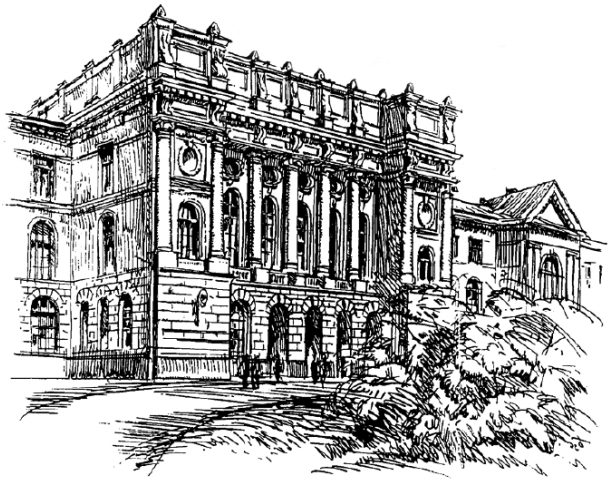
Министерство науки и высшего образования Российской Федерации

САНКТ-ПЕТЕРБУРГСКИЙ
ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО

СОВРЕМЕННЫЕ ТЕХНОЛОГИИ В ТЕОРИИ И ПРАКТИКЕ ПРОГРАММИРОВАНИЯ

Сборник материалов
научно-практической конференции студентов,
аспирантов и молодых ученых

26–27 апреля 2023 года



ПОЛИТЕХ-ПРЕСС
Санкт-Петербургский
политехнический университет
Петра Великого

Санкт-Петербург
2023

УДК 004
ББК 32.973
С56

Современные технологии в теории и практике программирования : сборник материалов научно-практической конференции студентов, аспирантов и молодых ученых, 26–27 апреля 2023 г. – СПб. : ПОЛИТЕХ-ПРЕСС, 2023. – 267 с.

В сборнике публикуются материалы докладов студентов, аспирантов и молодых ученых, представленных на научно-практической конференции, проводимой Санкт-Петербургским политехническим университетом Петра Великого и организованной Институтом компьютерных наук и технологий при поддержке группы компаний YADRO. Доклады отражают уровень подготовленности участников конференции в области применения современных средств и технологий разработки программного обеспечения.

Представляет интерес для специалистов в области информационных технологий, методов разработки программных проектов различного назначения, систем и средств автоматизации инженерного проектирования, а также для учащихся и работников системы высшего образования.

ISBN 978-5-7422-8099-6

Приветствие от группы компаний YADRO

Уважаемые участники конференции
«Современные технологии в теории и практике программирования»!

В этом году YADRO впервые выступает партнером конференции «Современные технологии в теории и практике программирования».

Сотрудничество нашей группы компаний и Политехнического университета только начинается, при этом основывается на уже налаженном экспертном взаимодействии: многие наши сотрудники ранее выступали наставниками для политехников, курировали совместные учебные и научные проекты. Многие бывшие участники данной конференции стали инженерами и руководителями в YADRO. Мы высоко оцениваем уровень подготовки выпускников Политеха, качество преподавания профильных дисциплин из области разработки ПО, а также создания систем на кристалле.

Целью данной конференции, в том числе и секции YADRO является, в первую очередь, поддержка молодых ученых, их научных исследований в сфере программной инженерии и обсуждение передовых технологических трендов в индустрии. Компаниям, занимающимся разработкой сложных комплексных технологических продуктов, сотрудничество с университетами жизненно необходимо для дальнейшего устойчивого развития.

Желаю вам успешной работы на конференции, профессионального роста и творческих побед!

С уважением,

Евгений Викторович Максимов

Директор по развитию экосистемы и образовательных инициатив YADRO

Тезисы докладов конкурса-конференции

Секция «Программная инженерия: приложения, продукты и системы»

УДК 004.416

Г.Р. Абисалов (4 курс бакалавриата),
В.В. Амосов, к.т.н., доцент

РАЗРАБОТКА ANDROID ПРИЛОЖЕНИЯ СЕРВИСА ДОСТАВКИ И СБОРА ПРОДУКТОВ

Целью работы является создание сервиса сбора, заказа и доставки продуктов и товаров с полок магазинов, создание работы платформы в качестве посредника между магазинами и клиентами – у нее нет своего склада, но есть курьеры.

В последнее время количество сервисов по доставке продуктов, одежды, даже услуг увеличилось многократно. Люди стали реже посещать торговые центры, выходить из дома и бродить по магазинам в поисках одного-двух нужных продуктов, незаметно сметая полку за полкой. Пользоваться такими сервисами очень удобно, поскольку, сидя дома, на работе или в любом другом месте, с помощью приложения можно выбрать интересующие нас продукты и доставить их куда захотим. Это существенно сокращает время на прогулки по магазинам, ведь специально обученные сборщики качественно выберут товары, а курьеры привезут их в выбранное нами время и место.

В процессе работы алгоритма происходит следующее:

При входе в приложение пользователь (клиент, сборщик, курьер, супервайзер) может либо зарегистрироваться в системе, либо войти в нее.

После входа в систему клиент может создать заказ, выбрав из предложенного меню необходимые для него товары и кладя из в корзину. В процессе он может менять количество товара, адрес доставки и время, убирать товары из корзины. После оформления заказа вся информация отправляется в БД. В личном кабинете пользователь может посмотреть информацию о своих заказах и изменить личные данные.

Курьер при входе в систему может увидеть информацию о своих текущих рейсах (список заказов, куда он должен за один заезд отвезти товары клиенту) и о конкретном заказе рейса, а также его завершить. Доставленные заказы хранятся в БД, тем самым после завершения рейса эти заказы сохраняются в таблице.

Сборщик, войдя в систему, может увидеть все текущие заказы от клиентов, узнать какие товары там находятся, а также время, адрес, номер и данные клиента. Собрав заказ, сборщик завершает сборку, а заказ отправляется в таблицу “Собранные”.

У супервайзера больше функционала, поскольку он отвечает за регистрацию новых сотрудников в системе, назначает смену работникам, а также может уволить подчиненных, создает рейс для курьера, добавляя туда n-ое количество заказов и видит абсолютно всю информацию о каждом заказе и его статусе.

Для написания программы сервиса доставки продуктов был выбран язык Java с JDK 17.0.1.

Так как язык Java, то и среда разработки выбрана Intellij IDEA, это одна из самых популярных сред разработки для приложений на данном языке. Данная IDE удобна, в ней есть всё необходимое для комфортной и продуктивной работы.

Также для сохранения версионности разрабатываемого кода была выбрана система контроля версий git, а именно платформа gitlab. Git сейчас повсеместно распространён и является удобным инструментом для отслеживания и сохранений истории изменений в коде. В случае ошибки всегда можно вернуться на более раннюю версию кода. Также можно просмотреть историю изменений в виде списка или графика.

Хранение данных можно реализовать с помощью простой и удобной СУБД SQLite, поскольку она хорошо работает с Java и ничем не уступает аналогам.

ЛИТЕРАТУРА

1. Эккель Б. Философия Java. Библиотека программиста. 4-е изд. СПб.: Питер, 2011
2. Кей Хорстманн, Гари Корнелл "Java. Библиотека профессионала. Том 1". 11-е издание
3. Гриффитс Дэвид, Гриффитс Дон - Head First. Программирование для Android. 2-е издание (Head First O'Reilly)

УДК 004.032.26

Р. Ж. Айдаров (4 курс бакалавриата),
Н. В. Воинов, к.т.н., доцент,
В. С. Тутыгин, к.т.н., доцент

РАЗРАБОТКА МОДУЛЯ АВТОМАТИЗИРОВАННОЙ ОБРАБОТКИ ФОТОГРАФИЙ В СИСТЕМЕ ПОДДЕРЖКИ ФОТОПРОТОКОЛОВ

Ортодонты используют фотопротоколы, чтобы документировать ход лечения своих пациентов. Эти фотопротоколы обычно включают стандартизированные фотографии лица, зубов и челюстей пациента, сделанные под разными углами, которые используются для оценки эффективности лечения и для связи с другими медицинскими работниками, участвующими в уходе за пациентом.

Актуальной проблемой является автоматизация обработки фотографий для экономии времени и сокращение ошибок, улучшения качества диагностики, улучшения взаимодействия между пациентами и врачами.

В решении данной проблемы может помочь применение нейронных сетей благодаря высокой точности распознавания изображений на фото, автоматизации процесса создания фотопротокола и адаптивности к изменениям (нейронные сети могут быть обучены на большом объеме данных и могут адаптироваться к изменениям в данных и условиях съемки).

Таким образом, целью работы является разработка модуля автоматизированной обработки фотографий в системе поддержки фотопротоколов на основе методов машинного обучения. Для достижения поставленной цели необходимо решение следующих задач:

1. Изучение различных методов и моделей машинного обучения применительно к обработке фотографий.
2. Формирование требований к разрабатываемому модулю.
3. Реализация модели для обнаружения ориентиров лица.
4. Реализация модели для классификации изображений.
5. Тестирование моделей.

Одним из основных компонентов является модель классификации изображений, которая будет основана на глубокой нейронной сети (Deep neural network) с помощью глубокого обучения (Deep learning) [1]. Эта модель обучается распознавать и классифицировать объекты на изображениях, что позволяет автоматически анализировать фотографии, полученные при диагностике пациента в ортодонтии.

Модель классификации изображений будет основана на сверточной нейронной сети (Convolutional Neural Networks, CNN) [2,3], которая является наиболее эффективной для задач анализа изображений. Они обучаются на большом количестве размеченных данных, чтобы научиться распознавать образцы на изображениях.

Вторым основным компонентом является модель нахождения ориентиров лица (Facial Landmark Detection Model) [4], которая также будет основана на сверточной нейронной сети, которая обучается на большом количестве размеченных изображений, чтобы находить ключевые точки на лице человека, такие как глаза, нос, рот, уши и другие.

После обучения модель сможет автоматически определять положение и форму лица на изображении, а также локализовать ключевые точки, что позволяет более точно измерять и анализировать данные о состоянии зубов и челюстей пациента.

Для реализации применяется фреймворк PyTorch [5] на языке Python [6]. PyTorch предоставляет удобный и гибкий интерфейс для создания и настройки сложных моделей нейронных сетей, а также обладает мощным инструментарием для обработки изображений [7].

ЛИТЕРАТУРА

1. Aston Zhang, Zachary C. Lipton, Mu Li, and Alexander J. Smola "Dive into Deep Learning" [Электронный ресурс] Режим доступа: https://d2l.ai/index.html
2. Szymon Płotka, Tomasz Włodarczyk, Ryszard Szczerba, Przemysław Rokita, Patrycja Bartkowska, Oskar Komisarek, Artur Matthews-Brzozowski, Tomasz Trzciński “Convolutional Neural Networks in Orthodontics: a review” [Электронный ресурс] Режим доступа: https://arxiv.org/pdf/2104.08886.pdf
3. Зайцев, В. А. Реализация программного модуля для распознавания городских сооружений на основе сверточной нейронной сети / В. А. Зайцев, Н. В. Воинов // Современные технологии в теории и практике программирования : сборник материалов конференции, Санкт-Петербург, 19 апреля 2019 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2019. – С. 183-184. – EDN BTPQOS.
4. Kostiantyn Khabarlak, Larysa Koriashkina “Fast Facial Landmark Detection and Applications: A Survey” [Электронный ресурс] Режим доступа: https://arxiv.org/pdf/2101.10808.pdf
5. PyTorch documentation [Электронный ресурс] Режим доступа: https://pytorch.org/docs/stable/index.html
6. Python documentation [Электронный ресурс] Режим доступа: https://docs.python.org/3.10/
7. Сараджишвили, С. Э. Введение в обработку изображений на языке Python / С. Э. Сараджишвили, В. В. Леонтьев, Н. В. Воинов. – Санкт-Петербург : Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2020. – 36 с. – ISBN 978-5-7422-6955-7. – DOI 10.18720/SPBPU/2/id20-76. – EDN GEJJDU.

УДК 004.413

А. Д. Алексеев, В. Кудравец (2 курс бакалавриата),
Д. С. Эйзенах (2 курс аспирантуры),
А.П. Маслаков, ст. преподаватель

РАЗРАБОТКА КОНВЕЙЕРА СБОРКИ И НЕПРЕРЫВНОЙ ИНТЕГРАЦИИ ДЛЯ СЕРВИСА ПОМОЩИ ПРИЮТАМ И ДОМАШНИМ ЖИВОТНЫМ “ISHELT”

В современном мире, где скорость и качество разработки программного обеспечения играют ключевую роль DevOps-подход [1] становятся все более популярным. Он позволяет автоматизировать процессы разработки, тестирования и доставки изменений в сложных, многокомпонентных продуктах. Данная статья описывает опыт настройки DevOps-инфраструктуры на проекте iShelt. В статье будут рассмотрены следующие аспекты:

- Создание репозитория, где могли бы без помех друг другу работать несколько команд разработчиков;
- Настроить процесс непрерывной интеграции и непрерывной поставки (CI/CD) для бесперебойной сборки, тестирования и доставки приложения на сервер, где оно всегда было бы доступно.

Все операции, связанные с разработкой проекта, проводятся в системе управления репозиториями и программным кодом GitLab, где была создана группа репозиториев для разделения работы между командами, ответственными за графический интерфейс и серверную часть приложения.

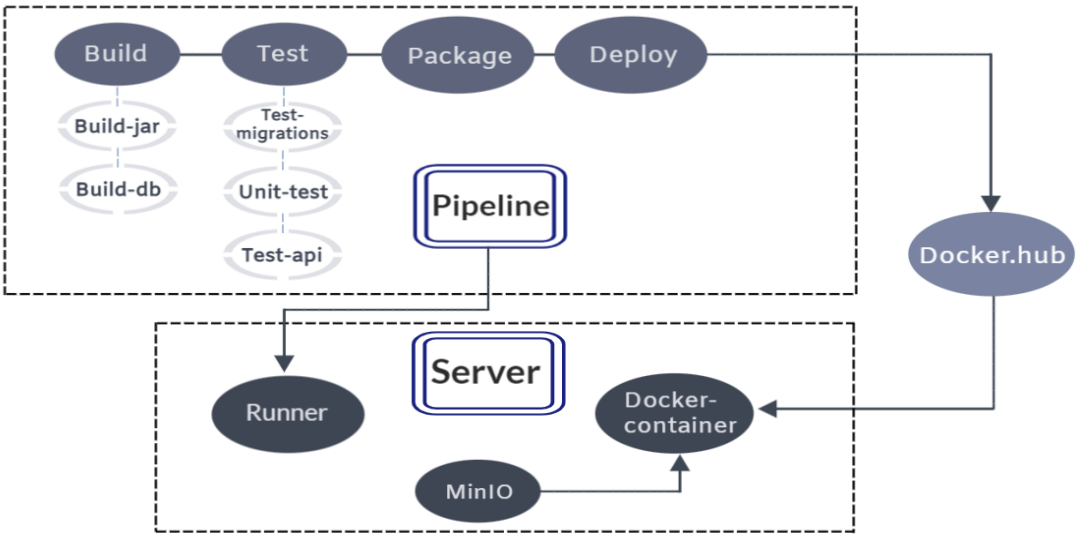


Рисунок 1 – Схема работы конвейера

Автоматизация сборки и тестирования проекта решается путем применения конвейера, изображен на Рис. 1, предоставляемого GitLab через инструмент gitlab-ci [2].

Для того, чтобы отказаться от использования коммерческих функций платформы GitLab было решено установить свой GitLab Runner на собственный сервер. В качестве среды для выполнения конвейера был выбран Docker, так как для разных задач конвейера можно будет выбрать разные docker-образы. Для сборки образов внутри конвейера использовалась технологией Docker-in-Docker [3], предоставляемой образом docker:dind. Но тут возникла проблема, что по умолчанию у сервиса Docker не открыты нужные внешним программам порты, из-за этого при создании нового образа внутри контейнера возникала ошибка соединения. Поэтому нужно было прописать в настройках сервиса Docker, какие порты ему нужно слушать. Для этого используется файл daemon.json, по умолчанию лежащий по пути /etc/docker. В нем прописывается нужны порт: {"hosts": ["tcp://0.0.0.0:2375", "unix:///var/run/docker.sock"]}. Для того, чтобы при запуске изменения вступили в силу, нужно добавить файл: /etc/systemd/system/docker.service.d/override.conf и прописать в нём путь к файлу исполнения процесса Docker. После перезапуска процесса также важно заново создать ссылку на исполняемый файл, поскольку по какой-то причине после перезагрузки она исчезает.

Настроив сервис для выполнения конвейера, следует настроить сам конвейер. Чтобы не перегружать сервер, конвейер запускается только при запросе на слияние с основной веткой. Для каждой отдельной задачи запускается отдельный контейнер. Для сборки запускается контейнер на основе официального образа gradle, внутри которого выполняется сборка всех элементов приложения в один jar архив [4]. После этот архив передается как артефакт в контекст конвейера и начинается стадия тестирования, происходящая внутри контейнера openjdk. Сначала проверяется исполнение юнит-тестов приложения. Далее проверяются миграции в базу данных, для чего параллельно запускается контейнер PostgreSQL. После дополнительно запускается контейнер, содержащий программу Newman от Postman для тестирования корректной работы API приложения. После успешного прохождения тестирования, на основе jar-архива собирается образ и посылается в репозиторий docker.hub, откуда на стадии развёртывания он выгружается на сервер и сразу там и запускается.

В заключение можно сказать, что настройка CI/CD упростило работу команды, добавило автоматизацию тестирования, а также сборку и доставку готового программного обеспечения на стенд.

ЛИТЕРАТУРА

1. DevOps / C. Ebert [и др.] / IEEE Software. 2016. С. 94-100
2. GitLab. (n.d.). GitLab Documentation. https://docs.gitlab.com/
3. Nemeth, E., Snyder, G., Hein, T., Whaley, B., & Mackin, D. (2017). Unix and Linux System Administration Handbook: The comprehensive guide to managing your systems. Addison-Wesley Professional.
4. Chin, S. (2016). DevOps Tools for Java Developers: A practical guide to building, testing, and deploying Java applications using Jenkins, Travis CI, and Docker. Apress.

УДК 004.453

Е.О. Алферова (4 курс бакалавриата),
О.В. Прокофьев, ст. преподаватель

АВТОМАТИЗАЦИЯ РАБОТЫ ТЕПЛИЦЫ С ИСПОЛЬЗОВАНИЕМ ТЕХНОЛОГИИ INTERNET OF THINGS (BACKEND)

В мире, где все становится все более и более связанным с интернетом, технология Internet of Things (IoT) [1] получает все большее значение в различных сферах жизни, включая сельское хозяйство. Применение IoT для автоматизации процессов в тепличном хозяйстве может значительно повысить эффективность производства и уменьшить трудозатраты. В данной работе мы рассмотрим, как технология IoT может быть применена для автоматизации теплицы, а также о том, какие преимущества она может предоставить фермерам и садоводам.

Для автоматизации работы теплицы использованы датчики, которые измеряют определенные параметры, такие как температура, влажность, давление и освещенность. Исполнительные устройства, такие как вентиляторы, обогреватели и системы полива, регулируют все эти параметры, чтобы обеспечить оптимальные условия для роста растений.

С помощью IoT все эти данные передаются на устройство управления, где они обрабатываются и анализируются. Это позволяет нам отслеживать все параметры работы теплицы в режиме реального времени и принимать необходимые меры для поддержания оптимальных условий роста растений или делегировать это нашей программе.

Благодаря этой системе управления, можно значительно снизить риск возникновения различных проблем, таких как перегрев или переохлаждение теплицы, а также уменьшить расходы на ресурсы и трудозатраты, что в конечном итоге приведет к повышению эффективности работы теплицы без использования человеческого труда.

Существует несколько готовых решений для автоматизации систем умного дома, таких как NodeRed [4] и HomeAssistant [3], которые предоставляют инструменты для создания графических интерфейсов и управления устройствами. Однако, эти системы содержат много ненужного для тепличного хозяйства, а также не всегда гарантируют полную совместимость с требуемыми датчиками и исполнительными устройствами [2].

В связи с этим, для нашего проекта мы решили разработать свою собственную систему управления, которая будет адаптирована к специфическим требованиям тепличного хозяйства и позволит нам полностью контролировать все процессы внутри теплицы. Такой подход не только позволил нам уменьшить количество ненужных функций и сделать систему более гибкой, но и дал возможность полностью интегрировать все наши датчики и исполнительные устройства в единую систему управления.

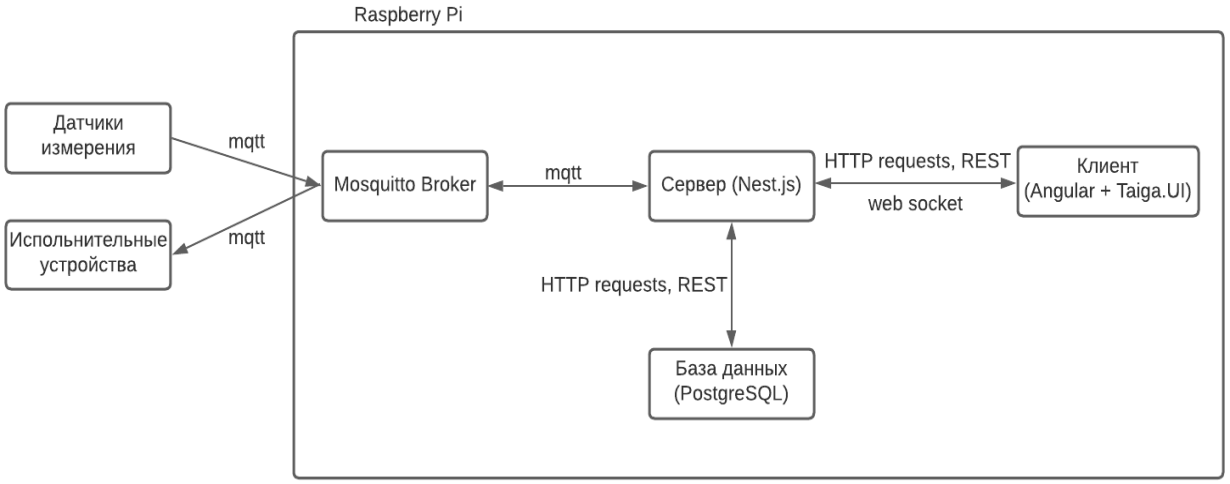


Рисунок 1 – Архитектура системы

Для написания веб-сервиса использована монолитная архитектура [5], представляющая собой backend и frontend в одном проекте. В данном случае нам очень важна простота развертывания и тестирования проекта, а также единая база кода для избегания повторного написания одних и тех же API. На Рис. 1 представлена схема взаимодействия всех элементов системы.

Для хранения данных использована СУБД PostgreSQL 11 [6]. Postres в отличие от других систем управления базами данных обеспечивает удобное хранение разных типов сетевых адресов, в том числе MAC-адресов, что очень важно в нашем проекте, так как нам надо хранить MAC-адрес каждого датчика. Так же одним из самых главных преимуществ Postgres я считаю возможность хранения многомерных массивов и поддержку JSON.

Взаимодействие с базой данных и принятие решений на основе получаемых данных реализовано на Node.js [7]. Так, наша база кода становится удобочитаема, исходя из того, что frontend написан на языке JavaScript.

Обмен информацией с датчиками настраивается по MQTT протоколу с использованием инструмента Eclipse Mosquitto [8]. MQTT – это одно из самых распространенных и логичных решений, когда дело касается Internet of Things.

Наш проект по автоматизации теплицы на базе IoT представляет собой эффективный инструмент для повышения производительности и качества урожая в тепличном хозяйстве. Он позволяет сократить затраты на ручную работу и обеспечивает точный контроль над всеми процессами, что в свою очередь повышает эффективность всей конструкции.

ЛИТЕРАТУРА

1. Что такое IoT? [Электронный ресурс] Режим доступа: https://aws.amazon.com/ru/what-is/iot/
2. Home Assistant + Node-RED = Простая автоматизация [Электронный ресурс] Режим доступа: https://sprut.ai/article/homeassistant-node-red-prostaya-avtomatizaciya
3. Home Assistant [Электронный ресурс] Режим доступа: https://community.home-assistant.io/
4. Node-RED [Электронный ресурс] Режим доступа: https://nodered.org/
5. Сравнение микросервисной и монолитной архитектур [Электронный ресурс] Режим доступа: https://www.atlassian.com/ru/microservices/microservices-architecture/microservices-vs-monolith
6. PostgreSQL [Электронный ресурс] Режим доступа: https://www.postgresql.org/
7. Node.js [Электронный ресурс] Режим доступа: https://nodejs.org/en/
8. Eclipse Mosquitto [Электронный ресурс] Режим доступа: https://mosquitto.org/

УДК 004.415.25

С. Т. Алиев (4 курс бакалавриата),
Н. В. Воинов, к.т.н., доцент,
В. С. Тутыгин, к.т.н., доцент

РАЗРАБОТКА СИСТЕМЫ ПОДДЕРЖКИ ФОТОПРОТОКОЛОВ ДЛЯ ВРАЧЕЙ-СТОМАТОЛОГОВ

Фотопротокол - это подборка фотографий челюсти пациента с заметками от врача. Он помогает отслеживать прогресс во время исправления прикуса или коррекции положения зубов. Врач детально изучает состояние зубов и черты лица в целом, это особенно важно при комплексной реабилитации здоровья полости рта.

Существующие программные решения не позволяют создавать фотопротоколы, их функционал ограничен хранением и обработкой отдельных рентгеновских снимков, обычно врачи компонуют их вручную.

Большинство современных программных сервисов, которые облегчают и улучшают нашу жизнь, реализованы в виде приложений или сайтов [1-3]. Очень удобно, когда можно получить решение своей проблемы в одном месте и не распыляться между несколькими приложениями и проходить там разные этапы своей работы.

Целью данной работы является предоставление врачам среды для поддержки фотопротоколов, которая будет использоваться врачами-стоматологами и поможет вести пациентов и формировать фотопротоколы.

Для достижения данной цели необходимо решить следующие задачи:

- Обзор аналогов.
- Формулирование требований к собственному решению.
- Создание архитектуры.
- Реализация выбранного подхода.
- Тестирование полученного решения.

Архитектура разрабатываемой системы представлена на Рис. 1.

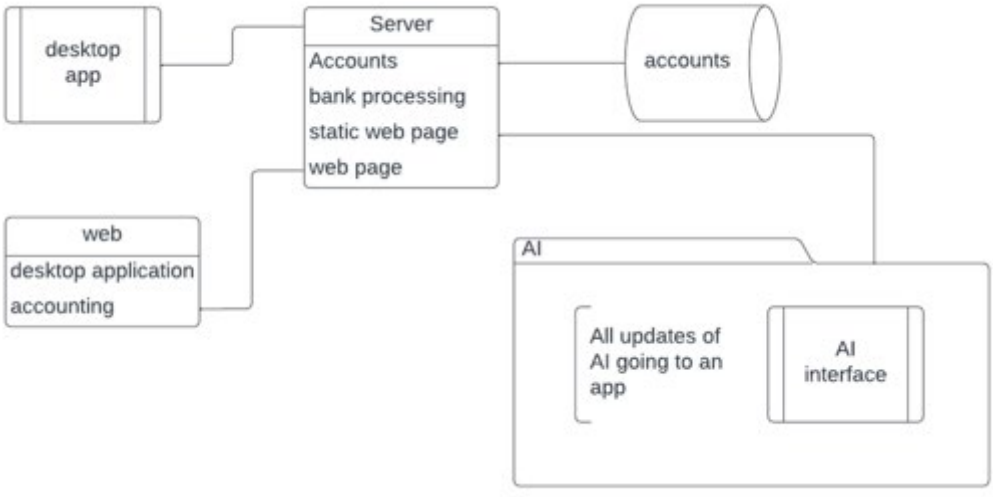


Рисунок 1 – Архитектура разрабатываемой системы

Взаимодействие с системой будет осуществляться через web-интерфейс, что даст доступ к своим данным в любом месте и в любое время, но у хранения медицинских данных есть некоторые ограничения, при которых данные не могут храниться у третьей стороны. Поэтому хранение фотографий и протоколов возможно только на компьютерах пользователей.

Electron [4-8] является фреймворком, который предоставляет кроссплатформенность, возможность обновлять некоторые аспекты приложения на лету, не заставляя пользователей скачивать обновления. В написании интерфейсной части были использованы React и Redux.

Серверная часть реализована с помощью фреймворка Express.js, в роли базы данных используется нереляционная база данных MongoDB, также для работы с базой данных используется Prisma ORM.

ЛИТЕРАТУРА

1. Смирнов, П. А. Клиент-серверное приложение для донорской службы / П. А. Смирнов, В. В. Малиновская, Н. В. Воинов // Современные технологии в теории и практике программирования : Сборник материалов конференции, Санкт-Петербург, 26 апреля 2022 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2022. – С. 44-46. – EDN XPPPRC.
2. Smirnov, P. A. A Web Application to Promote Blood Donation in Russia / P. A. Smirnov, V. V. Malinovskaya, N. V. Voinov // Proceedings of the Institute for System Programming of the RAS. – 2022. – Vol. 34, No. 2. – P. 179-190. – DOI 10.15514/ISPRAS-2022-34(2)-14. – EDN SGENXV.
3. Бойков, К. М. Разработка клиент-серверной архитекутры для сервиса по управлению интерактивными подписками / К. М. Бойков, О. С. Командина, Н. В. Воинов // Современные технологии в теории и практике программирования : сборник материалов конференции, Санкт-Петербург, 23 апреля 2020 года / Санкт-Петербургский политехнический университет Петра Великого; Dell Technologies; EPAM Systems. - Санкт-Петербург: ПОЛИТЕХ-ПРЕСС, 2020. - С. 113-114. - EDN DTYFAF.
4. Kredpattanakul, K., Limpiyakorn, Y. (2019). Transforming JavaScript-Based Web Application to Cross-Platform Desktop with Electron. In: Kim, K., Baek, N. (eds) Information Science and Applications 2018. ICISA 2018. Lecture Notes in Electrical Engineering, vol 514. Springer, Singapore. https://doi.org/10.1007/978-981-13-1056-0_56
5. M. Jasim, “Building cross-platform desktop applications with electron,” Packt Publishing Ltd, 2017
6. Toman, Zinah Hussein et al. “An In-Depth Comparison Of Software Frameworks For Developing Desktop Applications Using Web Technologies.” *Journal of Southwest Jiaotong University* (2019): n. pag.
7. A. Popov, J. Bilokin, T. Solianyk and K. Vasylchenko, "Development of the system to provide cross-browser compatibility of web application," *2018 IEEE 9th International Conference on Dependable Systems, Services and Technologies (DESSERT)*, Kyiv, UKraine, 2018, pp. 117-122, doi: 10.1109/DESSERT.2018.8409111.
8. How to handle blockers in Node.js ? [Электронный ресурс] Режим доступа: https://levelup.gitconnected.com/how-to-handle-blockers-in-node-js-1966d0399703

УДК 004.738.05:004.451.55

В.А. Андрейченкова (1 курс магистратуры),
Т.С. Горавнева, к.т.н., доцент

РАЗРАБОТКА ПРИЛОЖЕНИЯ ДЛЯ СИСТЕМАТИЗАЦИИ ТЕМАТИЧЕСКИХ ЖУРНАЛОВ

Целью работы является разработка Интернет-проекта для использования научными работниками (научной организацией) в поиске журналов, подходящих к опубликованию научной статьи определенной тематики, а также наглядное представление информации об индексированных тематических журналах и о публикациях научных сотрудников в виде динамического одностраничного WEB-приложения.

