

# Разработка высоконагруженных приложений на языке JavaScript

Бакалаврская работа  
студента 411 группы А. С. Низамутдинова

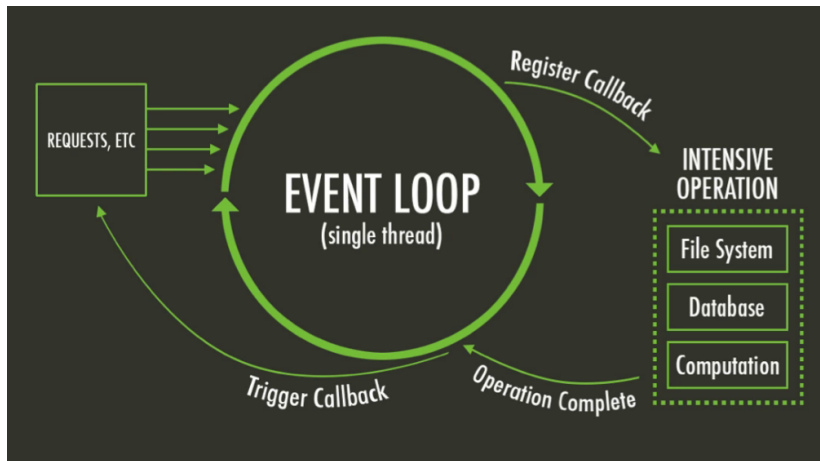
Саратовский национальный исследовательский  
государственный университет  
им. Н. Г. Чернышевского

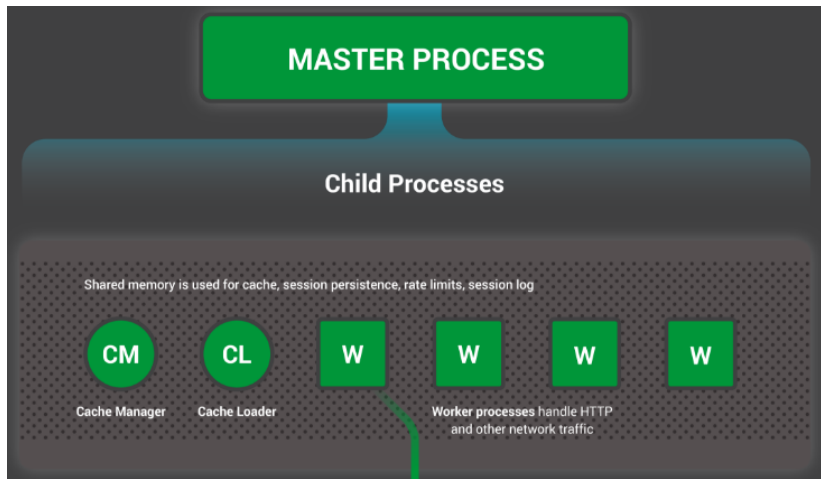
Кафедра математической кибернетики  
и компьютерных наук

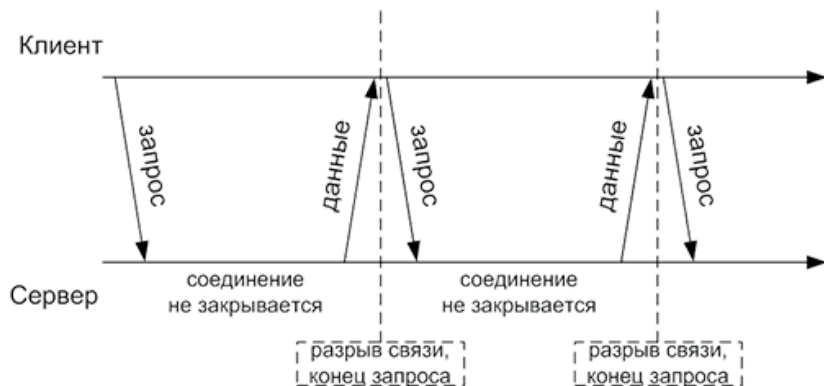
Научный руководитель: доцент, Борзов И. А.

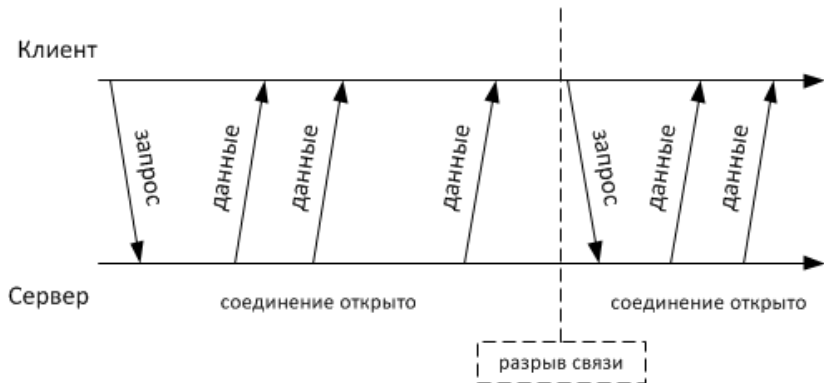
2016г.

