Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«Пермский национальный исследовательский**

**политехнический университет»**

**(ПНИПУ)**

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЕТ**

**о научно-исследовательской работе на тему**

Разработка веб-сервера, используя архитектурный стиль REST

Выполнил: студент группы РИС-17-1б

Ерохин Николай Владимирович

Проверил: доцент кафедры ИТАС

Курушин Даниил Сергеевич

**г. Пермь, 2021**

**РЕФЕРАТ**

Отчёт 15 с., 10 источн.

REST, ВЕБ-СЕРВЕР, ВЕБ-СЕРВИС, ПРОТОКОЛ, КЛИЕНТ, РАСПРЕДЕЛЁННАЯ СЕТЬ.

Целью научно-исследовательской работы является выбор средств и технологий разработки сервиса подачи заявок на куплю/продажу для возможности реализовать приложение-клиент данного сервиса.

Объектом исследования в рамках данной работы является применимость архитектурного стиля REST к условиям, приближенным к реальным.

Предметом исследования является выбор средств и технологий разработки сервиса подачи заявок на куплю/продажу.

В процессе работы будет произведена попытка рассмотрения альтернатив для реализации веб-сервера для сервиса подачи заявок на куплю/продажу, а также выбор средств для данной реализации.

В результате работы были выявлены преимущества архитектурного стиля REST и его наилучшая область применения.

**СОДЕРЖАНИЕ**

[ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ 4](#_Toc70330626)

[ВВЕДЕНИЕ 5](#_Toc70330627)

[1 Исследование архитектурного стиля REST 7](#_Toc70330628)

[2 Изучение смежных с архитектурным стилем REST технологий 9](#_Toc70330629)

[3 Описание веб-сервиса и выбор технологий 11](#_Toc70330630)

[ЗАКЛЮЧЕНИЕ 14](#_Toc70330631)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 15](#_Toc70330632)

ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ

REST (Representational State Transfer) – архитектурный стиль взаимодействия компонентов распределённого приложения в компьютерной сети.

Клиент-сервер – вычислительная или сетевая архитектура, в которой задания или сетевая нагрузка распределены между поставщиками услуг, называемых серверами, и заказчиками услуг, называемых клиентами.

Веб-сервер (в контексте разработки программного обеспечения) – программное обеспечение, выполняющее функции поставщика услуг, выполняя приём и обработку запросов, а также выдачу ответов на них.

Веб-сервис – комплекс программного обеспечения, предоставляющий доступ к данным и функциям системы в распределённой компьютерной сети.

ВВЕДЕНИЕ

Компьютерные технологии начали своё непрерывное развитие в середине XX века. С течением времени накапливается всё больше знаний в области информационных технологий, растёт производительность вычислительной техники с одновременным упрощением её использования. Богатые возможности вычислительной техники и способов её коммутации позволяет разрабатывать разнообразные распределённые системы обработки информации, а непрерывное развитие вычислительных систем позволяет проектировать всё более сложные системы обработки информации.

Растущую сложность систем необходимо упорядочивать по специальным правилам, вводить некоторые ограничения и следовать им. В противном случае процесс разработки информационной системы и сама информационная система становятся не управляемыми, ещё более сложными и, в конце концов, не пригодными к использованию. Такими правилами могут быть различные сетевые протоколы, паттерны проектирования и программирования, архитектурные стили. Накопленный за прошлые десятилетия опыт необходимо изучать, чтобы иметь возможность разрабатывать эффективные информационные системы.

В настоящее время активно заполняется ниша программных приложений, которые используют сетевую архитектуру «клиент-сервер». Данная архитектура открывает огромные возможности по созданию самых разнообразных систем, которые не исчерпаны до сих пор, поэтому знание основ и методологий построения таких систем позволит успешно конкурировать на рынке программных приложений. Хотя протоколы и архитектурные стили разработаны относительно давно, большинство разработчиков программного обеспечения в полной мере не знают или не понимают их, ошибочно принимают те или иные решения, и, как следствие, получают растущую сложность своих продуктов с точки зрения разработки и поддержки, что отрицательно сказывается на их конкурентоспособности.