Вся данные о тематических научных журналах и о публикациях научных сотрудников хранится в XML-базе данных, информация из которой выводится в приложение с помощью языка XSLT без необходимости перезагрузки всей страницы.

В состав базы данных входит следующая информация:

– о журналах:

- название журнала;
- категория (предметная область);

- страна;
- квартиль;
- индексация;
- ключевые слова;
- аннотация,

– о публикациях научных сотрудников:

- ФИО;
- факультет;
- кафедра;
- название журнала;
- название публикации;
- категория (предметная область).

Для работы с XML-базой данных был использован динамический скриптовый язык программирования JavaScript, а также технология AJAX, позволяющая изменять содержимое WEB-приложения, не перезагружая его путем динамического обращения к серверу посредством объекта XMLHttpRequest. Кроме того, был создан XSD-документ для структурного представления базы данных и XSL-документы для преобразования и манипулирования данными. Вместе с тем, для добавления информации о публикациях в базу данных был разработан парсер на языке программирования C# с использованием библиотеки AngleSharp на интегрированной среде разработки Microsoft Visual Studio.

Для отображения данных из каталога, а также для работы с созданным каталогом (организации поиска по заданным критериям фильтрации, поиск по ключевым фразам, возможность сортировки данных) было выполнено наглядное представление информации в одностраничном динамическом WEB-приложении.

В процессе работы были выполнены требования, предъявляемые к WEB-приложению:

– WEB-приложение должно предоставлять интерфейс для просмотра, фильтрации и поиска информации из базы данных;
– обновление частей WEB-приложения, полученных с помощью XSLT-преобразований, должно выполняется без перезагрузки WEB-страницы;
– отображения форм и результатов вывода по заданным критериям должно осуществляется в модальных окнах.

При запуске WEB-приложения отображается приветственная WEB-страница, в верхней части которой располагается раздел заголовка, в котором находится кнопка «Обновить каталог» и строка поиска. Ниже расположен вступительный текст и кнопка перехода к основной информации сайта (Рис. 1).

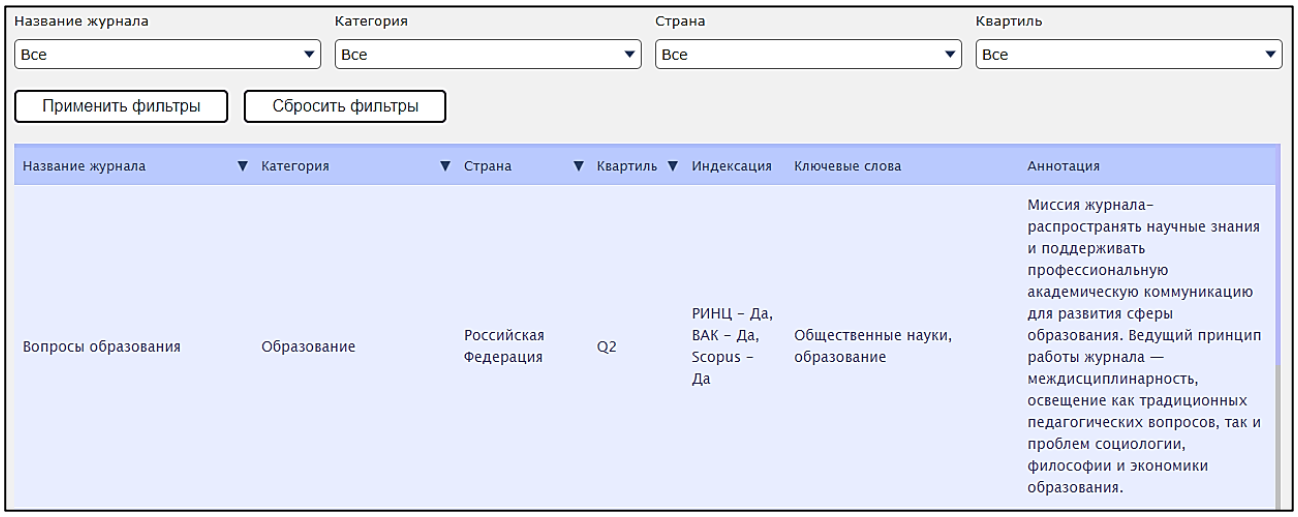


Рисунок 1 – Фрагмент модального окна с таблицей

Кнопка «Обновить каталог» позволяет пользователю послать запрос на обновление информации в каталоге данных, то есть запустить парсер. Строка поиска позволяет по ключевым фразам произвести поиск в базе данных. Если введённая фраза/слово обнаружена в базе данных, то информация на WEB-странице автоматически обновляется, отображая модальные окна в виде таблицы с информацией о том узле, в котором найдено данное вхождение. Кнопка перехода к основной информации сайта автоматически обновляет содержимое WEB-страницы, предоставляя доступ к модальным окнам и функциональным элементам фильтрации информации.

Для фильтрации выводимой информации нужно выбрать в выпадающем списке нужный критерий. После выбора всех необходимых критериев следует нажать на кнопку «Применить фильтры». После этого информация на WEB-странице автоматически обновится в соответствии с необходимой информацией. Кнопка «Сбросить фильтры» выполняет сброс выбранных критериев и при необходимости возвращает содержащуюся в модальных окнах информацию в состояние по умолчанию.

Для сортировки столбца таблицы по возрастанию или убыванию нужно щелкнуть по заголовку столбца таблицы необходимое количество раз для сортировки в разных направлениях.

Средства создания разработки:

– Язык программирования JavaScript;
– Язык гипертекстовой разметки HTML5;
– Каскадные таблицы стилей CSS3;
– Технология AJAX;
– База данных в форме XML-файла;
– Язык преобразования XML-документов XSLT;
– Язык программирования C#;
– .NET Framework и библиотека AngleSharp.

ЛИТЕРАТУРА

1. W3Schools Reference. JavaScript and HTML DOM Reference [URL]: https://www.w3schools.com/jsref/default.asp – (дата обращения: 10.03.2023).

2. MDN Web Docs. Document [URL]: https://developer.mozilla.org/ru/docs/Web/API/Document – (дата обращения: 10.03.2023)

УДК 004.42

А. А. Андрианов (4 курс бакалавриата),
Б. М. Медведев, к.т.н., доцент

ПОЗИЦИОНИРОВАНИЕ ПОЛЬЗОВАТЕЛЯ В ПОМЕЩЕНИЯХ С ИСПОЛЬЗОВАНИЕМ BLUETOOTH LOW ENERGY УСТРОЙСТВ

Системы позиционирования пользователя в помещениях сегодня применяются во многих сферах, таких как маркетинг, сбор аналитики, контроль персонала и безопасности, а также для навигации пользователей по торговым, промышленным или социальным площадям. Для решения задачи определения позиции используются различные подходы. Популярным способом поиска положения пользователя является применение спутниковых навигационных систем GPS или ГЛОНАСС, которые обеспечивают погрешность порядка 10–15 метров на открытых пространствах, однако их практически невозможно использовать в зданиях [1]. Еще одним подходом является использование станций операторов сотой связи [2]. Однако, большая погрешность измерений (более 200 метров) позволяет применить это решение только для грубого определения координат. Позиционирование на базе магнитного поля земли хоть и

является перспективным [3], однако напрямую зависит от магнитных полей, излучаемых устройствами в помещениях, подход сложен в реализации и обладает низкой точностью. Решения на базе Wi-Fi позволяют получить высокую точность вплоть до 2–3 метров [4], однако с появлением iOS 8 использование данного подхода является затруднительным, так как mac-адреса Apple-устройств постоянно меняются с целью предотвращения «рекламной» слежки.

Одним из перспективных и быстроразвивающихся направлений в навигации внутри помещений является использование Bluetooth Low Energy (BLE) [5, 6]. Из основных преимуществ данного подхода можно выделить распространенность поддержки BLE в большинстве современных устройств (смартфоны, планшеты и проч.), длительное время работы, высокая точность позиционирования, которая может достигать до 1.5–3 метров в зависимости от помещения и покрытия маяками [7]. При реализации навигации на базе BLE-маяков возникают сложности, связанные с необходимостью адаптировать математические модели распространения сигналов под разные помещения, а также с потребностью учитывать скорость движения пользователя для достижения малых значений погрешности позиционирования.

Целью работы является создание кроссплатформенных программных средств позиционирования пользователя на базе BLE-маяков.

Для достижения поставленной цели необходимо решение следующих задач:

1. Обзор существующих подходов в реализации систем позиционирования пользователя в помещении с использованием BLE-маяков.

2. Разработка алгоритмов позиционирования пользователя в помещении по мощности сигналов BLE-маяков.

3. Разработка программных средств для позиционирования пользователя в помещениях с использованием BLE-маяков.

Функциональная схема системы позиционирования приведена на Рисунке 1. Система позиционирования пользователя состоит из клиент-приложения, собирающего информацию о мощности сигнала маяков и производящего расчеты положения пользователя, хранилища данных о маяках и пользователях и API-сервера, являющегося провайдером данных между хранилищем и клиент-приложением.

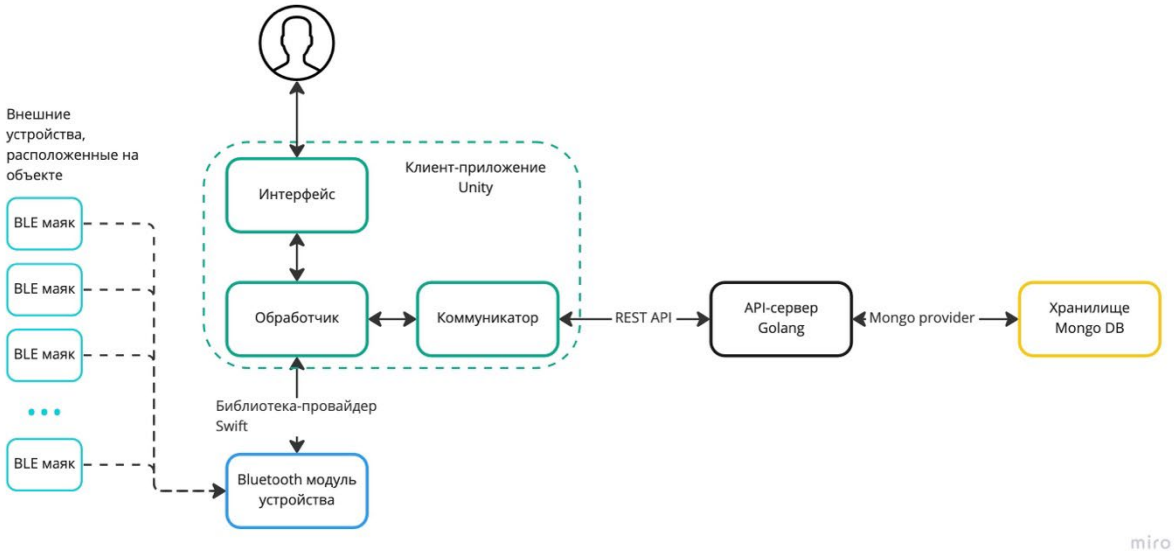


Рисунок 1 – Функциональная схема системы позиционирования

Для позиционирования пользователя измеряется мощность сигнала от BLE-маяков, выполняется фильтрация мощности каждого из них с использованием фильтров (фильтр Калман, бегущее среднее, медианный фильтр и др.), осуществляется выборка по наибольшей мощности сигнала лучших маяков, рассчитывается расстояние до маяка с использованием

математических моделей распространения сигнала в пространстве и трилатерация на основе полученных расстояний для определения положения пользователя.

Программные средства были разработаны с использованием редактора Unity на языке программирования C#. Для работы с внутренними модулями (Bluetooth модуль устройства) на базе iOS была реализована библиотека, написанная на языке Swift. Для хранения и работы с данными были использованы документно-ориентированная база данных MongoDB и API-сервер на языке Golang.

Разработанные программные средства могут быть использованы для создания системы навигации в аэропортах, в торговых и выставочных комплексах, а также в производственных помещениях.

ЛИТЕРАТУРА

1. Серапинас Б.Б. Глобальные системы позиционирования. - М.: ИКФ "Каталог", 2002. – 106 с.
2. Медведев Б.М., Тышкевич А.И. Модели телекоммуникационных систем: учеб. пособие. - СПб.: Изд-во Политехн. ун-та, 2022. – 77 с.
3. Желамский М.В. Электромагнитное позиционирование. Преимущества и области применения. // ЭЛЕКТРОНИКА: НТБ. - 2007. - №5. - С. 105-111.
4. Использование Wi-Fi для indoor позиционирования и навигации // Navigine URL: https://nvgn.ru/blog/wi-fi-dlya-vnutrenney-navigatsii-v-pomeshenii/ (дата обращения: 10.03.2023).
5. F. Zafari, A. Gkelias and K. K. Leung, "A Survey of Indoor Localization Systems and Technologies," in IEEE Communications Surveys & Tutorials, vol. 21, no. 3, pp. 2568-2599, thirdquarter 2019, doi: 10.1109/COMST.2019.2911558.
6. Mohammad Afaneh Intro to Bluetooth Low Energy: The easiest way to learn BLE. – 1 изд. - NOVEL BITS, 2018
7. Determining The Right Lift Truck Navigation System // Jungheinrich URL: https://www.supplychain247.com/papers/determining_the_right_lift_truck_navigation_system (дата обращения: 05.02.2023).

УДК 004.42

М. А. Белова (4 курс бакалавриата),
А. В. Самочадин, к.т.н., доцент

АВТОМАТИЗИРОВАННАЯ СИСТЕМА ПЛАНИРОВАНИЯ ПЕРЕВОЗОК ГРУЗОВ АВТОТРАНСПОРТОМ

Сегодня приложения для решения логистических задач, связанных с организацией транспортировки грузов, пользуются большим спросом. Внедрение автоматизированных систем позволяет усовершенствовать работу логистических отделов, предоставить информацию для своевременного принятия управленческих решений [1]. Целью этой работы является разработка автоматизированной системы планирования перевозок грузов автотранспортом (АС ППГ) на основе выбора совокупности предложений по перевозке, обеспечивающих максимальную прибыль компании перевозчика.

Поставленные задачи:

– Разработать требования к системе;

– Рассмотреть существующие решения;

– Разработать архитектуру и интерфейс;

– Разработать приложение;

– Провести тестирование приложения.

АС ППГ должна быть реализована в виде независимого клиент-серверного приложения со своими средствами регистрации и аутентификации пользователей. Для обеспечения планирования АС ППГ должна обеспечивать выполнение следующих функций:

1. Учет предложений на перевозку грузов. Должна быть обеспечена следующая функциональность:

– Ввод информации о новых предложениях из различных источников:

– Редактирование предложений.

– Удаление (архивацию) предложений.

– Просмотр предложений. Должны быть предусмотрены следующие возможности просмотра предложений: сортировка, фильтры, поиск.

– Формирование статистических отчетов по предложениям.

2. Формирование плана по поездкам для конкретного автомобиля. Должна быть обеспечена следующая функциональность:

– Формирование плана по поездкам для выбранного логистом автомобиля на основе выбора из совокупности предложений по перевозке, предложения, которое обеспечивает максимальную выручку компании перевозчика. План, формируется с учетом следующей информации:

– Список предложений;

– Информация о конкретном автомобиле (транспортное средство и водитель) и его местонахождении.

Данные, необходимые для формирования себестоимости маршрута, включающие длину маршрута, среднее время погрузки и выгрузки, запрашиваются у системы ATrucks.

– Формирование заявок на транспортное средство (поля: клиент, дата погрузки, адрес погрузки, даты выгрузки, адрес выгрузки, транспортное средство, водитель, логист);

– Формирование уведомлений о принятых заявках.

Уведомление должно приходить сотруднику, который этот заказ загрузил в систему. Форма уведомления: Уведомление в почту, уведомление на рабочей странице системы.

– Формирование статистических отчетов.

3. Формирование отчетов о работе.

Отчетная подсистема должна обеспечивать формирование отчетов по работе системы в целом.

Основной алгоритм работы приложения реализуется в соответствии со следующей блок-схемой:

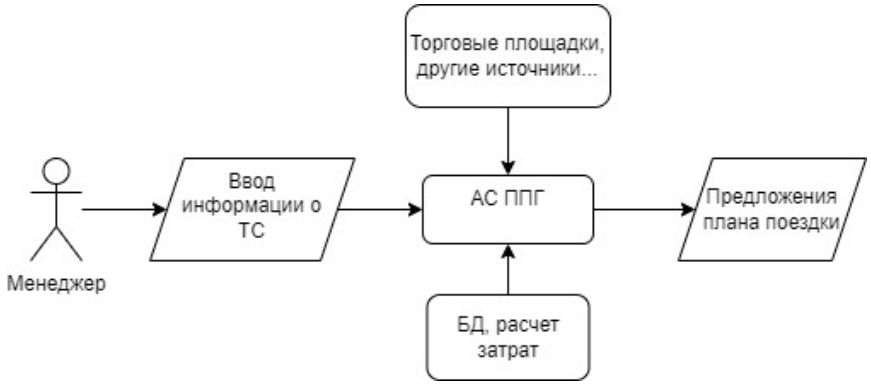


Рисунок 1 – Блок-схема, отображающая принцип работы приложения

В качестве средств для реализации АСППГ был выбран технологический стек MERN (MongoDB [2], Express [3], React [4], NodeJS [5]) и среда разработки VS Code.

ЛИТЕРАТУРА

1. Рустамова, А. Н. Обзор программного обеспечения для управления логистическими процессами / А. Н. Рустамова // Вестник магистратуры. – 2019. – № 1-2(88). – С. 78-81
2. MongoDB. [Электронный ресурс] Режим доступа: https://www.mongodb.com/
3. Express. [Электронный ресурс] Режим доступа: https://expressjs.com/
4. React. [Электронный ресурс] Режим доступа: https://react.dev/
5. NodeJS. [Электронный ресурс] Режим доступа: https://nodejs.org/en/about

УДК 004.774

М. Г. Брежо (4 курс бакалавриата),
Т. В. Коликова, ст. преподаватель

РЕАЛИЗАЦИЯ ВЕБ-ВЕРСИИ КОМАНДНОЙ РОЛЕВОЙ ИГРЫ «МИРОВОЕ ГОСПОДСТВО»

При проведении дистанционных мероприятий (игр) такими организациями, как Объединённый студенческий совет общежитий СПбПУ, постоянно встаёт вопрос необходимости автоматизации этих частей этих игр, например, оповещение игроков, подсчёт внутриигровой статистики и т. д. Одной из популярных на данный момент игр является «Мировое господство». Она включает две основные части:

1. Распределение командами ресурсов на развитие своей команды и ухудшения состояния команд-соперников.

2. Переход одного из участников команды в другую для переговоров (осуществляется посредством программы для организации конференций (ZOOM, Discord и др.)

Ниже представлена сравнительная таблица различных способов проведения игры.

Таблица 1 – Сравнение способов проведения игры

Критерии	Ручной подсчёт	Google-таблицы	Веб-сервис
Необходимость программирования	Не требуется программирование	Как правило, не требуется написание кода, но есть возможность использовать Google Apps Script для создания макросов	Необходимо написание программного кода для создания API
Поддержка командной игры	Ограничена возможностью проведения игры только в реальном времени (меньшее количество участников)	Может быть неудобен в использовании организаторами и пользователями	Дает возможность реализовывать дистанционные игры и синхронизировать действия игроков
Цена	Не требует затрат (без учёта деятельности ведущего и помощников)	Бесплатно, но требует наличия Google-аккаунта (без учёта деятельности ведущего и помощников)	Разная ценовая политика в зависимости от размера проекта и уровня сложности реализации
Безопасность	Отсутствует возможность повлиять на игру извне, так не используется программный продукт	Хорошо защищен от взломов, но требует наличия Google-аккаунта для доступа	Высокая степень безопасности с помощью использования соответствующих протоколов и технологий защиты данных
Масштабируемость	Ограничена возможностью проведения игры только в реальном времени	Хорошо масштабируется за счет добавления новых пользователей	Обладает высокой масштабируемостью за счет поддержки веб-сервисов и различных технологий разработки приложений.

Таблица 1 – Сравнение способов проведения игры (продолжение)

Критерии	Ручной подсчёт	Google-таблицы	Веб-сервис
Человеческий фактор	Высокая вероятность ошибки со стороны организаторов, так как все расчёты проводятся вручную	Средняя вероятность ошибки со стороны организаторов за счёт частичной автоматизации	Отсутствует, так как все расчёты проводятся без участия человека
Необходимость организаторов в проведении игры	Необходимость участия обязательна	Необходимость участия обязательна	Может быть полностью автономна
Возможность проведения нескольких игр одновременно	Есть, при условии нескольких групп организаторов, но количество игр сильно ограничено	Есть, при условии нескольких групп организаторов, но количество игр сильно ограничено	Есть, количество игр может быть достаточно большим

Несмотря на то, что данную таблицу можно дополнять и дальше, очевидно, что веб-сервис гораздо удобнее для пользователя, хоть и требует навыков для его реализации.

Для создания веб-сервиса для проведения дистанционных игр понадобится ряд инструментов и технологий:

1. Язык программирования. Выбран Java.[1]
2. База данных для хранения игровых данных. PostgreSQL.[2]
3. Фреймворк. Java Spring Framework[3], проект будет разработан в соответствии со структурой MVC.[4]
4. Хостинг для размещения веб-сервиса и базы данных. [4]
5. Интерфейс пользователя для облегчения взаимодействия с пользователем.
6. API для взаимодействия веб-сервиса и клиентского приложения.
7. Авторизация для обеспечения безопасности игры.
8. Тестирование чтобы убедиться в работоспособности приложения.

В ходе работы был реализован макет веб-сервиса.

ЛИТЕРАТУРА

1. Роль Java в разработке в веб- и мобильных приложений. Функциональное программирование — за и против [Электронный ресурс] URL: https://javarush.com/groups/posts/3832-kofe-breyk-131-roljh-java-v-razrabotke-v-veb--i-mobiljhnihkh-prilozheniy-funkcionaljhnoe-progra (дата обращения: 20.03.2023)
2. Postgresql для веб приложения [Электронный ресурс] URL: https://compsovet.com/postgresql-dlya-veb-prilozheniya/ (дата обращения: 20.03.2023)
3. В Java всегда весна: почему фреймворк Spring считается самым популярным у Java-разработчиков [Электронный ресурс] URL: https://practicum.yandex.ru/blog/framework-spring-java/ (дата обращения: 20.03.2023)
4. MVC [Электронный ресурс] URL: https://developer.mozilla.org/en-US/docs/Glossary/MVC (дата обращения: 20.03.2023)
5. Рейтинг хостингов 2022, платные и бесплатные хостинги, какой лучший хостинг сайта в России? [Электронный ресурс] URL: https://vc.ru/u/609590-ruslan-iyun/353431-reyting-hostingov-2022-platnye-i-besplatnye-hostingi-kakoy-luchshiy-hosting-sayta-v-rossii (дата обращения: 20.03.2023)

УДК 004.42

Д.Е. Булатов, А.А. Шожат (3 курс бакалавриата),
Т.В. Леонтьева, к.т.н., доцент

РАЗРАБОТКА КОМПЛЕКСНОЙ СУДЕЙСКОЙ СИСТЕМЫ ДЛЯ ПРОВЕДЕНИЯ СПОРТИВНЫХ СОРЕВНОВАНИЙ

В современном мире цифровизация входит во все области нашей жизни, а спорт для многих является большой ее частью. На данный момент судейство многих соревнований начального и среднего уровня происходит в устаревшем виде – используются неэффективные и трудоемкие инструменты, в следствие чего могут возникать ошибки в итоговых оценках судей, а зрители и сами спортсмены вынуждены долгое время ожидать подсчета результатов. Решением вышеупомянутых проблем является использование нашей электронной судейской системы.

Изучая конкурентов нашей системы, мы выяснили, что все существующие аналоги используются международными комитетами и направлены на конкретный вид спорта. Например, существующая судейская система ИСУ [1] нацелена на соревнования по фигурному катанию, и никак не может быть применена к другим соревнованиям. Кроме того, ни одна из таких систем не предоставляется для проведения соревнований на более низких уровнях - например, для детских или юношеских соревнований.

Исходя из описания проблемы, мы сформировали цель проекта – разработать комплексную систему судейства соревнований, с возможностью адаптации под разные виды спорта, а также разработали архитектуру [2] проекта по принципу Interface Segregation Principle [3] (Рисунок 1).

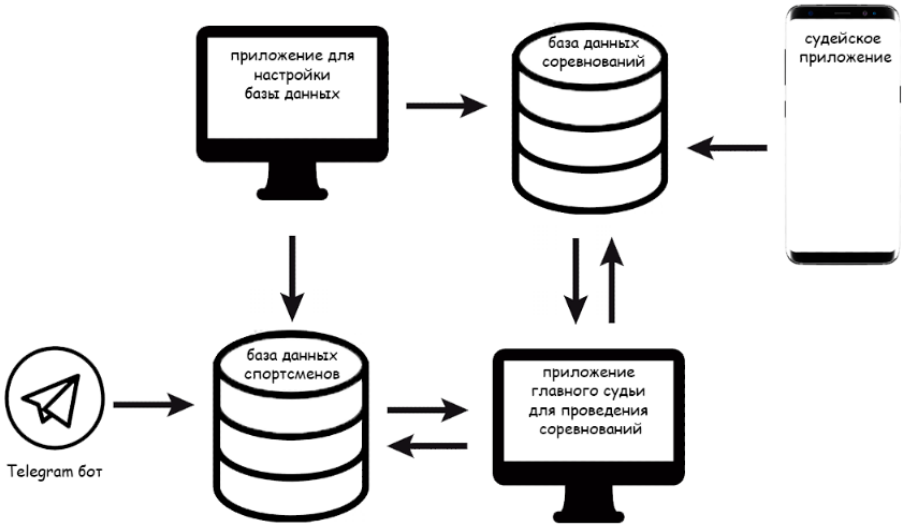


Рисунок 1 – Архитектура системы

В архитектуру программной системы входят 2 базы данных PostgreSQL, в первой хранятся все данные о спортсменах, тренерах и результатах прошедших соревнований, доступ к которым можно получить через Telegram-бот. Поскольку результаты соревнований должны быть всегда зафиксированы, на случай, если интернет-соединение пропадет, чтобы данные сохранились, вторая база данных является локальной и разворачивается на компьютере главного судьи во время проведения соревнования. В нее заносятся текущие результаты спортсменов и после окончания соревнований загружаются в общую базу спортсменов. Для управления конфигурацией баз данных предусмотрено отдельное приложение, оно позволяет сделать нашу систему гибкой к соревнованиям по абсолютно разным видам спорта. Судейское

приложение реализовано в виде android-приложения, такой вариант позволяет судье легко заносить результат в базу данных и не быть привязанным к рабочему месту, где установлен компьютер, что является важным параметром в судействе некоторых видов спорта.

Для реализации сформированной архитектуры разработаны клиентские приложения на языке программирования Java с использованием фреймворков Spring и Swing для UI десктопных приложений, а также средств Android API для приложения на смартфон. Кроме того, создан Telegram-бот на языке Python с применением API Telebot [4].

ЛИТЕРАТУРА

1. Судейская система ИСУ. [Электронный ресурс] – URL: https://www.isu.org/isu-statutes-constitution-regulations-technical-rules-2/isu-judging-system
2. Роберт Мартин, Чистая архитектура. Искусство разработки программного обеспечения. 2022, - с. 232-235
3. Создание архитектуры программы. [Электронный ресурс] – URL: https://habr.com/ru/post/276593/
4. pyTelegramBotAPI. [Электронный ресурс] – URL: https://pypi.org/project/pyTelegramBotAPI/

УДК 004.4

М. В. Головин (4 курс бакалавриата),
Т. В. Коликова, ст. преподаватель

РАЗРАБОТКА СЕРВЕРНОЙ ЧАСТИ ДЛЯ ВЕБ-ПРИЛОЖЕНИЯ НА ANDROID С ИСПОЛЬЗОВАНИЕМ ЯЗЫКА JAVA И ФРЭЙМВОРКА SRING HIBERNATE

Существует малое количество, в которых любой может стать автором и выкладывать свои произведения, которые другие смогут читать на этом же ресурсе. Само приложение будет полезно и удобно многим людям, ведь среди неизвестных авторов и людей, которые хотят заняться написанием книги, может быть много таких, которые имеют большой талант в данной области, и могут быть оценены аудиторией по достоинству. Именно поэтому реализованное приложение может помочь людям в начале их авторской карьеры, а также пользователям найти интересующие их произведения в открытом доступе. Для работы программного продукта необходимо создание серверной части приложения, для чего будет использоваться язык Java, и фрэймворк Hibernate [1] в частности. Существует несколько фрэймворков для работы с базами данных на языке JAVA:

1. Hibernate.
2. EclipseLink[2].
3. MyBatis [3].
4. OpenJPA [4].
5. JOOQ [5].

Из данного списка после проведения сравнения и уточнения информации был выбран фрэймворк Hibernate. Данное решение можно обосновать рядом преимуществ:

1. Hibernate является полноценным ORM-фреймворком, который позволяет разработчикам работать с объектами и базами данных на более высоком уровне абстракции.
2. Hibernate предоставляет простой и удобный API для работы с базами данных, что делает его привлекательным для разработчиков.
3. Hibernate поддерживает множество баз данных, включая MySQL, Oracle, PostgreSQL и многие другие.
4. Hibernate имеет большое сообщество разработчиков, которые активно работают над его улучшением и поддержкой.