- 1 изучить архитектуру NodeJS и NGINX;
- 2 рассмотреть способы асинхронного обмена данными с сервером с использованием websocket, comet, iframe и jsonp;
- 3 реализовать веб-сервер на websocket и comet;
- 4 настроить NGINX как прокси-сервер и балансировщик нагрузки;
- 5 провести нагрузочное тестирование для comet и websocket соединений.





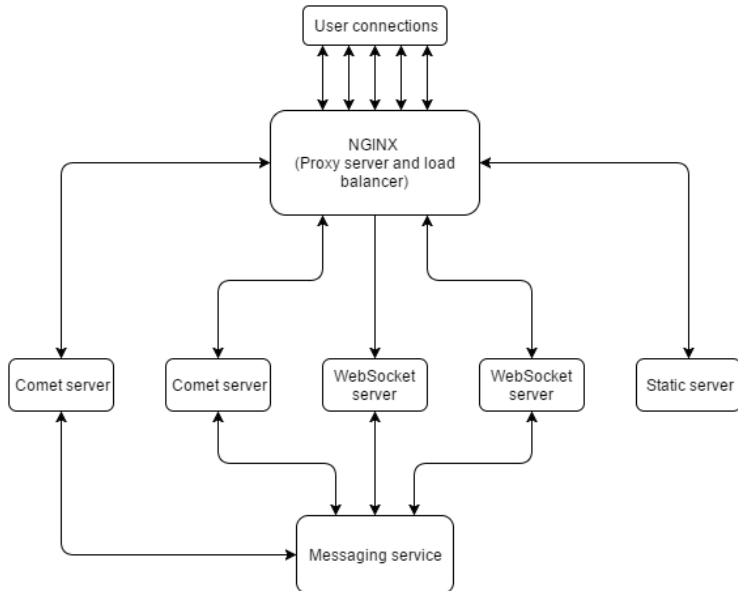


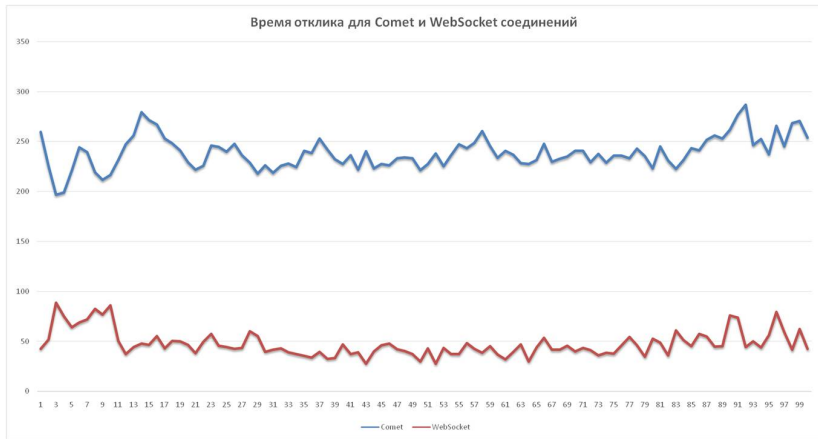


- 1 На клиентской стороне создается тег `<script>` на специальный URL, в котором в качестве параметра передается имя функции обратного вызова, которая будет вызываться при получении данных;
- 2 В свою очередь сервер формирует ответ в виде вызова этой функции с данными переданными в ней в качестве параметров, и отправляет ответ клиенту;
- 3 Сразу после того как клиентская сторона получает ответ от сервера, полученный скрипт начинает немедленно выполняться, таким образом вызывая функцию обратного вызова, которая располагается на стороне клиента.







- 1 Клиентская сторона отправляет GET запрос с заголовком `Upgrade: websocket`, указывая, что хочет переключиться на WebSocket протокол;
- 2 Сервер возвращает ответ с заголовками `HTTP/1.1 101 Switching Protocols` и `Upgrade: websocket`;
- 3 Затем данные передаются по специальному протоколу с использованием «фреймов». И это уже совсем не HTTP.
- 4 При закрытии соединения сторона, желающая это сделать (обе стороны в WebSocket равноправны) отправляет закрывающий «фрейм» (с опкодом `0x8`), в теле которого указывает причину закрытия.







В настоящей работе были изучены основы архитектуры NodeJS и NGINX, также были рассмотрены способы асинхронного обмена данными с сервером посредством использования websocket, comet, iframe и jsonp. На основе этих данных было создано приложение, позволяющее обмениваться данными между клиентами и сервером в режиме реального времени, а также в достаточной мере устойчивое к высоким нагрузкам (к большому числу одновременно подключенных клиентов). Кроме этого по результатам нагрузочного тестирования удалось выяснить, что передача данных с использованием WebSocket протокола является наиболее оптимальной для двунаправленной передачи данных.

-  <https://learn.javascript.ru/ajax-jsonp>  
Протокол JSONP
-  <https://tools.ietf.org/html/rfc6455>  
The WebSocket Protocol
-  <https://learn.javascript.ru/xhr-longpoll>  
COMET с XMLHttpRequest: длинные опросы
-  <https://learn.javascript.ru/ajax-iframe>  
IFRAME для AJAX и COMET
-  Сухов К. К.  
Node.js. Путеводитель по технологии  
ДМК Пресс: 2015.
-  <https://habrahabr.ru/post/260065/>  
NGINX изнутри: рожден для производительности и масштабирования

СПАСИБО ЗА ВНИМАНИЕ!