В любой сфере, не только информационных технологий, перед промышленным применением приёмов и технологий, их следует изучить и применить в упрощённых условиях. Узнав об архитектурном стиле REST, было принято решение о его изучении, выявлении условий для его успешного применения, создания задачи на разработку веб-сервера с использованием этого архитектурного стиля, чтобы увидеть его работу в действии.

**Объектом исследования** в рамках данной работы является применимость архитектурного стиля REST к условиям, приближенным к реальным.

**Предметом исследования** является выбор средств и технологий разработки сервиса подачи заявок.

**Задачи**, которые необходимо решить для достижения цели:

* изучить архитектурный стиль REST;
* изучить альтернативные подходы к разработке веб-серверов;
* выявить условия, при которых наиболее подходящим архитектурным стилем является REST;
* сделать обзор инструментов, с помощью которых можно разработать веб-сервер;
* выбрать наиболее подходящий для исследования и разработки инструмент.

1 Исследование архитектурного стиля REST

REST – архитектурный стиль взаимодействия компонентов распределённого приложения в компьютерной сети. REST представляет собой согласованный набор ограничений, учитываемых при проектировании распределённой медиа-системы. В определённых случаях, таких как интернет-магазины, поисковые системы и прочие системы, основанные на данных, это приводит к повышению производительности и упрощению архитектуры. Из определения REST становится очевидна наилучшая область применения данной архитектуры – системы, основанные на данных. Это означает, что если необходимо обеспечить централизованное хранение данных и распределённый доступ к ним, то выбор архитектурного стиля REST можно считать наилучшим решением.

Свойства архитектуры REST:

* производительность – взаимодействие компонентов системы может являться доминирующим фактором производительности и эффективности сети с точки зрения пользователя;
* масштабируемость – для обеспечения большого числа компонентов и взаимодействий компонентов.

Как и любой архитектурный стиль, REST накладывает ограничения для построения распределённых систем. Выполнение этих ограничений обязательно. Накладываемые ограничения определяют работу сервера в том, как он может обрабатывать и отвечать на запросы клиентов. Действуя в рамках этих ограничений, система приобретает свойства архитектуры REST. Любое нарушение этих ограничений не позволяет считать систему как построенную по архитектуре REST:

* Модель «клиент-сервер». Разграничение потребностей является принципом, лежащим в основе данного ограничения.
* Отсутствие состояния. Протокол взаимодействия между клиентов и сервером требует соблюдения следующего условия: в период между запросами клиента никакая информация о состоянии клиента на сервере не хранится. Все запросы от клиента должны быть составлены так, чтобы сервер получил всю необходимую информацию для выполнения запроса. Состояние сессии сохраняется на стороне клиента.
* Кэширование. Клиенты могут выполнять кэширование ответов сервера. Ответы сервера, в свою очередь, должны иметь явное или неявное обозначение как кэшируемые или некэшируемые с целью предотвращения получения клиентами устаревших или неверных данных в ответ на последующие запросы.
* Единообразие интерфейса. Наличие унифицированного интерфейса является фундаментальным требованием дизайна REST-сервисов. Унифицированные интерфейсы позволяют каждому из сервисов развиваться независимо. К унифицированным интерфейсам предъявляются следующие ограничительные условия: идентификация ресурсов (все ресурсы идентифицируются в запросах), манипуляция ресурсами через представление, «самоописываемые» сообщения (каждое сообщение содержит достаточно информации, чтобы понять, каким образом его обрабатывать).
* Слои. Клиент обычно не способен (и не должен) точно определить, взаимодействует он напрямую с сервером или же с промежуточным узлом. Применение промежуточных сервером способно повысить масштабируемость за счёт балансировки нагрузки и распределённого кэширования.
* Код по требованию (необязательное ограничение)

В распределённых системах используется понятие «Удалённый вызов процедур» (RPC). Это класс технологий, позволяющих программам вызывать функции или процедуры в другом адресном пространстве – на удалённых узлах, либо в независимой сторонней системе на том же узле. Обычно реализация RPC-технологии включает два компонента: сетевой протокол для обмена в режиме клиент-сервер и язык сериализации объектов. Архитектурный стиль REST можно считать альтернативной заменой RPC. Следует заметить, что архитектурный стиль REST не включает в себя компоненты «сетевой протокол» и «сериализация объектов», а использует их.