5. Hibernate легко интегрируется с другими фреймворками, такими как Spring и Struts, что делает его еще более привлекательным для разработчиков.

Для разрабатываемого продукта была разработана база данных с использованием объектно-реляционной базы данных PostgreSQL[6] так как данная СУБД является бесплатной, удобной в пользовании, используется в большом количестве продуктов. Также на выбор повлиял личный опыт использования двух популярных СУБД MySQL[7] и PostgreSQL, из которых более удобной в пользовании и работе с данными оказалась PostgreSQL.

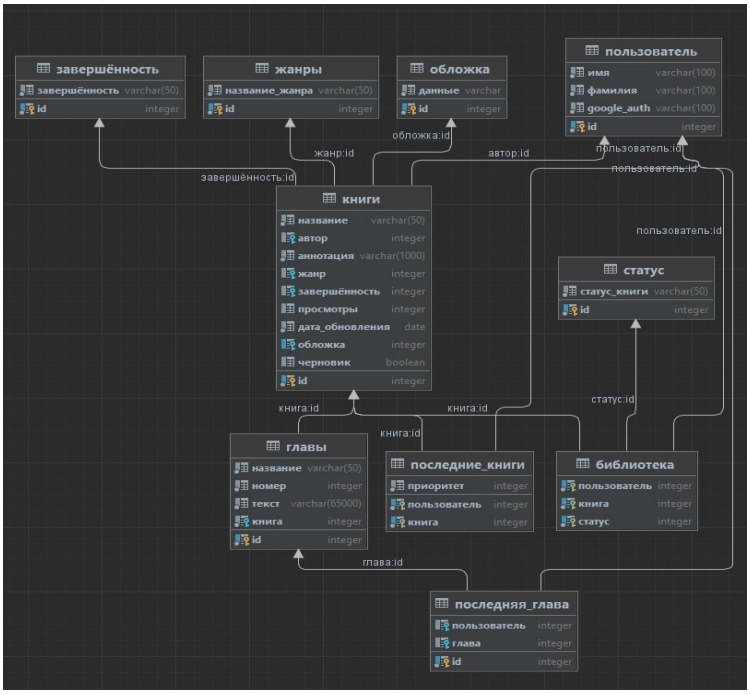


Рисунок 1 – Реализованная база данных

Взаимодействие Hibernate с PostgreSQL помогает максимизировать преимущества базы данных, такие как расширяемость, высокую производительность и надежность. Hibernate также обеспечивает удобный способ работы с базой данных, что упрощает разработку и поддержку приложений.

В Java с помощью Hibernate были созданы определенные группы классов для работы с СУБД и связи с клиентской частью приложения:

1. Модели – классы для таблиц базы данных, поля в которых соотносятся столбцам в таблице базы данных. Происходит соответствие между классами Java и таблицами БД, что даёт возможность работать с объектами из БД уже с помощью языка Java.

2. DAO (Data Access Object) являются прослойкой между БД и системой. С помощью данных классов происходит обращение к базам данных и получение, а также изменение данных внутри самой БД. Так же в методах DAO написаны сложные запросы к базе данных с использованием языка HQL, работающий с объектами Java вместо таблиц базы данных.

3. Сервисы используются для вызова методов в DAO и чтения данных, полученных в запросах, изменения, создания объектов, которые потом с помощью DAO будут отправлены в базу данных и сохранены там.

4. API для связи с клиентской частью приложения и отправки или получения данных. API вызывают методы в сервисах и отправляют полученные данные клиентской части для обработки и показа пользователю.

ЛИТЕРАТУРА

1. Hibernate [Электронный ресурс]. -Режим доступа: https://hibernate.org/
2. EclipseLink [Электронный ресурс]. -Режим доступа: https://www.eclipse.org/eclipselink/

3. MyBatis [Электронный ресурс]. -Режим доступа: https://mybatis.org/mybatis-3/
4. OpenJPA [Электронный ресурс]. -Режим доступа: https://openjpa.apache.org/
5. JOOQ [Электронный ресурс]. -Режим доступа: https://www.jooq.org/
6. PostgreSQL Documentation [Электронный ресурс]. -Режим доступа: https://www.postgresql.org/docs/
7. MySQL Documentation [Электронный ресурс]. -Режим доступа: https://dev.mysql.com/doc/

УДК 004.4

А. А. Винокуров (4 курс бакалавриата),
И. В. Шошмина, к.т.н, доцент

РАЗРАБОТКА ПРОГРАМНЫХ МОДУЛЕЙ КОНСТРУКТОРА ИНДИВИДУАЛЬНЫХ ОБРАЗОВАТЕЛЬНЫХ ТРАЕКТОРИЙ

Конструктор индивидуальных образовательных траекторий (КИОТ) – система, предназначенная для построения для каждого обучающегося индивидуальной образовательной траектории на платформе «Moodle». Система предоставляет пользователям набор инструментов для автоматизации подбора элементов дисциплин в соответствии с заданным профессиональным профилем обучающегося.[1]

Целью работы является разработка программных модулей в проекте, которые предоставляют инструментарий, позволяющий ввести функцию управления ИОТ дисциплины. Данная функция позволит автоматизировать подбор элементов дисциплины в соответствии с текущим профессиональным профилем обучающегося.

Рассмотрим процесс дистанционного образования на платформе Moodle таким, какой он сейчас есть: преподаватель создает электронный курс и этот курс доступен всем обучающимся в исходном виде. Приложение КИОТ предлагает добавить следующий функционал: для каждого обучающегося создаются их профессиональные профили в зависимости от их успехов при прохождении тестов. Благодаря созданию профессиональных профилей, обучающихся будут построены индивидуальные траектории обучения, и у каждого студента будет персонализированный исходный курс, содержащий те образовательные дисциплины, которые ему необходимы. Тем самым на выходе для каждого обучающегося мы получаем персонализированный курс.[1]

Архитектура КИОТ – это классическое традиционное клиент-серверное приложение. Приложение состоит из клиентской части и серверной, где находятся разрабатываемые модули, состоящие из слоя бизнес-логики и слоя передачи данных. Важнейшей частью проекта является разработка программных модулей для бекэнд части. В частности, модуль курса предполагает выгрузку курса с платформы Moodle и дальнейшее взаимодействие с ним для получения индивидуальной образовательной траектории. [2]

Для достижения этой цели необходимо решить следующие задачи:

1. Реализация модуля межсистемного взаимодействия для выгрузки сущности курса с платформы Moodle
2. Реализация модулей вложенных сущностей курса
3. Реализация модуля метрики ЗУН (Знания-Умения-Навыки)
4. Реализация связующего модуля между ЗУНами, другими метриками и вопросами
5. Реализация модуля абстрактного профиля обучающегося
6. Реализация модуля профессионального профиля обучающегося

Из этих задач были выполнены все задачи, связанные с реализацией модуля курса и метрик. В дальнейшем планируется разработка взаимодействия ИОТ и профилем обучающегося. Возникшие трудности были из-за несоответствия архитектуры базы данных

Moodle и нашего представления связи курса с метриками, в следствие чего приходилось продумывать необходимые шаги для перепроектирования некоторых модулей взаимодействия с платформой Moodle. [3][4]

Еще одной проблемой при разработке стали компетенции и индикаторы, а именно их представление на платформе Moodle, которое также отличается от того, которое планировалось при проектировании конструктора индивидуальных образовательных траекторий. Эта проблема была решена предоставляем альтернативы их отображения заказчику и его последующим одобрением. [2]

Для выполнения моих задач и для реализация всего проекта в целом был использован следующий технологический стек:

1. Spring Framework [5]
2. Micronaut Framework
3. Pure Java
4. H2 (Postgres в будущем)
5. Lombok

Все разрабатываемые модули имеют клиентское отображение и представлены в клиенте приложения – веб-части сайта КИОТ.

В ходе проделанной работы были выполнены следующие поставленные задачи: был разработан модуль межсистемного взаимодействия с платформой «Moodle» для выгрузки сущности курса, организовано взаимодействие курсов, модулей и других вложенных сущностей. Кроме этого реализована выгрузка, обновление, удаление компетенций и индикаторов с платформы «Moodle». Наконец, была реализована метрика ЗУН на основе вопроса и привязанного к нему индикатора.

В дальнейших планах – реализация абстрактных и профессиональных профилей обучающихся и связующего модуля между ЗУНами, индикаторами, компетенциями и вопросами. Разработка ведется для ОС Linux на языке C++.

ЛИТЕРАТУРА

1. Документация КИОТ [Электронный ресурс] Режим доступа: https://gitlab.com/SPBPU_KIOT/kiot-wiki/-/wikis/home
2. Разработка клиент-серверны приложений с помощью Micronaut Framework [Электронный ресурс] Режим доступа: https://habr.com/ru/post/418117/
3. Moodle для разработчиков [Электронный ресурс] Режим доступа: https://moodledev.io/docs/
4. Micronaut Framework документация [Электронный ресурс] Режим доступа: https://docs.micronaut.io/index.html
5. Spring Framework [Электронный ресурс] Режим доступа: https://docs.spring.io/spring-framework/docs/current/reference/html/

УДК 004.4'412

А. В. Дамбиева (4 курс бакалавриата),
А. В. Самочадин, к.т.н., доцент

КАЛЕНДАРЬ С ВОЗМОЖНОСТЬЮ ДОБАВЛЕНИЯ СОБЫТИЙ ИЗ ТЕКСТОВ

Социальные сети стремятся заменить собой почти все существующие приложения. В том числе это касается и календаря, для осуществления тайм-менеджмента. Запланированные напоминания уже давно встроены в любые календари во всех социальных сетях, но у них крайне ограниченный функционал. Более того, главная задача любой социальной сети заключается в том, что пользователи могут делиться своими новостями в текстовом формате.

Заказчик заинтересован в расширении функционала календаря в своей социальной сети. Пользователь должен иметь возможность осуществлять полноценный тайм-менеджмент через

календарь в своей социальной сети. Календарь также должен извлекать информацию о событиях из текстов или же публикаций и уведомлять пользователя об этих событиях.

Таким образом целью работы является разработка технологии, способной анализировать входящий неструктурированный текст [1][3]; извлекающей из текста необходимую для календаря информацию [2]. А также необходимо реализовать уведомления с расширенным функционалом для этого календаря.

Для достижения этой цели необходимо решить следующие задачи:

1. Обзор существующих приложений, использующих функцию календаря.
2. Сравнение существующих решений.
3. Написание детальных требований для проекта.
4. Определение инструментов, необходимых для реализации проекта.
5. Реализация алгоритма анализа неструктурированного текста.
6. Реализация алгоритма уведомлений для календаря.
7. Реализация пользовательского интерфейса.
8. Демонстрация решения, обозначенных ранее проблем.

Будет проведён анализ самых востребованных приложения для ПК и для смартфонов, которые предоставляют возможности напоминаний.

Для разработки был выбран язык Python, для создания интерфейса используется GUI-фреймворк DearPyGUI [4]. Для работы с уведомлениями используется библиотека Plyer. Для получения информации об активности пользователя на Windows используется WinAPI. Для хранения информации в календаре будет использована база данных постгресс SQL. [5]

Для расширения функционала напоминаний планируется реализовать алгоритм, отслеживающий активность пользователя, а также напоминания с возможностью учитывать время активности или неактивности пользователя. Повторяющиеся напоминания с нестабильным графиком, в которых можно выбрать любые интервалы, разное количество повторов и пропусков. А также стандартные пользовательские напоминания, в которых задается событие и время.

ЛИТЕРАТУРА

1. Xueqing Wu, Jiacheng Zhang, Hang Li: Text-to-Table: A New Way of Information Extraction // Long Papers. - 2022. - №1. - С. 2518 - 2533.
2. Weikum, G., Hoffart, J., Suchanek, F.: Ten Years of Knowledge Harvesting // Lessons and Challenges. - 2016. - Data Engineering 5. - С. 41–50
3. Ismini Lourentzou, Anna Lisa Gentile, Daniel Gruhl, Jane Fortner, Michele Freemon, Kendra Grande.: Difficult Relations: Extracting Novel Facts from Text // ISWC-Posters-Demos-Industry. – 2018
4. Dear PyGui’s Documentation: [Электронный ресурс] // Read the doc. URL: https://dearpygui.readthedocs.io/en/latest/index.html
5. XScreenSaver: [Электронный ресурс] // ArchWiki. URL: https://wiki.archlinux.org/title/XScreenSaver

УДК 004.422.81

А. А. Дмитриев (4 курс бакалавриата),
С. Э. Сараджишвили, к.т.н., доцент

РАЗРАБОТКА ПРИЛОЖЕНИЙ ДЛЯ ВУЗОВ НА ПРИМЕРЕ ПРИЛОЖЕНИЯ "МОЙ ПОЛИТЕХ"

В современном мире получение актуальной информации об университете является одной из проблем каждого студента. Интерфейс ВУЗовских сайтов часто оказывается перегруженным, что сильно осложняет поиск ответов. Кроме того, по статистике [1], с каждым годом люди всё больше и больше отдают предпочтение мобильным приложениям, нежели

браузерам и другим источникам информации, из-за чего новости и важные объявления не доходят до значительной части студентов.

Рисунок 1 – Статистика использования браузеров и мобильных приложений

Одним из способов увеличения информационной доступности ВУЗов является наличие собственного мобильного приложения, отвечающего запросам студентов.

Таким образом *целью* данной работы является разработка методики разработки приложения для ВУЗов, а также разработка мобильного приложения на основе данной методики.

Для достижения этой цели необходимо решение следующих *задач:*

– Обзор существующих подходов к разработке приложений для ВУЗов.

– Проведение сравнительного анализа найденных подходов.

– Проведение опроса среди студентов СПбПУ с целью определения информационных потребностей и запросов.

– Предложение методики разработки приложения для ВУЗов.

– Реализация мобильного приложения на основе разработанной методики.

– Демонстрация результатов на основе заинтересованности среди студентов.

Архитектура разрабатываемого приложения включает в себя язык программирования Kotlin [2] и шаблон проектирования Model-View-ViewModel [3]. В качестве сервера используется Firebase [4].

ЛИТЕРАТУРА

1. Статья Статистика использования браузеров и мобильных приложений. [Электронный ресурс] Режим доступа: https://it-guru.kz/news/statistika-ispolzovaniya-brauzerov-i-mobilnykh-prilozheniy/
2. Kotlin Programming Language. [Электронный ресурс] Режим доступа: https://kotlinlang.org
3. Alvin Ashcraft. “Learn WinUI 3.0” – 2021. – 82pp.
4. Firebase. [Электронный ресурс] Режим доступа: https://firebase.google.com

УДК 004.93

С. Р. Дзуцева (4 курс бакалавриата),
Д.С. Эйзенах, (2 курс аспирантуры),
А. П. Маслаков, ст. преподаватель

РАЗРАБОТКА СКАНЕРА ФОТОГРАФИЙ ДЛЯ МОБИЛЬНОГО ПРИЛОЖЕНИЯ «ОДНОКЛАССНИКИ» ПОД ОС ANDROID

Одноклассники [1] – одна из крупнейших социальных сетей в России и странах ближнего зарубежья. Наибольшее число пользователей площадки используют мобильные устройства на базе операционной системы Android. Очевидно, популярность этой ОС не ограничивается аудиторией социальной сети: подавляющее большинство современных смартфонов во всем мире используют Android [2].

Широкая распространенность мобильных устройств способствовала стремительному развитию мобильной фотографии. Как результат, подавляющее большинство личных снимков пользователей социальных сетей сделаны с помощью смартфонов. Однако так было не всегда, и у огромного числа людей до сих пор хранятся альбомы с напечатанными фотографиями. При желании опубликовать старые бумажные фотографии с социальной сети, пользователям приходится сталкиваться со сложным процессом оцифровки: сначала необходимо сфотографировать изображение, стараясь не исказить перспективу, затем выполнить кадрирование.

В работе был реализован раздел в приложении «Одноклассники» для ОС Android для удобного переноса бумажных фотографий пользователей в социальную сеть. Раздел должен освободить пользователей от монотонных действий, оставляя за собой выполнение распознавания фотографии на снимке, последующего кадрирования и исправления перспективы так, как если бы снимок был отсканирован с помощью специализированного устройства, а не сделан на камеру телефона.

На первом этапе разработки раздела был реализован алгоритм распознавания границ фотографий и алгоритм трансформации перспективы с помощью OpenCV [3]. Алгоритм распознавания границ в качестве результата возвращает координаты углов фотографии, которые используются в расчетах для исправления перспективы [4].

Следующий этап заключался в разработке самого сканера. Код программы был написан на языке Kotlin. При проектировании раздела был выбран архитектурный шаблон проектирования MVVM.

Пользовательский интерфейс состоит из трех экранов, реализованных с помощью Fragment (см. Рис. 1):

- камера, с помощью которой пользователь делает снимок
- редактирование распознанной области
- предварительный просмотр фотографии обрезанной и откорректированной фотографии

Для работы фрагмента камеры была использована библиотека CameraX, предоставляющая согласованный, простой в использовании API, который работает на подавляющем большинстве устройств Android.

На фрагменте редактирования отображается сделанная пользователем фотография, поверх которой рисуется распознанная область в виде Custom View [5] – четырехугольника с двигающимися углами. Таким образом, пользователь сможет перемещать углы необходимым образом в случае, если алгоритм распознал область фотографии недостаточно точно. С этого фрагмента пользователь может либо перейти к просмотру результата сканирования, либо вернуться к камере.

Фрагмент предпросмотра демонстрирует фотографию, к которой был применен алгоритм трансформации перспективы и кадрирование. Отсюда пользователь может либо

вернуться к редактированию области фотографии, либо принять результат. Во втором случае фотография будет загружена на устройство пользователя, и откроется встроенный редактор приложения.

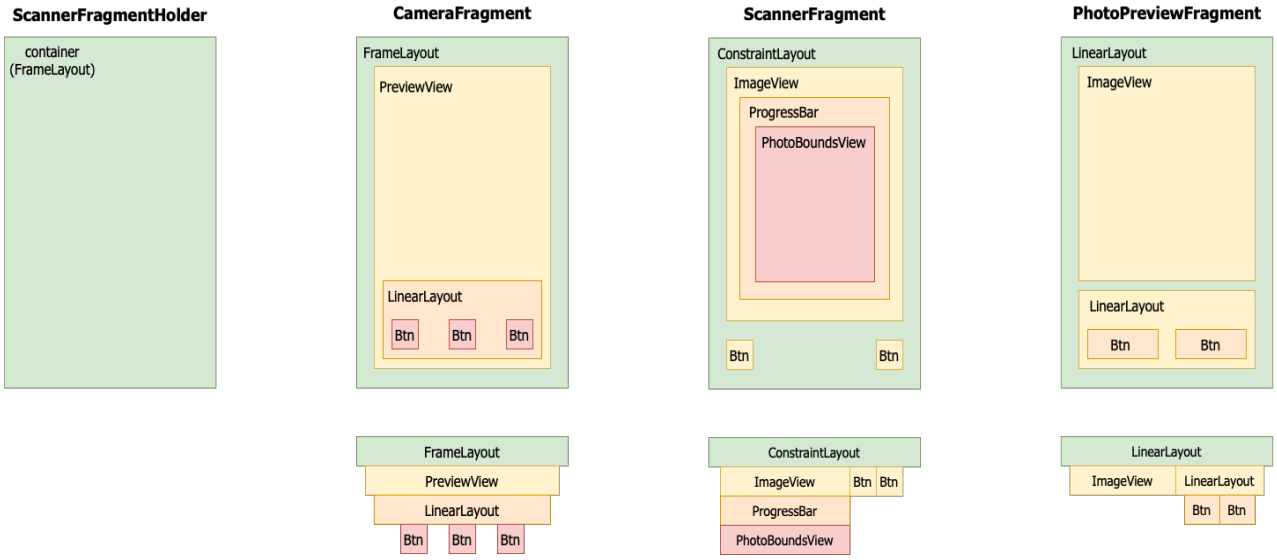


Рисунок 1 – Схема верстки фрагментов

За обновление состояния пользовательского интерфейса отвечают наследники класса ViewModel [6]: одна общая для всех фрагментов ViewModel, сохраняющая актуальное состояние, и ViewModel, содержащая логику распознавания границ фотографии. Полная схема взаимодействия компонентов приведена на Рис. 2.

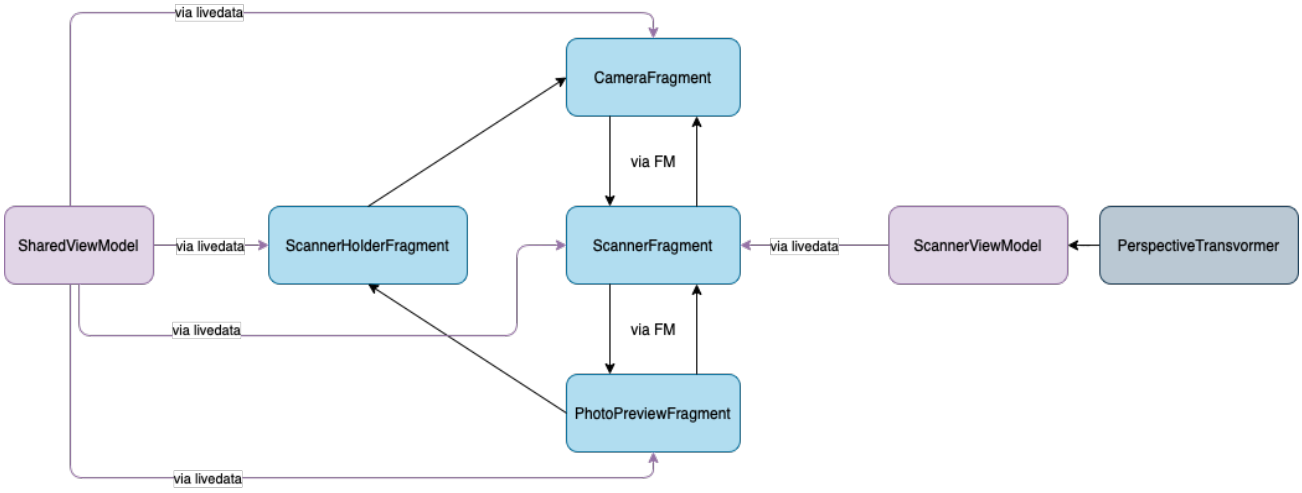


Рисунок 2 – Схема взаимодействия компонентов

Код сканера фотографий разрабатывался в отдельном модуле и был интегрирован в приложение «Одноклассники»: в разделе «Фото» добавились несколько точек входа в раздел сканирования. Так, у пользователей приложения появилась возможность использовать для переноса бумажных фотографий в социальную сеть удобную функциональность, реализованную в результате данной работы.

ЛИТЕРАТУРА

1. История и этапы развития соцсети Одноклассники [Электронный ресурс], URL: https://insideok.ru/info/
2. Mobile Operating System Market Share Worldwide | Statcounter Global Stats [Электронный ресурс], URL: https://gs.statcounter.com/os-market-share/mobile/worldwide
3. OpenCV: Introduction [Электронный ресурс], URL: https://docs.opencv.org/4.7.0/

4. С. А. Молодяков, Применение функций OpenCV в компьютерном зрении (60 примеров на Python): монография, Санкт-Петербург: Политех-Пресс, 2022.
5. Custom View Components | Android Developers [Электронный ресурс], URL: https://developer.android.com/develop/ui/views/layout/custom-views/custom-components
6. ViewModel overview | Android Developers [Электронный ресурс], URL: https://developer.android.com/topic/libraries/architecture/viewmodel

УДК 004.421

Г.В. Дорохов (4 курс бакалавриата),
Е.С. Орлов, ст. преподаватель

ИСПОЛЬЗОВАНИЕ БЛОКЧЕЙН ТЕХНОЛОГИЙ ДЛЯ ХРАНЕНИЯ И ЦИФРОВИЗАЦИИ УНИВЕРСИТЕТСКИХ ДИПЛОМОВ

Данный проект упростит всю сложную процедуру заполнения диплома, позволит не потерять диплом и также даст возможность проверять подлинность диплома, решив большую проблему с подделками дипломов. Это упростит жизнь людям, которые заполняют дипломы. А также позволит бывшим студентам всегда иметь в доступе заверенную версию своего диплома. И еще это может помочь при устройстве на работу, поскольку работодатель при устройстве на работу сможет проверить диплом на достоверность.

В ходе выполнения НИР будет разработано веб-приложение, с помощью которого можно будет загружать и просматривать университетский диплом. Хранится он будет в собственноручно написанном блокчейне.

Интерес к данной теме возник на основании изучении блокчейна и его применений в реальной жизни, кроме криптовалют. И тогда я обнаружил для себя, что, по сути, использование блокчейна, как хранения важных документов в полной мере оправдывает себя. И обратив внимание какие важные документы имеют проблемы в получении. Было обращено внимание на диплом. Изучив, как получается диплом, сколько процедур возникает при его оформлении, передаче и хранении. Стало понятно, что этой сфере нужна цифровизация, которая будет закрывать текущие проблемы. Ведь, например, диплом может быть утерян или может быть развалена сама образовательная система. И такие случаи были в современной истории. Также очень часто происходит подделывание дипломов или кража пустых дипломов.

Программный продукт, который будет разработан, позволит намного облегчить и обезопасить процедуру выдачи дипломов. И также это даст развитие сфере блокчейна в сфере образования. И также это даст определенный плюс в будущем.

Начиная с 2015 года эту идею постепенно применяют в разных университетах всего мира. Все началось с MIT и постепенно эта технология набирает популярность. Однако в нашей стране данная технология еще не успели прижиться и не стала такой популярной. И поэтому для Санкт-Петербургского политехнического университета это могло бы стать прорывной технологией, а впоследствии и другие университеты воспользовались бы разрабатываемым продуктом.

На данный момент нет конкретных продуктов, которые бы расширяли данную сферу. И этому есть причина, на данный момент из-за законодательных ограничений диплом в блокчейне не является юридической заменой. Однако, если будет реализован данный продукт и покажет свою успешность. Это может продвинуть законодательную сторону данного вопроса.

Для написания блокчейна и веб-приложения на стороне бэкенда я выбрал язык Kotlin [1]. Этот язык был разработан компанией Jetrbains в 2011, он работает поверх Java Virtual Machine [2]. Данный язык набирает свою популярность, тесня старые языки программирования. А также у него есть обратная совместимость с языком Java [3]. Это дает возможность библиотеки, написанные под Java. Язык Kotlin молодой и современный. Он вмещает в себя множество парадигм, которые и делают его моим выбором:

1) Null безопасность
2) Огромное количество синтаксического сахара, представленного в огромном функционале
3) Полное функциональное программирование и ООП
4) Дженерики
5) Корутины (легковесные потоки)
6) Интерпретируемость с JavaScript [4]

Среду разработки взята также от создателей языка - от компании Jetbrains. Intellij IDEA [5] - представляет из себя современную, мощную среду разработки, которая обладает огромным функционалом и при этом простотой в использовании.

Также для хранения и интеграции кода я выбрал платформу Github [6]. Основные его преимущества - простой, бесплатный и удобен для отслеживания и сохранения историй изменений.

ЛИТЕРАТУРА

1. Kotlin [Электронный ресурс] https://kotlinlang.org/
2. Java Virtual Machine [Электронный ресурс] https://www.java.com/en/download/
3. Java [Электронный ресурс] https://docs.oracle.com/javase/7/docs/technotes/guides/language/
4. JavaScript [Электронный ресурс] https://262.ecma-international.org/
5. Intellij Idea[Электронный ресурс] https://www.jetbrains.com/ru-ru/idea/
6. Github [Электронный ресурс] https://github.com/

УДК 004.9

М. С. Драцкая (4 курс бакалавриата),
Д.С. Эйзенах, аспирант,
А. П. Маслаков, ст. преподаватель

РАЗРАБОТКА ВЕБ-ПРИЛОЖЕНИЯ ДЛЯ ВЫБОРА ОБРАЗОВАТЕЛЬНЫХ КУРСОВ В СФЕРЕ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

В условиях стремительно меняющегося мира среди специалистов в сфере информационных технологий особенно востребованы различные способы повышения квалификации и приобретения новых знаний. На данный момент, одним из самых популярных вариантов решения задачи получения дополнительных навыков являются курсы. Особенно актуальна становится проблема выбора из большого количества образовательных программ.

При исследовании конкурентов, было выявлено, что существует ряд агрегаторов, предоставляющих возможность сравнить курсы, ознакомиться с отзывами об организациях, которые их проводят, а также получить данные о стоимости и длительности. Но такие сервисы не решают проблему получения достоверной информации от профессионалов, прошедших ту или иную образовательную программу. Пользователи, принимающие решение, не могут быть уверены в корректности отзывов. Также ни на одном существующем ресурсе не было отзывов о конкретном курсе, только о школах. В этой работе стоит задача решить эти проблемы.

Было разработано веб-приложение, предоставляющее доступ к отзывам людей, действительно прошедших курс, что позволяет быстро принять решение о покупке. Факт прохождения того или иного курса подтверждается организатором, поэтому в достоверности можно не сомневаться. Также внутри сервиса разработана функциональность личного кабинета пользователя, в котором можно отслеживать свой прогресс. Есть и возможность удобного поиска по курсам с фильтрацией, а также возможность оставить отзыв о пройденном курсе.

Кроме того, реализована страница с информацией о школе, на которой организаторы могут делиться актуальной информацией о новых курсах.

Таким образом, веб-приложение является не только эффективным способом найти отзывы, но и средством отслеживания личной истории обучения.

Веб-приложение было разработано с использованием таких языков программирования, как Java, JavaScript, а также СУБД MySQL. Помимо этого, использовались следующие фреймворки: Spring Framework, Spring Security [1], Spring Boot [2], Hibernate, Vue.js.

