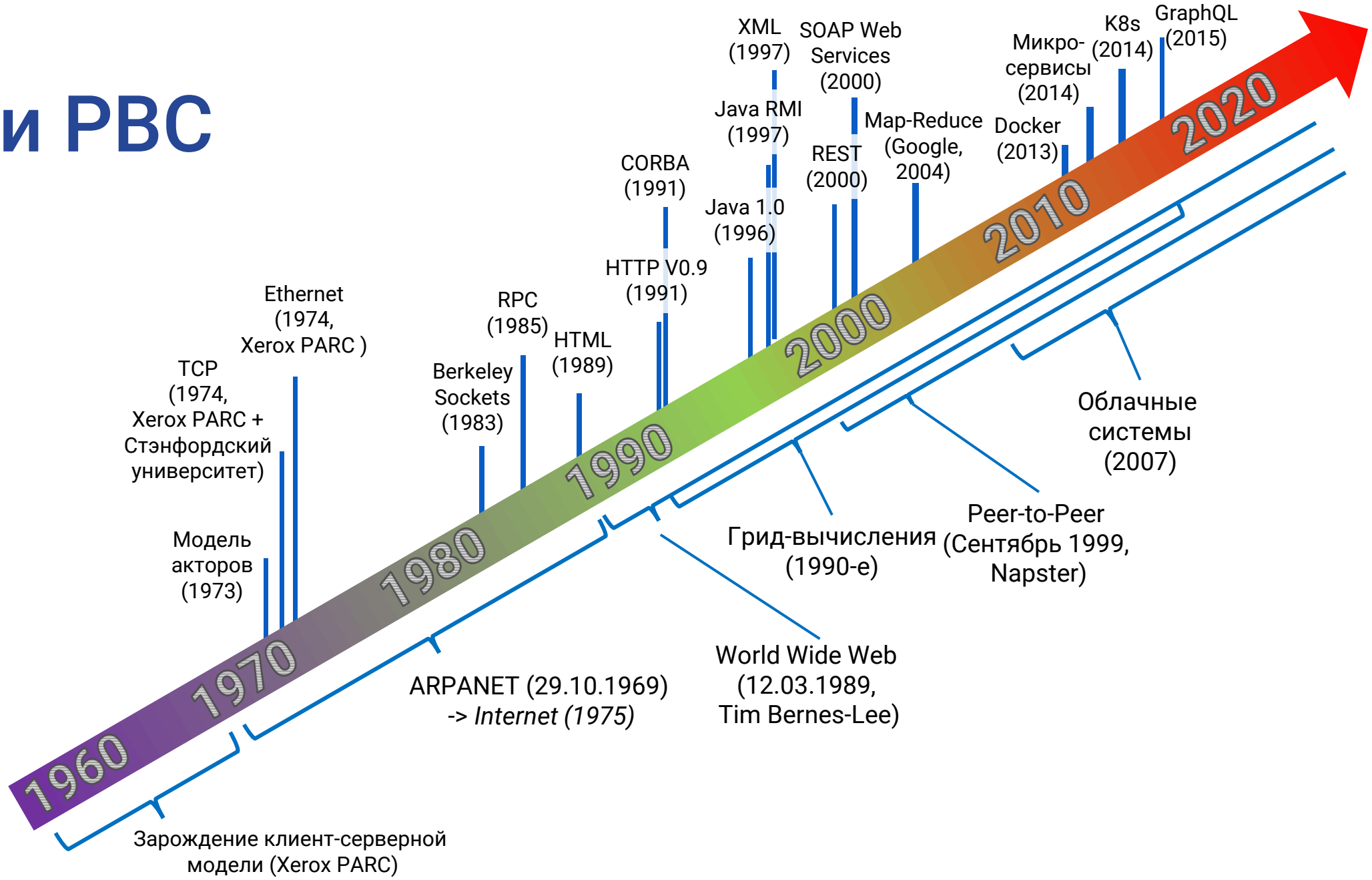
Сервис-ориентированные архитектуры

Презентация курса

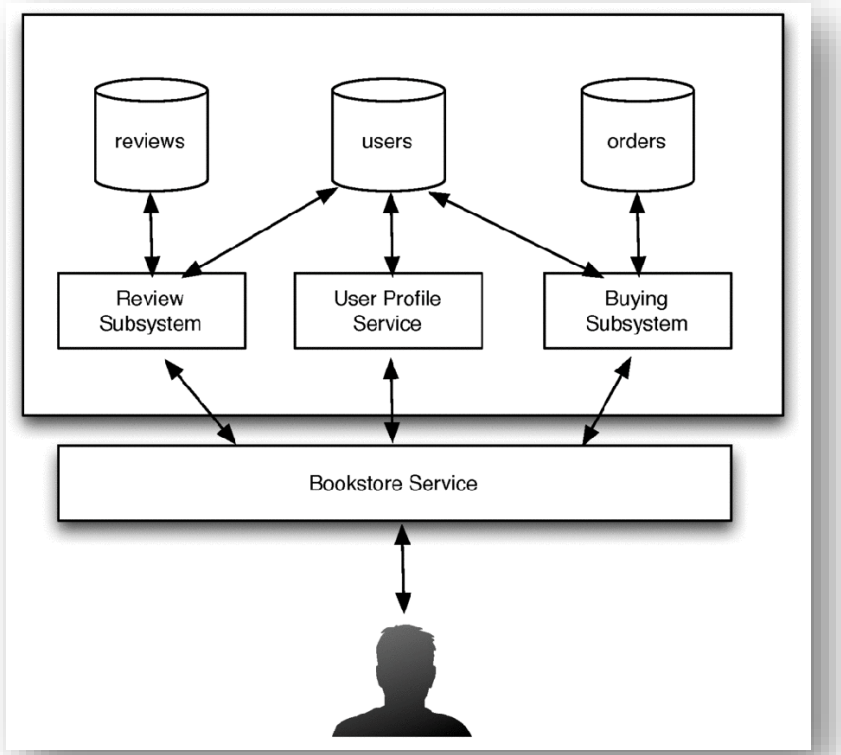
Глеб Игоревич Радченко

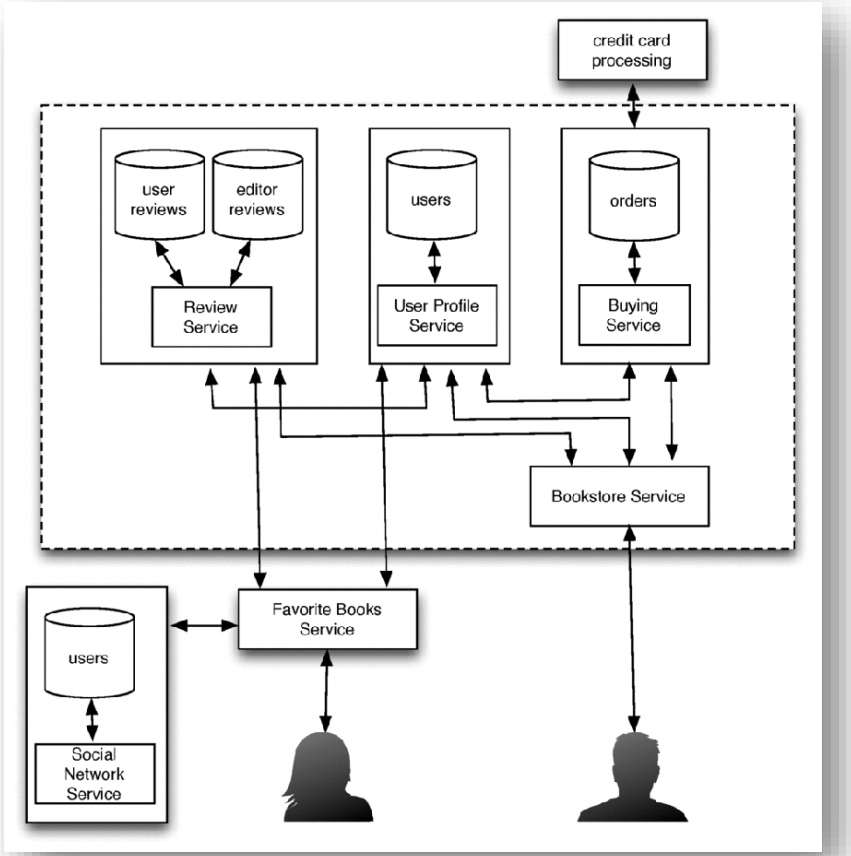
12.01.2021

Вехи РВС



Монолитная VS Сервис-ориентированная архитектура





Цель курса

- Изучить современные подходы к проектированию API сервис-ориентированных распределенных систем
- На практике освоить технологии разработки систем на базе технологий и подходов ZeroMQ, RPC/RMI, gRPC, REST, GraphQL
- Изучить технологии, обеспечивающие авторизацию и безопасность при организации сервис-ориентированных систем

Структура курса

- Основы распределенных вычислительных систем, протоколы передачи данных, форматы сериализации данных;
- Технологии организация связи в распределенных системах: сокеты, RPC/RMI, ZeroMQ.
- Понятие сервис-ориентированной архитектуры (COA). Типы COA API. Общие принципы организации COA.
- RPC API на примере JSON RPC, gRPC, SOAP XML Веб-сервисов
- API Ресурсов на примере REST
- API Сообщений и очереди сообщений.
