УДК 004

Черников В.С.
студент 4 курс
Институт инженерных и цифровых технологий БелГУ
Белгородский государственный национальный исследовательский университет
Россия, г. Белгород

ВЫБОР ИНСТРУМЕНТАЛЬНЫХ СРЕДСТВ ПРИ РАЗРАБОТКЕ ПРОГРАММНОГО МОДУЛЯ СБОРА И ОБРАБОТКИ АНАЛИТИЧЕСКИХ ДАННЫХ ПО ЭКСПЛУАТАЦИИ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ УПРАВЛЕНИЯ НАРУЖНЫМ ОСВЕЩЕНИЕМ
SELECTION OF TOOLS WHEN DEVELOPING A SOFTWARE MODULE FOR COLLECTING AND PROCESSING ANALYTICAL DATA FOR THE OPERATION OF AN AUTOMATED EXTERNAL LIGHTING CONTROL SYSTEM

В данной статье автор обосновывает важность выбора инструментальных средств при разработке программного модуля сбора и обработки аналитических данных по эксплуатации автоматизированной системы управления наружным освещением. Автор описывает архитектуру решения по сбору и обработке информации и выделяет 4 подмодуля. Далее в ходе статьи автор осуществляет выбор инструментальных средств для каждого из подмодулей.

Ключевые слова: инструментальные средства; сбор и обработка данных; аналитика; автоматизированные системы управления.

In this article, the author substantiates the importance of choosing tools when developing a software module for collecting and processing analytical data for the operation of an automated outdoor lighting control system. The author describes the architecture of the solution for collecting and processing information and identifies 4 submodules. Further in the course of the article, the author selects tools for each of the submodules.

Keywords: tools; data collection and processing; analytics; automated control systems.

В настоящее время аналитика данных стала одним из ключевых инструментов для принятия стратегических решений во многих сферах. Поэтому выбор инструментальных средств для разработки программного модуля сбора и обработки аналитических данных имеет большое значение. Важно учитывать следующие аспекты при выборе инструментов:

1. Функциональность: необходимо выбирать инструменты, которые позволяют собирать, хранить, обрабатывать и визуализировать данные различного формата и объема.
2. Производительность: важно выбирать средства, которые обеспечивают быструю и эффективную обработку данных для минимизации времени отклика и увеличения производительности системы.
3. Масштабируемость: выбранные инструменты должны быть способны масштабироваться в зависимости от изменяющихся потребностей и объемов данных.
4. Надежность и безопасность: следует обращать внимание на защиту данных и обеспечение их конфиденциальности, а также на отказоустойчивость системы.
5. Совместимость: важно выбирать инструменты, которые легко интегрируются с другими средствами и технологиями, используемыми в организации. В общем, правильный выбор инструментальных средств при разработке программного модуля сбора и обработки аналитических данных играет критически важную роль и является актуальной задачей для любой организации или команды разработчиков.

Первоначально все инструменты должны быть выбраны исходя из целей и задач проекта, а также учитывать специфику исходных данных и требования

по обработке этих данных. Выбранные инструменты в значительной степени определяют скорость и эффективность разработки модулей, их универсальность, масштабируемость и применимость в разных условиях. Существует большое количество инструментов для работы с аналитическими данными, каждый из которых имеет свои особенности, преимущества и недостатки. Некоторые из них предназначены для работы с большими объемами данных (например, Hadoop или Apache Spark), другие могут предоставить удобные инструменты для визуализации результатов (Tableau, Power BI), еще другие хорошо подходят для работы с реального времени данными (Apache Kafka), определенных типах аналитики (R и Python для статистического анализа и машинного обучения). Учитывая важность этого вопроса и сложность выбора, необходимо заранее провести детальное изучение всех возможных инструментов и технологий, их функционала, стоимости, требований к обучению персонала и поддержке, а также рассмотреть возможность их интеграции с существующей IT-инфраструктурой компании. Это поможет выбрать наиболее подходящие инструменты, обеспечит успешное выполнение проекта, высокое качество и эффективность работы с аналитическими данными.