Архитектура приложения строится по принципу доменной модели [3]. Система состоит из слоев, как бы уложенных друг на друга [4]. В многоуровневой архитектуре, управляемой доменом, уровень пользовательского интерфейса располагается наверху. Этот уровень, в свою очередь, взаимодействует с уровнем контроллеров, который взаимодействует с сервисами. Внизу есть уровень репозиториев, который взаимодействует с базой данных, в которой хранятся данные о пользователях, курсах и отзывах.

Между составными частями сервера данные передаются с помощью объектов передачи данных (DTO) [5]. В базу данных передаются сущности доменной модели (Entity for Domain Model).

Ниже приведено схематичное изображение архитектуры проекта (см. Рис.1):

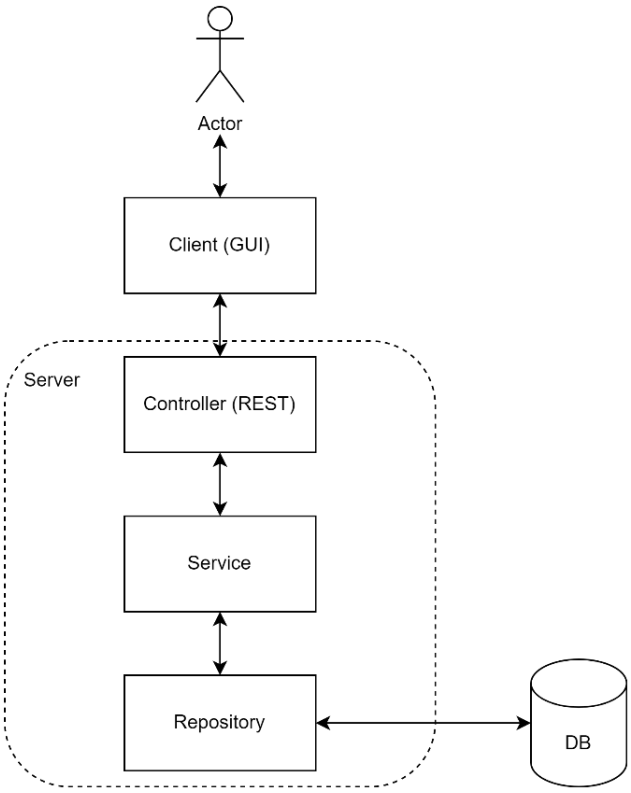


Рисунок 1 – Архитектура приложения

Взаимодействие клиент-сервера осуществляется по REST API [6] (см. Рис.2).

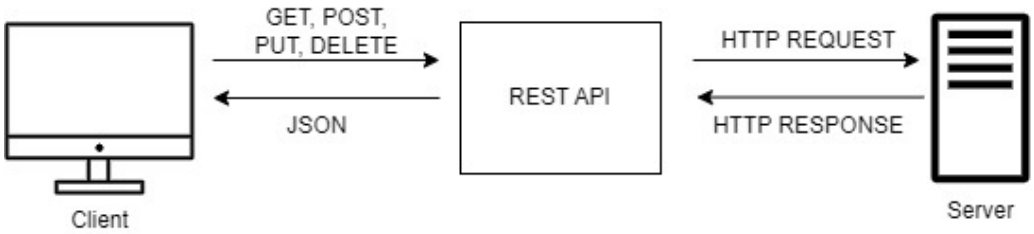


Рисунок 2 – Взаимодействие по REST API

В результате выполнения работы был получен готовый программный продукт, удовлетворяющий поставленным требованиям, сформированным исходя из особенностей

предметной области и анализа конкурентов. Была достигнута цель создать уникальный программный продукт. Стоит отметить и то, что инструменты, которые были использованы в работе, позволили наиболее эффективно выполнить поставленную задачу. В дальнейшем возможно развитие и усовершенствование продукта.

ЛИТЕРАТУРА

1. Spring Security // Spring URL: https://spring.io/projects/spring-security (дата обращения: 15.02.2023).
2. Spring Boot // Spring URL: https://spring.io/projects/spring-boot (дата обращения: 15.03.2023).
3. Элементы архитектуры веб-приложений // Medium URL: https://medium.com/nuances-of-programming (дата обращения: 15.02.2023).
4. Архитектура фронтенда и какой она должна быт // Habr.com: https://habr.com/ru/post/667214/ (дата обращения: 15.02.2023).
5. Pros and Cons of Data Transfer Objects // Learn.microsoft.com: https://learn.microsoft.com/en-us/archive/msdn-magazine/2009/august/pros-and-cons-of-data-transfer-objects (дата обращения: 15.02.2023).
6. E. John, M. Siddique, “Efficient Semantic Web Services Development Approaches using REST and JSON”, International Conference on Decision Aid Sciences and Application (DASA), 2021 (дата обращения: 15.02.2023).

УДК 004.031.42

С. В. Дроздов (2 курс магистратуры),
Н. В. Воинов, к.т.н., доцент

РАЗРАБОТКА МОБИЛЬНОГО ПРИЛОЖЕНИЯ УЧЕТА ЛИЧНЫХ ФИНАНСОВ С ПРИМЕНЕНИЕМ МИКРОСЕРВИСНОЙ АРХИТЕКТУРЫ

Приложения учёта личных финансов полезны людям, оказавшимся в трудном финансовом положении и желающим лучше контролировать свои траты. Ежедневный учёт финансов является рутинной задачей, из-за чего пользователи откладывают и забывают вносить свои траты в приложение. Это подрывает процесс выработки привычки вести учёт своих финансов у пользователя приложения, который в среднем занимает 2 месяца. Для решения этой проблемы было решено разработать систему поощрения пользователя на основе подходов геймификации приложения [1]. Система поощрений основана на очках опыта, уровне аккаунта и достижениях. Очки и достижения начисляются за действия пользователя в приложении, например, добавление расходов, создание и выполнение бюджетов. Также было решено реализовать возможность вести бюджет по тегу, например, что позволит вести единый бюджет для, например, проектов, требующих трат на разные категории товаров и услуг.

Приложениям, нацеленным на рынок потребителей, необходимы постоянные развитие и поддержка, чтобы выдерживать конкуренцию с приложениями-аналогами. Микросервисная архитектура имеет определенные преимущества в поддержке и расширении приложения в сравнении с монолитной. Объём исходного кода отдельного микросервиса в разы меньше монолитного приложения, что делает внедрение обновлений проще и быстрее [2]; чёткие границы ответственности и жёсткое разделение на отдельные приложения способствуют слабой связности микросервисного приложения и уменьшают сложность кода [3], тогда как в больших монолитах легче допустить тесную связь элементов и нарушить архитектурную согласованность приложения. Также в большинстве случаев микросервисное приложение показывает схожую с монолитным производительность [4].

Поддержка инфраструктуры микросервисного приложения сложна и требует серьёзных компетенций, что может быть невыгодно для выпуска новых приложений на рынке, в котором сложно оценить будущий успех приложения у потребителей. Эту проблему можно решить развёртыванием микросервисного приложения в serverless среде. Клиент serverless среды платит за время обработки запросов, а не время аренды вычислительных ресурсов, что может

уменьшить расходы на аренду облачной инфраструктуры [5]. В сравнении с популярным инструментом оркестрации контейнеров Kubernetes [6-8] настройка и поддержка облачной serverless инфраструктуры проще.

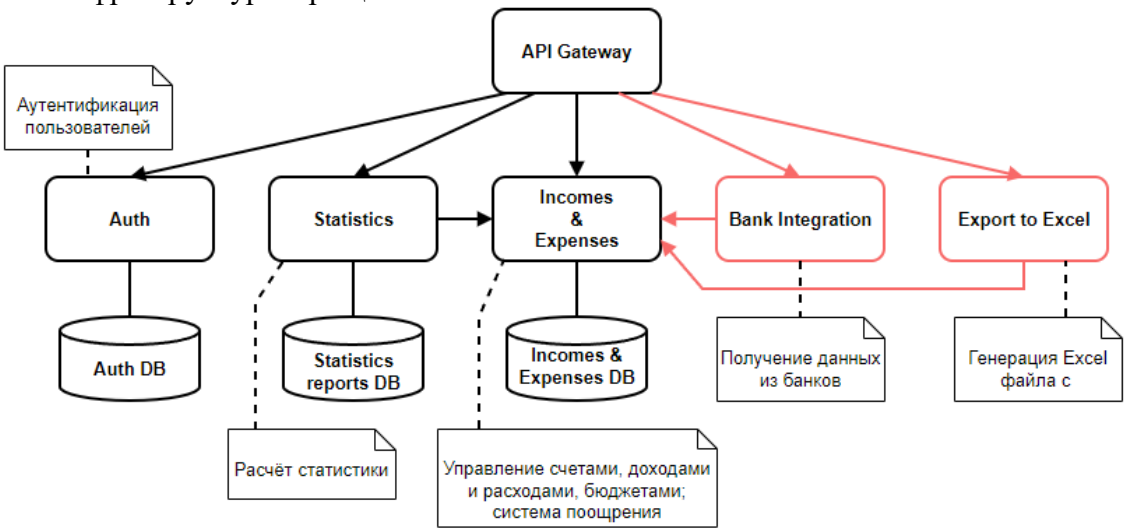


Рисунок 1 – архитектура бэкенда приложения

Согласно изложенным выше идеям был разработан бэкенд мобильного приложения, состоящий из трёх микросервисов и API-шлюза. На Рис. 1 изображена схема архитектуры бэкенда, красным обозначены микросервисы, которые могут быть разработаны в будущем. Все микросервисы разработаны с помощью языка Java [9] и фреймворка Spring. Бэкенд развёртывается в сервисе Yandex Cloud Functions [10].

Так как все микросервисы реализованы с помощью одних и тех же инструментов, возможно реализовать монолитную версию бэкенда для сравнения микросервисной и монолитной версий бэкенда. Сравнение будет осуществлено на основе результатов нагрузочного тестирования, расчёта стоимости аренды вычислительных ресурсов облачного сервиса и стоимости обработки запросов, и других количественных метрик, например, объёма исходного кода и дублирования кода.

ЛИТЕРАТУРА

1. Jeni Miles. The right app rewards to boost motivation. [Электронный ресурс] URL: https://medium.com/googleplaydev/the-right-app-rewards-to-boost-motivation-c1ec86390450 (дата обращения: 10.03.2023).
2. D. Pianini, A. Neri. Breaking down monoliths with Microservices and DevOps: an industrial experience report. 2021 IEEE International Conference on Software Maintenance and Evolution (ICSME), Luxembourg, 2021, pp. 505-514. – DOI: 10.1109/ICSME52107.2021.00051.
3. Florian Auer, Valentina Lenarduzzi, Michael Felderer, Davide Taibi. From monolithic systems to Microservices: An assessment framework. Information and Software Technology, Volume 137, 2021, 106600, ISSN 0950-5849. – DOI: 10.1016/j.infsof.2021.106600.
4. O. Al-Debagy, P. Martinek. A Comparative Review of Microservices and Monolithic Architectures. 2018 IEEE 18th International Symposium on Computational Intelligence and Informatics (CINTI), Budapest, Hungary, 2018, pp. 000149-000154. – DOI: 10.1109/CINTI.2018.8928192.
5. Villamizar, M., Garcés, O., Ochoa, L. и др. Cost comparison of running web applications in the cloud using monolithic, microservice, and AWS Lambda architectures. SOCA 11, 233–247 (2017). – DOI: 10.1007/s11761-017-0208-y.
6. Шемякинская, А. С. Реализация паттерна "оператор" для отслеживания состояния дисков в системе оркестрации контейнеров Kubernetes / А. С. Шемякинская, И. В. Никифоров // Современные технологии в теории и практике программирования : сборник материалов конференции, Санкт-Петербург, 23 апреля 2020 года / Санкт-Петербургский политехнический университет Петра Великого; Dell Technologies; EPAM Systems. – Санкт-Петербург: ПОЛИТЕХ-ПРЕСС, 2020. – С. 167-169. – EDN DQJEZO.

7. Сафронов, Д. Автоматическая балансировка нагрузки между потоковой обработкой данных и внутренними задачами кластера с использованием Kubernetes / Д. Сафронов, К. М. Стоноженко, И. В. Никифоров // Современные технологии в теории и практике программирования : сборник материалов конференции, Санкт-Петербург, 23 апреля 2020 года / Санкт-Петербургский политехнический университет Петра Великого; Dell Technologies; EPAM Systems. – Санкт-Петербург: ПОЛИТЕХ-ПРЕСС, 2020. – С. 165-167. – EDN VOXGIM.
8. Shemyakinskaya, A. S. Hard drives monitoring automation approach for Kubernetes container orchestration system / A. S. Shemyakinskaya, I. V. Nikiforov // Proceedings of the Institute for System Programming of the RAS. – 2020. – Vol. 32, No. 2. – P. 99-106. – DOI 10.15514/ISPRAS-2020-32(2)-8. – EDN ABZBLX.
9. Oracle Java. [Электронный ресурс] URL: https://www.oracle.com/java/
10. Yandex Cloud Functions. [Электронный ресурс] URL: https://cloud.yandex.ru/services/functions.

УДК 004.9

П. В. Елисеев (1 курс магистратуры),
А. Н. Борисов, ст. преподаватель

СЕРВИС АВТОМАТИЗАЦИИ И ПРОДВИЖЕНИЯ В СОЦИАЛЬНОЙ СЕТИ «ВКОНТАКТЕ»

На сегодняшний день социальные сети глубоко внедряются во множество сфер жизни человеческого общества. Спектр их применения только растет: от повседневного общения и распространения развлекательного контента до организации корпоративного взаимодействия, и интернет-коммерции. При этом вне зависимости от конкретной социальной сети, как правило, в основе её использования с целью решения коммерческих задач лежит принцип увеличения аудитории, связанной с тем или иным аккаунтом.

Зачастую в этом контексте пользователю, который управляет подобным аккаунтом, требуется выполнять повторяющиеся рутинные действия. Их выполнение может быть не привязано к временной зоне данного пользователя, что может создать дополнительные сложности в реализации поставленных задач, приводящие к снижению общей эффективности.

Разрешение подобных проблем приводит к потребности в автоматизации работы пользователя с социальной сетью. В свою очередь, на территории Российской Федерации большой популярностью обладает социальная сеть «ВКонтакте». На фоне тенденций к импортозамещению именно она была выбрана для создания сервиса автоматизации и продвижения.

Для реализации такого сервиса я постарался использовать подход Clean Architecture [1] и микросервисную архитектуру.

Серверную часть приложения составляет микросервис, который реализует интерфейс взаимодействия с логикой приложения через API [2] и содержит в себе специальный Worker для выполнения распределенных во времени задач. Для этого я использовал язык программирования PHP [3] и адаптацию веб-приложения на фреймворке Laravel [4].

Бизнес-логика серверной части построена на нескольких созданных пакетах: VK-API; VK-automatic, VKFLOW-Core. Так получается структурная схема, показанная на Рис. 1.

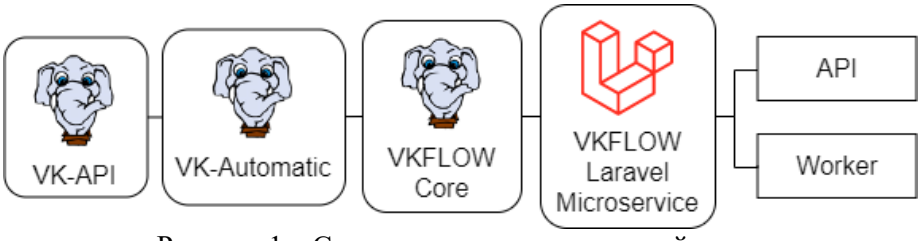


Рисунок 1 – Структурная схема серверной части

Клиентскую часть я построил на основе собственной мультисервисной платформы, в основе которой лежит приложение на Laravel. Оно передает пользователю JavaScript

приложение, выполненное с помощью ReactJS и стилизованное посредством TailwindCSS [5]. Для взаимодействия браузерного приложения с API микросервиса был использован созданный в прошлом проекте NPM-пакет MSVC-API. Структурная схема клиентской части представлена на Рис. 2.

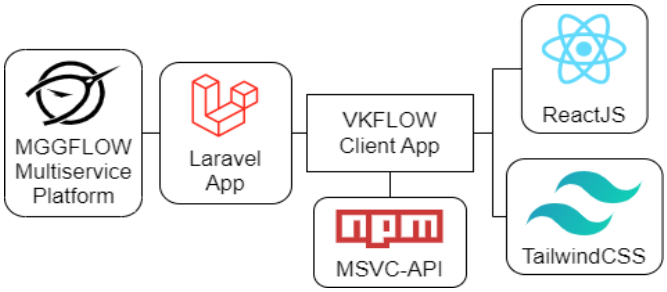


Рисунок 2 – Структурная схема клиентской части

С учетом ограниченности ресурсов сервера, который является хостом, было принято решение развернуть Worker обслуживания аккаунтов сервиса ещё и на домашнем сервере. Для этого был создан соответствующий контейнер Docker [6]. За счет этого была повышена отказоустойчивость системы: в случае, если домашний сервер недоступен, задачи автоматизации в меньших объемах выполняются на сервере хостинга; а в случае, если оба сервера в рабочем состоянии, то большую часть распределенных задач выполняет Worker в контейнере. Полученная общая структурная схема отображена на Рис. 3.

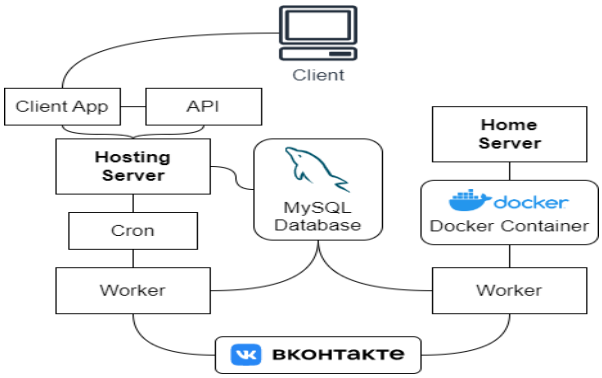


Рисунок 3 – Общая структурная схема

Сервис был запущен в формате минимально жизнеспособного продукта и сейчас доступен для общего бесплатного использования. Тестирование сервиса в целом и его алгоритмов проводилось на 8 специально созданных аккаунтах. За общее время работы суммарное количество друзей достигло 47 000. При этом на этих страницах не было периодических публикаций и специального контента за исключением начального оформления и добавления нескольких публикаций.

Однако стоит отметить, что число 47 000 – это не чистая аудитория, так как оно не учитывает пересечения друзей разных аккаунтов. Также при работе была обнаружена проблема заморозки страниц. Хотя алгоритмы сервиса настроены таким образом, чтобы не превышать лимиты сети. Основной причиной таких блокировок являются жалобы со стороны пользователей, к которым поступают заявки в друзья от незнакомых аккаунтов.

В качестве основного решения этой и подобных проблем выступает повышение правдоподобности поведения управляемых аккаунтов. Например, добавление возможности отвечать на входящие сообщения. Также будет не обойтись без мониторинга аккаунтов.

Таким образом, был создан и развернут сервис автоматизации и продвижения в социальной сети «ВКонтакте». Была рассмотрена эффективность его работы и были определены ориентиры его развития.

ЛИТЕРАТУРА

1. Мартин Р. Чистая архитектура. Искусство разработки программного обеспечения. СПб.: Питер, 2018, с. 203.

2. Тельо-Родригес М., Очаран-Эрнандес Х.О., Перес-Арриага Х.К., Лимон К., Санчес-Гарсия А.Х. ПУТЕВОДИТЕЛЬ ПО ПРОЕКТИРОВАНИЮ УДОБНЫХ WEB-API // Труды ИСП РАН. 2021. №1. URL: https://cyberleninka.ru/article/n/putevoditel-po-proektirovaniyu-udobnyh-web-api (дата обращения: 03.03.2023).
3. Никсон Робин, Создаем динамические веб-сайты с помощью PHP, MySQL, JavaScript, CSS и HTML5. 5-е изд. СПб.: Питер, 2019, с. 63.
4. Стаффер Мэп, Laravel. Полное руководство. 2-е изд. - СПб.: Питер, 2020, с. 30-35.
5. Берьянов Максим Сергеевич, Салахов Илья Равильевич, Иванов Михаил Дмитриевич ИССЛЕДОВАНИЕ СОВРЕМЕННОГО СТЕКА REACT РАЗРАБОТКИ В 2022 ГОДУ // Столыпинский вестник. 2022. №8. URL: https://cyberleninka.ru/article/n/issledovanie-sovremennogo-steka-react-razrabotki-v-2022-godu (дата обращения: 03.03.2023).
6. Иан Милл, Эйдан Хобсон Сейерс, Docker на практике. Перевод с англ. Д. А. Беликов. М.: ДМК Пресс, 2020, с. 22-26.

УДК 004.021

В. Д. Ершов (4 курс бакалавриата),
Н. Г. Смирнов, ассистент

РАЗРАБОТКА ПЕСОЧНИЦЫ ПО ОПЛАТЫ НА ОСНОВЕ СПБ B2B И ОПЛАТА ПО QR-КОДАМ

СБП B2B дает возможность замены текущих платежных поручений в банке между счетами корпоративных клиентов на более совершенное и быстрое решение, которое не будет занимать много времени для клиентов. Текущее решения с платежками занимает от 1–3 дней, но оплата через СБП B2B занимает 1–3 минуты. Также это позволяет совершать платежи простым методом, через сканирование QR-кода, который может быть статическим, динамическим. QR-код генерируется получателем, в зависимости от типа по нему может быть несколько платежей или только один, после этого QR-код становится недействительным. Отправитель сканирует QR-код, его перенаправляют на выбор банка, с которого можно произвести оплату, после выбора банка открывается приложение банка, и клиент может совершить платеж.

Многим корпоративным клиентам не удобно, когда перевод средств занимает несколько дней, в течение этого времени клиенты не могут пользоваться данными средствами. Также многие сталкиваются со сложность создания платежных поручений, и оплаты по ним. Данное решение не только ускоряет процесс оплаты, но и дает максимально масштабирование своих платежей: создания для каждого клиента уникальный QR-код с фиксированной суммой и налогом, а также статического QR-кода, по которому клиенты смогут самостоятельно оплачивать в любой момент и на любую сумму.

Многие компании активно начали пользоваться данным функционалом через банковское приложение. Но у каждой компании есть возможность интеграции с сервисом оплаты через REST API, это позволяет клиентам самим регулировать используемые технологии, реализовывать свой интерфейс, который будет удобен своим сотрудникам и тд.

Для написания сервера был выбран язык программирования Kotlin, фреймворк Spring и база данных PostgreSQL. Выбраны современный и популярные технологии, которые начали активно внедряться в разработку серверных приложений.

Для разработки было решено использовать среду разработки IntellijIDEA, которая считается идеальной ide для написания кода подобных приложений.

Для отслеживания и сохранения истории изменений кода выбрана система контроля версий git с платформой GitLab. Это популярный и простой инструмент, который позволяет в случае ошибки вернуться к более ранней версии кода, а также просмотреть историю изменений в виде списка или графика.

Основой данного этапа заключается разработка песочницы для клиентов, в который они

смогут начать процесс интеграции и настройки своих приложений. Посмотреть весь функционал, без реального создания платежей и перевода денежных средств. Реализованы получатель и отправить со своими REST API методами (GET, POST). В базе данных хранятся все созданный QR-код и платежи, который были совершены.

Фреймворк Spring MVC обеспечивает архитектуру паттерна Model — View — Controller (Модель — Отображение (далее — Вид) — Контроллер) при помощи слабо связанных готовых компонентов. Паттерн MVC разделяет аспекты приложения (логику ввода, бизнес-логику и логику UI), обеспечивая при этом свободную связь между ними.

- Model (Модель) инкапсулирует (объединяет) данные приложения, в целом они будут состоять из POJO («Старых добрых Java-объектов», или бинов).
- View (Отображение, Вид) отвечает за отображение данных Модели, — как правило, генерируя HTML, которые мы видим в своём браузере.
- Controller (Контроллер) обрабатывает запрос пользователя, создаёт соответствующую Модель и передаёт её для отображения в Вид.

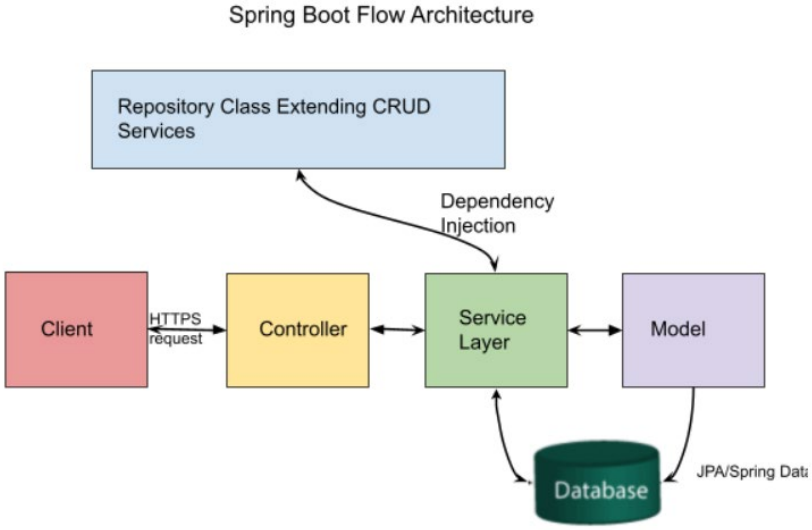


Рисунок 1 – Архитектура приложения

ЛИТЕРАТУРА

1. Spring Boot – Architecture. [Электронный ресурс] https://www.geeksforgeeks.org/spring-boot-architecture/ (Дата обращения: 23.01.23)
2. Spring MVC — основные принципы. [Электронный ресурс] https://habr.com/ru/post/336816/ (Дата обращения: 23.01.23)
3. GitLab https://about.gitlab.com/
4. СПБ B2B. [Электронный ресурс] https://xn--90aifddrld7a.xn--p1ai/smi/s-2022-goda-banki-zapuskayut-perevody-mezhdu-yurlitsami-cherez-sbp (Дата обращения: 23.01.23)
5. НСПК. [Электронный ресурс] https://sbp.nspk.ru/api/ (Дата обращения: 23.01.23)

УДК 004.415.25

К. А. Жукова (2 курс магистратуры),
С. А. Молодяков, д.т.н., доцент

РАЗРАБОТКА СИСТЕМЫ ПОДБОРА ЛЕКАРСТВЕННЫХ ПРЕПАРАТОВ

На сегодняшний день, когда цифровые технологии активно внедряются в сфере здравоохранения, существует потребность в создании медицинских информационных систем (МИС) для российских медучреждений [1]. В 2019 г. был составлен прогноз тенденций развития МИС в России [2], и большое внимание в нем уделяется развитию систем поддержки принятия врачебных решений (СППВР). В зависимости от потребностей конкретного

медучреждения и специалиста эти системы могут быть реализованы по-разному и могут выполнять разные функции. Например, СППВР может опираться на алгоритмы машинного обучения, а может представлять из себя систему сбора, анализа или визуализации данных. СППВР предназначены быть вспомогательным звеном, упрощающим и ускоряющим работу специалиста.

Целью данной работы является разработка системы подбора лекарственных препаратов (ЛП) по заданной болезни.

Актуальность этой задачи обосновывается тем, что при огромном объеме информации и новых данных для врачей зачастую бывает сложно отследить момент, когда у назначаемого препарата меняется форма выпуска, появляется аналог или новый эффективный препарат. Разрабатываемая система может значительно упростить поиск актуальной информации. При этом важно, чтобы болезни, задаваемые системе, классифицировались в соответствии с международными стандартами, чтобы врач мог задать название в привычной стандартной форме, то есть в соответствии с международной классификацией болезней (МКБ). Также необходимо, чтобы препараты, предлагаемые системой по заданной болезни, были включены в список ЛП, утвержденных для использования в РФ, то есть были включены в государственный реестр лекарственных средств для медицинского применения (ГРЛС).

Для выполнения поставленной задачи был проведен анализ возможных программных решений. Рассматривалась возможность разработки рекомендательной системы [3,4], а также системы сбора и обработки данных. Выяснилось, что система сбора и обработки данных позволит получить более точные результаты, поэтому была реализована именно она.

Система состоит из двух блоков. Первый представляет из себя web-скрапер, получающий информацию о том, какие действующие вещества для лечения каких болезней применяются.

Второй блок является основным – по действующему веществу он осуществляет поиск ЛП, включенных в ГРЛС, и формирует список торговых наименований препаратов, в основе которых лежит заданное действующее вещество, с указанием этого вещества, а также информацией о дозировках и форме выпуска.

Блоки могут работать независимо друг от друга. Наиболее целесообразно единожды собрать информацию (запустить первый блок), а потом по необходимости запускать поиск ЛП (то есть второй блок).

Аналогами разработанной системы являются интернет-справочники с возможностью поиска лекарства по болезни. Помимо привязки к конкретному региону, аптеке или производителю, сложность работы с ними часто заключается в разрозненности предоставляемой информации. Но основной проблемой является то, что вся информация представлена посредством веб-интерфейса, то есть в виде, удобном для просмотра через браузер. Не всегда у врача есть стабильный доступ в интернет с рабочего компьютера, поэтому было бы удобно, если бы часть функций поиска работала бы локально вне зависимости от интернета. Кроме того, врачу было бы удобно иметь доступ к списку актуальных препаратов изнутри той МИС или того интерфейса, которым он уже пользуется.

Все эти возможности может предоставить разработанная система подбора ЛП. Подключение к интернету требуется только для работы первого блока (сборщика информации по действующим веществам), а конкретные поисковые запросы по болезни могут обрабатываться локально. Также система имеет минимум внешних зависимостей и ее не составит труда встроить в более крупную систему. Кроме того, информация, собранная системой на каждом этапе, представляется в удобном для дальнейшей обработки виде – ее легко можно отобразить в интерфейсе или использовать для работы другой системы.

Для разработки был выбран язык Python версии 3.7.

Web-скрапер получает информацию о том, какие действующие вещества для лечения каких болезней применяются, из интернет-справочника ГЭОТАР [5], где болезни классифицированы в соответствии с МКБ-10. HTTP-запросы выполняются с помощью библиотеки requests, а анализ и извлечение данных из HTML-документов – с помощью пакета

BeautifulSoup версии 4. Собранная информация сохраняется в виде json-файла для дальнейшего использования.