- Графовый API на примере GraphQL
- Обеспечение безопасности COA: технологии OAuth, JSON Web Token.
- Обзор технологий облачных вычислительных систем. Концепция микросервисов как развитие COA. Технология Docker.

Лектор

- Глеб Игоревич Радченко (к.ф.-м.н., доц.)
- Директор Высшей школы электроники и компьютерных наук ЮУрГУ (г. Челябинск)
- Заведующий кафедрой Электронных вычислительных машин
- Занимаюсь исследованиями в области распределенных вычислительных систем, облачных и туманных вычислений

Семинарист

- Дмитрий Игоревич Савченко (PhD, LUT University, Финляндия)
- Архитектор программного обеспечения в компании Orkestr.io
- Занимается практикой внедрения и управления микросервисными системами

Практические/домашние задания

В рамках курса вам будет выдано 8 заданий, которые частично будут разобраны на практике, частично вам необходимо будет выполнить дома. В каждом задании вы изучите и примените одну из технологий организации распределенных и/или сервис-ориентированных систем для создания прототипа приложения:

1. Основы технологии Docker (без него сегодня ни куда)
2. Сравнение методов сериализации (сравниваем эффективность)
3. Реализация приложения на технологии TCP/UDP сокетов
4. Реализация приложения на ZeroMQ
5. Прототип приложения на RPC (gRPC или JSON RPC)
6. Основы REST
7. Расширение REST – асинхронная обработка запросов на базе очереди
8. Прототип приложения на GraphQL

Платформы и языки для разработки

- Java – EJB, Spring, …
- Python – Django, …
- C# - WCF, …
- Ruby – Rails, …
- Go
- JavaScript - Node JS, …
- C++

Расписание занятий по курсу

- Курс включает в себя 8 занятий (Лекция + Практика)
- Занятия запланированы по субботам, с 9:30 до 12:30
- Начало занятий: 23 января

Система оценивания

- Экзамена не предусмотрено
- Итоговая оценка вычисляется по формуле:

$$Total = 0.7 * Practice + 0.3 * Test$$

где

- $Practice$ – усредненная оценка за выполненные и сданные практические задания
- $Test$ – усредненная оценка за тесты по лекционным материалам, выполняемые на занятиях

Спасибо за внимание!

Готов ответить на ваши вопросы!