Так как АСУНО состоит из множества микросервисов и также имеются внешние системы, которые решают частные прикладные задачи, то в первую очередь необходимо найти такой механизм, помогающий централизовать и унифицировать сбор данных из этих всевозможных сервисов [1]. Так как разрабатываемый программный модуль (Потребитель данных) будет обрабатывать большое количество данных и в будущем при развитии системы количество данных будет расти и часть информации будет поступать в реальном времени, то архитектура приложения будет строиться на основе потоковой передачи данных. В связи с этим в качестве посредника между модулями систем анализа и потребителем будет использоваться брокер сообщений. Схема передачи данных представлена на рисунке 1.

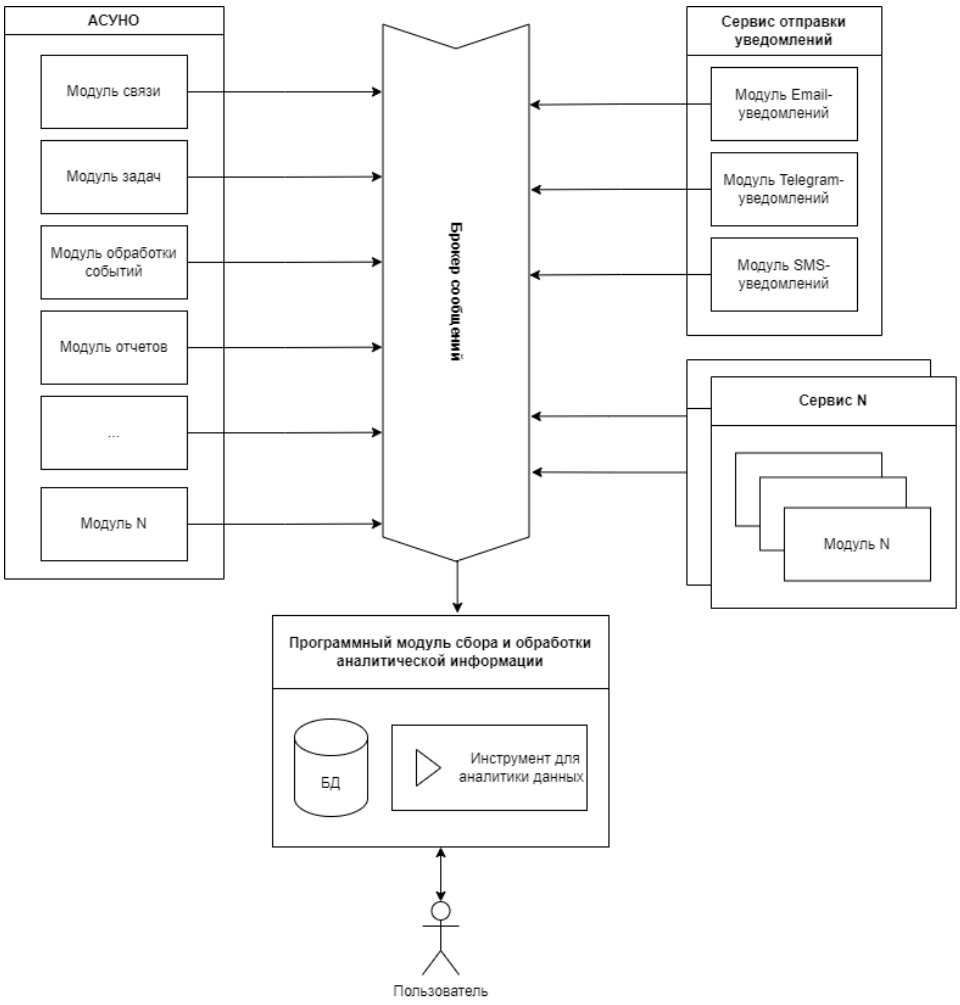

Рис. 1. Схема передачи данных в программный модуль сбора и обработки аналитической информации

Таким образом системы и сервисы, которые интересны для анализа могут в унифицированном формате передавать данные в брокер сообщений. Программный модуль сбора и обработки данных будет осуществлять обработку сообщений от брокера, производить при необходимости модификацию данных, сохранение данных в базу данных. Инструмент для аналитики данных будет производить анализ и вывод по определенным критериям информации.

