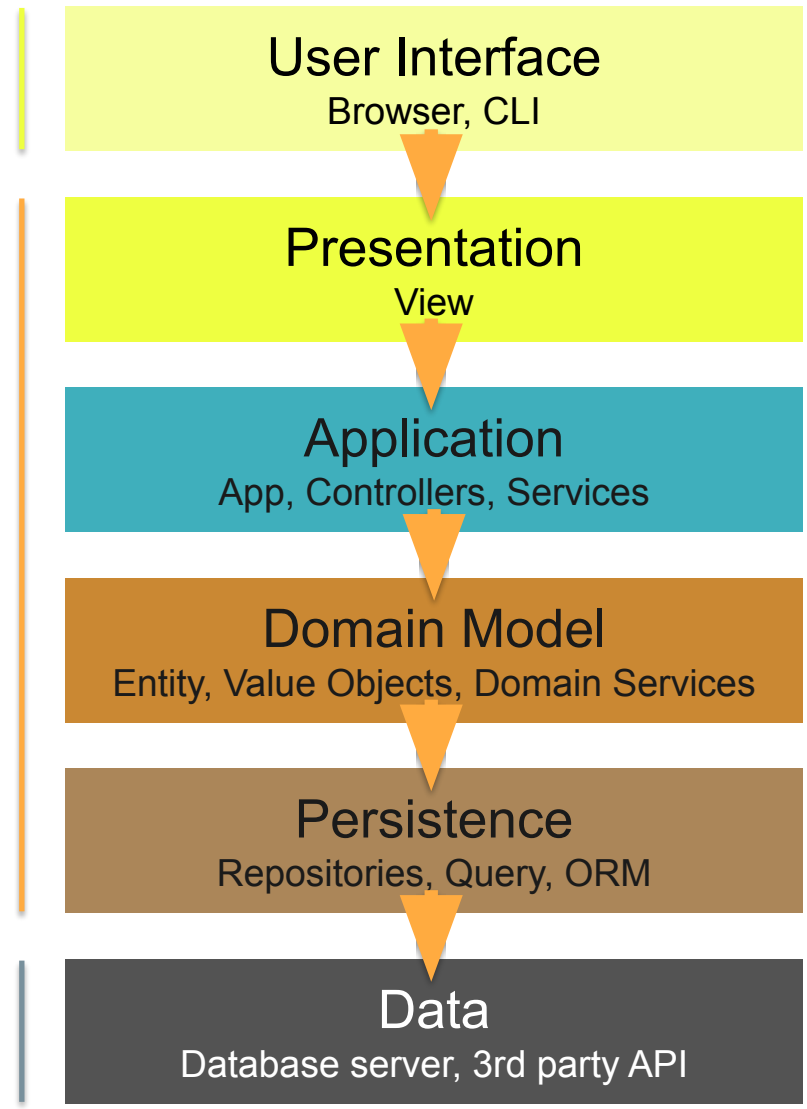
Skillbox

Three (N)-tier Architecture

1st tier — браузер

2nd tier — сервер приложений

3rd tier — сервер базы данных

