

# Лекция 3

## Гибкая методология. Git. UML

### Разработка интернет приложений

Канев Антон Игоревич

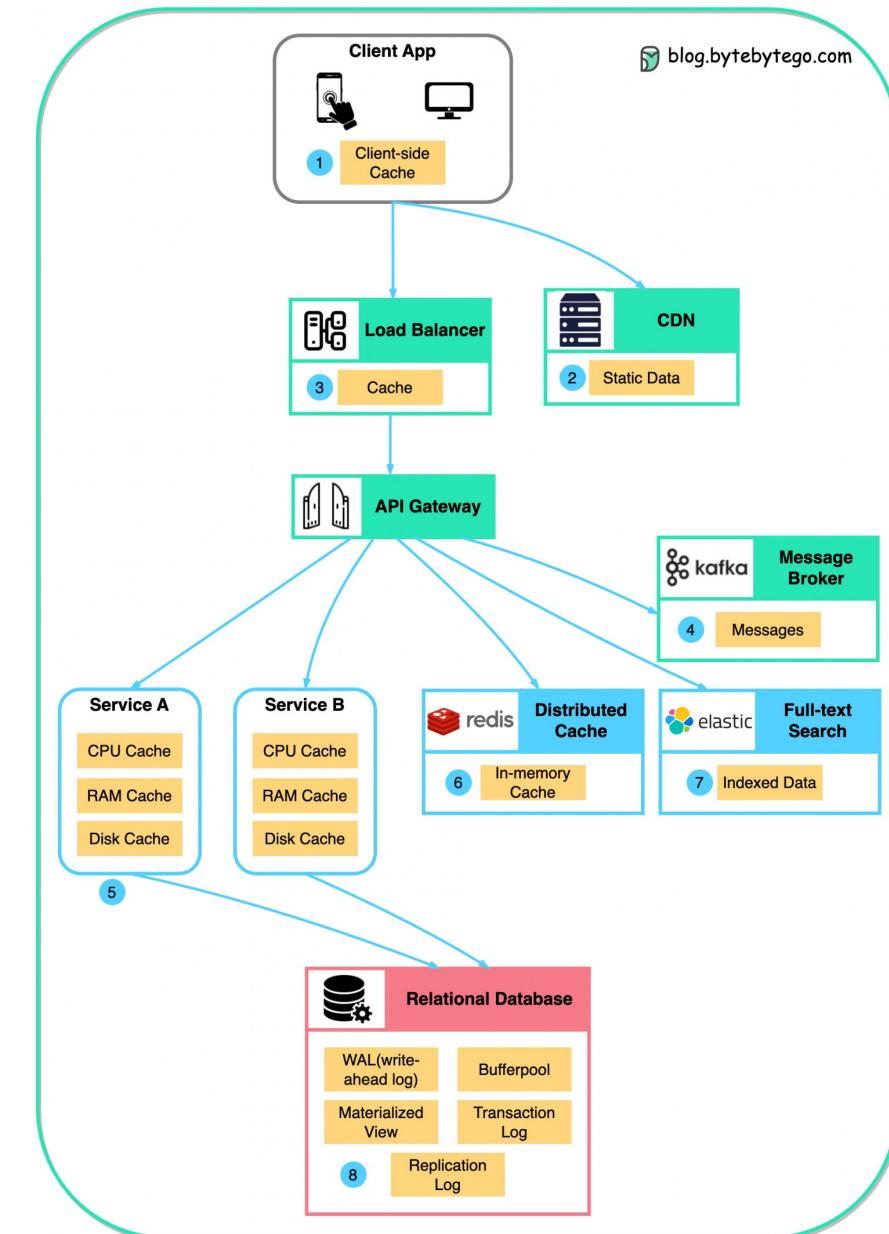
# Дизайн системы

## или зачем нужны диаграммы

- Архитектура больших систем очень сложна
- Очень хорошая шпаргалка приведена по ссылке ниже
- В курсе мы затронем часть аспектов, но чтобы понять их все требуется время и опыт

<https://habr.com/ru/articles/770564/>

Cache Systems Every Developer Should Know



# Зачем нужна документация

- Обучение новых сотрудников
- Снижение рисков при разработке продукта
- Универсальный язык, слабо зависит от языка программирования
- Описание сложных системы и взаимодействия их между собой
- Поддержка продукта и его развертывание
- Low-code разработка

# Роли в команде

- Заказчик
- Директор проекта
- Руководитель проекта
- UI/UX-дизайнер
- Аналитик (бизнес/системный)
- Руководитель команды
- Разработчик
- Тестировщик QA
- DevOps/SRE



# Проект, продукт и процесс

- Проект – временное предприятие, направление на получение результата
  - Процесс – повторяемая последовательность действий, направленная на достижение цели
  - Продукт – программный продукт, программа
  - Minimum viable product (MVP) – для гостя 1 лабораторная
- 
- Цель – конечное желание (повысить продажи)
  - Задачи – совокупность шагов для достижения цели

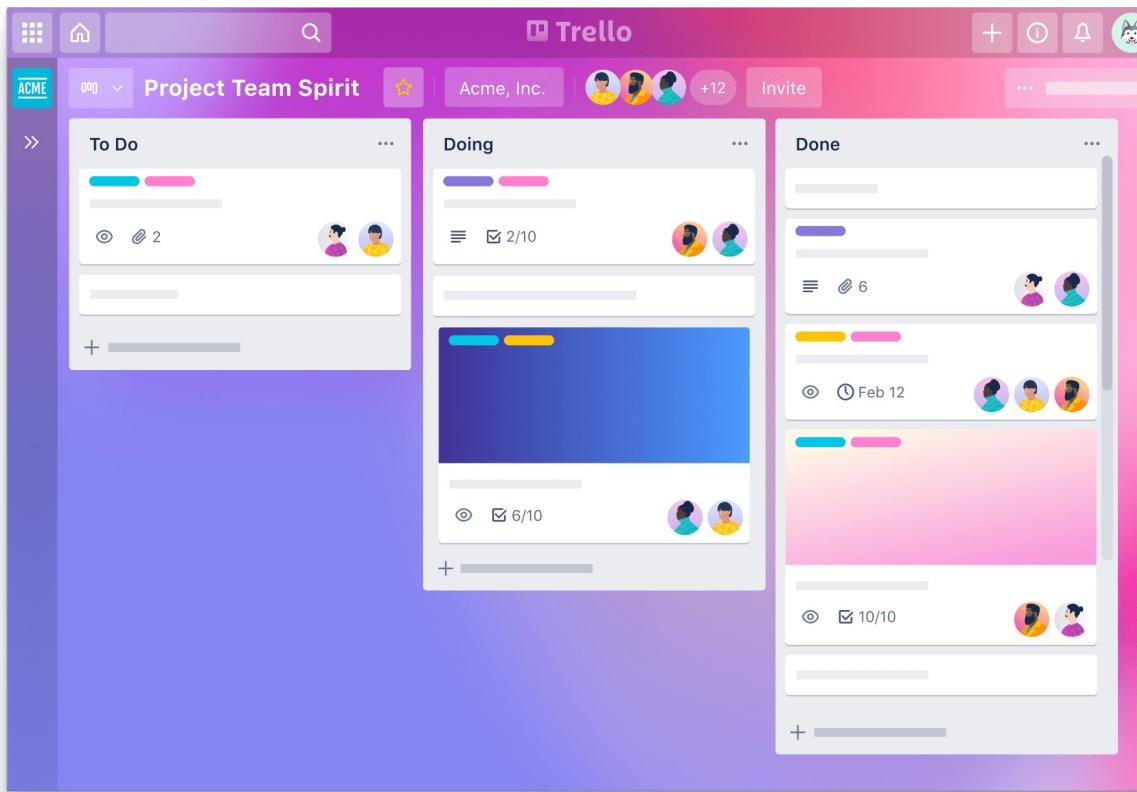
# Менеджер проекта

- Понимание конечной цели - в больших проектах часто все против вас: сотрудники заказчика (бизнес), другие команды разработчиков
- Протоколы совещаний
- Контроль сроков: вовремя утвердить ТЗ, макет figma, вовремя показать MVP и тд
- Дорожная карта
- Kanban-доска



# Kanban

- Способ организации множества задач
- То что нужно сделать описано в user stories (я-хочу-результат)