Прежде чем приступить к разработке, необходимо определить, какие технологии будут использоваться. Разработку такой системы можно разбить на четыре части: выбор брокера сообщений, разработка потребителя данных, выбор или разработка инструмента для аналитики данных, доработка анализируемых систем и сервисов по потоковой передаче данных и метрик.

Выбор брокера сообщений при построении системы важен, потому что он отвечает за передачу информации между компонентами системы. Брокер сообщений обеспечивает эффективную коммуникацию между различными узлами сети, гарантируя доставку сообщений. Благодаря этому, система работает стабильно и эффективно, повышается общая производительность и управляемость. Правильный выбор брокера сообщений является важным шагом при проектировании и разработке системы.

На сегодняшний день на рынке представлено огромное количество различных брокеров сообщений. Например, Apache Kafka, RabbitMQ, NATS (NATS Messaging System), ActiveMQ, Redis Pub/Sub.

Так как в нашем проекте предполагается большой объем сообщений и АСУНО представляет собой масштабную систему, использующую множество микросервисов, то выбор остановим на Apache Kafka.

В задачи потребителя данных входят получение и обработка сообщений от брокера, сохранение данных в базу данных. Для такой задачи подойдут множество языков программирования, такие как java, python, go, C++.

Каждый из этих языков программирования имеет свои сильные стороны и экосистему, которые могут быть использованы для эффективной реализации потребителя данных. Однако, выбор технологического стека должен учитывать не только технические аспекты, но и контекст проекта и существующую инфраструктуру. В данном случае, если АСУНО и большинство сервисов уже разработаны на языке Java [3] с применением фреймворков – Spring [4] и Hibernate, то выбор этого же стека технологий.

Java и Spring хорошо зарекомендовали себя в создании масштабируемых и высокопроизводительных приложений. Эти технологии предоставляют механизмы для эффективной обработки больших объемов данных и обеспечения высокой доступности сервисов. Таким образом, выбор Java и стека Spring/Hibernate для реализации потребителя данных в контексте АСУНО является обоснованным и согласуется с существующей технологической экосистемой проекта.

При создании этой системы были приняты во внимание многие системы управления базами данных (СУБД), а именно PostgreSQL, MySQL, MariaDB и Firebase. Они будут подробно рассмотрены ниже, начиная с MySQL.

После проведения анализа было принято решение использовать СУБД PostgreSQL [5], причины которого будут представлены ниже. В сравнении с MySQL и MariaDB, которые имеют ограничение размера строки в 65 535 байт, PostgreSQL предлагает возможность хранить табличные данные в нескольких файлах меньшего размера. Вследствие этого, хотя ограничение размера файла операционной системы может быть обойдено, важно учитывать, что излишнее количество файлов может негативно повлиять на общую производительность. MySQL и MariaDB, однако, поддерживают большее количество столбцов в таблице (до 4096, в зависимости от типа данных) и большие размеры отдельных таблиц, чем PostgreSQL. В редких случаях возможно необходимо превышать существующие ограничения в PostgreSQL.

Делая вывод на основании имеющихся данных, следует отметить, что PostgreSQL представляет собой мощный инструмент с обширными возможностями. Применяя объектно-реляционную модель, данная система поддерживает сложные структуры и разнообразные типы данных, как встроенные, так и определенные пользователем. PostgreSQL способен обрабатывать большие объемы информации, обладает высокой целостностью данных и надежностью. Не все функции хранения данных, упомянутые выше, могут быть необходимы в каждом конкретном случае, однако гибкость и доступность широкого спектра функций под рукой представляют значительное преимущество. В связи с вышеперечисленным, выбор PostgreSQL в качестве системы управления базами данных является обоснованным.