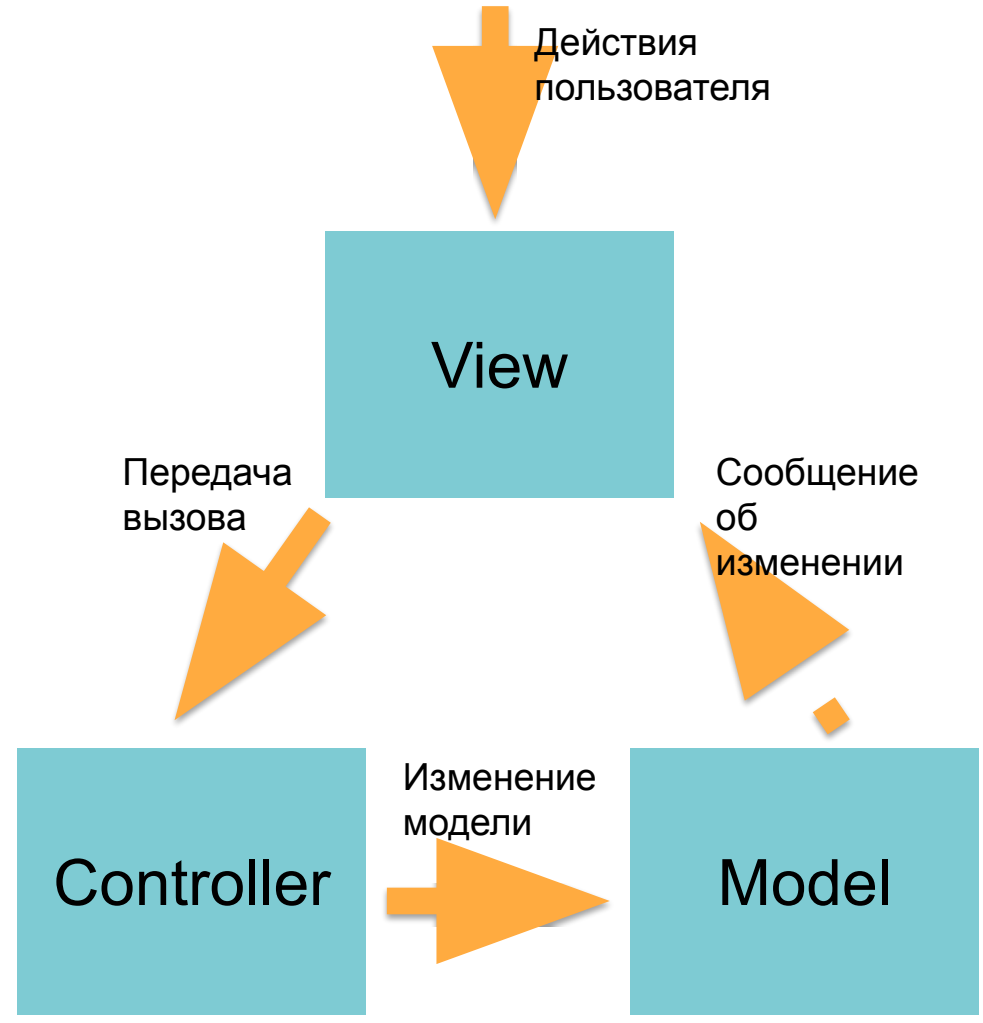


Anti-Pattern: Big Ball of Mud

Skillbox

Model-View-Controller

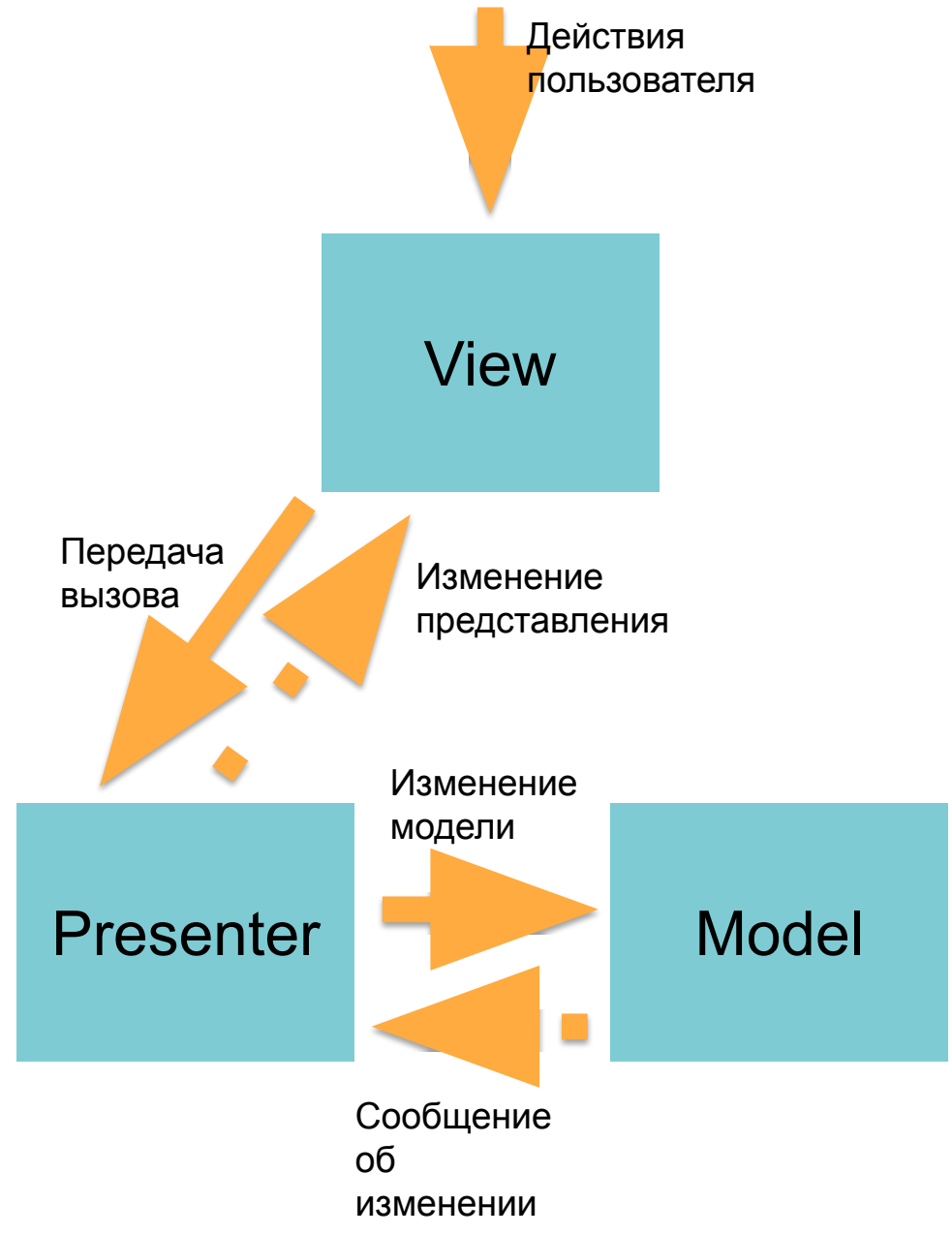
- Структурирует код
- Разделяет ответственность