Второй блок использует этот файл и Единый справочник-каталог ЛП (ЕСКЛП), который представлен в виде Excel-таблицы, для поиска конкретных ЛП. Для работы с Excel-таблицей не подключаются отдельные специализированные библиотеки, такие как openpyxl или xlrd, а применяются только средства библиотеки Pandas, которая далее используется для обработки полученного массива данных. Результат работы блока сохраняется также в json-файл – таким образом доступ к информации на любом этапе прост и удобен, ее легко отобразить в интерфейсе или передать в другой блок, если система подбора лекарств является частью более крупной системы. Работа системы с json-файлами производится с помощью библиотеки json.

Система была протестирована с использованием Pytest.

Таким образом, в ходе работы были рассмотрены возможные программные решения для выполнения поставленной задачи и выбрано наиболее подходящее – система сбора и обработки данных. Система подбора лекарственных препаратов была разработана с минимальным количеством внешних зависимостей, после чего было проведено ее тестирование. Разработанная система позволяет, задав болезнь в соответствии с классификацией МКБ-10, получить актуальный список лекарственных препаратов, включенных в ГРЛС, причем работа системы не требует подключения к интернету на этапе обработки поисковых запросов, то есть основная часть работы выполняется локально. Следовательно, разработанная система удовлетворяет заданным критериям и может облегчить поиск актуальной информации о лекарственных препаратах, применяемых в РФ.

ЛИТЕРАТУРА

1. Пугачев, П. С. Мировые тренды цифровой трансформации области здравоохранения / П. С. Пугачев, А. В. Гусев, О. С. Кобякова, Ф. Н. Кадыров, Д. В. Гаврилов, Р. Э. Новицкий, А. В. Владзимирский // Национальное здравоохранение. – 2021. – Т. 2. – № 2. – С. 5-12.
2. Гусев, А. В. Тренды и прогнозы развития медицинских информационных систем в России / А. В. Гусев, М. А. Плисс, М. Б. Левин, Р. Э. Новицкий // Врач и информационные технологии. – 2019. – №2. – С. 38-49.
3. Ковалев, И. Ю. Эффективность алгоритмов коллаборативной фильтрации / И. Ю. Ковалев // Вестник студенческого научного общества ГОУ ВПО "Донецкий национальный университет". – 2022. – Т. 1. – № 14. – С. 156-159.
4. Жуманова, А. О. Виды рекомендательных систем и их простейшие алгоритмы / А. О. Жуманова, Л. К. Найзабаева // Актуальные научные исследования в современном мире. – 2021. – № 5-2(73). – С. 88-90.
5. ЛС ГЭОТАР – Электронный лекарственный справочник. – URL: https://www.lsgeotar.ru/ (дата обращения 07.03.2023)

УДК 004.42

Р.А. Золотарев (4 курс бакалавриата),
А. В. Самочадин., к.т.н., доцент

ВИЗУАЛИЗАЦИЯ И АНАЛИЗ ГЕОГРАФИЧЕСКИХ ДАННЫХ

Для создания карт морского дна применяются методы исследования, базирующиеся на высокоточной измерительной аппаратуре и сложном программном обеспечении — геоинформационных системах (ГИС) [1]. Применение ГИС-технологий дает совершенно новые возможности для решения любых научных и практических задач в этой области. В этой связи необходимы практические шаги по исследованию и картографированию подводного рельефа. Для различных задач требуются карты рельефа дна разного пространственного разрешения и точности. В частности, для научных исследований незначительных по площади, но имеющих большое практическое или природоохранное значение, морских объектов значительно повышаются требования к детальности и наглядности карт морского дна.

Целью данной работы является реализация сервиса, который будет представлять функционал обработки и визуализации батиметрических логов морского дна.

Для достижения этой цели необходимо решение следующих задач:

1. Обзор существующих решений, - GIS систем, с функционалом визуализации батиметрических данных.
2. Проведение сравнительного анализа, найденных решений.
3. Выбор необходимых технологий и алгоритмов.
4. Проектирование схемы базы данных.
5. Реализация сервиса.
6. Нагрузочное и интеграционное тестирование реализованного сервиса.

Разрабатываемый сервис является частью большой системы, общий архитектурный план сервиса и взаимодействующих с ним элементов системы представлен с помощью диаграммы на Рис. 1.

Сервис предоставляет свой функционал в форме API, являющимся ASGI приложением, написанным на языке программирования Python.

Для построения сервиса применялся архитектурный паттерн Domain Driven Design. DDD [2] — это набор принципов и схем, направленных на создание оптимальных систем объектов. Сводится к созданию программных абстракций, которые называются моделями предметных областей. В эти модели входит бизнес-логика, устанавливающая связь между реальными условиями области применения продукта и кодом. В программном коде, через ООП реализованы 3 типа субдоменов [2]:

– Input - отвечает за хранение и форматирование входных данных
– Processor – объекты этого типа реализуют в себе части бизнес-логики, связанной с процессингом геоданных и хранением промежуточной информации
– Output – отвечает за форматирование ответа сервиса

Важным компонентом системы является спроектированная схема базы данных, реализованная в СУБД PostgreSQL и с применением расширения Postgis. В базу данных записывается промежуточная информация процесса обработки, конвертированные координаты точек батиметрии, а также создаются бизнес сущности для отображения в GIS-системе. Помимо этого, сервис оперирует ресурсами удаленного дискового пространства, записывая в файловую систему результаты обработки в различных форматах.

Для реализации алгоритма 2D визуализации использовалась утилита PostGIS и метод регулярных сеток. 3D визуализация включала в себя использование утилит Open3D и Blender API, с основным алгоритмом Poisson reconstruction или методом неявного построения сетки.

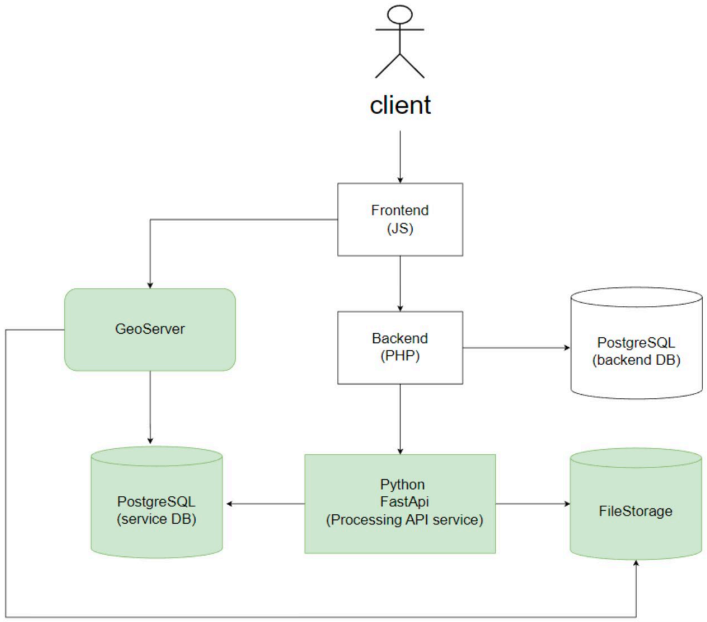


Рисунок 1 – Общая архитектура проекта

Общий архитектурный стек системы выглядит следующим образом:

В качестве языка программирования использован Python 3.10. Для реализации API используется асинхронный фреймворк FastApi [3]. Для доступа к базе данных используются асинхронная ORM – SqlAlchemy, в сочетании с инструментом сериализации – Alembic.

Для реализации ядра обработки и визуализации батиметрии был выбран программный комплекс GDAL [4], способный с высокой степенью параметризации выполнять задачи растеризации [5] батиметрии быстро и качественно. Для 3d визуализации применялись утилиты Blender API [6] и Open3D [7]

Для хранения данных была выбрана СУБД PostgreSQL, которая отлично подходит под формат хранения географических данных. Также, для расширения возможностей СУБД в рамках хранения географических данных, используется специальный плагин PostGIS [8], реализующий в себе огромное количество полезных функций для запросов и создания таблиц.

ЛИТЕРАТУРА

1. Введение в геоинформационные системы: учебное пособие / Я.Ю. Блиновская, Д.С. Задоя – 2022 (дата обращения: 03.09.2022)
2. Clean Architecture. A Craftsmans Guide to Software Structure and Design / Robert Martin (дата обращения: 01.09.2022)
3. FastAPI documentation. [Электронный ресурс] Режим доступа: https://fastapi.tiangolo.com
4. GDAL documentation. [Электронный ресурс] Режим доступа: https://gdal.org
5. Efficient Point Cloud Rasterization for Real Time Volumetric Integration in Mixed Reality Applications // IEEE. - 2019 (обращения: 25.11.2022)
6. Blender 3.4 Python API Documentation. [Электронный ресурс] Режим доступа: https://docs.blender.org/api/current/index.html
7. Open3D documentation. [Электронный ресурс] Режим доступа: http://www.open3d.org/docs/release
8. PostGIS Documentation. [Электронный ресурс] Режим доступа: https://postgis.net/documentation

УДК 004.4

Н. А. Казимиров (4 курс бакалавриата),
Т.В. Леонтьева, к.т.н., доцент

СОЗДАНИЕ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ УПРАВЛЕНИЯ ПРОФЕССИОНАЛЬНЫМИ РИСКАМИ НА ПРЕДПРИЯТИИ

В производственной сфере всегда присутствует риск возникновения нештатных ситуаций, и для обеспечения безопасности сотрудников необходимо иметь эффективную систему управления профессиональными рисками. Идея создания данной системы возникла после прочтения различных новостей о несчастных случаях и ЧП на предприятиях. Согласно подсчетам Международной Организации Труда ежегодно на различного рода предприятиях погибают около 2,3 миллионов мужчин и женщин в результате несчастных случаев или воздействия вредных факторов производственной среды. Наиболее распространёнными причинами травмирования сотрудников являются:

– технические причины (конструктивные недостатки и плохое техническое состояние оборудования);

– организационные причины (нарушение правил эксплуатации оборудования, недостатки содержания территории и рабочих мест);

– санитарно-гигиенические причины.

Целью работы является разработка веб-приложения для управления профессиональными рисками. На данный момент у сотрудников большинства предприятий уходит большое количество времени и сил на идентификацию и анализ рисков. А весь документооборот происходит на бумажных носителях, что ведет к тому, что на

классифицирование выявленных рисков и разработку мероприятий для их устранения требуются значительные трудозатраты от всех участников процесса управления профессиональными рисками. Для достижения поставленной цели необходимо решить следующие задачи:

– изучить предметную область;
– провести анализ существующих решений;
– составить требования к системе;
– спроектировать и разработать архитектуру системы;
– спроектировать и разработать веб-приложение;
– протестировать работу всей системы.

В создаваемой системе пользователю будут доступны вкладки “Реестр рисков”, “Мероприятия”, “Аналитика” и личный кабинет пользователя. Все риски, выявленные и устраненные за последний месяц, а также их текущее состояние отображаются на вкладке “Реестр рисков”. На этой же вкладке предусмотрена следующая основная функциональность:

– фильтрация данных в реестре рисков;
– внесение в реестр выявленных рисков;
– просмотр и редактирование подробной карточки риска;
– добавление корректирующих мероприятий в подробной карточке риска;

Вкладка “Мероприятия” предназначена для хранения проведённых мероприятий по воздействию на профессиональный риск. Вкладка “Аналитика” предназначена для эффективного анализа предпринимаемых мероприятий направленных на устранение рисков. На этой же странице можно посмотреть графическую информацию о рисках, получить статистические сведения о выявленных и устранённых рисках, а также узнать, какова степень влияния и среднее время необходимое на устранение угроз.

Разрабатываемая система позволит оптимизировать документооборот, а также приведет к снижению рисков возникновения нештатных ситуаций, повышению оперативности реагирования и принятию корректирующих мер по устранению профессиональных рисков.

В настоящее время доступная информация о подобных приложениях и системах ограничена. Большинство программных продуктов сфокусированы на управлении финансовыми, проектными и операционными рисками, но лишь часть из них оценивает профессиональные риски на предприятиях различных отраслей. Например, существует решение от компании «1С», под названием «1С:Управление холдингом 8. Управление рисками и мероприятиями», которое позволяет работать с профессиональными рисками, но данная подсистема может не подойти многим малым и средним предприятиям ввиду ее сложности и дороговизны.

Реализуемое приложение основывается на трехуровневой клиент-серверной архитектуре, в которую включены серверная и клиентская часть приложения, а также сервер базы данных.

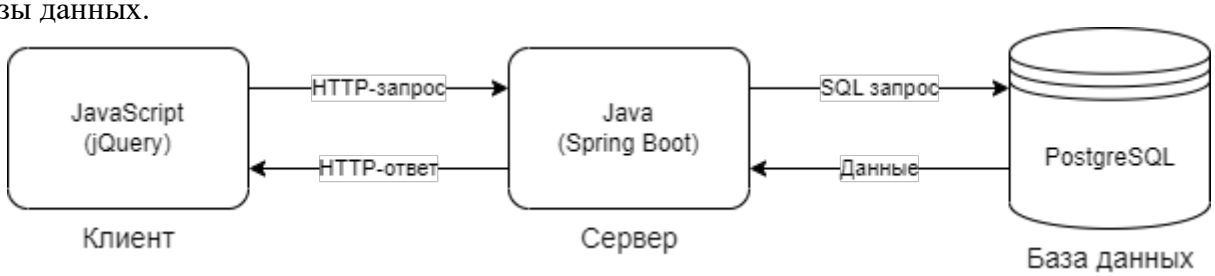


Рисунок 1 – Архитектура приложения

Для реализации серверной части был выбран язык программирования Java и фреймворка Spring Boot. Spring Boot – это один из наиболее популярных фреймворков для создания различных веб-приложений на языке Java.

Для создания клиентской части приложения используется язык программирования JavaScript и фреймворк jQuery, который упрощает написание JavaScript-кода. Взаимодействие клиентской части и серверной будет организовано с помощью REST API.

Для хранения данных используется СУБД PostgreSQL. Данная СУБД имеет открытый исходный код и хорошую документацию, а также поддерживает сложные структуры данных.

Разрабатываемая система управления профессиональными рисками будет представлять собой решение, которое отлично подойдет для небольших компаний. Основными преимуществами будут простота и удобство использования, а также наличие всего необходимого функционала для оценки и контроля профессиональных рисков.

ЛИТЕРАТУРА

1. Мировая статистика. [Электронный ресурс] Режим доступа: https://www.ilo.org/moscow/areas-of-work/occupational-safety-and-health/
2. Оценка и управления профессиональными рисками [Электронный ресурс]. Режим доступа: https://eisot.rosmintrud.ru/otsenka-i-upravlenie-professionalnymi-riskami
3. Виды и причины несчастных случаев на производстве [Электронный ресурс]. Режим доступа: https://www.protrud.com/
4. PostgreSQL Documentation [Электронный ресурс]. Режим доступа: https://www.postgresql.org/docs/
5. Spring Boot [Электронный ресурс]. Режим доступа: https://spring.io/
6. jQuery [Электронный ресурс]. Режим доступа: https://jquery.com/

УДК 004.4

А. Е. Карпов (4 курс бакалавриата),
А. П. Маслаков, ст. преподаватель

РАЗРАБОТКА НОВОСТНОГО САЙТА С СИСТЕМОЙ ОТБОРА И ОТСЕИВАНИЯ НЕДОСТОВЕРНОЙ ИНФОРМАЦИИ

На сегодняшний день на просторах интернета каждый день создаются новые новостные и информационно-развлекательные порталы откуда любой желающий может получить сведения по интересующим его вопросам. Однако, несмотря на широкий доступ к информации, в сети также присутствует множество посредственных сайтов, на которых публикуется непроверенная и неотфильтрованная информация. Несмотря на усилия модераторов и правоохранительных органов, в сети Интернет все еще можно встретить множество ошибок и неточностей.

В этой связи возникает необходимость в создании надежных и качественных новостных сайтов, которые бы предоставляли пользователям только проверенную и достоверную информацию. Для этого необходимо разработать новостной портал, который бы позволял пользователям самостоятельно выбирать интересующую их категорию новостей и получать доступ к проверенным и достоверным источникам информации.

На основании вышеперечисленного были сформулированы требования к системе. Пользователи могут жаловаться на недостоверные публикации, предоставляя доказательства. Пользователи с высоким рейтингом (100 и более) могут просматривать подозрительные публикации и получать бонусы за верное определение недостоверности. Если публикация ошибочно заблокирована, она возвращается в список открытых публикаций, а рейтинг пользователя, который на нее жаловался, снижается. Пользователи могут терять возможность жаловаться или оценивать, если их рейтинг становится равен 0. Авторы публикаций могут лишаться возможности публиковать при частых нарушениях. Для поддержания интереса пользователей создается страница с рейтингами всех пользователей. В ходе выполнения проектной работы был проведен социальный опрос, в котором было выяснено, что заинтересованность пользователей в выявлении недостоверной информации повышается при добавлении их рейтингов в открытый доступ.

Приложение разрабатывается на основе клиент-серверной архитектуры. На Рисунке 1 приведено схематическое изображение модулей данного программного продукта.

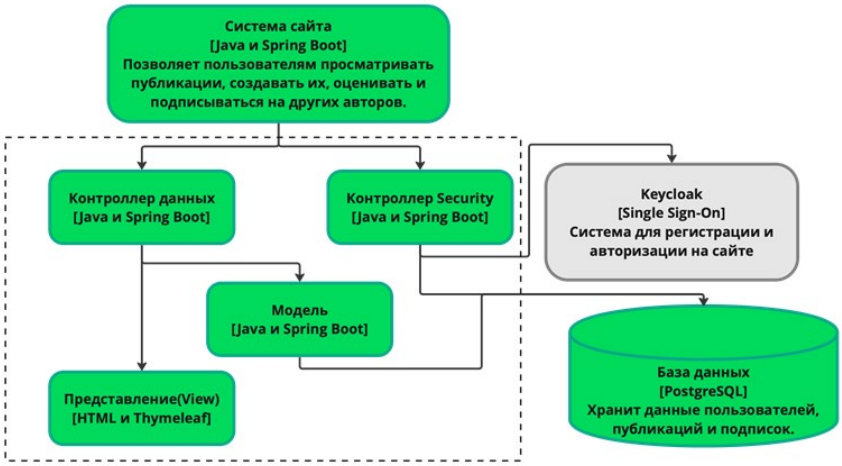


Рисунок 1 – архитектура приложения

Серверная часть приложения реализована на языке Java 15 [4] с использованием фреймворка Spring [1]. В качестве системы хранения выбрана база данных PostgreSQL, взаимодействие с которой происходит с помощью Spring Data JPA.

Для регистрации и авторизации была использована современная платформа для управления идентификацией и доступом Keycloak [2].

В проекте присутствует четыре контроллера, соответственно отвечающих за:

1. MainContorller - правильное отображение информации на страницах.
2. WorkController - создание, обновление и удаления информации в базе данных.
3. AdminController - работу администраторов.
4. FakeController - данные о недостоверной информации и все что с ней связано.

Spring Data понадобился для интеграции Keycloak с PostgreSQL, поэтому данный модуль подключается к двум базам данных, одна из которых отвечает за авторизацию и регистрацию пользователей.

В проекте RESTfull API формируется с помощью Spring MVC. Запросы передаются в контроллеры в зависимости от их направления (пользователи, публикации, алминистраторы и недостоверности). Формат запросов - JSON.

Клиентская часть приложения реализована в виде веб-интерфейса, созданного при помощи языка разметки HTML 5.3, CSS3, языка JavaScript и фреймворка Thymeleaf [3].

Архитектура фронтенда состоит из трех модулей, которые отвечают за соответствующие части приложения:

1. Publications Editor предназначен для добавления, редактирования, удаления и оценивания публикаций.
2. Users Editor используется для заполнения, обновления и удаления информации о пользователях, а также создание подписок между ними.
3. Fakes Editor отвечает за работу с данными о недостоверной информации в публикациях и на страницах пользователей.

Главным результатом создания данного программного продукта должна стать заинтересованность пользователей в более внимательном изучениии информации от незнакомых источников и самостоятельном выявлении недостоверности.

ЛИТЕРАТУРА

1. Spring//[документация]/ — URL: https://spring.io/ (дата обращения:12.12.2022).
2. Keycloak//[документация]/ — URL: https://www.keycloak.org/docs/latest/server_admin/ (дата обращения:25.01.2023).
3. Учебник Thymeleaf// [научная статья]/ — URL: https://habr.com/ru/post/350864/ (дата обращения:31.01.2023).

4. ava//[документация]/ — URL: https://www.oracle.com/java/technologies/javase/15all-relnotes.html (дата обращения:12.02.2023).
5. Томас Коннолли, Каролин Бегг Базы данных. Проектирование, реализация и сопровождение. Теория и практика. - 1-е изд. - Москва: Вильямс, 2003. - 1436 с.

УДК 004.6

К. А. Каширин (4 курс бакалавриата),
И. В. Шошмина, к.т.н., доцент

РАЗРАБОТКА СИСТЕМЫ ХРАНЕНИЯ И ВЫГРУЗКИ ДАННЫХ КИОТ НА ОСНОВЕ МЕЖСИСТЕМНОГО ВЗАИМОДЕЙСТВИЯ С ПЛАТФОРМОЙ MOODLE

Конструктор индивидуальных образовательных траекторий (КИОТ) – система, предназначенная для построения индивидуальных образовательных траекторий на платформе «Moodle» [1]. Чтобы выполнять автоматический подбор элементов дисциплин, необходимо непрерывно взаимодействовать с платформой «Moodle»: от выгрузки и сохранения данных курсов с платформы, до сокрытия модулей курса для студента в соответствии с его индивидуальной образовательной траекторией.

Целью данной работы является создание и настройка средств хранения данных, разработка методов автоматической выгрузки и обработки данных курсов с платформы дистанционного образования «Moodle» для сервиса конструктора индивидуальных образовательных траекторий.

Задачи:

1. Создание архитектуры базы данных сервиса.
2. Реализация сущностей основных модулей в базе данных.
3. Реализация плагина к платформе «Moodle» с функциями ограничения доступа к модулям курса посредством внешней службы системы.
4. Реализация методов автоматической выгрузки, сохранения и выдачи данных с платформы «Moodle».
5. Реализация связи между сущностями основных модулей в backend части сервиса.

В качестве используемой системы управления базами данных (СУБД) на момент активной разработки используется реляционная H2 [2], так как она легко интегрируется с Java-приложениями. Взаимодействие с платформой «Moodle» производится посредством протокола REST API [3]. Платформа позволяет авторизованным пользователям с определенными правами доступа осуществлять HTTP [4] запросы к ограниченному числу её сервисов. Для активного взаимодействия сервиса КИОТ с сервисами «Moodle» реализован плагин на языке программирования PHP, добавляющий возможность использования функций платформы. В плагине реализованы функции, расширяющие функционал платформы: создание локальных групп курса для формирования групп студентов с открытым доступом к определенному модулю, добавление студента в определенную локальную группу курса и открытие доступа модуля курса только для определенной группы. Использование средств плагина осуществляется через зарегистрированного пользователя с указанием необходимых прав доступа.

В результате работы реализованы все необходимые методы для автоматической выгрузки, сохранения и выдачи данных с платформы «Moodle», позволяющие строить для обучающего индивидуальную образовательную траекторию. Предстоит оптимизировать часть запросов сервиса к платформе, а также перейти на объектно-реляционную СУБД PostgreSQL, обладающую большими возможностями расширения и оптимизации по сравнению с H2.

ЛИТЕРАТУРА

1. Moodle – Open-source learning platform. [Электронный ресурс] Режим доступа: https://moodle.org/
2. H2 Database Engine. [Электронный ресурс] Режим доступа: https://h2database.com/

3. Moodle Web service API functions. [Электронный ресурс] Режим доступа: https://docs.moodle.org/dev/Web_service_API_functions
4. Hypertext Transfer Protocol -- HTTP/1.1. [Электронный ресурс] Режим доступа: https://ietf.org/rfc/rfc2616.txt

УДК 004.421

И. М. Киряков, аспирант,
С. А. Молодяков, д.т.н., доцент

ПОСТРОЕНИЕ МОДУЛЬНОЙ МЕДИЦИНСКОЙ ИНФОРМАЦИОННОЙ СИСТЕМЫ НА ПРИМЕРЕ РАЗРАБОТКИ ПРОГРАММЫ ПОДБОРА ЛЕКАРСТВЕННОГО ПРЕПАРАТА

Рассматриваются вопросы построения модульных медицинских информационных систем, в которых обеспечивается возможность не только подключения новых, но и замена старых модулей новыми более совершенными модулями. Программное обеспечение медицинской системы разрабатывается с использованием микросервисной архитектуры. Представлена архитектурная схема системы. Проведен анализ технологий обмена данными, проведено тестирование, выделена технология удаленного вызова процедур gRPC. Разработаны шаблон и первые модули, связанные с задачей подбора лекарственного препарата.

На Рис. 1 представлена схема медицинского модульного комплекса, который, предназначенный для автоматизации процесса медицинского обслуживания пациента.

Рисунок 1 – Архитектура системы

Данный модульный комплекс будет иметь возможность динамического приседания и отсоединения нужных модулей, которые могут быть использовано как последовательно, так и напрямую по отдельности. Каждый модуль представляет самостоятельную единицу, которая в полной мере выполняет свою задачу и имеет возможность передавать нужные данные по API «следующему модулю». В качестве «следующего модуля» может выступать модуль Диспетчер, который будет поддерживать интерфейс пользователя, организовывать вызов других модулей. В данной работе излагается разработка конкретного модуля «Подбор лекарственного препарата», также подобным образом может быть преобразована в модуль разработанная нами программа обнаружения COVID-19 [4].

При проектировании приложения на микросервисной архитектуре одним из основных вопросов является взаимодействие между ее модулями. Рассмотрены известные решения и определено наилучшее для обмена данными между приложением-клиентом и приложением-сервером. Обмен данными между приложениями относится к элементу программного интерфейса API (Application Programming Interface), который может быть вызван или выполнен на различных уровнях абстракции в системе.

Для рассмотрения были выбраны следующие используемые технологии [5, 6]:

- протокол простого доступа к объектам (SOAP)
- передача репрезентативного состояния (REST)
- удаленный вызов процедур (gRPC)

Проведено измерение времени обмена от количества передаваемых данных при разных технологиях (Рис. 2). Определено, что REST является легковесной и быстрой технологией, но проигрывает по времени при передаче больших данных по причине того, что у REST отсутствует встроенная система сериализации и десериализации данных. SOAP и gRPC обладают встроенной системой сериализации и десериализации данных, но SOAP уступает gRPC по причине того что gRPC использует более быстрый механизм конвертации данных на основе Protobuf. Использование gRPC является лучшим вариантом.

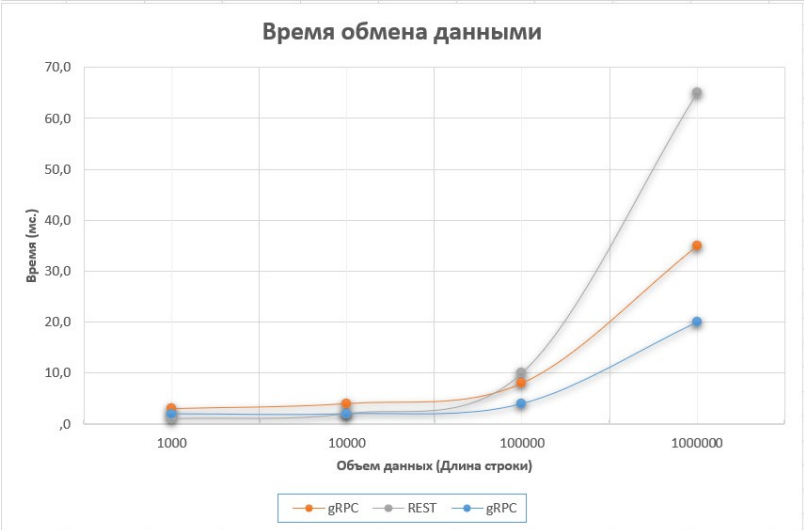


Рисунок 2 – График времени обмена данными

Представлены первые результаты разработки модуля подбора лекарственного препарата как элементы медицинской системы. В качестве API используется gRPC. Информация о лекарственном препарате берется из единого структурированного справочника лекарственных препаратов (ЕСКЛП) [7].

ЛИТЕРАТУРА

1. Мазаев В. П., Рязанова С. В. Основные направления развития искусственного интеллекта в медицине. // Научное обозрение. Медицинские науки. 2020.- №5.- С.33-40. doi: 10.17513/srms.1141.
2. А. Э. Порфильева, Р. Ф. Шайхутдинов, Нуриева Г. А. Эффективная разработка приложений при микросервисной архитектуре // Электронные библиотеки. – 2018. – Т. 21. – № 3-4. – С. 357-368. – EDN XVASHB.
3. Chris Richardson (2017). Pattern: Microservice Architecture. URL: http://microservices.io/patterns/microservices.html
4. Думаев Р. И., Киряков И. М., Молодяков С. А. Особенности предобработки и сегментации изображений в задаче обнаружения COVID-19 по рентгеновским снимкам // Современная наука: актуальные проблемы теории и практики. Серия: Естественные и Технические Науки. -2022. -№09. -С. 88-95. doi: 10.37882/2223-2966.2022.09.08

5. Различия REST и SOAP [Электронный ресурс]. – URL: https://habr.com/ru/post/483204/ (19.01.2023)
6. Авельцов, Д. О. Применение протокола сериализации структурированных данных Protobuf в микросервисной архитектуре / Д. О. Авельцов // Проблемы автоматики и управления. – 2022. – № 3(45). – С. 185-196. – EDN SJWCHJ.
7. Единый структурированный справочник лекарственных препаратов (ЕСКЛП) [Электронный ресурс]. – URL: esklp.egisz.rosminzdrav.ru

УДК 004.4

А.П. Ковалева (4 курс бакалавриата),
Т.В. Леонтьева, к.т.н., доцент

РАЗРАБОТКА МОБИЛЬНОГО ПРИЛОЖЕНИЯ - ГИДА ПО УЛИЧНОМУ ИСКУССТВУ

На текущий момент тема стрит арта не имеет широкого распространения. Очень мало проектов и, тем более, мобильных приложений, которые показывают, что уличное искусство — это не вандализм и оно может быть согласованным и красивым.