В качестве инструмента для аналитики данных возможно применение готовых решений или разработка своего в случае если готовые имеют недостатки. В качестве готовых инструментов можно рассмотреть Яндекс Datalens, Microsoft Power BI, Apache Superset [11].

Яндекс Datalens, Microsoft Power BI и Apache Superset это мощные инструменты аналитики данных, каждый из которых имеет свои сильные стороны и особенности.

Среди данных продуктов для нашей задачи особняком выделяется Apache Superset [10] за счет своей гибкости и бесплатности. Остановимся на данном продукте подробнее.

Apache Superset [2] - это передовое, масштабируемое и визуально привлекательное платформо-независимое веб-приложение для анализа данных на уровне предприятия, созданное Apache Software Foundation. Этот инструмент нацелен на обеспечение максимально детального и точного анализа данных. Apache Superset обеспечивает совместимость со всеми основными SQL-запросами и большинством баз данных, работающих с SQL. Открытый код приложения поощряет сообщество разработчиков к постоянному совершенствованию и обновлению функционала приложения. Integration Superset позволяет пользователю подключиться к множеству источников данных, включая все главные SQL и NoSQL базы данных.

Superset предлагает обширный набор средств для визуализации данных, позволяя пользователям представить информацию в самых разных форматах - от стандартных графиков и диаграмм до геопространственных карт и графиков временных рядов. Интерфейс Superset характеризуется интуитивностью в использовании, предлагая функции "drag and drop" для создания графиков и панелей мониторинга.

Apache Superset подчеркивает свою гибкость в настройке, позволяя определить интерфейс и функциональность в соответствии с уникальными нуждами ваших бизнес-задач.

Apache Superset обеспечивает расширенные функции безопасности, включая ролевой контроль доступа, аутентификацию LDAP, поддержку OAuth и другие. Однако, стоит отметить, что Apache Superset может требовать определенной технической подготовки и времени для настройки и управления, но так как данные знания имеются, то это не является проблемой. В связи с

этим останавливаем выбор на Apache Superset. Пример отчета в системе Apache Superset представлен на рисунке 2.