## Roadmap

By Status ▾

Properties Group by Status Filter Sort Search ... New ▾

Not Started 12 ... + In Progress 5 ... + Complete 5 ... + Hidden

>Login Bugs

Camille Ricketts, Nate Martins

Bug, Sprint 20

Onboarding

Andrea Lim, Nate Martins

Epic

Mobile Start Up Time

Camille Ricketts, David Tibbitts, Andrea Lim

Epic

Rewriting Flow

David Tibbitts, Camille Ricketts, Nate Martins

Task, Sprint 23

Error Codes

Andrea Lim, Cory Etzkorn

Bug, Sprint 21

Landing Page Redesign

Cory Etzkorn, Nate Martins, Andrea Lim

Sharing Permissions

David Tibbitts, Cory Etzkorn

Sprint 23, Sprint 24

Mobile

Ben Lang, Cory Etzkorn

Task, Sprint 23

- Notion, Trello, Taiga
- Jira и тд

# Дорожная карта

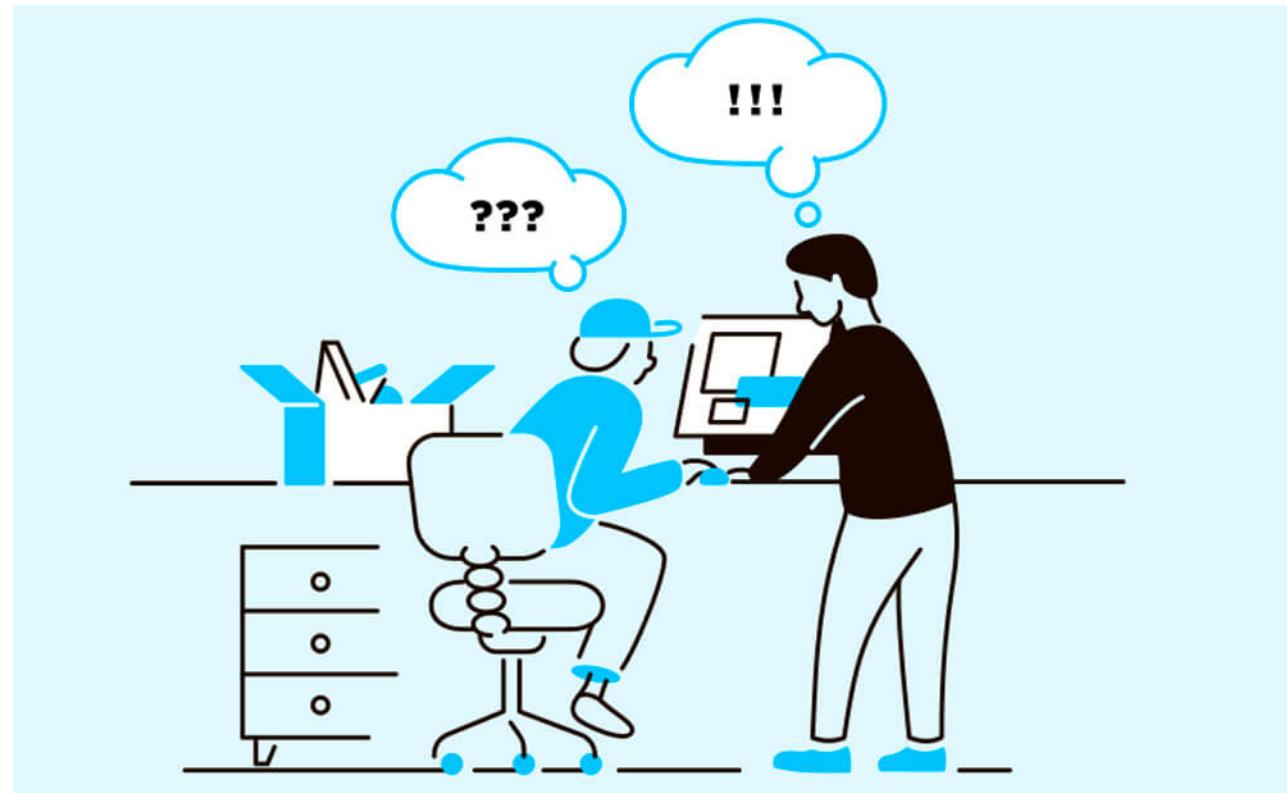
Это более высокий уровень

- Пользовательские истории группируются в эпики – конкретные результаты
- В эпике может участвовать несколько команд
- В конечном итоге нас интересует тема (цель) – оцифровать кампус университета
- Для ее достижения формируются инициативы



# Руководитель команды

- Набор и развитие команды
- Административные вопросы
- Отдельный вид – технический руководитель (архитектура)
- Развитие платформы, архитектура системы



# Разработчик

- Frontend – Java Script, TS, React, Vue.js
- Backend – Java, C#, Go, Python и тд. SQL
- Дата инженер – создание хранилища данных, SQL+ETL
- Дата аналитик – выявление зависимостей в данных, построение графиков и тд
- Data Scientist – специалист по ML моделям



# Тестировщик

- QA – ручное, но чаще автоматическое через Python
- Не путать с unit-тестированием и автотестами, его пишет сам разработчик
- нагрузочное тестирование



# Бизнес аналитик

- ближе к заказчику
  - описание функций системы
  - описание пользователей и их ролей
  - интерфейс системы
- 
- Артефакты
  - Описание бизнес процессов
  - BPMN или Activity
  - Use-case



# Системный аналитик

- ближе к разработчикам
  - описание данных
  - описание алгоритмов
  - архитектура системы
- 
- ER диаграмма и описание БД
  - UML диаграммы: развертывание, последовательности
  - Описание списка запросов: таблицей или swagger



# ТЗ

- Техническое задание описывает все аспекты реализации системы
- Фиксирует функционал системы на понятном заказчику и разработчикам языке
- Включает ряд приложений: описание БД, swagger, figma

## Таблица атрибутов пользователя (surdoapi\_userattrs)

Используется для добавления атрибутов с сохранением стандартной модели пользователя

Столбец	Тип	Модификаторы	Описание
id	BIGINT	PK	id атрибута
user_id	BIGINT	FK	id пользователя
key	VARCHAR		Название атрибута
value	VARCHAR		Значение атрибута
end_date	TIMESTAMP (6)		Дата, до которой действительна запись

# Т3. Описание методов

- Описание методов таблицей. Также описание можно подготовить в виде swagger
- Методы группируются по их домену url: «/user/..», «/lesson/..», «/request/..»
- Выходные данные часто аналогичны GET, чтобы сразу отобразить данные в интерфейсе

Название	Описание	Method	Path	Вход	Выход
	Добавление нового задания в занятие	POST	/lesson/<int:lesson_id>/task	{ id: integer task_text: string, task_audio: string <URL> task_type: string }	[ { id: integer task_text: string, task_audio: string <URL> task_type: string } ]

# Т3. Функциональные требования

- Функциональные требования -
- Описание элементов интерфейса и доступных действий для разных ролей пользователей
- Группировка по страницам интерфейса
- При описании интерфейса либо лучше сослаться на методы, либо в методах сослаться на конкретные пункты функций

– Неавторизованному пользователю доступен только список акций.

