Министерство образования Республики Беларусь
Учреждение образования
Белорусский государственный университет
информатики и радиоэлектроники

УДК 004.93.1

Ледяев
Владимир Павлович

Система учета пассажиропотока

АВТОРЕФЕРАТ
на соискание степени магистра информатики и вычислительной техники
по специальности 1-40 81 04 «Обработка больших объемов информации»

_____

Научный руководитель
Хотеев Александр Леонидович
доцент кафедры Информатики
кандидат физико-математических наук

_____

Минск 2018

ВВЕДЕНИЕ

Пассажирские перевозки представляют собой одну из важных отраслей экономики. Одной из проблем предприятий пассажирского транспорта является учет пассажиропотока, для анализа и построения экономически выгодного графика движения общественного транспорта. Осуществить контроль пассажиропотока можно несколькими способами. Один из них — использование систем видеонаблюдения.

Видеонаблюдение уже давно является неотъемлемой частью нашей жизни. Оно активно используется для обеспечения безопасности коммерческих объектов, общественных мест и частных домовладений. С развитием Интернета вещей и информационных технологий появилась возможность строить достаточно интеллектуальные системы на их основе. С точки зрения Интернета вещей, главный элемент системы видеонаблюдения, а именно видеокамеру, можно рассматривать как IoT устройство.

Подсчет пассажиропотока с использованием систем видеонаблюдения требует определенного количества вычислительных мощностей, а также наличие соответствующей инфраструктуры. Еще пару лет назад для осуществления этой задачи предприятию необходимо было бы производить закупку серверов и настройки той самой инфраструктуры.

С появлением облачных технологий задача значительно упростилась. Поставщики облачных технологий берут на себя обязанности по построению отказоустойчивой инфраструктуры одновременно с высоким уровнем качества. Таким образом задача для предприятия сводится только к приобретению или разработке специализированного ПО, которое будет впоследствии запущено в облачном сервисе.

В данной работе будет рассмотрена возможность создания системы, позволяющей осуществлять мониторинг пассажиропотока в реальном времени. Целью диссертационной работы является анализ, проектирование и создание решения масштабируемого, отказоустойчивой и слабо связной распределенной системы.

Для выполнения цели диссертационной работы были поставлены следующие задачи:

1) Обзор предметной области, анализ технологий и алгоритмов, необходимых для реализации системы.

2) Разработка архитектуры системы.

3) Разработка прототипа системы.

Объектом исследования является пассажиропоток, а также облачные сервисы. Предметом исследования является учет пассажиропотока в реальном времени.

Основным вопросом, затронутым в исследовании, является не только теоретическое исследование, но и его практическая реализация. Существует масса подходов при проектировании сложных систем, множество технологий, способствующих процессу реализации на практике поставленной задачи. В результате необходимо получить приложение, которое будет развернуто на облачной платформе. Следовательно, нужно выбирать соответствующие средства разработки.

КРАТКОЕ СОДЕРЖАНИЕ

Для того, чтобы надежно управлять и эффективно обрабатывать большие данные видеопотока, требуется масштабируемая, отказоустойчивая, слабо связная распределенная система.