Skillbox

MVC family

- Model-View-Presenter
- Model-View-ViewModel
- Model-View-Presenter-ViewModel
- …



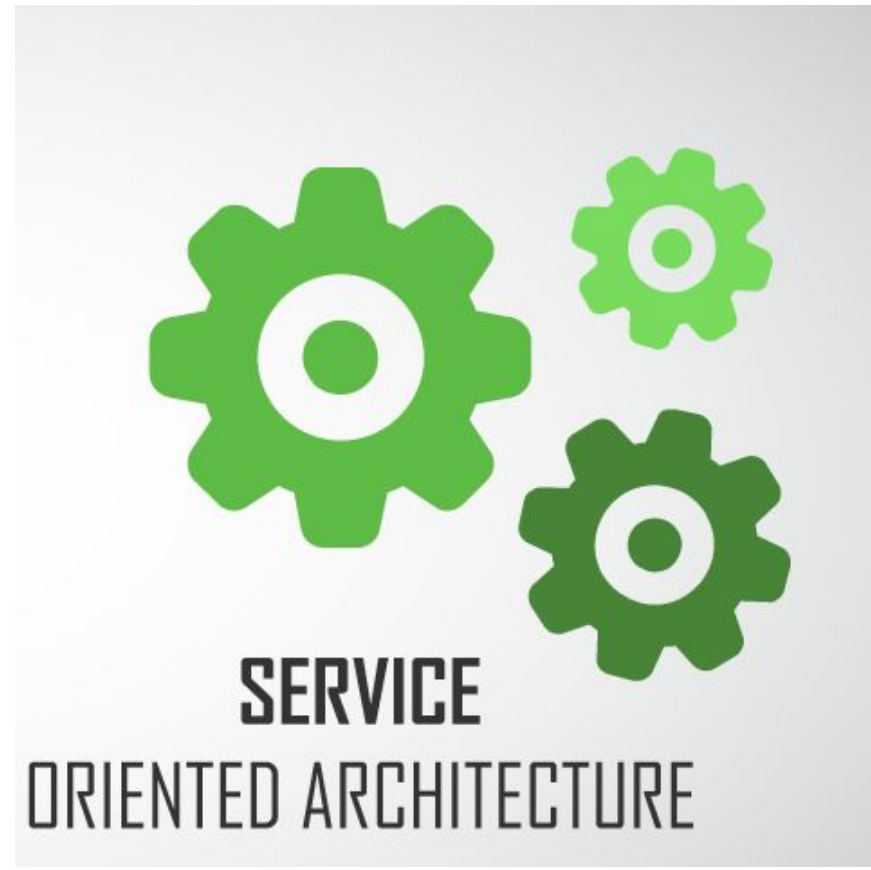
MVC family

- Слабая связанность компонентов
- Разделение ответственности

Skillbox

SOA (Service Oriented Architecture)

- Как сделать систему проще?
- SOA — это набор принципов
- Ключевая концепция — сервисы