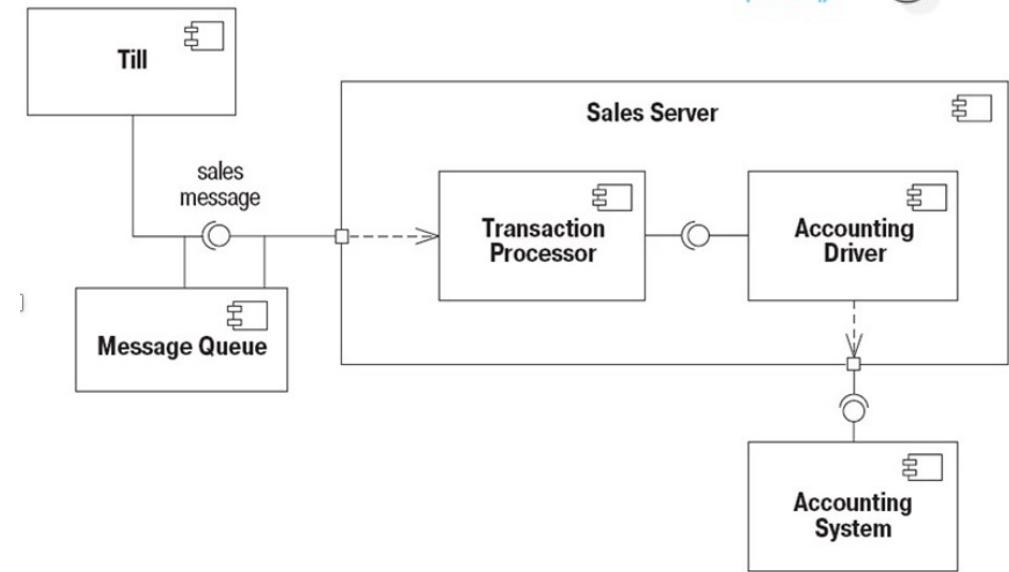
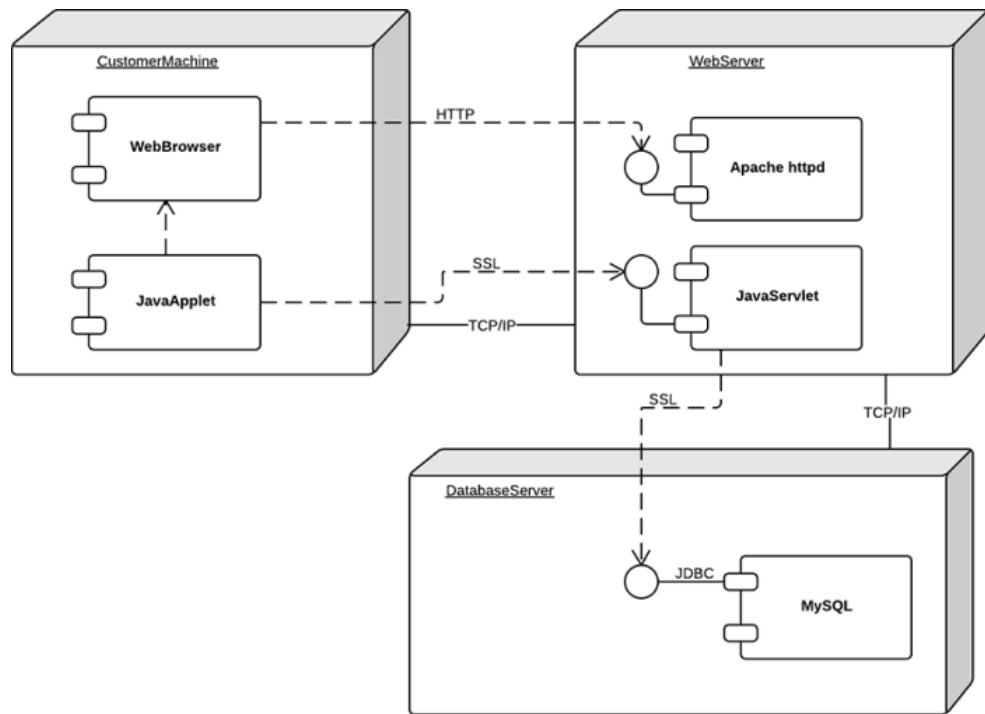
5.1.1.4 Разработка страницы просмотра котировок и графика цены акций конкретной компании:

- На странице пользователь получает данные о компании: текущую цену акции, график и описание компании;
- Авторизованному пользователю также доступны функции покупки и продажи.

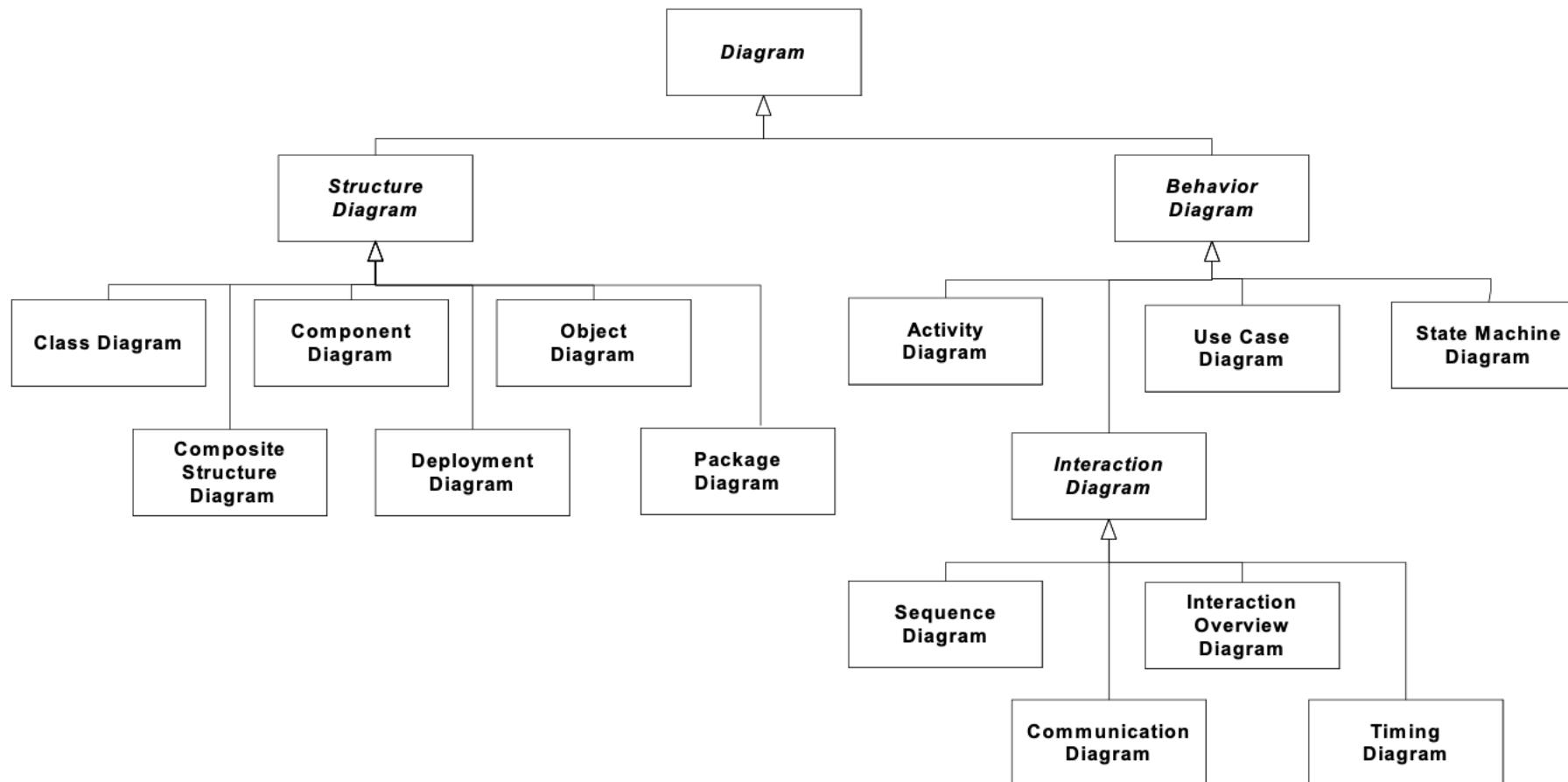
5.1.1.5 Разработка страницы добавления и редактирования компаний:

- Доступ к странице имеют только менеджеры;
- На странице можно добавлять, удалять и редактировать имеющиеся компании.