2 Изучение смежных с архитектурным стилем REST технологий

Так как архитектурный стиль REST использует компоненты сетевого протокола и сериализации объектов, необходимо определиться с этими понятиями, а также рассмотреть альтернативы, которые применимы в рамках REST.

Сетевой протокол – набор правил и действий, позволяющий установить соединение и обмен данными между двумя и более включенными в сеть устройствами. Примеры сетевых протоколов:

* HTTP (Hyper Text Transfer Protocol) – протокол передачи гипертекста. Позволяет передавать различные данные в текстовом виде.
* FTP (File Transfer Protocol) – протокол передачи файлов со специального файлового сервера на компьютер пользователя.
* POP3 (Post Office Protocol) – стандартный протокол почтового соединения.
* SMTP (Simple Mail Transfer Protocol) – протокол, который задаёт набор правил для передачи почты.
* TELNET – протокол удалённого доступа, который даёт возможности абоненту работать на любой ЭВМ, находящейся с ним одной сети, как на своей собственной.

Для передачи различных типов данных по сети необходимо использовать процесс сериализации объектов – это процесс перевода структуры данных в последовательность байтов. Обратной к операции сериализации является операция десериализации – создание структуры данных из битовой последовательности.

Протокол HTTP накладывает ограничение на структуры данных – это должны быть данные в текстовом формате, поэтому необходимо рассмотреть, какие именно существуют текстовые форматы:

* Простой текст (plain text) – обычный набор символов, который воспринимается как текст. Не позволяет структурирование данных
* Двоичные данные (binary data) – последовательность символов, в которой закодированы двоичные данные. Как правило, используется для передачи файлов в виде текста
* XML (extensible markup language) – расширяемый язык разметки. Позволяет структурирование. С физической точки зрения XML-документа состоит из сущностей, из которых каждая может ссылаться на другую сущность. Единственный корневой элемент – документная сущность. Содержание сущностей – символы. С логической точки зрения документ состоит из комментарием, объявлений, элементов, ссылок на сущности и инструкций обработки. Всё это в документе структуризуется разметкой.
* JSON – текстовый формат обмена данными, основанный на JavaScript.

3 Описание веб-сервиса и выбор технологий

Сервис приёма заявок купли/продажи предоставляет возможность различным пользователям системы создавать объявления о продаже своего имущества, а также просматривать объявления на продажу других пользователей системы. Пользователи могут находиться в разных городах, поэтому единственной объединяющей их компьютерной сетью является сеть Интернет. Все пользователи должны иметь доступ к данным сервиса, таким как:

* информация об объявлениях на продажу;
* информация о заявителях объявлений;
* прочая информация, необходимая для совершения сделки купли/продажи (цена, адрес и другое).

Говоря о преимуществах самого архитектурного стиля REST, можно выделить следующие:

* надёжность, за счёт отсутствия необходимости сохранять информацию о состоянии клиента, которая может быть утеряна;
* производительность, за счёт использования кэширования;
* масштабируемость;
* прозрачность системы взаимодействия;
* простота и унификация интерфейсов;
* портативность компонентов;
* лёгкость внесение изменений и поддержки системы;
* способность эволюционировать, приспосабливаясь к новым требованиям.