В процессе изучения рынка мобильных приложений я выделила только 3 приложения, затрагивающих тему стрит арта: “Street Wise”, “Street Art” и “AR Hunter”. Есть приложения, в которых есть интерактивная карта, есть приложения, в которых есть AR, но нет приложения, которое сочетало бы в себе эти функции и плюс ко всему этому еще рассказывало о последних новостях в мире стрит арта и об интересных проектах. Но ни одно из этих приложений не содержит в себе столько функционала как в PublicARt.

Цель работы – создать мобильное приложение на платформах Android и iOS, позволяющее окунуться в мир уличного искусства. С помощью интерактивной карты пользователь сможет найти арт объекты недалеко от себя, узнать интересную информацию о них. Некоторые работы можно “оживить” с помощью технологии дополненной реальности (AR). В разделе “Авторы” можно узнать интересные факты о деятелях стрит арта. Также в приложении имеется раздел “Новости”, в котором освещены последние новости и события мира уличного искусства. Помимо перечисленного в приложении есть специально составленные маршруты по интересным арт объектам и раздел “Проекты” в котором описаны интересные коллаборации, нацеленные на популяризацию уличного искусства и не только.

Для реализации мобильного приложения был выбран игровой движок Unity. На нем без особых усилий можно сделать кроссплатформенное приложение, которое будет работать и на Android и на iOS.

Данные для приложения (информация об арт объектах, авторах, новостях и т.д) хранятся на сервере. Взаимодействия с бекендом реализованы на основе REST API. Приложение отправляет запросы на специальные эндпоинты и получает обратно данные в формате JSON. Данные кешируются на устройстве пользователя и обновляются при необходимости. Изображения загружаются по принципу “ленивой загрузки” - загрузка начинается только тогда, когда необходимо (когда картинка должна появиться на экране).

Для построения архитектуры пользовательских интерфейсов использовался паттерн MVC - Model-View-Controller (Модель-Вид-Контроллер). У каждого экрана есть модель данных, которая на нем отображается. View (вид) отвечает за визуальную репрезентацию данных на экране. Controller (контроллер) отвечает за бизнес логику и все действия, которые происходят с данными. Пользовательские интерфейсы сделаны на новой UI системе Unity, которая называется UI Toolkit, и работает по принципу верстки в вебе.

Также для Unity существует много SDK для дополненной реальности. Я выбрала AR SDK Vuforia. Так как в приложении будет использоваться только технология отслеживания изображений (image tracking) Vuforia идеально подходит. Она бесплатная и не использует нативных SDK для отслеживания изображения, соответственно приложение будет работать на

большем количестве устройств. Фотографии граффити используются в качестве таргетов — картинок, на которые будет активироваться дополненная реальность. Поверх таргетов накладывается видео анимация. Граффити плоские и обычно весьма контрастные, их легко отличить друг от друга. Это способствует стабильной работе дополненной реальности.

К примеру, в приложении “AR Hunter” в качестве таргетов используются фотографии зданий, которые не очень для этого подходят, т. к. они достаточно похожи. Из-за этого приложение иногда путает таргеты и показывает неверный контент.

Можно сказать, что приложение PublicARt не имеет аналогов. Оно позволяет пользователю с головой окунуться в мир уличного искусства, узнать интересную информацию о работах и их авторах, на карте найти ближайшие арт объекты, “оживить” их в дополненной реальности и быть в курсе последних новостей и интересных событий, связанных со стрит артом.

ЛИТЕРАТУРА

1. Документация Unity https://docs.unity3d.com/Manual/index.html
2. Документация UI Toolkit https://docs.unity3d.com/Manual/UIElements.html
3. Документация Vuforia SDK для Unity https://library.vuforia.com/getting-started/getting-started-vuforia-engine-unity
4. Сергей Тепляков “Паттерны проектирования на платформе .NET”. Издательский дом "Питер" 2015
5. Мартин Роберт “Чистая архитектура. Искусство разработки программного обеспечения”. Издательский дом "Питер" 2018

УДК 004.522: 004.921

В.Ю. Климкин (2 курс магистратуры),
Т.С. Горавнева, к.т.н., доцент

РАЗРАБОТКА МОБИЛЬНОГО ПРИЛОЖЕНИЯ ДЛЯ СОЗДАНИЯ МАРШРУТА МЕЖДУ АУДИТОРИЯМИ УНИВЕРСИТЕТА

Целью работы является разработка мобильного приложения, определяющего оптимальный маршрут от одной точки на карте Морского технического университета до заданной аудитории с учетом многоэтажности здания СПбГМТУ на Ленинском проспекте, д. 101.

Требования к мобильному приложению:

– мобильное приложение должно работать под ОС Android не ниже 4.4,
– мобильное приложение должно работать, не используя доступ в Интернет и GPS,
– мобильное приложение должно предоставлять пользователю выбор начальной и конечной аудитории без явного указания этажа,
– мобильное приложение должно сообщать пользователю об успешной прокладке маршрута.

Приложение было создано в среде разработки Android Studio [1]. Данная среда позволяет быстро создавать XML-разметку для интерфейса и данных, ускоряет создание Java-кода путём автогенерации имён методов и переменных, а также упрощает сборку и компиляцию приложения.

Средства создания разработки:

– для демонстрации интерфейса – язык разметки XML,
– для работы алгоритма – язык программирования Java,
– для получения контуров – Python, imagemagick [2], Inkscape,
– для рисования графа маршрутов – Inkscape.

Приложение содержит 3 вкладки: «Параметры» (Рис. 1), «Этаж начальной аудитории» (Рис. 2), «Этаж конечной аудитории».

Рисунок 1 –Экран «Параметры»

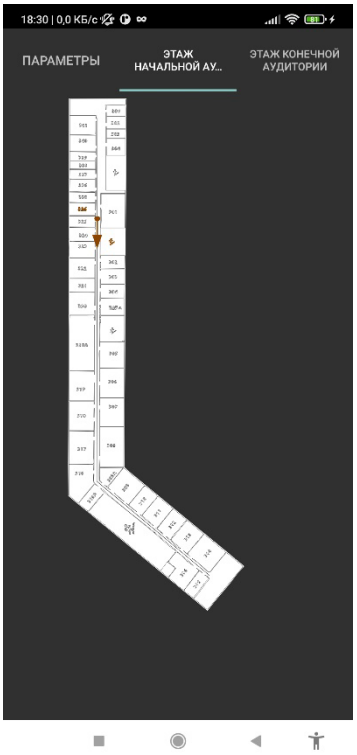


Рисунок 2 – Экран «Этаж начальной аудитории»

После запуска приложения отображается диалог, на котором можно выбрать начальную и конечную аудитории из раскрывающегося списка.

После нажатия кнопки «Составить маршрут» осуществляется выбор наиболее оптимальной траектории движения с использованием доступных лестниц. Отображение плана производится только при переключении на следующую вкладку.

На данном экране отображаются схема этажа, граф маршрутов, маршрут с указанием направления движения, а также контуры начальной и конечной аудиторий.

Экран «Этаж конечной аудитории» аналогичен предыдущему.

Вся информация, необходимая для создания маршрута, хранится в папке в виде файлов.

Любой файл начинает с номера этажа, для которого он применяется. Следующие постфиксы определяют тип хранимых данных:

– csv – последовательности аудиторий, определяющие направление движения,
– -0.csv – идентификаторы контуров начальной и конечной аудитории,
– -1.svg – графы маршрутов,
– -4.svg – контуры аудиторий.

Отображение маршрута осуществляется с помощью загрузки объектов drawable, а также преобразования строк в SVG-объекты [3], полученные в результате работы программы.

Контуры аудиторий были сгенерированы с помощью imagemagick, а также скриптов, написанных на Python, на основе поэтажных планов с аудиториями университета.

ЛИТЕРАТУРА

1. Программирование под Android на Java // METANIT.COM, 2023. [URL]: https://metanit.com/java/android/ (дата обращения: 20.02.2023).
2. Man imagemagick // Opennet, 2005 [URL]: https://www.opennet.ru/man.shtml?topic=ImageMagick&category=1&russian=3 (дата обращения: 18.02.2023)
3. Справочник SVG элементов — SVG // MDN, 2023. URL: https://developer.mozilla.org/ru/docs/Web/SVG/Element (дата обращения: 14.02.2023).

УДК 004.42

М. А. Коновалов (4 курс бакалавриата),
А. В. Самочадин к.т.н., доцент

РАЗРАБОТКА ЧАТ БОТА ПО СБОРУ АНАМНЕЗА

Анамнез - совокупность сведений, получаемых при медицинском обследовании путём расспроса самого обследуемого и/или знающих его лиц. Сбор анамнеза — неотъемлемый этап осмотра пациента. Врач опрашивает больного о патологиях, оперативных вмешательствах, полученных травмах, течении заболевания. Часто данный процесс не автоматизирован и занимает большую часть всей консультации [1].

Чтобы уменьшить время онлайн консультации и тем самым сэкономить время врача, пациента и уменьшить нагрузку на сервера, было принято решение разработать чат бот по сбору анамнеза и интегрировать его в чат консультации. Благодаря чат-боту, сбор анамнеза можно вынести за рамки приема, на этап, во время которого пациент ожидает подключения дежурного врача.

Кроме того, необходимо учитывать, что данные, которые были собраны во время анамнеза, являются медицинской тайной. Поэтому необходимо применять шифрование при передаче их с клиента на сервер и с сервера на клиент.

Таким образом *целью* работы является проектирование и реализация чат-бота по сбору анамнеза для мобильных устройств на операционной системе iOS.

Для достижения данной цели необходимо решение следующих *задач*:

1. Обзор существующих решений и подходов к реализации чат-ботов на мобильных платформах и реализации передачи данных в зашифрованном виде. Выявить недостатки и сформулировать преимущества.
2. Проведение сравнительного анализа найденных решений.
3. Проектирование архитектуры чат-бота.
4. Реализовать передачу данных между клиентом и сервером в зашифрованном виде.
5. Реализовать чат-бот по сбору анамнеза для устройств на операционной системе iOS

Логика чат-бота была реализована на языке программирования Swift [2] – родном языке для платформы iOS. Перед консультацией с сервера приходят все вопросы и варианты ответов в зашифрованном виде, а также в каком порядке они должны задаваться. Клиент отображает их, собирает ответы от пользователя, зашифровывает их и отправляет на сервер (Рис 1.).

Для общения с сервером был выбран формат JWE[3] токена, внутри которого данные находятся в зашифрованном виде. Для шифрования были выбраны следующие алгоритмы: для шифрования ключа используется AES-256-GCMKW, где 256 — это размер ключа с помощью, которого происходит шифрование, в качестве алгоритма для контента используется AES-256-GCM. Для шифрования выбран фреймворк CryptoKit [4] предоставляемый компанией Apple. Он удобен в использовании, а также достаточно быстрый, так как эффективно использует чип Secure Enclave, разработанный компанией Apple для аппаратной поддержки шифрования на их устройствах.

В ходе обзора выяснилось, что для работы с форматом JWE в iOS существовали различные решения. Но все они использовали собственные реализации алгоритмов шифрования, а не фреймворк CryptoKit. Поэтому возникла потребность в реализации своего пакета для работы с JWE и фреймворком CryptoKit.

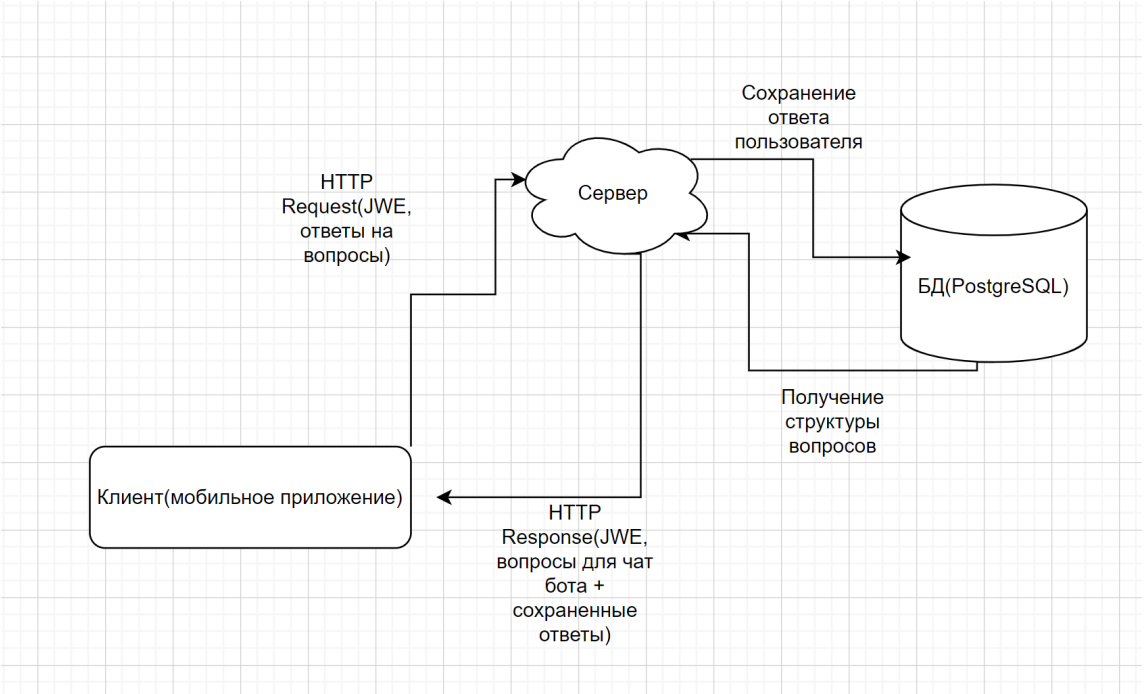


Рисунок 1 – Схема взаимодействия клиента и сервера

К архитектуре чат-бота предъявлялось несколько требований [5]:

1) Система должна быть масштабируемой. То должна быть возможность варьировать вопросы в зависимости от специализации врача, а также легко добавлять новые типы вопросов.
2) Система должна быть отделена от чата и интегрироваться в него как модуль, чтобы не усложнять систему онлайн консультаций.

Для выполнения данных требований использовались различные паттерны проектирования такие как: Фасад, Делегат, Билдер [6]. Кроме того, использовался паттерн MVP (Model-View-Presenter) отделяющее реализацию интерфейса от бизнес-логики и бизнес-логику от данных. Так же архитектура строилась по принципам SOLID.

Так как система онлайн консультации достаточно нагруженная, то работа чат-бота вынесена на отдельный очередь с помощью фреймворка GCD [7].

ЛИТЕРАТУРА

1. Как устроены медицинские чат-боты — разбираемся на примере бота DOC+. [Электронный ресурс] Режим доступа: https://habr.com/ru/company/docplus/blog/374493/
2. The Swift Programming Language Book [Электронный ресурс] Режим доступа: https://www.swift.org/documentation/
3. RFC 7516. JSON Web Encryption (JWE), M. Jones, J. Hildebrand. 2015
4. Apple CryptoKit [Электронный ресурс] Режим доступа: https://developer.apple.com/documentation/cryptokit/
5. Зянчурин Ильяс Вильсурович Архитектура Корпоративного Чат-Бота // ИНТЕРНАУКА. - 2019. - №44-1. - С. 8-11.
6. Karoly Nyisztor, Monika Nyisztor Design Patterns in Swift 5. - Razeware LLC , 2019. - 277 с.
7. Marin Todorov Modern Concurrency in Swift. - Razeware LLC , 2021. - 273 с.

УДК 004.42

И. С. Кухоткин (4 курс бакалавриата),
И. В. Шошмина, к.т.н., доцент

РАЗРАБОТКА ПРИЛОЖЕНИЯ ДЛЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ЦИФРОВОЙ ПЛАТФОРМЫ РАЗВИТИЯ РЕГИОНОВ

При разработке цифровой платформы развития регионов потребовалось обрабатывать большие массивы данных с высокой скоростью, чтобы принимать статистически обоснованные управленческие решения [1,2]. В мои задачи при выполнении проекта входила предоставление данных в формат, удобном для человека, а также дальнейшее интерактивное представление обработанной информации. Текущий вариант цифровой платформы можно изучить по ссылке: https://rscf.spbpu.com.

Цель - спроектировать архитектуру базы данных и разработать приложения для программного обеспечения цифровой платформы развития регионов.

Задачи:

1. Проектирование базы данных.
2. Заполнение базы данных статистическими данными.
3. Реализация API:
   - 3.1. Реализация запросов для получения необходимых статистических данных. Данные представлены в виде excel таблиц, загруженных с сайтов “Росстат” и “ЭМИСС”.
   - 3.2. Реализация модуля выгрузки статистических данных
   - 3.3. Реализация личного кабинета.
4. Визуализация статистических данных.

Архитектура портала:

1. Front-end: NextJS[3] основан на React – JavaScript библиотеке, используемой для создания пользовательских интерфейсов, что позволяет реализовать индексацию страниц, а также отобразить на странице необходимые компоненты интерфейса, синхронизируя его с данными полученными через API[5].
2. API: фреймворк NestJS[4] позволяет реализовать API (application programming interface), который описывает и предоставляет протоколы взаимодействия серверной части и базы данных. NestJS такжЬIe предлагает архитектурные решения, которые хорошо масштабируются и поддерживаются.
3. BD: система управления базами данных PostgreSQL использует реляционную модель хранения данных, которая хорошо подходит для информации из нашей предметной области. Данные представляют из себя наборы различных показателей для каждого региона Российской Федерации, представленные в виде таблицы excel.

ЛИТЕРАТУРА

1. Schallmo D.R.A., Williams C.A. History of Digital Transformation// Digital Transformation Now! Springer Briefs in Business. Springer, Cham. 2018.
2. Бойков А. В., Успенский М. Б., Болсуновская М. В. Комплексное решение по цифровизации на примере производственной ячейки //Системный анализ в проектировании и управлении. – 2021. – Т. 25. – №. 3. – С. 314-325.
3. NextJS. [Электронный ресурс] Режим доступа: https://nextjs.org
4. NestJS. [Электронный ресурс] Режим доступа: https://nestjs.com
5. API. [Электронный ресурс] Режим доступа: https://ru.wikipedia.org/wiki/API

УДК 004.514

Ю. В. Ловкачева (4 курс бакалавриата),
А. П. Маслаков, ст. преподаватель

РАЗРАБОТКА ПРИЛОЖЕНИЯ ДЛЯ ПЛАНИРОВАНИЯ ЗАНЯТИЙ ПО ИГРЕ НА ГИТАРЕ

Целью работы является разработка графического приложения для планирования занятий по игре на гитаре, позволяющего создавать и редактировать планы занятий, содержащие несколько этапов, к каждому из которых есть возможность приложить материалы и заметки. В настоящее время для обучения игре на гитаре доступно очень большое количество ресурсов – книг, видеоуроков с разборами песен и техник игры, табулатур и нот в различных форматах и их редакторов. Из-за этого возникает потребность в решении, позволяющем удобно организовать эту разрозненную информацию и использовать ее во время занятий. Таким образом, актуальна разработка нового решения, позволяющего планировать занятия на основе имеющихся материалов, с учетом недостатков существующих решений.

Реализация приложения ведется с использованием языка Java [2], в клиентской части приложения используется платформа JavaFX [4]. Планируется реализация режимов прохождения и редактирования для каждого планируемого занятия, возможности добавить новое занятие и удалить существующее. Каждый из этапов занятия имеет имя, содержит настройки метронома и набор материалов (диаграмм, отформатированного текста, изображений). В режиме прохождения требуется реализовать возможность перехода между этапами, запуска метронома, просмотра материалов каждого этапа. В режиме редактирования необходимы возможности удаления, добавления и редактирования отдельных материалов каждого из этапов, редактирования имени самого занятия и имен его этапов, настройка на каждом этапе наличия метронома и, в случае его наличия, его параметров по умолчанию – темпа и тактового размера. На Рисунке 1 представлен редактор одного из этапов занятия, реализованный на данный момент. Содержимое (материалы) представленного этапа занятия содержит отформатированный текст и изображение.

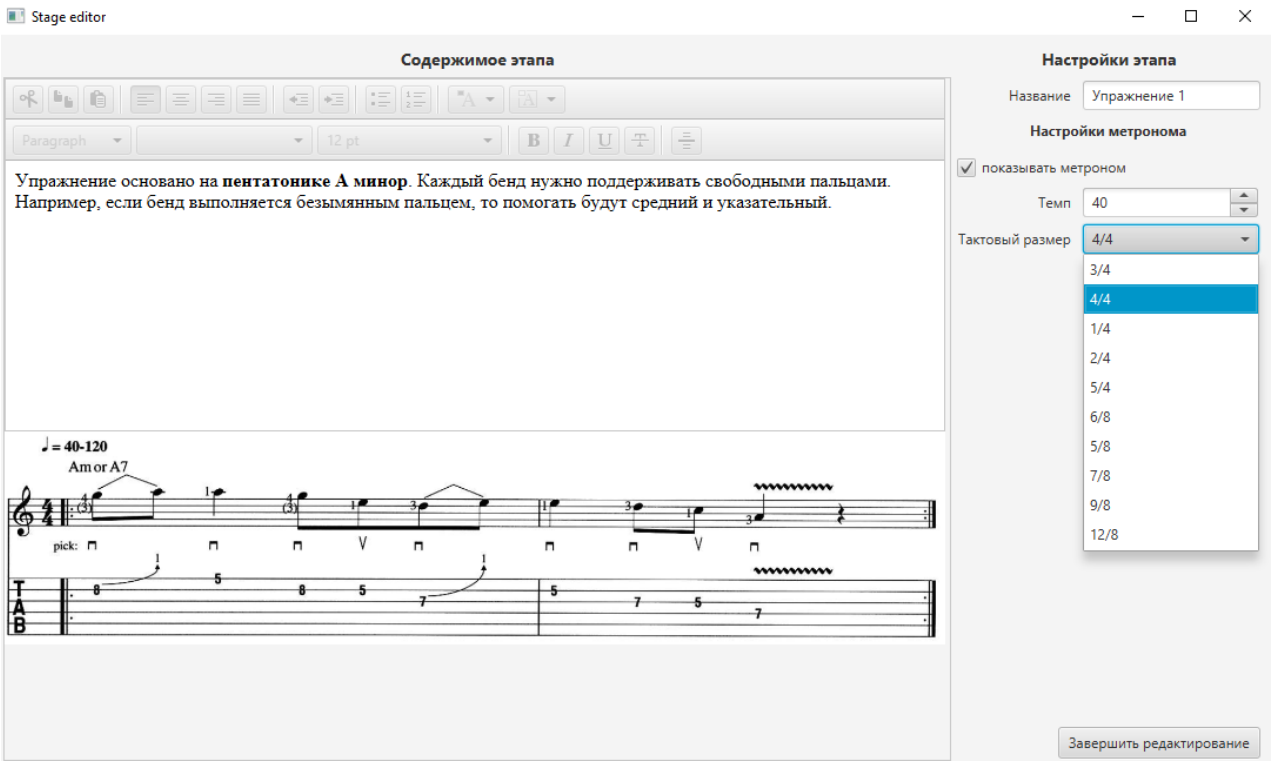


Рисунок 1 – Окно редактора этапа занятия

Разработка серверной части приложения ведется с использованием Spring Framework [5], а именно Spring Boot, Spring Web MVC, Spring Data JPA и Spring Security (для аутентификации

и авторизации пользователей). Для серверной части необходима функциональность сохранения выбранных пользователем занятий в базе данных для доступа к ним с помощью клиента на другом устройстве. В качестве базы данных в серверной части будет использоваться PostgreSQL[1].

Клиентская и серверная части приложения будут взаимодействовать по REST API с помощью запросов, содержащих данные о занятиях в формате JSON. Для сериализации данных в JSON планируется использование библиотеки Jackson [3].

Таким образом, разрабатываемое приложение позволит планировать занятия, составляя их из этапов с приложенными к ним материалами, и проходить их. За счет функциональности серверной части работа с занятиями будет доступна с нескольких устройств.

ЛИТЕРАТУРА

1. PostreSQL [Электронный ресурс]. URL: https://www.postgresql.org/ (Дата обращения: 15.03.2023).
2. Herbert Schildt, Java: The Complete Reference, Twelfth Edition: McGraw-Hill Education, 2021. 1280 с.
3. Jackson Project Home [Электронный ресурс]. URL: https://github.com/FasterXML/jackson (Дата обращения: 15.03.2023).
4. JavaFX [Электронный ресурс]. URL: https://openjfx.io/ (Дата обращения: 15.03.2023).
5. Spring Projects [Электронный ресурс]. URL: https://spring.io/projects (Дата обращения: 15.03.2023).

УДК 004.93'12

Ли Ицзя (4 курс бакалавриата),
О. Г. Малеев, к.т.н., доцент

ПРИМЕНЕНИЕ НЕЙРОННЫХ СЕТЕЙ ДЛЯ ОЦЕНКИ ТРАФИКА МАГАЗИНА

Статистика посещаемости магазинов и анализ времени пребывания клиентов находятся в центре внимания продавцов. Собирая и анализируя эти данные, продавцы могут лучше понять покупательское поведение и потребности клиентов, чтобы оптимизировать план работы магазина и стратегии продвижения товаров, а также увеличить продажи.

В настоящее время широко используемые методы статистики трафика магазинов включают использование датчиков, видеонаблюдение, данные Wi-Fi[1] и т.д. Хотя эти методы могут собирать большое количество релевантных данных, их статистическая точность и эффективность имеют множество ограничений[2].

Нейронные сети способны обрабатывать большие объемы информации и сложные взаимосвязи, а также могут адаптивно повышать точность прогнозов и обобщение за счет обучения, поэтому они имеют широкие перспективы применения в статистике трафика магазинов.[3] В этой работе мы рассмотрим предлагаемое решение для контроля посещаемости магазинов с использованием архитектуры YOLO (You Only Look Once) - одной из самых популярных нейронных сетей для обнаружения объектов.

В научно-исследовательской работе " People Detection System Using YOLOv3 Algorithm" [4] авторы предлагают метод для реал-тайм отслеживания и счёта людей в магазине с использованием YOLOv3 и глубокого обучения. Данный метод показал высокую точность и быстрое время работы.

Много существенных исследований показывает, что использование нейронных сетей и YOLO для отчётности посещаемости магазинов является эффективным и точным методом. Использование OpenCV в качестве интерфейса для ввода изображения с камеры может улучшить производительность и надежность системы.[5]

Предлагаемая архитектура решения задачи:

1. Использовать OpenCV для захвата видеопотока и предварительной обработки каждого кадра потока, включая такие операции, как обрезка, масштабирование и нормализация.

2. Использовать алгоритм обнаружения человека на видео (HaarCascade или детектор

объектов на основе глубокого обучения), чтобы обнаружить человека на предварительно обработанном изображении и извлечь ROI (область интереса) человека.

3. Чтобы выполнить извлечение признаков для каждой извлеченной области интереса, можно использовать предварительно обученную свёрточную нейронную сеть (например, ResNet) для извлечения вектора признаков человека.

4. Извлеченные векторы признаков группируются, и похожие векторы признаков классифицируются в одну и ту же категорию, то есть идентифицируются одни и те же люди.

5. По информации о положении и времени каждого человека в видеоряде рассчитываются такие показатели, как время входа и выхода и время пребывания в магазине.

6. Сохранить рассчитанные показатели в базу данных или файл для последующего анализа и статистики.

Применение предлагаемой архитектуры позволяет эффективно собирать статистику потоков покупателей в магазинах, конфиденциально и не используя контактные датчики. В дальнейшем планируется использовать собранную статистику для улучшения эффективности маркетинговых акций и оптимизации потоков покупателей.

ЛИТЕРАТУРА

1. Y. Yang, J. Cao, X. Liu and X. Liu, "Wi-Count: Passing People Counting with COTS WiFi Devices," 2018 27th International Conference on Computer Communication and Networks (ICCCN), Hangzhou, China, 2018, pp. 1-9, doi: 10.1109/ICCCN.2018.8487420.
2. Гао Ченцян, Юй Диху, Ли Цян и др. Статистика потоков людей на основе сопоставления характеристик видеоизображений [J] Телевизионные технологии, 2011(15):20-23.
3. Антонов С А. 2. Разработка интеллектуальной системы контроля передвижений людей[J]. Научно-технический семинар кафедры МОЭВМ, 2021: 8.
4. N. I. Hassan, N. M. Tahir, F. H. K. Zaman and H. Hashim, "People Detection System Using YOLOv3 Algorithm," 2020 10th IEEE International Conference on Control System, Computing and Engineering (ICCSCE), Penang, Malaysia, 2020, pp. 131-136, doi: 10.1109/ICCSCE50387.2020.9204925.
5. A. H. Ahamad, N. Zaini and M. F. A. Latip, "Person Detection for Social Distancing and Safety Violation Alert based on Segmented ROI," 2020 10th IEEE International Conference on Control System, Computing and Engineering (ICCSCE), Penang, Malaysia, 2020, pp. 113-118, doi: 10.1109/ICCSCE50387.2020.9204934.

УДК 004.51

Д. К. Мальцева, М. В. Швыдкий (2 курс бакалавриата),
Д. С. Эйзенах (2 курс аспирантуры),
А.П. Маслаков, ст. преподаватель

РАЗРАБОТКА ГРАФИЧЕСКОГО ИНТЕРФЕЙСА ДЛЯ СЕРВИСА ПОМОЩИ ПРИЮТАМ И ДОМАШНИМ ЖИВОТНЫМ “ISHELT”

Идея создания приложения основана на анализе деятельности приютов и проблем, с которыми они сталкиваются, среди которых можно выделить ограниченные возможности для коммуникации, ограниченное финансирование, трудности во взаимодействии с волонтерами. Целью разработки приложения является создание кросс-платформенного графического интерфейса для приложения iShelt. Исходя из целей работы были сформированы следующие задачи:

- Составить карту переходов между экранами приложения.
- Сделать прототип графического интерфейса.
- Разработать макет.
- Реализовать функциональность проекта.

В данной статье мы рассмотрим стадии процесса разработки графического интерфейса, архитектуру и карту приложения, а также особенности реализации дизайна концепции.

Для начала было принято решение разработать карту приложения и утвердить все экраны и переходы между ними. Карта приложения состоит из четырех основных разделов: главная страница, новости, карта местности и аккаунт пользователя. На главной помимо просмотра объявлений, можно настроить фильтры и просмотреть «избранные объявления». В ленте новостей публикуются анонсы от приютов. На карте отмечены геопозиции приютов, животных и других объектов, опубликованных в объявлении. В разделе аккаунт пользователя много возможностей: можно создавать объявления, редактировать информацию о себе, устанавливать статус волонтера и перемещаться между аккаунтами: личным и приюта (если таковой создан).

