ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Факультет компьютерных наук

Департамент программной инженерии

|  |  |
| --- | --- |
|  | УТВЕРЖДАЮ  Академический руководитель образовательной программы «Программная инженерия»,  профессор департамента программной инженерии, канд. техн. наук  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.В. Шилов  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2019 г. |

Выпускная квалификационная работа

на тему «Сервер для мокирования асинхронных протоколов»

по направлению подготовки 09.03.04 «Программная инженерия»

|  |  |
| --- | --- |
| Научный руководитель  профессор  департамента программной инженерии ФКН НИУ ВШЭ  кандидат технических наук  Гринкруг Е.М.  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Подпись, Дата | Выполнил  студент группы БПИ152  4 курса бакалавриата  образовательной программы  «Программная инженерия»  Данилин П.И.  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Подпись, Дата |

Москва 2019

# Реферат

Использование мок серверов при разработке приложений дает разработчику определенные возможности. Среди них: разработка прототип программы без настоящего сервера и воспроизведение поведения клиентского приложения в зависимости от заданного поведения мок сервера. На данный момент существуют решения для мокирования HTTP сервисов. Такие решения получают от пользователя конфигурацию поведения сервера, обычно это JSON, и на заданном порту отдают ответы на HTTP запросы согласно описанной конфигурации. Однако текущие решения для мокирования асинхронных протоколов WebSocket и Server-Sent Events не поддерживают их асинхронную природу. В результате данной работы был разработан сервер для мокирования протоколов WebSocket и Server-Sent Events.

Работа содержит \* страниц, 3 главы, \* рисунков, \* источников, 4 приложения.

**Ключевые слова:** сервер, мокирование, асинхронный протокол, WebSocket, Server-Sent Events.

# Abstract

Using a mock server in development gives a developer a certain amount of possibilities. Among them are prototyping without a real server or testing a client application depending on a given mock server behavior. There are implemented solutions for HTTP services mocking currently. Such solutions receive a behavior configuration from a user, which is usually a JSON configuration, and give out responses to HTTP requests according to the given configuration. But contemporary solutions for mocking asynchronous protocols like WebSocket and Server-Sent Events do not support their asynchronous nature. In the course of this work we developed a server for mocking protocols WebSocket and Server-Sent Events.

The paper contains \* pages, 3 chapters, \* images, \* references, \* appendices.

**Keywords:** server, mocking, stubbing, asynchronous protocol, WebSocket, Server-Sent Events.

**Содержание**

[Реферат 2](#_Toc5914811)

[Abstract 3](#_Toc5914812)

[Определения 5](#_Toc5914813)

[Введение 6](#_Toc5914814)

[Актуальность работы 6](#_Toc5914815)

[Цель и задачи работы 7](#_Toc5914816)

[Структура выпускной квалификационной работы 7](#_Toc5914817)

[Глава 1. 8](#_Toc5914818)

[Глава 2. 9](#_Toc5914819)

[Глава 3. 10](#_Toc5914820)

[Заключение 11](#_Toc5914821)

[Список источников 12](#_Toc5914822)

# Определения

**Мокирование** –

**Сервер** –

**WebSocket** –

**Server-Sent Events** –

# Введение

## Актуальность работы

На текущий момент при разработке программных систем часто используются клиент-серверный и микросервисный подходы к архитектуре таких систем. В упомянутых подходах есть сервисы-потребители и сервисы-провайдеры. Сервис-потребитель обращается к сервису-провайдеру по сети для получения данных или выполнения какого-то действия. В подобных условиях разработчикам сервиса-потребителя может быть полезным использовать мок сервер в качестве сервиса-провайдера. Особенно уместным использование сервера для мокирования станет в следующих сценариях. Например, сервис-провайдер может еще не быть разработан или не поддерживать работу с данным сервисом-потребителем. Тогда разработчики могут договориться и сконфигурировать мок сервер, чтобы он отвечал на запросы сервиса-клиента, как будто он и есть сервис-провайдер. Разработчики смогут дорабатывать свои приложения без ожидания доработок других сервисов. Также с помощью сервера для мокирования можно сильно ускорить создание прототипа приложения, поскольку часть доработок будет идти параллельно, а часть – можно отбросить и заменить на конфигурацию мок сервера. Помимо уже упомянутых ???, мок сервер может использоваться при тестировании сервиса-потребителя. Тогда не потребуется создание условий (таких как создание пользователя с нужными ролями) для сервиса-провайдера, чтобы получить от него нужный ответ. Достаточно правильно сконфигурировать мок сервер, что сделать потенциально проще и быстрее.

Так как одним из наиболее распространенных протоколов взаимодействия сервисов является HTTP, есть несколько популярных реализаций мок серверов для данного протокола. Среди них Mountebank [1], Mock Server [2]. Данные реализации дают возможность отвечать на заданных портах согласно описанной конфигурации. Однако упомянутые решения работают только с протоколами HTTP(S) и TCP. Существует еще семейство асинхронных протоколов, для которых есть потребность в подобном мок сервере. В результате поиска подобного решения для протоколов WebSocket и Server-Sent Events был сделан вывод, что текущие решения обладают сильными недостатками, которые необходимо преодолеть для удовлетворения потребностей пользователей.

## Цель и задачи работы

## Структура выпускной квалификационной работы

# Глава 1.

# Глава 2.

# Глава 3.

# Заключение

# Список источников