Из обзора сетевых протоколов и их возможностей можно сделать вывод, что наиболее общим по назначению является протокол HTTP, так как можно передавать произвольные данные. Основой HTTP является технология «клиент-сервер», то есть предполагается существование потребителей (клиентов), которые инициируют соединение и посылают запрос, и поставщиков (серверов), которые ожидают соединения для получения запроса, производят необходимые действия и возвращают обратно сообщение с результатом. Основным объектов манипуляции в HTTP является ресурс. Обычно такими ресурсами являются хранящиеся на сервере файлы, но ими могут быть логические объекты или что-то абстрактное. Особенностью протокола HTTP является возможность указать в запросе и ответе способ представление одного и того же ресурса по различным параметрам: формату, кодировке, языку. Благодаря возможности указания способа кодирования сообщения клиент и сервер могут обмениться двоичными данными, хотя данный протокол является текстовым.

При ближайшем рассмотрении протокола HTTP становится понятно, что именно его лучше всего выбрать в качестве сетевого протокола при использовании архитектурного стиля REST, так как и там и там:

* основой является технология «клиент-сервер»;
* имеется возможность работы над произвольными данными;
* имеется возможность организовывать произвольные запросы и ответы на них.

Таким образом, в качестве вызова удалённой процедуры будет выступать обычный HTTP-запрос, в котором необходимые данные будут передаваться в качестве параметров запроса. Интересно заметить, что термин «REST» был введён одним из создателей протокола «HTTP» Роем Филдингом в своей диссертации «Архитектурные стили и дизайн сетевых программных архитектур», в которой он подвёл теоретическую основу под способ взаимодействия клиентов и сервером, абстрагировав его и назвав «передачей представительного состояния». Филдинг описал концепцию построения распределённого приложения, при которой каждый запрос (REST-запрос) клиента к серверу содержит в себе исчерпывающую информацию о желаемом ответе сервера (желаемом представительном состоянии), и сервер не обязан сохранять информацию о состоянии клиента.

Необходимость передачи данных различных структур ограничивает выбор текстовым форматов теми, которые поддерживают структурирование – это форматы XML и JSON. Целесообразно сделать выбор в пользу популярного в настоящее время формата JSON, так как у него есть ряд преимуществ перед форматом XML: простота, читабельность, компактность.

ЗАКЛЮЧЕНИЕ

В результате выполнения исследовательской работы был произведён обзор архитектурного стиля REST, выявление наилучших условий для его применения. Был произведён обзор смежных технологий и понятий, а также выбор альтернативных способов и технологий для возможной последующей реализации сервиса подачи заявок на куплю/продажу.

Результаты работы позволяют приступить к проектированию компонентов системы в соответствии с архитектурным стилем REST, их состав и обязанности, в соответствии с выбранными технологиями и методами вызова процедур и передачи данных. На основе проектирования можно будет приступить к выбору средств реализации сервиса подачи заявок на куплю/продажу.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Онлайн-документация по формату данных JSON. Режим доступа: <http://www.json.org/json-ru.html>
2. Онлайн-статья о понятии «веб-сервер». Режим доступа: <https://en.wikipedia.org/wiki/Web_server>
3. Онлайн-статья по архитектурному стилю REST. Режим доступа: <https://ru.wikipedia.org/wiki/REST>
4. Онлайн-урок по архитектуре «Клиент-сервер». Режим доступа: <http://www.4stud.info/networking/lecture5.html>
5. Онлайн-документация по системе управления БД PostgreSQL. Режим доступа: <https://www.postgresql.org/docs/9.5/static/index.html>
6. Онлайн-статья о платформе JavaEE. Режим доступа: <https://ru.wikipedia.org/wiki/Java_Platform,_Enterprise_Edition>
7. Онлайн-статья о протоколе HTTP. Режим доступа: <https://ru.wikipedia.org/wiki/HTTP>
8. Онлайн-статья о веб-сервисах. Режим доступа: <https://habrahabr.ru/post/46374/>
9. Онлайн-документация о библиотеке Android SDK. Режим доступа: <https://developer.android.com/reference/packages.html>
10. Энтони Гонсалвес. Изучаем Java EE 7. Питер, 2014.