Далее был разработан интерактивный макет интерфейса в графическом редакторе Figma [1]. При создании макета приложения учитывались современные тенденции [2] и требования созданию удобных и “дружелюбных” интерфейсов. После его разработки было проведено юзабилити-тестирование, что помогло выявить недочеты и доработать их.

Дизайн разрабатывался для IOS. В Human Interface Guidelines (HIG) [3] рекомендуется избегать смешения шрифтов и использовать жирность и курсив только для выделения. Поэтому в дизайне используется всего один шрифт, но в 5 стилях, что помогает пользователю различать функциональные значения текстов. Кнопки по HIG должны быть не меньше 44 точек в ширину и высоту, текст должен быть коротким и ясным, и стиль кнопок должен быть последовательным по всему приложению. Исходя из этого, кнопки понятны и разборчивы, выполняют свою функцию. Также были учтены рекомендации по использованию цвета [4]. Была выбрана зеленая палитра и дополнительный желтый цвет, фон белый (Рис. 1). Размер иконок панели навигации 20x20 пикселей с соотношением 1:1.

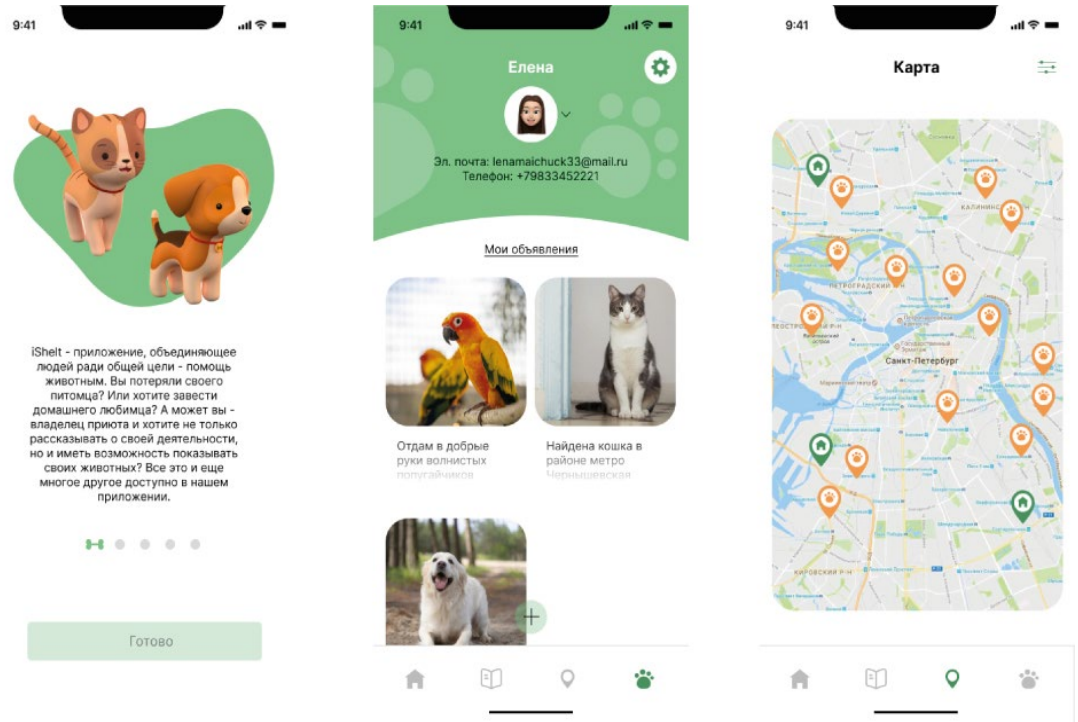


Рисунок 1 – Слайдер-анонс, Аккаунт, Карта

Разработка приложения началась с выбора архитектуры приложения, выбор пал на BLoC (Business Logical Component), после передачи данных с сервера, BLoC передает состояние страницы в UI и уже после происходит вывод ее на экран. Таким образом приложение формирует красивую страницу совмещая продуманный дизайн и данные с сервера. Благодаря системе виджетов, на которой основан Flutter, можно легко вносить правки в UI. Создавая собственные виджеты и компонуя их с уже существующими, достаточно легко реализовать задуманный дизайн.

По итогу проведенной работы, был реализован дизайн мобильного приложения, созданы прототип и макет в количестве 40 экранов.

ЛИТЕРАТУРА

1. Staiano, Fabio. Designing and Prototyping Interfaces with Figma: Learn essential UX/UI design principles by creating interactive prototypes for mobile, tablet, and desktop. Packt Publishing Ltd, 2022.
2. Sherin, Aaris. Introduction to Graphic Design: A Guide to Thinking, Process, and Style. Bloomsbury Publishing, 2023.
3. iPhone Human Interface Guidelines. Available at: https://developer.apple.com/design/human-interface-guidelines/guidelines/overview/
4. Ebner, Martin, Christian Stickel, and Josef Kolbitsch. "iPhone/iPad human interface design." HCI in Work and Learning, Life and Leisure: 6th Symposium of the Workgroup Human-Computer Interaction and Usability Engineering, USAB 2010, Klagenfurt, Austria, November 4-5, 2010. Proceedings 6. Springer Berlin Heidelberg, 2010.

УДК 004.421

А.С. Матус (4 курс бакалавриата),
П.Д. Дробинцев, к.т.н., доцент

POLYVENT. РАЗРАБОТКА МОБИЛЬНОГО ПРИЛОЖЕНИЯ ПОД ПЛАТФОРМУ IOS ДЛЯ ОРГАНИЗАЦИИ СТУДЕНЧЕСКИХ МЕРОПРИЯТИЙ

В наше время студенческие мероприятия являются неотъемлемой частью активной жизни студенческого сообщества. Однако, организация таких мероприятий может стать непростой задачей для студентов-организаторов. Они сталкиваются с проблемами в поиске участников, организацией прохода и многих других аспектов, которые могут затруднить успешную реализацию идеи мероприятия.

В связи с этим, создание мобильного приложения, которое поможет студентам легко и быстро организовывать свои мероприятия, является актуальной и перспективной задачей. В данной работе представляется разработка мобильного приложения под платформу iOS "PolyVent" для студентов и студенческих объединений Санкт-Петербургского политехнического университета Петра Великого, позволяющее им легко и быстро создавать и публиковать информацию о своих мероприятиях, а также находить мероприятия, которые им интересны. Приложение "PolyVent" будет удобным и полезным инструментом для студентов, которые желают организовать свое мероприятие, а также для тех, кто хочет находить интересные мероприятия в удобном формате.

Интерес к данной теме возник после посещения мероприятий, посвященных началу учебного года, где проводилось огромное количество собраний студентов. Организаторы мероприятий просили подписаться на свои социальные сети, для отслеживания их будущих активностей, что оказалось не очень практичным, потому что студенту приходится лично следить за всеми группами и велика вероятность пропустить одно из мероприятий, которое он хотел посетить. Также есть вероятность того, что группа, за которой следит студент, может быть заброшена, удалена или пересоздана, соответственно связь с данной группой пропадет полностью.

Программа позволит сократить время поиска и регистрации на мероприятия, а также создавать их. При этом каждый студент имеет личный QR-код, которые легко сканируются в приложении на входе в мероприятие и проверяются, записан ли данный студент или нет. Это упрощает процесс прохода на мероприятие и позволяет организаторам быстрее и эффективнее контролировать участников мероприятия.

Мобильные приложения в настоящее время являются неотъемлемой частью жизни молодого поколения, и студенческие мероприятия не являются исключением. Однако на рынке отсутствует мобильное приложение, специализированное на организации студенческих мероприятий, что делает организацию более сложной и трудоемкой.

Таким образом, разработка мобильного приложения "PolyVent" является актуальной задачей, которая позволит повысить эффективность организации студенческих мероприятий и повысить уровень активности студенческой жизни.

Приложение решает следующие проблемы:

– Отсутствие единой платформы для организации студенческих мероприятий: существующие приложения предназначены для широкой аудитории, и не учитывают специфические потребности студентов.

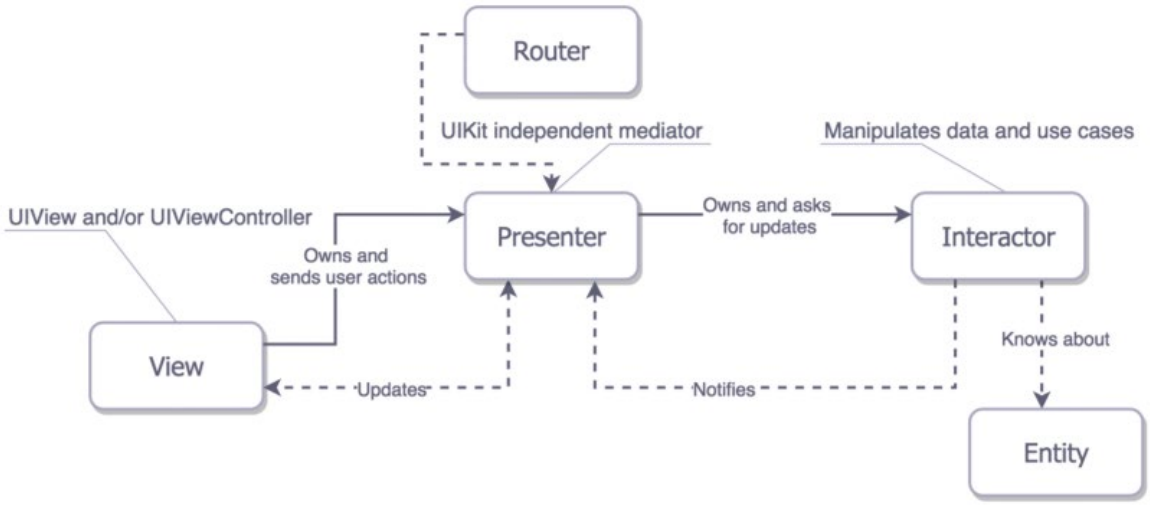
– Низкий уровень информированности студентов о предстоящих мероприятиях: студенты могут пропустить важное мероприятие из-за отсутствия информации о нем.

– Неэффективная коммуникация и сбор информации: студенты могут столкнуться с трудностями в коммуникации с организаторами мероприятий и сборе информации, что затрудняет организацию мероприятий.

– Низкая эффективность продвижения мероприятий: студенты могут столкнуться с трудностями в продвижении своих мероприятий, что может привести к малому количеству участников.

– Необходимость организации удобного фейс-контроля на мероприятиях. В настоящее время, проверка билетов и фейс-контроль могут производиться вручную, что затрудняет процесс и приводит к задержкам в проходе на мероприятие.

Приложение будет написано на языке программирования Swift и использует архитектуру VIPER(View-Interactor-Presenter-Entity-Router).



Для реализации функционала приложения будут использоваться такие технологии, как Firebase Authentication для аутентификации пользователей, Firebase Realtime Database для хранения данных о мероприятиях и пользователях, а также QR-коды для контроля доступа на мероприятия. Визуальное оформление приложения будет реализовано с помощью фреймворка UIKit и Auto Layout.

ЛИТЕРАТУРА

1. Swift. Язык программирования с открытым кодом. Мощь, простота и потрясающие приложения [Электронный ресурс] https://www.apple.com/ru/swift/ (Дата обращения 10.01.2023)
2. XCode 14 [Электронный ресурс] https://developer.apple.com/xcode/(Дата обращения 10.01.2023)
3. GitHub https://github.com/
4. Paul Hudson. Pro Swift. https://www.hackingwithswift.com/store/pro-swift
5. FaceID Security. [Электронный ресурс] URL: https://www.apple.com/business-docs/FaceID_Security_Guide.pdf (Дата обращения 26.02.2023)
6. Firebase: Get started: write, test, and deploy your first functions. [Электронный ресурс] URL: https://firebase.google.com/docs/functions/get-started (Дата обращения 27.02.2023)

УДК 004.021

Г. А. Мещеряков (4 курс бакалавриата),
П.Д. Дробинцев, к.т.н., доцент

РАЗРАБОТКА BACKEND СОСТАВЛЯЮЩЕЙ ПРОГРАММНОГО ПРОДУКТА POLYHABR

В процессе обучения каждый сталкивался с проблемой того, что не знал, как сделать или с чего начать выполнение курсовой, лабораторной или научной работы. Данную проблему возможно решить путем использования поисковой системы Интернета, однако часто искомый ответ там отсутствует. Или же можно попросить помощи у одногруппников, но, возможно, некоторые испытывают дискомфорт в просьбе оказать помощь, а также не исключен отказ со стороны соучащихся. Данная тема заинтересовала меня в связи с моим личным опытом, связанным с этой проблематикой.

Целью этой работы является создание веб-приложения «Polyhabr», которое будет помогать студентам, затрудняющимся выполнить свою лабораторную, практическую, курсовую или научную работу. В рамках данного приложения они смогут посмотреть примеры прошлых лет, которые помогут им ответить на их вопросы и справиться с трудностями. Кроме того, пользователи могут публиковать свои собственные работы с целью научного обмена и обсуждения результатов с другими пользователями. Также есть возможность комментировать публикации, чтобы пользователи смогли коммуницировать между собой и обсуждать пост. Помимо этого, приложение обеспечивает возможность проведения аутентификации пользователей и последующими отметками о том, что статья понравилась. К тому же есть возможность добавления отмеченных статей в закладки. На сайте имеется возможность осуществления поиска по названию статьи, который будет выдавать их по рейтингу. Также реализована функциональность сортировки по дате публикации и рейтингу.

Данный проект поможет значительно облегчить жизнь многим студентам, сократит время, требуемое на выполнение работы, так как не придется часами искать информацию в интернете, ответит на вопросы, которые возникают у них в процессе выполнения работы, и, помимо этого, повысит качество обучения.

Особенность программного продукта заключается в том, что все опубликованные на сайте записи объединяются одним общим критерием - они связаны с Санкт-Петербургским Политехническим университетом имени Петра Великого, то есть они предназначены только для студентов этого университета. Кроме того, в продукт будет включен алгоритм, который будет подбирать материалы на основе интересов пользователя.

Разрабатываемое приложение основывается на серверной части приложения и базы данных.

Для реализации серверной части выбран язык программирования Kotlin+JPA+Spring Boot. Kotlin – статически типизированный язык программирования, который работает на платформе Java Virtual Machine. Он имеет ряд преимуществ перед Java: более короткий и понятный синтаксис, более строгая типизация и безопасность при работе с null-значениями, что снижает количество ошибок и облегчает поддержку кода и одним из главных преимуществ является поддержка функционального программирования. JPA (Java Persistence API) - обеспечивает механизм для соединения и манипулирования данными в базе данных, а также устраняет необходимость написания SQL-кода для выполнения запросов к базе данных. Spring Boot - позволяет быстро настроить и развернуть приложение с минимальными усилиями и конфигурацией.

Для хранения данных используется СУБД PostgreSQL. PostgreSQL - объектно-реляционная система управления базами данных, которая поддерживает SQL-запросы и расширяема пользовательскими функциями, считается одной из наиболее надежных и безопасных СУБД с открытым исходным кодом.

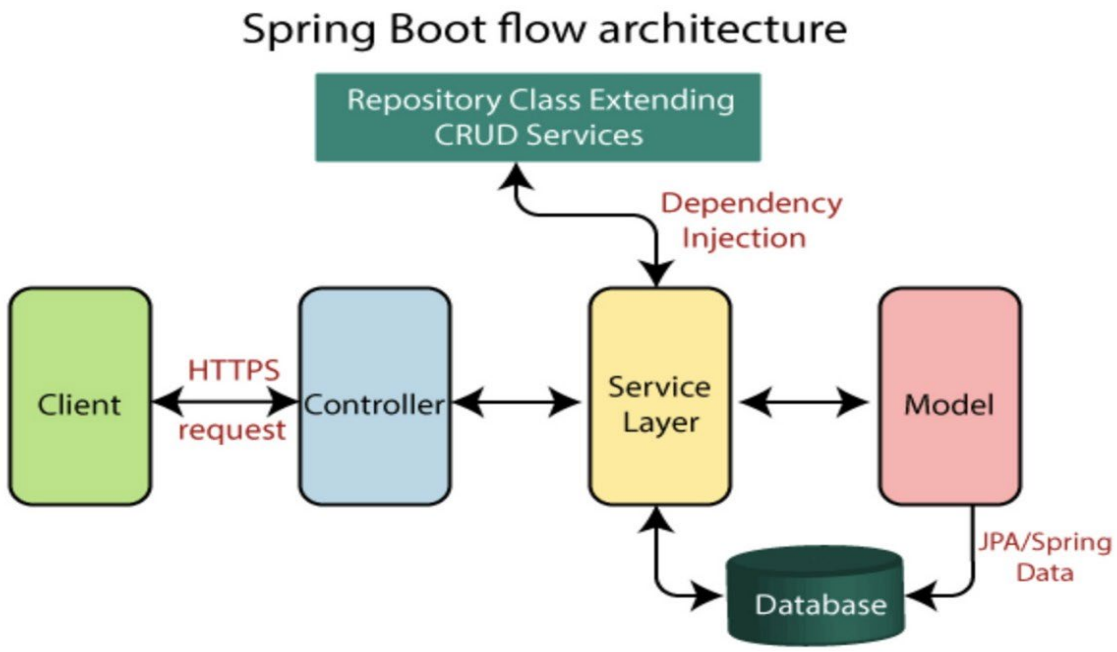


Рисунок 1 – Архитектура приложения

ЛИТЕРАТУРА

1. Технология программирования. Основные понятия и подходы [Электронный ресурс] Режим доступа: https://studfile.net/preview/2953279/
2. Java Persistence API (JPA) [Электронный ресурс] Режим доступа: https://www.ibm.com/docs/en/was-liberty/base?topic=overview-java-persistence-api-jpa
3. Kotlin. Basic syntax [Электронный ресурс] Режим доступа: https://kotlinlang.org/docs/basic-syntax.html
4. Spring Boot Reference Documentation [Электронный ресурс] Режим доступа: https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/
5. PostgreSQL Documentation [Электронный ресурс] Режим доступа: https://www.postgresql.org/docs/
6. JPA - Краткое руководство [Электронный ресурс] Режим доступа: https://translated.turbopages.org/proxy_u/en-ru.ru.5bd8d95e-6419aea7-21845570-74722d776562/https/www.tutorialspoint.com/jpa/jpa_quick_guide.htm

УДК 004.453

Г. В. Мироненков, Ф. Ю. Чемашкин (2 курс аспирантуры),
Н. В. Воинов, к.т.н., доцент

ПРИЛОЖЕНИЕ ДЛЯ ТЕХНИЧЕСКОГО ОБСЛУЖИВАНИЯ ПОЕЗДОВ С ПРИМЕНЕНИЕМ ТЕХНОЛОГИИ ДОПОЛНЕННОЙ РЕАЛЬНОСТИ

Дополненная реальность — это технология, которая в последнее время набирает популярность, особенно в области технического обслуживания, где она используется для повышения эффективности работ. Технология дополненной реальности накладывает цифровую информацию на реальные среды, тем самым предоставляя пользователям расширенное представление о реальности. Концепция дополненной реальности использовалась в различных областях, в том числе в играх и рекламе, но в сфере технического обслуживания эта технология также оказалась полезной.

Основная цель этого исследования — изучить возможность использования технологии дополненной реальности для обслуживания поездов. В частности, был проведен обзор существующих применений технологии дополненной реальности и изучены их

потенциальное применение в обслуживании поездов. Также было создано приложение для Android, которое можно использовать для предоставления решений на основе дополненной реальности для обслуживания поездов.

Для достижения вышеупомянутых целей был проведен обзор научных работ, связанных с применением технологии дополненной реальности в сфере технического обслуживания. Было обнаружено, что технология дополненной реальности уже используется в различных отраслях, таких как производство [1,2], аэрокосмическая и автомобильная промышленность [3-5], но количество исследований, специально посвященных обслуживанию поездов, было ограничено. Чтобы восполнить этот пробел, было спроектировано и разработано приложение для Android под названием «Train AR Assistant», которое использует технологию дополненной реальности для улучшения обслуживания поездов. «Train AR Assistant» предоставляет информацию о поезде и его компонентах в режиме реального времени в формате дополненной реальности. Приложение отображает информацию, когда камера направлена на определенный компонент поезда. Например, если камера направлена на неисправность в двигателе поезда, приложение отобразит соответствующую информацию о неисправности, а также инструкции по ремонту. Кроме того, приложение обеспечивает видеосвязь между обслуживающим персоналом и ремонтным центром для облегчения совместной работы и принятия решений по техническому обслуживанию. Эта функция позволяет экспертам тесно сотрудничать с удаленными командами для эффективного выявления и устранения неисправностей.

Исследование показало, что решения на основе дополненной реальности можно использовать для улучшения процедуры обслуживания поездов, тем самым повышая точность и эффективность процесса обслуживания. Использование технологии дополненной реальности обеспечивает детальное представление о составных частях поезда и их взаимосвязях, позволяя обслуживающему персоналу быстро выявлять неисправности и соответствующим образом устранять их. Кроме того, функция видеосвязи приложения расширяет возможности обслуживающего персонала общаться со специалистами по ремонту, тем самым расширяя их знания и навыки в области обслуживания.

В заключение, наше исследование показало, что применение технологии дополненной реальности для технического обслуживания поездов может повысить качество обслуживания и сократить время простоя. Наше приложение Train AR Assistant — это практический пример того, как AR можно применять для технического обслуживания поездов, предоставляя обслуживающему персоналу рекомендации по ремонту в режиме реального времени. Необходимы дальнейшие исследования, чтобы проверить эффективность решений на основе дополненной реальности при обслуживании поездов и изучить дополнительные функции, которые могут улучшить процесс обслуживания.

ЛИТЕРАТУРА

1. C. Botto et al., "Augmented Reality for the Manufacturing Industry: The Case of an Assembly Assistant," 2020 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW), Atlanta, GA, USA, 2020, pp. 299-304, doi: 10.1109/VRW50115.2020.00068.
2. Шевченко, Г. Г. Разработка мобильного приложения с поддержкой технологии дополненной реальности для получения информации о городских сооружениях / Г. Г. Шевченко, Н. В. Воинов // Современные технологии в теории и практике программирования : сборник материалов конференции, Санкт-Петербург, 19 апреля 2019 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2019. – С. 201-202. – EDN UZXOUU.
3. J. L. Gabbard, G. M. Fitch and H. Kim, "Behind the Glass: Driver Challenges and Opportunities for AR Automotive Applications," in Proceedings of the IEEE, vol. 102, no. 2, pp. 124-136, Feb. 2014, doi: 10.1109/JPROC.2013.2294642.
4. H. Regenbrecht, G. Baratoff and W. Wilke, "Augmented reality projects in the automotive and aerospace industries," in IEEE Computer Graphics and Applications, vol. 25, no. 6, pp. 48-56, Nov.-Dec. 2005, doi: 10.1109/MCG.2005.124.

5. Электрофизические и электрохимические технологии в машиностроении : учебное пособие для реализации основных профессиональных образовательных программ подготовки бакалавров по укрупненным группам специальностей и направлений подготовки 15.00.00 Машиностроение / Ю. М. Барон, Н. В. Воинов, В. С. Кобчиков [и др.] ; Санкт-Петербургский политехнический университет Петра Великого, Институт компьютерных наук и технологий. – Санкт-Петербург : ПОЛИТЕХ-ПРЕСС, 2019. – 788 с. – ISBN 978-5-7422-6463-7. – EDN ZDXDDF.

УДК 004.42

Е. А. Носкова (3 курс бакалавриата),
С. П. Воскобойников, к.ф.м.н., доцент,
Е. Н. Попов, к.ф.м.н., н.с.

РАЗРАБОТКА ПРОГРАММЫ ДЛЯ НАХОЖДЕНИЯ МОД ЭЛЕКТРОМАГНИТНОГО ПОЛЯ В ЭЛЛИПТИЧЕСКОМ ЦИЛИНДРЕ-РЕЗОНАТОРЕ

Решаемая задача является актуальной для квантовой радиофизики. Динамика излучателей в резонаторе имеет необычные свойства, например, спонтанный распад атома, расположенного между двумя зеркалами, становится медленнее по сравнению со скоростью спонтанного распада в свободном пространстве. Это называется эффектом Парсела. Его природа заключается в изменении структуры мод электромагнитного поля, с которым взаимодействуют излучатели. Аналогичной природой обладает эффект изменения диполь-дипольного взаимодействия между излучателями внутри резонатора, который существенно меняет структуру мод: если излучатели поместить в экстремальные точки мод электромагнитного поля, то обмен фотонами через резонансную моду может происходить намного более эффективно, чем между излучателями в пустом пространстве. Для поиска таких точек требуется найти пространственную форму мод резонатора. Эллиптический цилиндр выбран в связи с предположением о том, что в фокусах эллипса может наблюдаться увеличение константы связи между модой электромагнитного поля и диполем точечного излучателя.

Целью данной работы является разработка программы, позволяющей находить пространственную форму мод.

Математическая модель описывается системой уравнений Гельмгольца для вектора электрической напряжённости $\vec{E}$ в сечении эллиптического цилиндра, которое в плоскости (x, y) задается уравнением эллипса

$$\frac{x^2}{a^2}+\frac{y^2}{b^2}=1, \tag{1}$$

в котором надо найти нетривиальное решение системы уравнений

$$\frac{\partial^2 E_x}{\partial x^2}+\frac{\partial^2 E_x}{\partial y^2}+\lambda E_x=0 \tag{2}$$

$$\frac{\partial^2 E_y}{\partial x^2}+\frac{\partial^2 E_y}{\partial y^2}+\lambda E_y=0 \tag{3}$$

В качестве граничного условия рассматривается условие, что во всех точках эллипса (1) проекция вектора $\vec{E}$ на линию, касательную к эллипсу (1), равна нулю.

Кроме этого, решение должно удовлетворять дополнительному условию (4)

$$\frac{\partial E_x}{\partial x}+\frac{\partial E_y}{\partial y}=0 \tag{4}$$

Требуется найти наименьшее значение λ и соответствующие ему E_x и E_y.

Параметры a и b таковы, что площадь эллипса равна 1. Представляет интерес численное решение для четырёх случаев:

$$a/b=5, a/b=2, a/b=1/2, a/b=1/5. \tag{5}$$

В работе проведён обзор методов для аппроксимации эллиптических уравнений и на его основе выбран метод конечных разностей (МКР) на согласованной сетке [1] для эллиптической области, имеющей криволинейную границу. В результате применения МКР задача была сведена к алгебраической задаче на собственные значения, для которой применён QR-алгоритм для ленточных матриц [2] и его реализация из пакета LAPACK [3].

В работе приводятся результаты применения разработанной программы для расчёта мод электромагнитного поля в эллиптическом цилиндре-резонаторе для четырёх случаев, задаваемых отношениями (5). Для каждого случая проведены расчёты на нескольких сетках, чтобы проанализировать сходимость численного решения методом МКР.

Разработка программы осуществлена на языке программирования Fortran, входящего в состав MinGW gfortran, который интегрируется с Code::Blocks.

ЛИТЕРАТУРА

1. Самарский, А. А. Теория разностных схем / А. А. Самарский. - Москва : Наука, 1983. - 616 с.
2. Парлетт, Б. Симметричная проблема собственных значений : Числ. методы / Б. Парлетт; Перевод с англ. Х. Д. Икрамова, Ю. А. Кузнецова. - Москва : Мир, 1983. - 382 с.
3. LAPACK. [Электронный ресурс] Режим доступа : https://www.netlib.org/lapack

УДК 004.418

В. В. Орлов, М. А. Клименко (2 курс магистратуры),
Н. В. Воинов, к.т.н., доцент

FSM-СИСТЕМА ДЛЯ АВТОМАТИЗАЦИИ УПРАВЛЕНИЯ РЕМОНТОМ ДОРОЖНОГО ПОКРЫТИЯ

Современные системы взаимодействия через сеть интернет позволяют частично автоматизировать большое количество процессов для сокращения затрат человеко-часов ценных сотрудников. Именно для достижения этой цели было принято решение о разработке приложения, которое позволило бы сократить время, затрачиваемое на менеджмент работы бригад по ремонту дорожного покрытия.

Нашей целью было создание приложения, которое позволяло бы управлять бригадами рабочих дистанционно, эффективно и без постоянной связи с менеджером компании. Проблема менеджмента заключается в распределении рабочих бригад на большой площади и эффективном планировании их маршрута. Также проблемой является не полное покрытие интернетом областей вне населенных пунктов.

Разработанный продукт представляет из себя клиент-серверное приложение [1,2]. По итогу разработки приложение имеет 3 уровня пользовательского доступа: администратор, менеджер и рабочий. Выбор доступа происходит автоматически в момент авторизации на стартовом экране приложения. В функционал рабочего входит: просмотр списка доступных к выполнению объектов (отсортированных по близости к текущей локации), просмотр сведений об объекте, просмотр карты и построение маршрута к выбранному объекту, взятие объекта в работу и его завершение. В функционал менеджера входит: создание заявок, создание объектов, просмотр завершенных и активных заявок, просмотр карты объектов каждой заявки, построение маршрута к выбранному объекту, генерация отчета по полностью завершенной заявке. В функционал администратора входит: все функции менеджера с некоторыми ограничениями, регистрация аккаунтов пользователей, редактирование существующих аккаунтов, редактирование заявок и объектов, удаление объектов. Помимо основного функционала каждой из ролей, для менеджера и рабочего предусмотрена работа приложения в оффлайн режиме, поскольку эти два типа персонала часто работают в локациях с нестабильным доступом к сети интернет и необходимо обновлять данные на сервере в отложенном режиме при появлении сети.

Для разработки мобильного приложения под ОС Android [3,4] используется IDE – Android Studio, язык программирования – Kotlin [5]. Для локального хранения данных используется база данных SQLite, а доступ к ней осуществляется посредством библиотеки Room. Взаимодействие с сервером осуществляется посредством интерфейса REST API, для работы с этим интерфейсом используется библиотека Retrofit2 в связке с OkHttp3. Для организации работы программы в многопоточном режиме используется библиотека Coroutines [6]. Для отложенного выполнения кода по достижению триггера используется библиотека WorkManager. Архитектурный паттерн мобильного приложения – MVVM, наблюдаемые элементы ViewModel реализованы с использованием LiveData. Система контроля версий – Git, сервис – GitHub [7]. Для мониторинга состояния приложения на различных устройствах и баг-трекинга используется сервис компании Microsoft – App Center Analytics.