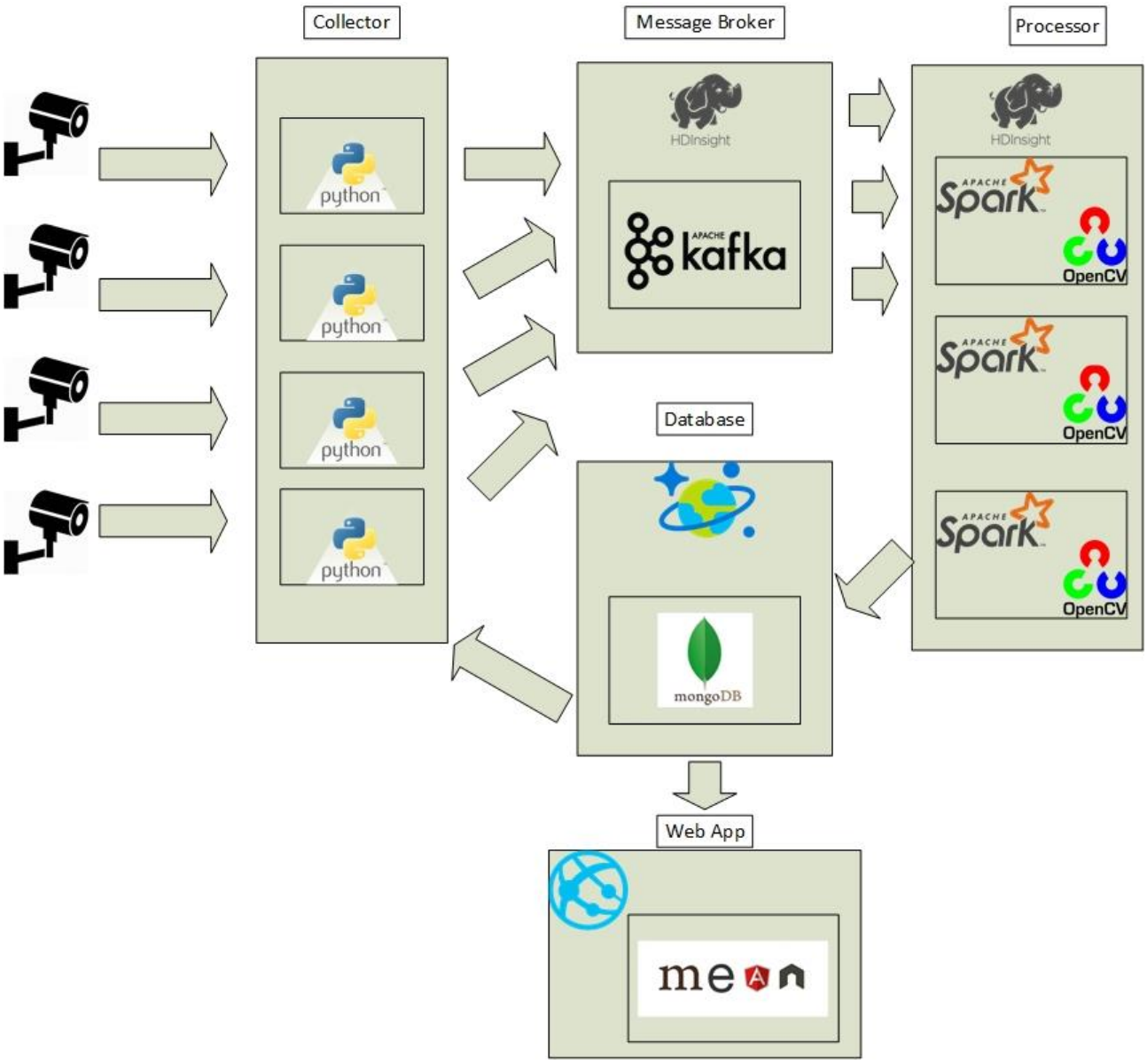

Рисунок 1 – Архитектура системы

Систему условно можно разделить на два главных модуля:

1. Passenger Accounting Module – модуль учета пассажиров.
2. Web Module – веб интерфейс.

Модуль учета пассажиров является основным в разрабатываемой системе и служит своего рода конвейером обработки видеопотока. Этот конвейер состоит из нескольких компонентов:

Video Stream Collector – сборщик видеопотоков. Компонент работает с кластером IP камер, которые предоставляют видеопоток в режиме реального времени. Компонент читает видеопоток и преобразует его в ряд видеокадров (фреймов). Информацию про доступные видеокамеры компонент берет из базы данных MongoDb.

Различные камеры могут предоставлять данные с различными спецификациями, такими как кодек, разрешение или количество кадров в секунду. Сборщик видеопотоков сохраняет эти данные при создании кадров из видеопотока.

Перед отправкой, каждый кадр подвергается изменению размера. Это не сказывается на качестве работы системы, но существенно экономит трафик. После этого, обработанное изображение сериализуется в JSON объект, к этому объекту добавляется служебная информация и после этого объект отправляется в сообщении брокеру сообщений Kafka с помощью клиента KafkaProducer.

Kafka в первую очередь предназначен для текстовых сообщений небольших размеров, но сообщение JSON, содержащее массив байт видеокадра будет большим (1.5 мегабайт например). Поэтому Kafka клиент потребует изменений конфигурации прежде чем сможет обрабатывать более крупные сообщение. Необходимо отрегулировать следующие свойства KafkaProducer:

– batch.size - размер пакета сообщений, который отправляется от производителя к брокеру. Производители умеют собирать эти “паки“, чтобы не отправлять сообщения по одному, т.к. они могут быть достаточно маленькими. В общем случае, чем больше этот параметр, тем:

– (Плюс) Больше степень сжатия, а значит выше пропускная способность.

– (Минус) Больше задержка в общем случае.

– max.request.bytes - максимальный размер запроса в байтах. Этот параметр ограничит количество партий записей, которые производитель отправит в одном запросе, чтобы избежать отправки огромных запросов. Это также эффективно ограничивает максимальный размер партии записи;

– compression.type - тип сжатия для всех данных, созданных производителем. По умолчанию используется значение none (без сжатия). Допустимые значения: none, gzip, snappy или lz4. Сжатие применяется на всю партию данных, поэтому размер партии будет также влиять на коэффициент сжатия (чем больше партия, тем больше сжатие).