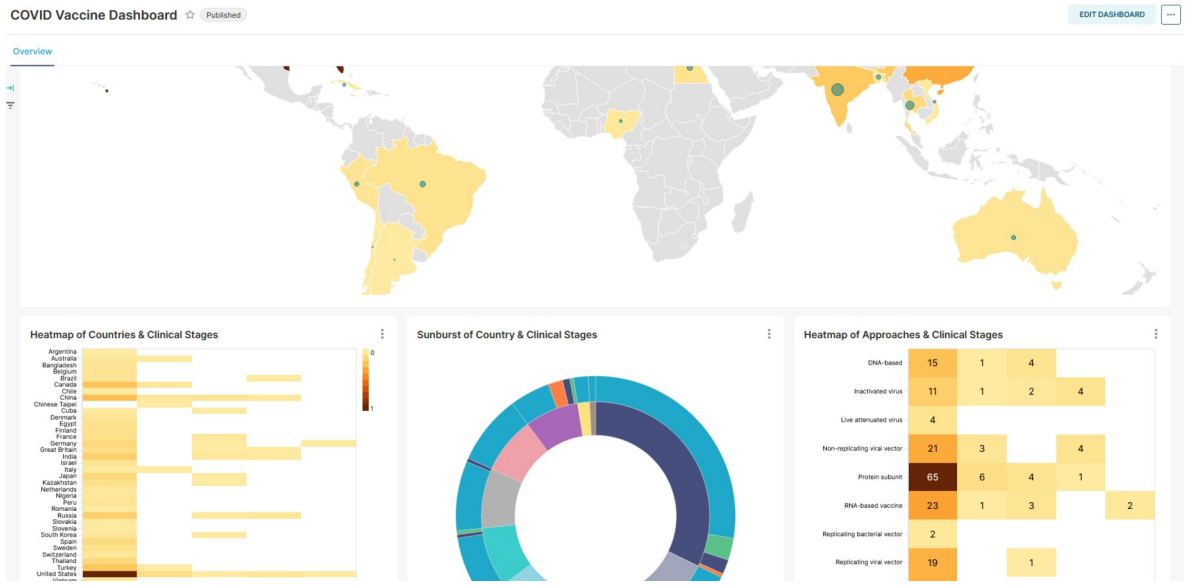

Рис. 2. Пример аналитического отчета в системе Apache Superset

Потоковая передача данных будет проводиться непосредственно из сервисов АСУНО, поэтому будет предстоять доработка в части передачи информации в брокер сообщений.

Рассматриваемая АСУНО разработана на современном кроссплатформенном языке программирования Java, также и практически все сервисы, взаимодействующие с АСУНО написаны на Java. В связи с этим все доработки анализируемых систем и сервисов по потоковой передаче данных и метрик также будут вестись на Java.

Кроме того, выбранный брокер сообщений Apache Kafka, с которым необходимо будет взаимодействовать также разработан на языке Java, что предоставляет взаимосовместимость и надежную производительность при использовании Java-клиентов. Apache Kafka хорошо интегрируется с экосистемой обработки данных на основе Java и обеспечивает надежную и согласованную обработку данных в распределенном режиме. Java, как статически типизированный язык, позволяет разработчикам определить явно типы данных и вносить строгость в обработку данных, что повышает надежность и прозрачность приложений на основе Kafka. Java предлагает

мощную многопоточность и широкий спектр библиотек и фреймворков, включая Spring, которые помогают разработчикам строить надежные и масштабируемые приложения с использованием Kafka. Spring Kafka предоставляет высокоуровневые абстракции, упрощающие работу с Kafka и обеспечивающие хорошую интеграцию с остальной экосистемой Spring. Помимо этого, Java Virtual Machine (JVM) обеспечивает управление памятью и автоматическую сборку мусора, что упрощает разработку приложений и увеличивает их надежность. Исходя из этих причин, выбор Java является адекватным и целесообразным подходом.

В данной статье описан процесс выбора инструментальных средств в конкретной задаче по разработке программного модуля для сбора и обработки аналитической информации по эксплуатации АСУНО. В статье изначально описана архитектура решения и выделены 4 части этой архитектуры для каждой из которых необходимо определить инструментальные средства. В ходе статьи было определено, что для данной задачи наиболее подходят для брокера сообщений – Apache Kafka, для потребителя данных – Java приложение с применением Spring, Hibernate, для СУБД – PostgreSQL, для трансляции данных – доработка приложений сервисов на Java. Выбранные инструментальные средства предоставят разработчикам гибкость и возможность масштабировать решение в дальнейшем наращивая объем данных и метрик, участвующих в аналитике.

СПИСОК ЛИТЕРАТУРЫ

1. Черников, В. С. Методика мониторинга качества связи в автоматизированных системах управления распределёнными объектами на примере автоматизированной системы управления наружным освещением / В. С. Черников // Аллея науки. – 2024. – Т. 1, № 2(89).

2. Superset.apache.org [Электронный ресурс]. - URL: https://superset.apache.org/docs/intro/. - (Дата обращения: 29.03.24).

3. Эккель, Б. Философия Java : Пер. с англ. / Б. Эккель ; Брюс Эккель. – 3. изд.. – М. [и др.] : Питер, 2003. – 970 с. – (Серия Библиотека программиста). – ISBN 5-88782-105-1.

4. Гутьеррес Фелипе. Spring Boot 2. Лучшие практики для профессионалов. Рук-во пользователя. / Под ред. Н. Гринчик. – Питер, 2020. – 464 с. – ISBN 978-5-4461-1587-7.

5. Obe, R. PostgreSQL: Up and Running: A Practical Guide to the Advanced Open Source Database/R. Obe, L.Hsu. – [Б.м]: O'Reilly Media, 2017. – 312 с.