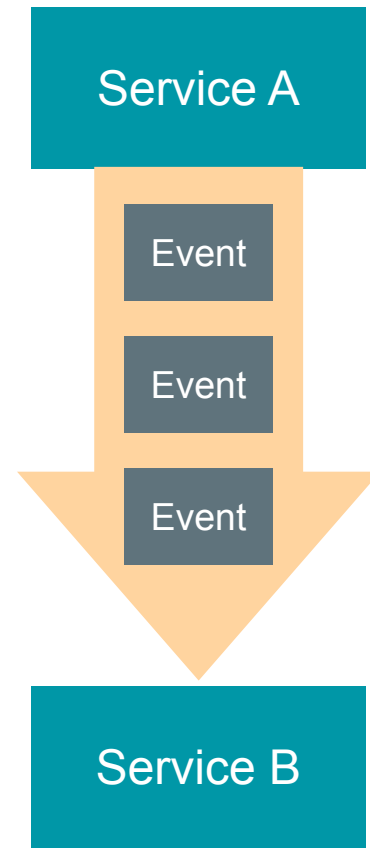
Примеры:

- Web Services
- ESB
- Микросервисы

Skillbox

EDA (Event-Driven Architecture)

- Ключевая сущность — событие
- Более слабая связанность между сервисами и асинхронная интеграция

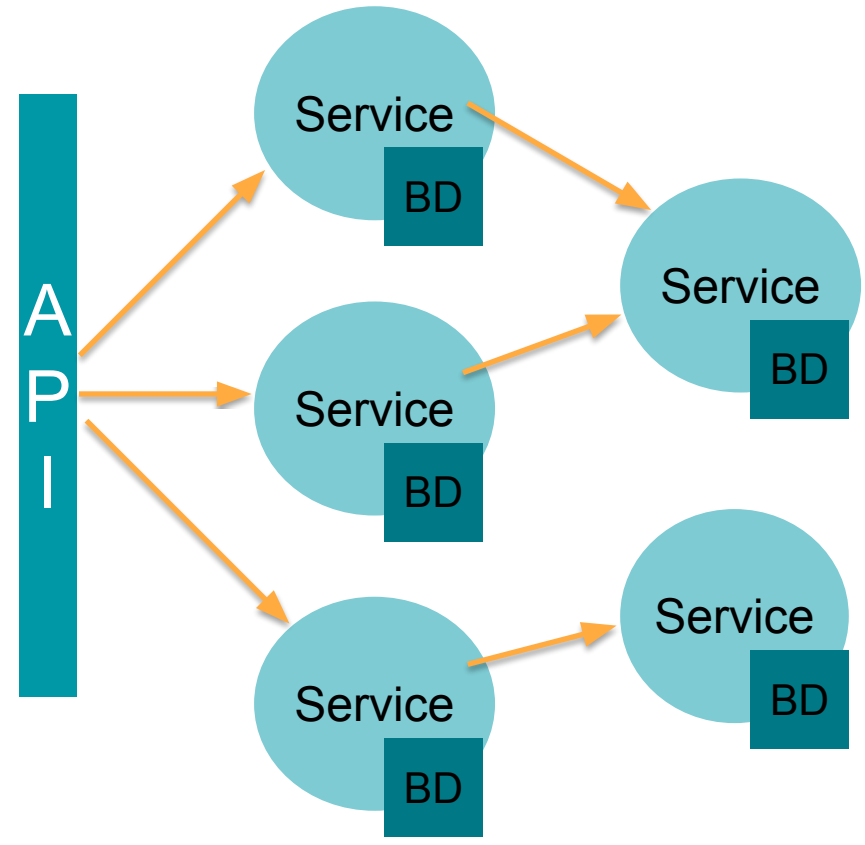


Включает подходы:

- Event Notification
- Event Sourcing

Микросервисы

- Ответ на вопрос: как должен правильно готовить SOA?
- Каждый сервис должен делать что-то одно. И делать это хорошо
- Набор паттернов
- Большой акцент на организационную структуру и инженерные практики



Skillbox

Cloud

- Предоставление ресурсов по подписке (как сервис)

- Облака — это про:
  - IaaS
  - PaaS
  - SaaS

- Стало драйвером для новых направлений архитектуры

- Community и стандарты

Serverless

- Драйверы:
  - Разумная утилизация ресурсов
  - Разделение ответственности
- Ваша задача — это код (вы вообще не думаете о среде исполнения)
- Позволяет достичь максимальной эластичности
- Из минусов:
  - Время старта приложения
  - Vender lock
  - Затрудняет отладку

Вывод

Рассмотрели, как развивалась архитектура.

На следующем занятии

Слабая связанность

Независимость компонентов

Монолиты