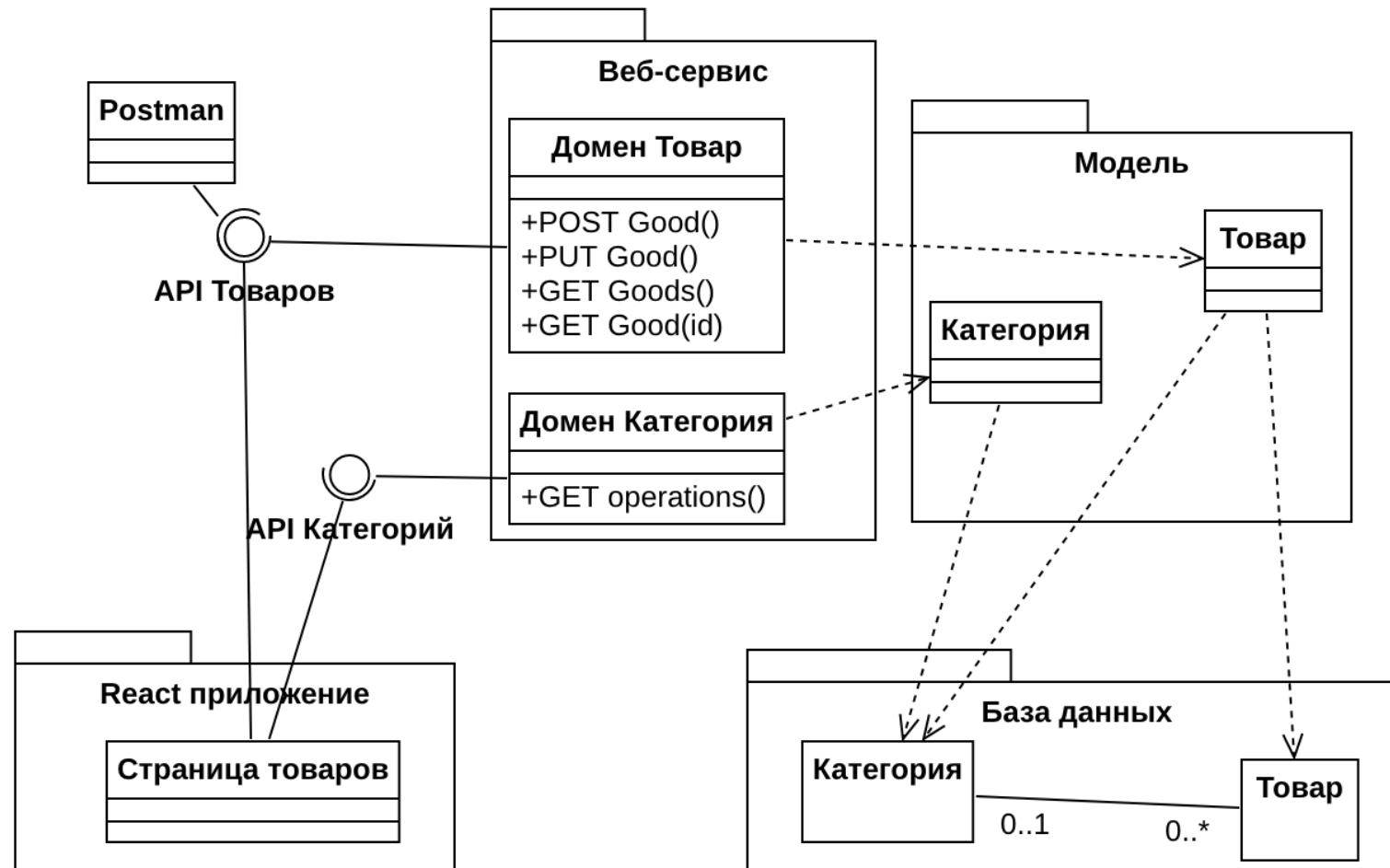
# UML



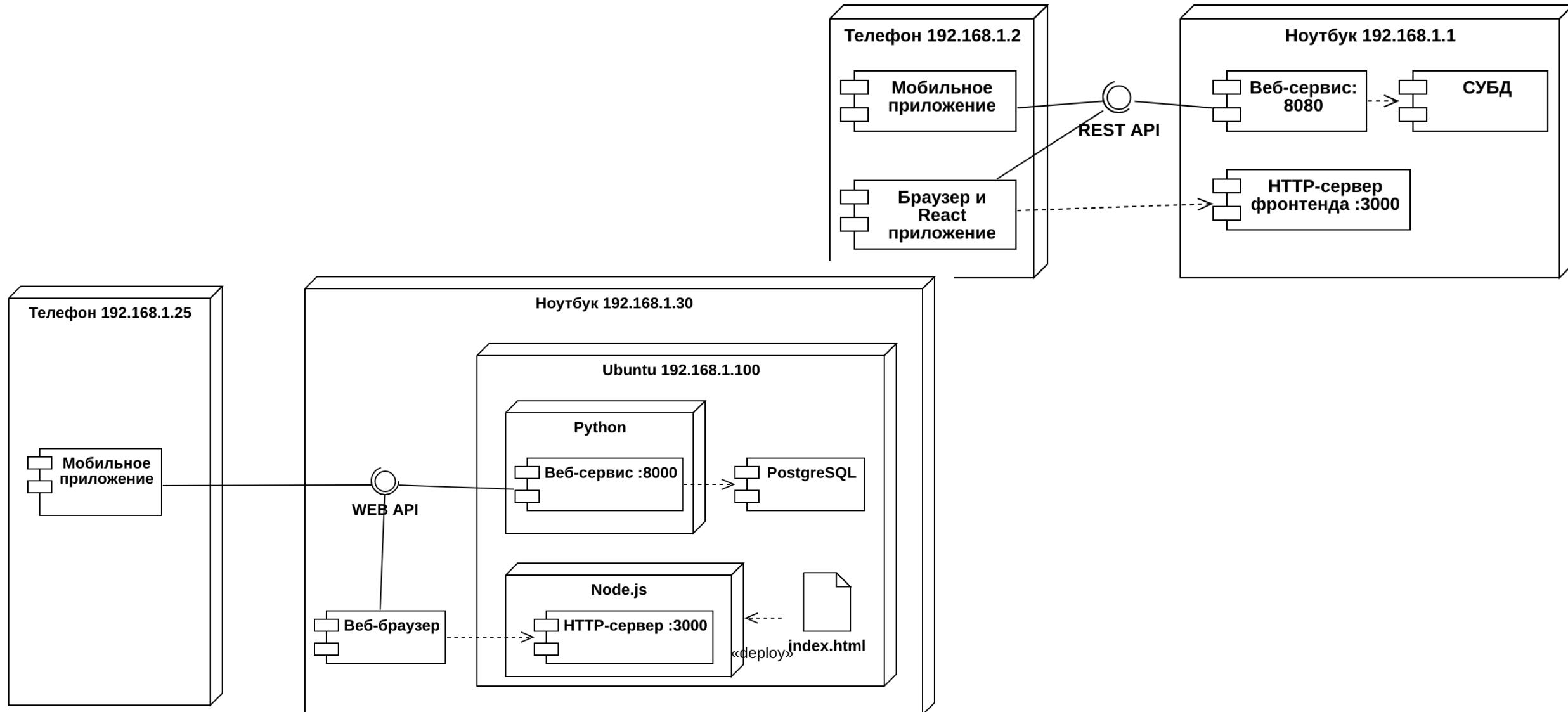
# Виды диаграмм



# Диаграмма классов со структурой системы

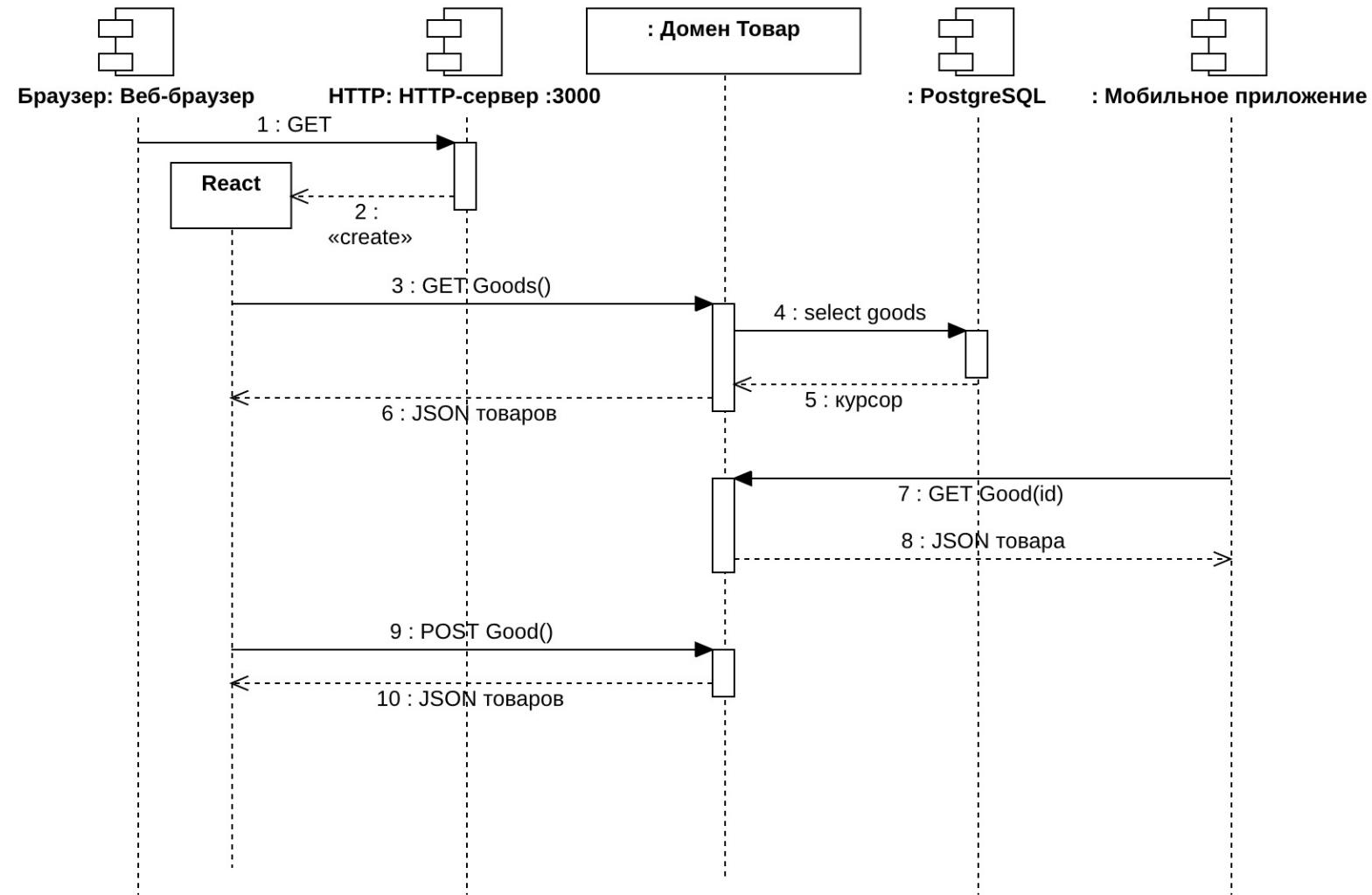


# Диаграмма развертывания



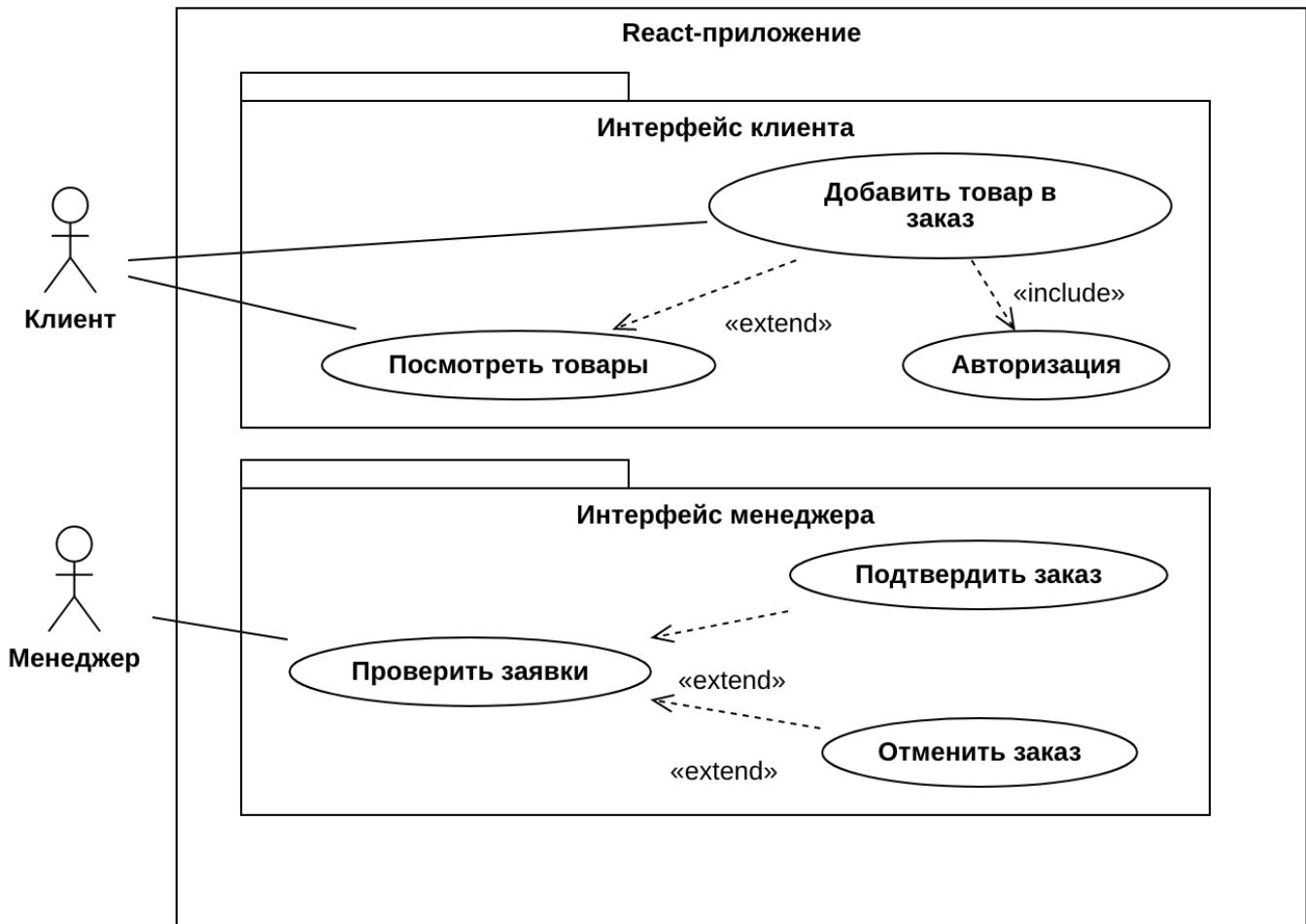
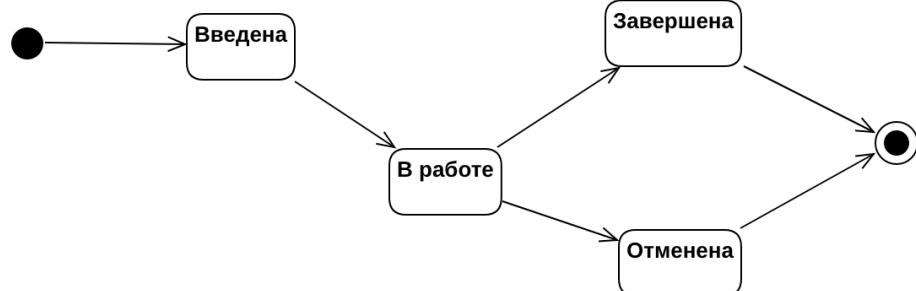
# AJAX запросы на диаграмме Sequence

- На диаграмме последовательности удобно представить совокупность HTTP запросов
- При этом в качестве сообщений можно использовать методы классов