Stream Data Buffer – буфер видеопотока. Чтобы обрабатывать огромное количество данных видеопотока без потерь, необходимо сохранить данные потока во временном хранилище. Брокер Kafka работает как буферная очередь

для данных, которые производит сборщик видеопотоков. Kafka использует файловую систему для хранения сообщений, а время, в течение которого они сохраняют эти сообщения, настраивается.

Хранение данных на диске перед обработкой обеспечивает его долговечность и улучшает общую производительность системы, так как процессоры могут обрабатывать данные в разное время и на разных скоростях в зависимости от нагрузки. Это повышает надежность системы, когда скорость производства данных превышает скорость обработки данных.

Kafka гарантирует порядок сообщений в одном разделе для данной темы. Это чрезвычайно полезно для обработки данных, когда порядок данных важен. Для хранения больших сообщений в файле server.properties сервера Kafka, необходимо настроить следующие конфигурации:

– message.max.bytes - самый большой размер партии, разрешенный Kafka.

– replica.fetch.max.bytes - количество потоков, используемых для репликации сообщений от исходного брокера. Увеличение этого значения может увеличить степень параллелизма ввода-вывода.

Video Stream Processor – процессор видеопотока выполняет следующие действия:

1 Читает JSON сообщение у брокера Kafka в виде RDD (Resilient Distributed Dataset).

2 Для каждого объекта в наборе данных производится десериализация видеокадров и их обработка.

3 Данные сохраняются в базу MongoDB.

Процессора видеопотока построен на Apache Spark. Spark предоставляет Spark Streaming API, которое использует дискретизированный поток или DStream и новый Structured Streaming API, основанный на датасете. Процессор видеопотока использует Structured Streaming API для получения и обработки JSON сообщений из Kafka.

Structured Streaming обеспечивает быструю, масштабируемую, отказоустойчивую, сквозную, однократную обработку. Именно поэтому процессор видеопотока разработан вокруг Structured Streaming Spark. Механизм Structured Streaming обеспечивает встроенную поддержку структурированных текстовых данных и управления состоянием для запросов агрегирования. Этот движок также предоставляет такие функции, как обработка неагрегированных запросов и внешнее управление состоянием набором данных.

Для обработки больших сообщений в движок Spark должны быть переданы следующая настройка консьюмера Kafka:

– max.partition.fetch.bytes – максимальный объем данных для каждого раздела, который вернет сервер.

Процессор видеопотока использует библиотеку OpenCV для обработки данных.

ЗАКЛЮЧЕНИЕ

В данной работе были рассмотрены различные технологии и средства, позволяющие создать систему учета пассажиропотока. В результате проведенного исследования был произведен анализ большого объема литературы по тематике облачных вычислений и компьютерного зрения. На основе анализа литературы была разработана архитектура решения. В результате реализации архитектурного решения были получены следующие результаты:

Разработана система, состоящая из двух модулей (модуль учета пассажиров и веб интерфейс) в которой:

Модуль учета пассажиров является масштабируемого, отказоустойчивой и слабо связной распределенной подсистемой, состоящей из сборщика видеопотоков (Kafka producer), буфера видеопотоков (Kafka server) и процессора видеопотоков (Kafka consumer). Для потоковой обработки сообщений процессор видеопотоков использует программные средства Spark и OpenCV.

Веб интерфейс реализован на основе современного стека MEAN.

Таким образом, разработанное программное средство обеспечивает рациональное решение возложенных на него задач, предоставляя необходимый уровень производительности и функциональности, а также возможности для его дальнейшего развития.

СПИСОК ОПУБЛИКОВАННЫХ РАБОТ

[1-А] Ледяев В.П. Обнаружение движущихся объектов с помощью компьютерного зрения и библиотеки OpenCV [Электронный ресурс]// Экономика и социум – 2017 – №12(43) (декабрь). – http://iupr.ru/domains_data/files/zurnal_43/Ledyaev%20V.P.(Informacionnye%20i%20kommunikativnye%20tehnologii).pdf

[2-А] Ледяев В.П. Определение объектов на изображениях с помощью гистограммы направленных градиентов и метода опорных векторов [Электронный ресурс]// Экономика и социум. – 2017. – №12(43) (декабрь). – http://iupr.ru/domains_data/files/zurnal_43/Ledyaev%20V.P.%20–1%20%20(Informacionnye%20i%20kommunikativnye%20tehnologii).pdf