Доклад

Лист 1. Цели и задачи

Здравствуйте, уважаемые члены государственной аттестационной комиссии. Вашему вниманию представляется выпускная квалификационная работа на тему «Система мониторинга парковочных мест». Данная система позволит пользователям в режиме реального времени получать актуальную информацию о состоянии парковок. А также реализована функция онлайн бронирования парковочного места. Владельцы же стоянок смогут удалённо управлять установленными датчиками МК-подсистемы.

В ходе выполнения работы были выполнены следующие задачи:

- Проведено исследование IoT технологий в сфере транспортной инфраструктуры. А также проведен анализ существующих систем.
- Спроектирована архитектура системы. А также разработана принципиальная схема МК-подсистемы.
- Разработано программное обеспечение для компонентов системы.
- А также проведено тестирование работоспособности всей системы.

Здесь показана структурная схема системы. Она состоит из: МК-подсистемы, вычислительного хаба и центрального сервера.

МК-подсистема представляет из себя микроконтроллер, которому подключаются до 15 сенсоров, каждый из которых следит за своим парковочным местом. Сенсоры крепятся к дорожному полотну под парковочным местом. Через радиомодуль LoRa информация о состоянии парковок передается в вычислительный хаб, для последующей обработки.

Основное назначение вычислительного хаба, состоит в осуществлении взаимодействия сервера, датчиков МК-подсистемы и других компонентов системы. Он с множества датчиков собирает дынные и отправляет центральному серверу.

Сервер осуществляет сбор данных с датчиков, управляет всеми компонентами системы и выводит данные пользователям о состоянии парковок.

Лист 2. Исследование

В рамках ВКР было проведено исследование IoT технологий в сфере транспортной инфраструктуры.

Архитектура интернет вещей состоит из 4х уровней: уровня сети сенсоров, уровня шлюзов и сетей, сервисного уровня и уровня приложений.

Для передачи данных между вычислительном хабом и сервером использовался протокол MQTT. MQTT организован по принципу издатель/подписчик: издатель является отправителем сообщения, которое публикуется в брокере сообщений, а подписчик получает сообщение из брокера. Данный протокол обладает небольшими накладными расходами на стороне устройства, что позволяет сократить расход батареи и увеличивает количество передаваемых в единицу времени сообщений.

Данные от датчиков передаются по сети LoRaWAN. У LoRaWAN имеется преимущества:

- большая дальность передачи радиосигнала до 10 — 15 км.
- Низкое энергопотребление.
- Высокая проникающая способность радиосигнала.

Но имеет недостатки:

- Низкую пропускную способность.
- Большую задержку передачи данных.

Лист 3. Функциональная схема системы

Здесь показана функциональная схема системы. Данные от датчиков о состоянии парковочных мест передаются по сети LoRaWAN на вычислительный хаб. Хаб преобразует принятый пакет в формат JSON и передаёт серверу по MQTT. Сервер принимает и сохраняет эти дынные в БД. Веб-приложение периодически отправляет данные о состояниях парковок пользователю по протоколу WebSockets.

Для реализации веб-приложения использовался фреймворк Ruby on Rails с веб-сервером Puma. Фронтенд-сервер Nginx осуществляет раздачу статики и проксирования запросов на веб-сервер. Данные о парковках хранятся в СУБД PostgreSQL. Для кэширования используется сервис Memcached. Key-value хранилище Redis применяется для реализации механизма Action Cable, позволяющий раздавать данные о парковках пользователям по протоколу WebSockets.

Лист 4. Принципиальная схема

Здесь показана реализованная принципиальная схема МК-подсистемы. Она состоит из:

- радиомодуля LoRa, подключаемый по шине SPI
- множества сенсоров подключаемые по шине I2C через расширитель портов PCF8574
- терминала оплаты состоящего из дисплея и клавиатуры

Лист 5. Диаграмма классов

Здесь показаны реализованные диаграммы классов для МК-подсистемы и вычислительного хаба. При разработке были использованы такие паттерны проектировании как СТРАТЕГИЯ и СИНГЛТОН. Например, абстрактный класс «*Driver*» у вычислительного хаба является базовым классом для «*LoRaConnection*» и «*SerialConnection*» и является интерфейсом для взаимодействия с датчиками. Потомки данного класса реализовывают подключение и приём-передачу данных, определяя соответствующие методы.

Лист 6. Схема БД и алгоритмы

Здесь показана разработанная схема базы данных PostgreSQL для хранения данных о состояниях парковочных мест на сервере. Для работы с картографическими данными используется расширение PostGIS. В целях оптимизации производительности на каждое поле участвующее в запросах были созданы соответствующие индексы, а также была применена денормолизация данных.

Ниже показан алгоритм выдачи парковок. Т.к. проектируемая система является высоконагруженной, карта была поделена на взаимно пересекающиеся квадраты, число которых ограничено. Размеры их определяются экспоненциальной функцией относительно масштаба. Таким образом, появляется возможность для кэширования данных.

Лист 7. Тестирование

Серверное ПО для тестирования было развернуто на облаке Heroku. Для нагрузочного тестирования была развернута соответствующая инфраструктура на облачной платформе Google Cloud, схема сети которой показана здесь. Она состоит из балансировщика Nginx, который распределяет нагрузку между множеством бекэнд-серверов.

На этих графиках показана распределение нагрузки при падении одного из бекэнд-сервера.

По результатам нагрузочного тестирования было выяснено, что при заполненной БД с 13 тыс. записями RPS для алгоритма выдачи парковок составил 846 запросов в секунду.

Для обеспечения качества для дальнейшей разработки были написаны 60 автоматизированных тестов серверного ПО.

Также было проведено комплексное тестирование системы, в ходе которого было выяснено, что все требования к системе были выполнены.

Спасибо за внимание!