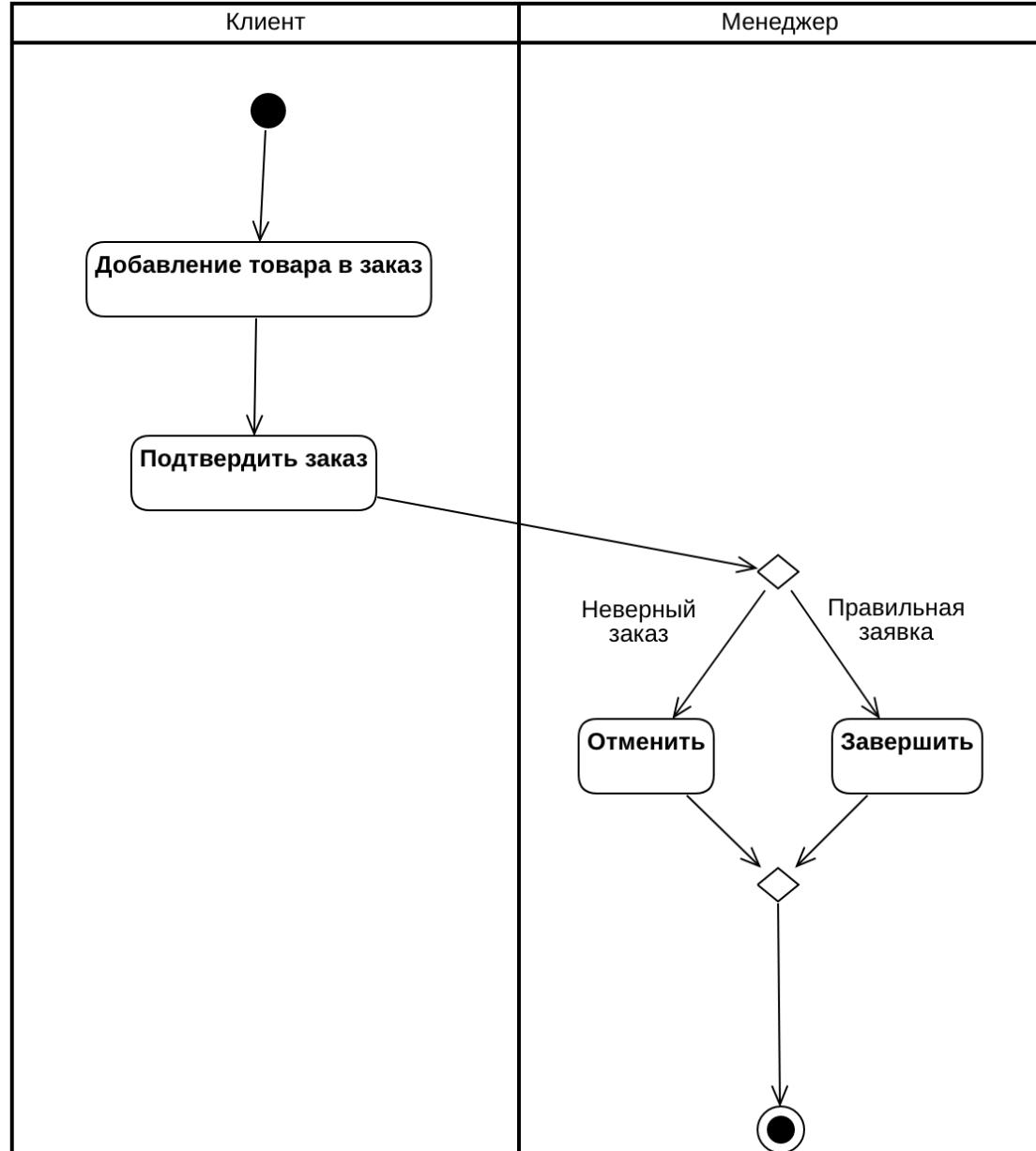
# Прецеденты и состояния

- Диаграмма прецедентов для функций системы
- Состояния для описания статусов отдельных сущностей



# Бизнес процесс

- Лучше показать с помощью BPMN
- Можно использовать более простую диаграмму деятельности



# Качество или скорость

- При фиксации двух параметров получаем третий
- Достичь сразу всех трех не получится



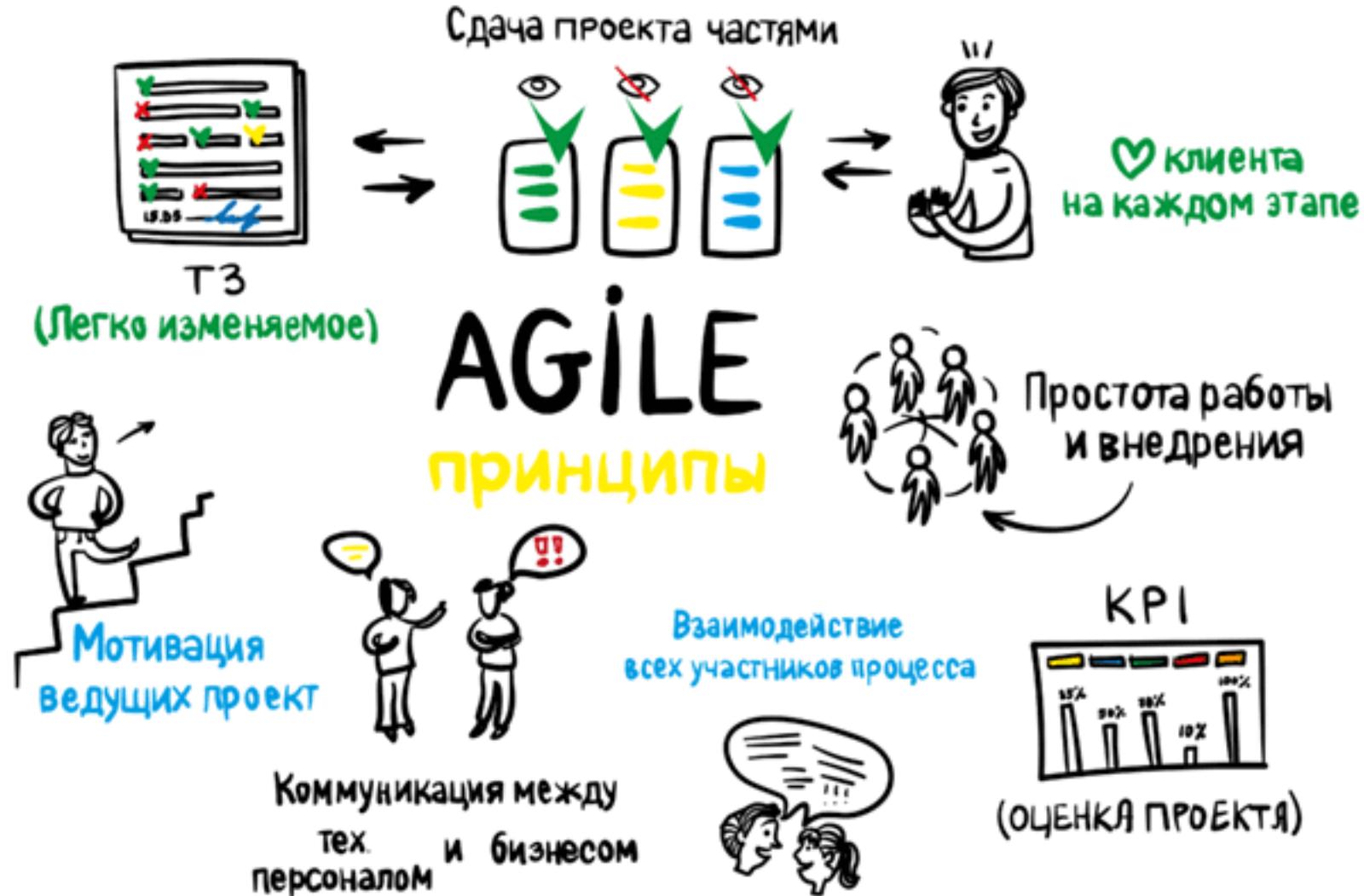
# Agile

- Гибкая методология разработки – альтернатива последовательной водопадной
- Разделение процесса разработки на короткие итерации и повторение



# Agile

- Гибкая методология разработки – альтернатива последовательной водопадной
- Разделение процесса разработки на короткие итерации и повторение



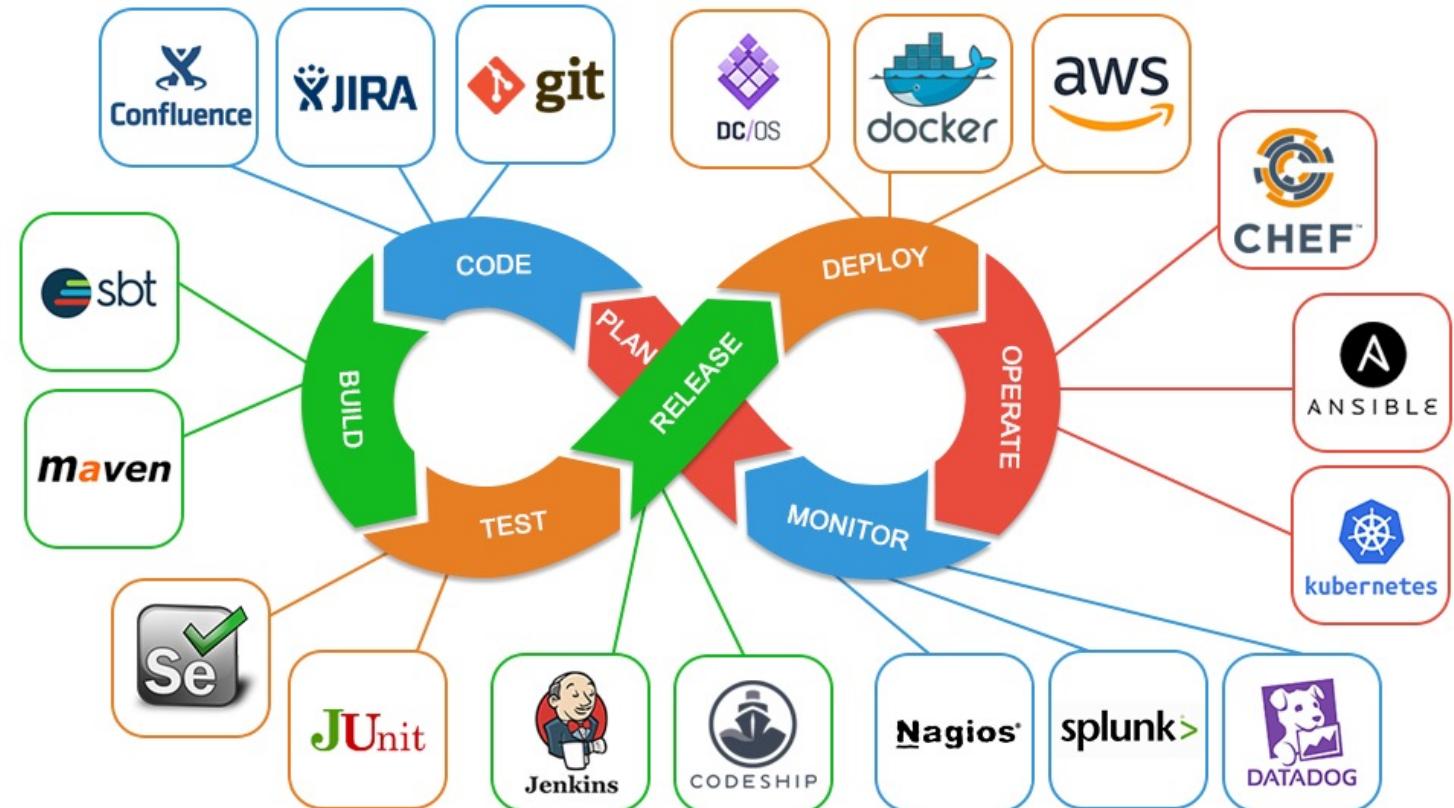
# Scrum

- Скрам-мастер
- Владелец продукта



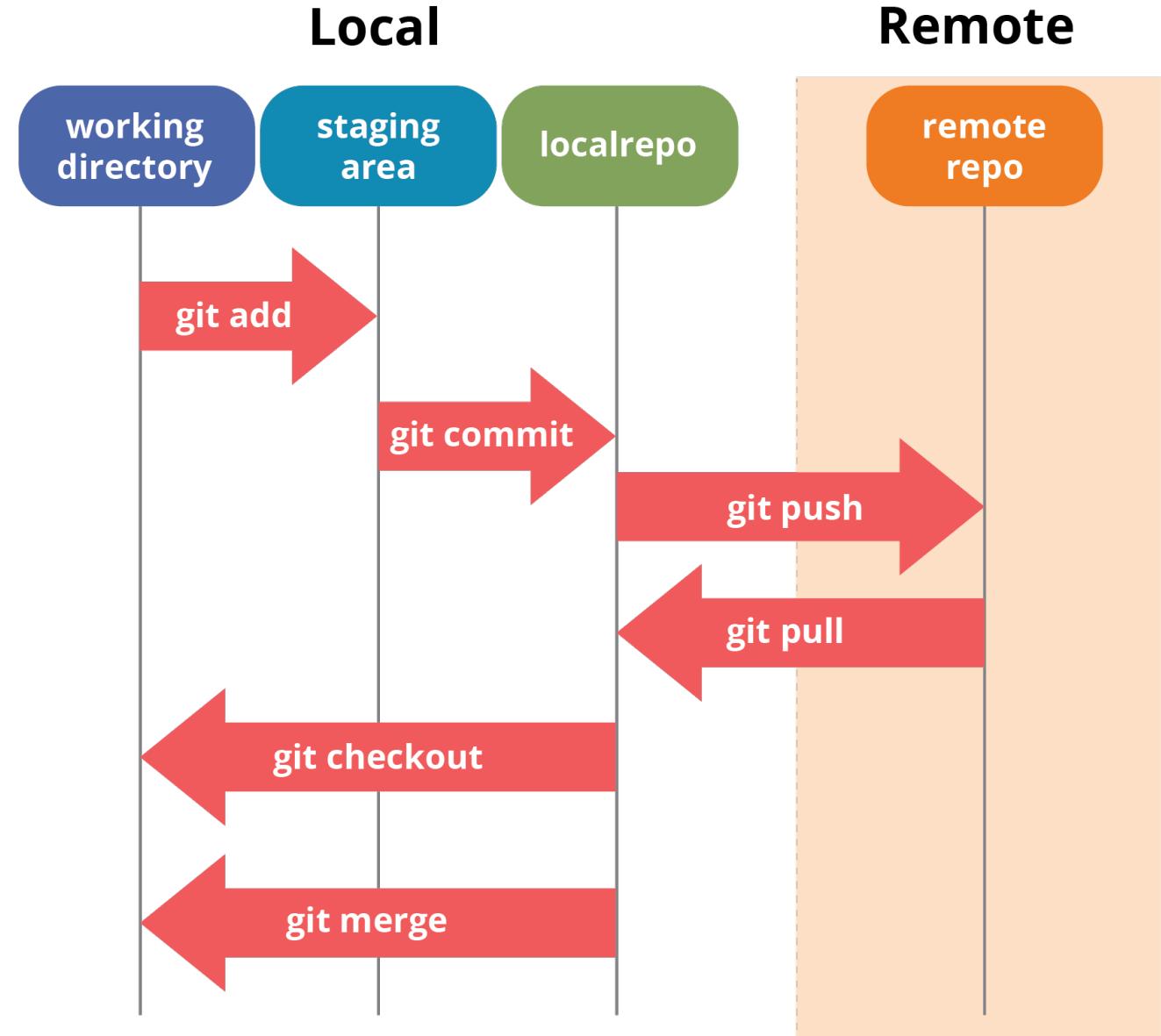
# DevOps – воплощение Agile на практике

- DevOps (development & operations) — методология автоматизации технологических процессов сборки, настройки и развёртывания программного обеспечения
- Быстрый перенос программного обеспечения между разными стадиями жизненного цикла ПО
- Снижение частоты отказов
- Сокращение времени доработок



# Git

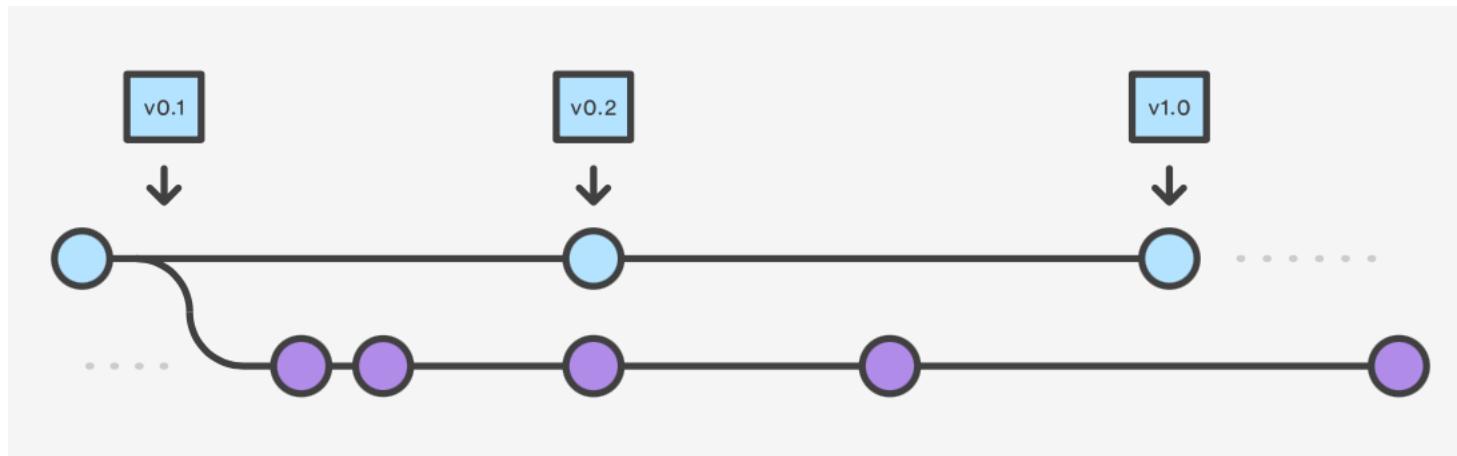
- Git – распределенная система управления версиями
- Позволяет хранить несколько версий одного и того же документа



# Develop

```
git branch develop
```

```
git push -u origin develop
```



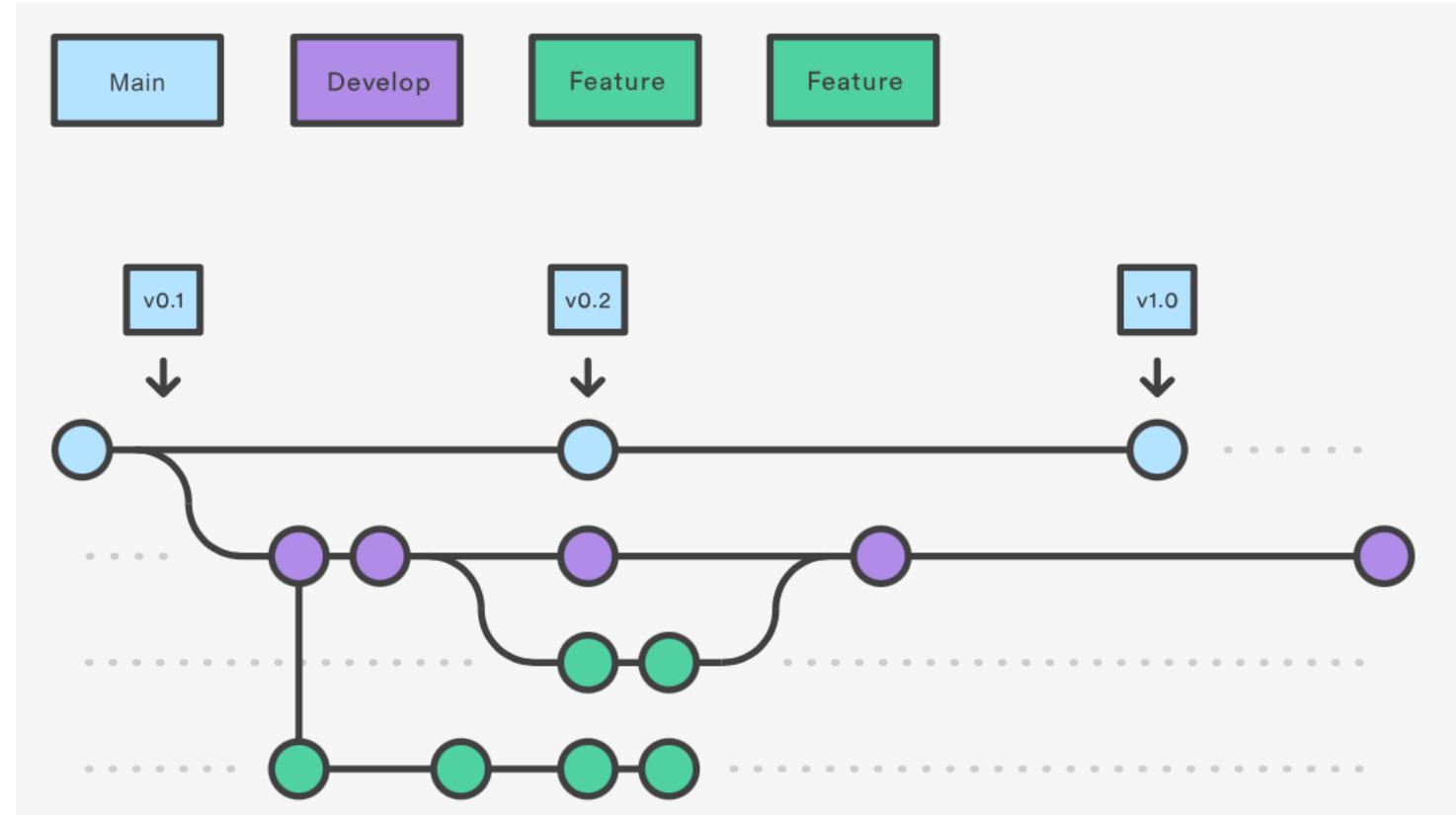
# Feature

```
git checkout develop
```

```
git checkout -b feature_branch
```

```
git checkout develop
```

```
git merge feature_branch
```



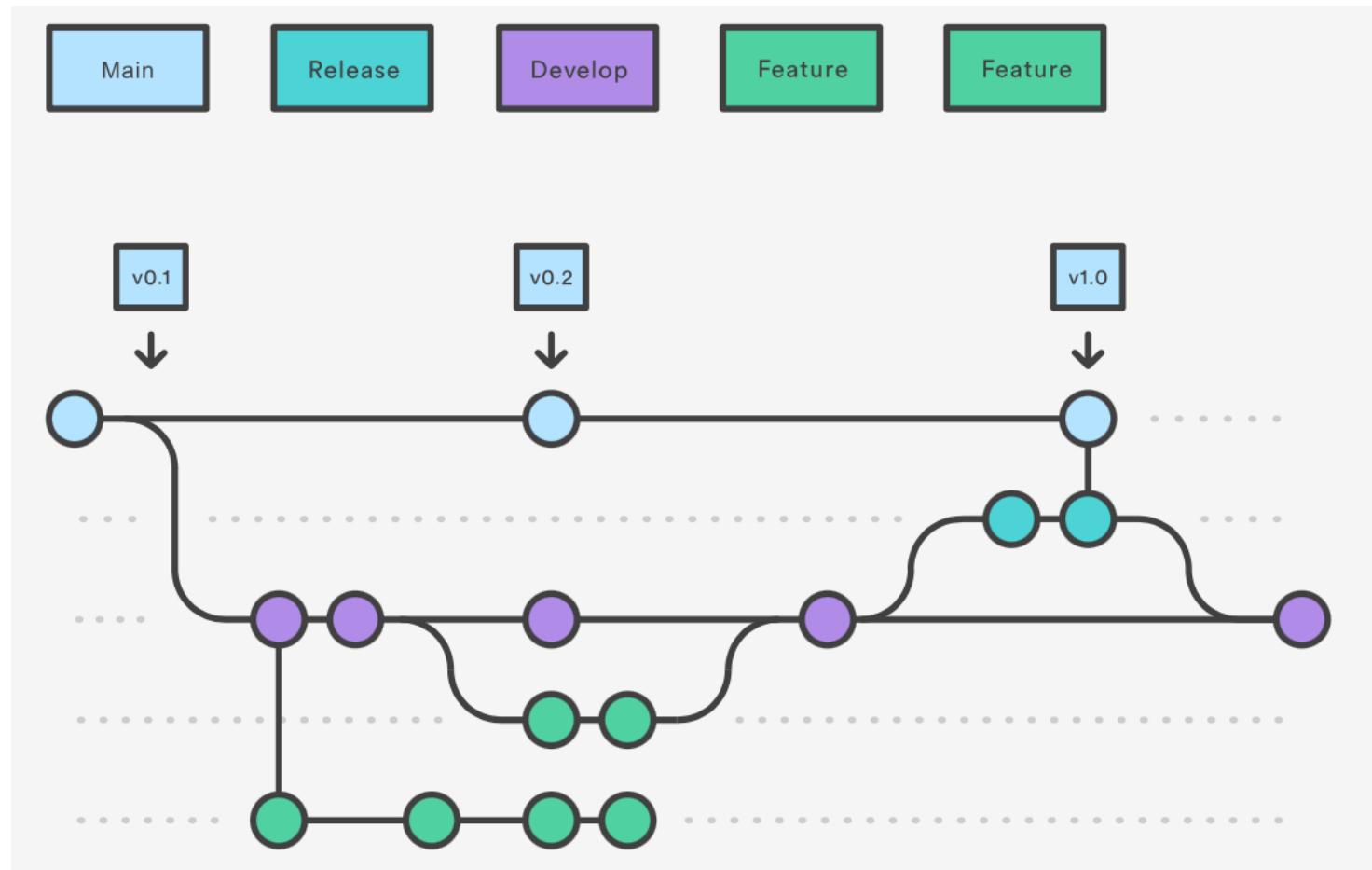
# Release

```
git checkout develop
```

```
git checkout -b release/0.1.0
```

```
git checkout main
```

```
git merge release/0.1.0
```



# Hotfix

```
git checkout main
```

```
git checkout -b hotfix_branch
```

```
git checkout main
```

```
git merge hotfix_branch
```

```
git checkout develop
```

```
git merge hotfix_branch
```

```
git branch -D hotfix_branch
```