Для разработки серверной части был выбран язык JavaScript совместно с фреймворком Node.js. Для данного фреймворка доступна библиотека Express, которая служит для обеспечения взаимодействия через web [8]. Серверное приложение было построено по архитектуре REST API, что подразумевает, что для каждой хранимой на сервере сущности реализована методология CRUD (Create, Read, Update, Delete), что предоставляет полный функционал для взаимодействия с объектами, хранимыми в базе данных. Взаимодействие с сервером происходит через протокол http при помощи запросов, запросы для обращения к серверу были описаны в документации продвинутого пользователя. В качестве базы данных была выбрана PostgreSQL, она была внедрена на сервер заказчика, так как было важно хранить данные на локальной машине, что послужило причиной отказа от баз данных, которые предоставляются как сервис. На сервере была реализована авторизация по JWT токенам, что позволяет оставаться в аккаунте на клиентской стороне при повторном входе в приложение. Также реализован механизм распределения ролей, что не позволяет пользователям более низкого уровня доступа совершать запросы, предназначающиеся для высокого уровня доступа, к серверу.

По результатам внедрения системы в рабочий процесс компании средняя производительность заметно выросла. Из-за того, что рабочим теперь показываются ближайшие к ним объекты, они более оптимально расходуют рабочее время и, следовательно, за рабочий день успевают выполнить в полтора раза больше объектов, чем до внедрения приложения. Менеджеру более не нужно тратить время на координацию рабочих и у них появилось свободное время, которое они могут потратить на поиск новых объектов, поэтому среднее количество новых объектов в день выросло в полтора раза. И, наконец, более нет необходимости в множестве чатов с рабочими, поэтому и необходимое количество администраторов уменьшилось до одного, который отвечает за контроль работы приложения.

ЛИТЕРАТУРА

1. Бойков, К. М. Разработка клиент-серверной архитекутры для сервиса по управлению интерактивными подписками / К. М. Бойков, О. С. Командина, Н. В. Воинов // Современные технологии в теории и практике программирования : сборник материалов конференции, Санкт-Петербург, 23 апреля 2020 года / Санкт-Петербургский политехнический университет Петра Великого; Dell Technologies; EPAM Systems. – Санкт-Петербург: ПОЛИТЕХ-ПРЕСС, 2020. – С. 113-114. – EDN DTYFAF.
2. Smirnov, P. A. A Web Application to Promote Blood Donation in Russia / P. A. Smirnov, V. V. Malinovskaya, N. V. Voinov // Proceedings of the Institute for System Programming of the RAS. – 2022. – Vol. 34, No. 2. – P. 179-190. – DOI 10.15514/ISPRAS-2022-34(2)-14. – EDN SGENXV.
3. Филлипс Б., Стюарт К., Марсикано К. Android. Программирование для профессионалов. 3-е изд. – СПб.: Питер, 2017. – 688 с.: ил. – (Серия «Для профессионалов»)
4. Documentation for app developers. URL: https://developer.android.com/docs, дата последнего обращения: 10.03.2023
5. Kotlin documentation. URL: https://kotlinlang.org/docs, дата последнего обращения: 10.03.2023

6. Coroutines documentation. URL: https://kotlinlang.org/docs/coroutines-overview.html, дата последнего обращения: 10.03.2023
7. Big data processing system for analysis of GitHub events / N. Voinov, K. Rodriguez Garzon, I. Nikiforov, P. Drobintsev // Proceedings of 2019 22nd International Conference on Soft Computing and Measurements, SCM 2019 : 22, St. Petersburg, 23–25 мая 2019 года. – St. Petersburg, 2019. – P. 187-190. – DOI 10.1109/SCM.2019.8903782. – EDN RDBGKV.
8. Express documentation. URL: https://expressjs.com/ru, дата последнего обращения: 10.03.2023

УДК 004.421

М.И. Позднова (4 курс бакалавриата),
Т.В. Леонтьева к.т.н., доцент

ВИЗУАЛИЗАЦИЯ РАЗЛОЖЕНИЯ СВЕТА НА СПЕКТР ЧЕРЕЗ ГРАНЕННЫЙ ОБЪЕКТ ДЛЯ СРЕДЫ РАЗРАБОТКИ UNITY

Тема работы - написание функции разложения света на спектр (дисперсия света) через граненый объект для среды разработки Unity.

Цель работы - воссоздать явление дисперсии, приближенное к реальному светорассеянию с помощью компьютерной графики. Объектом для разложения может быть призма, граненый минерал (бриллиант), стеклянная поверхность.

На данный момент очень тяжело найти реализацию разложения света в спектре в Open Source для Unity.

Есть множество сайтов, где можно бесплатно или заплатив получить эффекты для игр конкретно под данную платформу, потратив немалое количество времени я так и не нашла такого рода функцию, которую можно было бы применить для большей реалистичности графики.

Данная функция нужна, так как дисперсия встречается нам в жизни крайне часто – от обычного граненного стакана до радуги. Если разработчик захочет сделать свою игру детальнее – такая функция будет полезной.

Дисперсия света - совокупность явлений, обусловленных зависимостью абсолютного показателя преломления вещества от частоты (или длины волны) света (частотная дисперсия), или, что то же самое, зависимостью фазовой скорости света в веществе от частоты (или длины волны). Экспериментально открыта Исааком Ньютоном около 1672 года, хотя теоретически достаточно хорошо объяснена значительно позднее.

Белый свет разлагается в спектр в результате прохождения через дифракционную решётку или отражения от неё (это не связано с явлением дисперсии, а объясняется природой дифракции). Дифракционный и призматический спектры несколько отличаются: призматический спектр сжат в красной части и растянут в фиолетовой и располагается в порядке убывания длины волны: от красного к фиолетовому; нормальный (дифракционный) спектр — равномерный во всех областях и располагается в порядке возрастания длин волн: от фиолетового к красному [7].

Алгоритм разложения на спектр предполагает, что через прозрачный граненный объект будет проходить белый свет, который преломляется под разными углами через грани. С помощью законов преломления можно сделать расчёты относительно траектории лучей, а благодаря сведениям о преломлении разных цветов, узнаем под каким углом будет преломленный луч каждого цвета, тогда полученные данные можно перенести в систему RGB.

Unity сделан, чтобы дать возможности для создания лучших интерактивных развлечений или мультимедиа. Есть руководство, которое создано, чтобы помочь узнать, как использовать Unity, от базовых до продвинутых приемов. Его можно читать от начала до конца, или использовать в качестве справочника [2].

Unity — межплатформенная среда разработки компьютерных игр. Unity позволяет создавать приложения, работающие под более чем 20 различными операционными системами,

включающими персональные компьютеры, игровые консоли, мобильные устройства, интернет-приложения и другие. Выпуск Unity состоялся в 2005 году и с того времени идёт постоянное развитие.

Основными преимуществами Unity являются наличие визуальной среды разработки, межплатформенной поддержки и модульной системы компонентов. К недостаткам относят появление сложностей при работе с многокомпонентными схемами и затруднения при подключении внешних библиотек.

На Unity написаны тысячи игр, приложений и симуляций, которые охватывают множество платформ и жанров. При этом Unity используется как крупными разработчиками, так и независимыми студиями [5].

Vulkan - это низкоуровневый API, который предоставляет разработчикам прямой доступ к GPU для полного контроля над его работой. Отличаясь более простыми и легкими драйверами, Vulkan демонстрирует меньшие задержки и меньшие накладные расходы при обработке графических команд (overhead) по сравнению с традиционными API OpenGL и Direct3D. Vulkan также отличается эффективной поддержкой многопоточности и позволяет многоядерным центральным процессорам более эффективно загружать графический конвейер, поднимая производительность существующего оборудования на новый уровень.

Vulkan – это первый низкоуровневый API нового поколения, который является кроссплатформенным. Разработчики могут создавать приложения для ПК, мобильных и встроенных устройств, работающих под различными операционными системами. Как и OpenGL, Vulkan – это открытый бесплатный стандарт, доступный для любой платформы. Однако NVIDIA продолжит работу над OpenGL и OpenGL ES, чтобы поддержать тех разработчиков, которые предпочитают использовать традиционные API [6].

ЛИТЕРАТУРА

1. Дисперсия света. [Электронный ресурс] Режим доступа: https://online.mephi.ru/courses/physics/optics/data/course/6/6.2.html
2. Руководство Unity. [Электронный ресурс] Режим доступа: https://docs.unity3d.com/ru/530/Manual/UnityManual.html
3. Документация Unity. [Электронный ресурс] Режим доступа: https://docs.unity.com/
4. Дисперсия света в Cycles. [Электронный ресурс] Режим доступа: https://blender3d.com.ua/dispersiya-sveta-v-cycles/
5. Что такое Unity? . [Электронный ресурс] Режим доступа: http://web.spt42.ru/index.php/chto-takoe-unity-3d
6. Vulcan. [Электронный ресурс] Режим доступа: https://www.nvidia.com/ru-ru/drivers/vulkan-graphics-api-blog/
7. Большая российская энциклопедия : [в 35 т.] / гл. ред. Ю. С. Осипов. — М. : Большая российская энциклопедия, 2004—2017.

УДК 004.4

Р. Р. Попов (4 курс бакалавриата),
Н. В. Воинов, к.т.н., доцент,
Е. А. Павлов, ассистент

РАЗРАБОТКА СИСТЕМЫ УЧЕТА ПОКАЗАТЕЛЕЙ ДЕЯТЕЛЬНОСТИ НЦМУ СПБПУ

Научный центр мирового уровня (НЦМУ) «Передовые цифровые технологии» создан в рамках Национального проекта «Наука» в соответствии с Указом Президента Российской Федерации от 7 мая 2018 г. № 204 [1]. В 2020 году Министерством науки и высшего образования РФ был проведен конкурс по присвоению статуса научного центра мирового уровня (НЦМУ) научным или образовательным организациям – победителям конкурса.

В настоящее время отсутствует необходимый набор программного обеспечения, который в полной мере покрывает все потребности университетов, необходимые им для организации бизнес-процессов, связанных с научной деятельностью, в частности, ведения учета показателей деятельности НЦМУ.

К деятельности НЦМУ предъявляются высокие требования по выполнению показателей и отчётности [2]. На текущий момент в СПбПУ отсутствует централизованная система по ведению и учету необходимых показателей.

Таким образом, *целью* данной работы является реализация сервиса, который будет представлять собой программно-аппаратный комплекс и предназначаться для ведения учета показателей деятельности НЦМУ СПбПУ. Программно-аппаратный комплекс представляет собой сочетание аппаратной и программной части. Программная часть состоит из специально разработанного под конкретные нужды программного обеспечения, серверная часть устанавливается на сервер СПбПУ. Программная часть оснащена хранилищем данных.

Для достижения цели необходимо решение следующих *задач*:

1. Обзор существующих решений, которые могут быть использованы для ведения учета показателей деятельности НЦМУ СПбПУ.
2. Проведение сравнительного анализа найденных решений.
3. Предложение разработки собственной системы учета показателей.
4. Реализация данной системы.
5. Демонстрация полученного результата.

Архитектура системы ведения учета представлена на Рис. 1. Данная система реализована с применением языков Python [3] вместе с Django Rest Framework [4] для написания API составляющей данной системы, а также JavaScript с фреймворком React [5], через который реализован web-интерфейс.

Фронтэнд составляющая на React:

- веб-интерфейс пользователя для взаимодействия с системой;
- React-компоненты для визуализации данных и работы с API бэкэнда;
- роутинг и навигация между страницами;
- стилизация и верстка страниц с использованием CSS-фреймворков (Bootstrap).

Бэкэнд составляющая на Django:

- аутентификация пользователей и авторизация доступа к функционалу системы;
- API для обмена данными между клиентской и серверной частями системы;
- организация базы данных и работы с моделями данных;
- обработка запросов от клиентской части и выдача ответов;
- бизнес-логика системы (например, анализ и обработка показателей деятельности, генерация отчетов и т.д.);
- использование RESTful API для обмена данными [6];
- организация запросов и ответов в формате JSON;
- использование HTTP-протокола для передачи данных между клиентом и сервером [7].

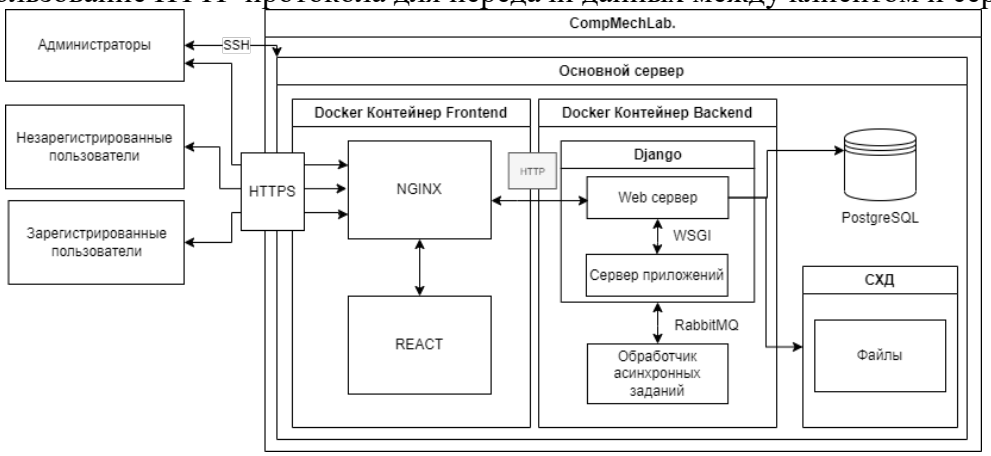


Рисунок 1 – Архитектура системы

Общий архитектурный стек системы выглядит следующим образом:

- клиентский код (React) будет размещен на сервере Nginx;
- серверный код (Django) будет размещен в контейнере Docker;
- общение между клиентским и серверным кодом будет осуществляться через API, реализованный в Django;
- Nginx будет использоваться как прокси-сервер для перенаправления запросов между клиентским и серверным кодом, а также для балансировки нагрузки и кеширования;
- база данных будет размещена на сервере.

Такая архитектура позволяет создать высокопроизводительную и масштабируемую систему, где каждый компонент выполняет свою задачу и может быть легко заменен или модифицирован в дальнейшем.

ЛИТЕРАТУРА

1. Научный центр мирового уровня «Передовые цифровые технологии». [Электронный ресурс] Режим доступа: https://ncmu.spbstu.ru/
2. Научные центры мирового уровня (НЦМУ). [Электронный ресурс] Режим доступа: https://minobrnauki.gov.ru/about/deps/dgnintp/nauka/ntcmu
3. Сараджишвили, С. Э. Введение в обработку изображений на языке Python / С. Э. Сараджишвили, В. В. Леонтьев, Н. В. Воинов. – Санкт-Петербург : Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2020. – 36 с. – ISBN 978-5-7422-6955-7. – DOI 10.18720/SPBPU/2/id20-76. – EDN GEJJDU.
4. Django Rest Framework documentation. [Электронный ресурс] Режим доступа: https://www.django-rest-framework.org
5. React documentation. [Электронный ресурс] Режим доступа: https://ru.reactjs.org/docs/getting-started.html
6. REST API (RESTful API) Alexander S. Gillis. [Электронный ресурс] Режим доступа: https://www.techtarget.com/searchapparchitecture/definition/RESTful-API
7. Бойков, К. М. Разработка клиент-серверной архитекутры для сервиса по управлению интерактивными подписками / К. М. Бойков, О. С. Командина, Н. В. Воинов // Современные технологии в теории и практике программирования : сборник материалов конференции, Санкт-Петербург, 23 апреля 2020 года / Санкт-Петербургский политехнический университет Петра Великого; Dell Technologies; EPAM Systems. – Санкт-Петербург: ПОЛИТЕХ-ПРЕСС, 2020. – С. 113-114. – EDN DTYFAF.

УДК 004.4

В. А. Почётный (4 курс бакалавриата),
Т. В. Леонтьева, к.т.н., доцент

МОБИЛЬНОЕ ПРИЛОЖЕНИЕ ДЛЯ ПОМОЩИ ЖИВОТНОВОДАМ

Целью работы является разработка специального мобильного приложения для животноводческого предприятия, которое существенно облегчит труд ветеринарного врача. С помощью предлагаемого приложения ветеринар сможет более эффективно оценивать здоровье животных, их способность к размножению, вести календарь прививок, отслеживать родственные связи между особями, создавать банк данных, планировать выведение потомства и выращивание молодняка, участвовать в селекционной работе. Такое приложение повысит качество ухода за животными и позволит вывести работу на животноводческих фермах на более высокий уровень.

На сегодня существует достаточно ограниченное количество мобильных приложений для работы в области животноводства, так как данный сегмент рынка ещё очень молод. Кроме того, много программ постоянно обновляется или исчезает из-за отсутствия поддержки. Большинство программ включают один, зачастую довольно узкий аспект, и для полноценного

ведения хозяйства приходится использовать сразу несколько приложений, что не слишком удобно. Очевидно, что развитие этой области будет идти по пути создания универсальных многофункциональных приложений.

Выбор и сравнение аналогов производился по следующим критериям: приложение или программа должно содержать базу данных о животных, помогать в планировании действий по уходу за животными и их разведением. Сравнение было проведено между следующими аналогами: Breed Manager by Moocall, Cattle Breeding Calculator R3 Consulting, Дайри Рацион, My Goat Manager, Коралл, Селэкс.

По результатам анализа можно сделать вывод, что основная часть приложений и программ разработана для крупного рогатого скота, свиней и птиц. Для козоводства разработано незначительное количество программ. Основная часть предлагаемых разработок предназначена для маленьких хозяйств и помогает в расчете рациона питания или учете общих расходов на содержание животных. Для таких направлений работы, как разведение и селекция, в разработках предлагаются лишь самые основные функции, не затрагивающие весь аспект необходимых мероприятий для успешного размножения. Кроме того, часть зарубежных разработок в данный момент недоступна для использования российскими хозяйствами.

На основе анализа предметной области, обзора аналогичных решений сформирован общий функционал разрабатываемой системы:

- регистрация пользователя в системе;
- авторизация по учетным данным;
- создание карточки животного и введение данных (имя, пол, дата рождения, место в стойле, вес, рост);
- ведение календаря вакцинации (указание дат планируемых и сделанных прививок);
- учет перенесенных заболеваний с указанием дат болезни и метода лечения;
- молочная продуктивность: лактации, учет удоев, лактационные кривые;
- формирование дат плановых случек;
- планирование проведения УЗИ, анализа крови / молока;
- планирование окота / отёла;
- push-уведомления о намеченных мероприятиях;
- оценка продуктивности разных поколений;
- возможность одновременной работы нескольких специалистов под одним аккаунтом;
- выход из аккаунта.

Для предотвращения близкородственного скрещивания следует произвести расчет коэффициента инбридинга по следующей формуле:

$$R_{xy} = \frac{\sum(1/2)^{n+n_1}(1+fa)}{\sqrt{(1+fx)(1+fy)}} \times 100$$

где R_{xy} - коэффициент генетического сходства между животными x и y, %;

n - ряд в родословной животного x, в котором встречается общий предок;

n_1 - ряд родословной животного y, в котором встречается общий предок;

fa - коэффициент инбридинга для общего предка (в десятичных дробях);

fx - коэффициент инбридинга для животного x (в десятичных дробях);

fy - коэффициент инбридинга для животного y (в десятичных дробях).

При коэффициенте 25% и более инбридинг считается тесным (кровосмешение), от 12,5 до 25% – близким, от 1,55 до 12,5% – умеренным, от 0,20 до 1,55% – отдалённым.

В наши дни на рынке мобильных устройств самыми распространенными являются устройства, разработанные на базе мобильных платформ iOS и Android. При этом продукты, разработанные на платформе Android, является более популярными среди потребителей по данным маркетингового исследования за 2022 год. Таким образом, предлагаемое мобильное

приложение целесообразно разработать для платформы Android. Для разработки понадобится использовать библиотеку Android SDK.

Компания-разработчик операционной системы Android рекомендует использовать среду разработки Android Studio для наиболее корректной работы разрабатываемого приложения.

Достоинства Android Studio:

- среда разработки поддерживает работу с несколькими языками программирования, к которым относятся C/C++, Java и Kotlin.
- тестирование корректности разработки происходит непосредственно в эмуляторе;
- большая библиотека с готовыми шаблонами и компонентами для разработки ПО;
- большой набор средств для тестирования каждого элемента приложения.

Android Studio предоставляет комплекс средств для написания кода на языке программирования Java. Java является основным языком для разработки Android-приложений. Соответственно, именно язык программирования Java был выбран для работы.

Для информационного обеспечения системы была выбрана СУБД MySQL версии 8.0. Преимущества MySQL:

- простота в работе: установить MySQL довольно просто;
- богатый функционал: MySQL поддерживает большинство функционала SQL;
- безопасность: большое количество функций, обеспечивающих безопасность, которые поддерживаются по умолчанию;
- масштабируемость: MySQL легко работает с большими объемами данных и легко масштабируется;
- скорость: упрощение некоторых стандартов позволяет MySQL значительно увеличить производительность.

Разработанное приложение предполагается использовать на животноводческом предприятии Ленинградской области АО ПЗ «Красноозерное». Приложение может быть модернизировано с учётом пожеланий и замечаний заказчика.

ЛИТЕРАТУРА

1. Рудаков А. В., Федорова Г. Н. Технологии разработки программных продуктов. Практикум. 4-е изд. М.: Академия, 2014. 192с.
2. Орлов С. А. Технологии разработки программного обеспечения. 5-е изд. СПб: Питер, 2016. 770с.
3. Android tools [Электронный ресурс]: Использование паттерна MVP в Android – Режим доступа: https://android-tools.ru/coding/ispolzovanie-patterna-mvp-v-android/
4. Компоненты сетевого приложения. Клиент-серверное взаимодействие и роли серверов. [Электронный ресурс]. Режим доступа: http://www.4stud.info/networking/lecture5.html
5. Помощь компьютера в животноводстве. [Электронный ресурс]. Режим доступа: https://soft-agro.com/korovy/pomoshh-kompyutera-v-zhivotnovodstve.html
6. Животноводческое хозяйство: разведение коз. [Электронный ресурс]. Режим доступа: https://www.openbusiness.ru/biz/business/zhivotnovodcheskoe-khozyaystvo-razvedenie-koz/

УДК 004.453

А.М. Потапова (4 курс бакалавриата),
Н.В. Воинов., к.т.н., доцент,
Ю.В. Юсупов, к.т.н

РАЗРАБОТКА IOS-ПРИЛОЖЕНИЯ ДЛЯ ЗАПИСИ ВОСПОМИНАНИЙ

Как известно, на нашу память влияет ряд факторов, такие как: недостаток сна, злоупотребление вредными привычками, стрессовые ситуации, нагруженный график и т.д. Кроме того, нередки случаи клинических заболеваний, вызывающих ухудшение или вовсе

потерю памяти. В обоих случаях людям важна оперативность записи воспоминаний и простой доступ к записанному.

Таким образом, была сформулирована цель работы: разработать приложение для оперативной записи воспоминаний на платформе iOS [1] с учетом устранения недостатков существующих решений, добавления новых функций и минимизации количества итераций от запуска приложения до начала записи.

Для достижения поставленной цели, был выделен ряд задач:

– провести анализ существующих аналогов;
– реализовать прототип для анализа поведения пользователя с целью улучшения функционала;
– разработать архитектуру полноценного приложения;
– реализовать приложение;
– провести тестирование.

Поскольку идея является не новой, был проведен анализ популярных аналогов с целью выявления недостатков и формирования принципиально новых решений (Табл.1). Среди них можно выделить Memoires, Live Diary, Card Diary и Alta.

Таблица 1 – Сравнение аналогов разрабатываемого приложения (РП)

	РП	Memoires	Live Diary	Card Diary	Alta
Адаптированный под различных пользователей понятный интерфейс	+	+/–	–	+/–	+
Отсутствуют проблемы с хранением данных	+	–	+	–	–
Отсутствуют программные ошибки	+	–	–	–	–
Русская локализация	+	+/–	+/–	–	+
Полноценный бесплатный функционал	+	+	–	–	–

Из выявленных недостатков самыми существенными оказались проблемы с интерфейсом, ненадежное хранение данных, платный функционал, проблемы с русской локализацией и программные ошибки, препятствующие комфортному использованию приложения. В разрабатываемом решении предлагается устранение данных недостатков и внедрение нового функционала в виде поддержки оперативной записи, добавления статистических данных и расширения взаимодействия с текстовыми и медиа данными.

В качестве инструмента разработки был выбран нативный язык Swift [2], который наиболее оптимизирован для разработки под операционную систему IOS. Для конструирования компонентов пользовательского интерфейса были выбраны библиотеки UIKit [3] и SwiftUI [4]. Использование двух библиотек обусловлено необходимостью реализации UI компонентов различных уровней сложности. Для простейших элементов интерфейса – SwiftUI, для более сложных – UIKit. Для доступа к основным типам данных, коллекциям и службам операционной системы был выбран фреймворк Foundation. В качестве шаблона проектирования архитектуры приложения был выбран MVVM [4]. Данный паттерн позволит упростить процесс модульного тестирования, повысит удобство обслуживания и обезопасит процесс взаимодействия с данными. Графическое представление связи модулей в архитектуре проекта представлено на Рис. 1.

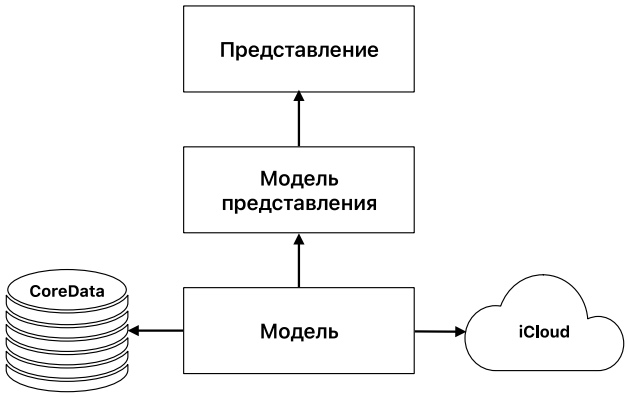


Рисунок 1 – Графическое представление связи модулей в архитектуре проекта

Контроллер представления получает и отображает информацию. В данном модуле расположены следующие контроллеры:

– представление со списком записей;
– представление для ввода воспоминания;
– представление со статистикой.

Модель представления получает информацию от контроллера представления, обрабатывает эту информацию и предоставляет выходные данные обратно в контроллер. Таким образом, в данном модуле реализовано поведение представлений, перечисленных выше.

Модель содержит описание данных, хранящихся в базе Core Data [6]. В этом модуле реализована база данных приложения, содержащая данные о записях и фото/видео материалах пользователя.

ЛИТЕРАТУРА

1. iOS 16 – Apple [Электронный ресурс]. URL: https://www.apple.com/ios/ios-16/
2. Apple Incorporated. About Swift // The Swift Programming Language (Swift 5.7 Edition), 2022. – С. 2-4.
3. UIKit | Apple Developer Documentation [Электронный ресурс]. URL: https://developer.apple.com/documentation/uikit (дата обращения: 12.03.2023)
4. SwiftUI Overview - Xcode - Apple Developer [Электронный ресурс]. URL: https://developer.apple.com/xcode/swiftui/ (дата обращения: 12.03.2023)
5. MVVM in iOS Swift – Medium [Электронный ресурс]. URL: https://medium.com/@abhilash.mathur1891/mvvm-in-ios-swift-aa1448a66fb4 (дата обращения: 11.03.2023)
6. Core Data | Apple Developer Documentation [Электронный ресурс]. URL: https://developer.apple.com/documentation/coredata (дата обращения: 14.03.2023)

УДК 004.912

И. О. Прищепа (4 курс бакалавриата),
А. В. Самочадин, к.т.н., доцент

ПРИЛОЖЕНИЕ ДЛЯ ПРОСМОТРА И РЕДАКТИРОВАНИЯ PDF-ДОКУМЕНТОВ

Формат документов PDF давно стал стандартом во многих областях деятельности. Изначально, основной целью его создания было сохранение идентичного отображения документа на различных платформах, независимо от аппаратного и программного обеспечения. Но, из-за возросшей популярности PDF, часто возникает потребность в таких действиях, как поиск по документу, копирование и редактирование содержимого. Так как формат не предназначен для таких целей, с этим часто возникают сложности [1].

Я выделил следующие основные проблемы:
– Поиск по документу часто позволяет найти искомую фразу только в том случае, если она целиком размещена на одной строке, без переносов;
– При копировании абзаца из нескольких строк, он разбивается на несколько отдельных абзацев-строк;
– Если текст на странице расположен в несколько колонок, либо если нужно скопировать текст из таблицы, ситуация ещё хуже – многие программы для работы с PDF могут считать одной строкой никак не связанные по смыслу фрагменты текста, расположенные в разных колонках или ячейках таблицы [2];
– При выделении и копировании текста с нескольких страниц сразу, копируются колонтитулы и номера страниц [3];
– При копировании блоков моноширинного текста (например, листингов кода), пропадают все отступы, и данные приходится форматировать заново;
– Копирование изображений, либо изображений вместе с текстом, зачастую не поддерживается.
– PDF-документы проблематично редактировать. Зачастую нужно быстро поправить ошибку или обновить какое-то значение в документе, но для этого приходится искать исходный документ в другом формате. Хотя программы для редактирования PDF и существуют, но они являются тяжеловесными комбайнами, а для простого редактирования хорошо бы подошел более легковесный и быстрый инструмент. К тому же, большинство таких программ платные.

Целью моей работы является реализация комплексного программного средства для просмотра PDF-документов, в котором отсутствовали бы обозначенные выше недостатки, присутствовали все стандартные функции работы с PDF, а также была возможность простого редактирования — изменение текста (в т.ч. выделение, изменение форматирования), замена изображений, перемещение элементов.

Для достижения этой цели требуется решение следующих задач:
1. Обзор существующих инструментов для работы с PDF-документами;
2. Выделение стандартных возможностей, которыми должен обладать инструмент;
3. Реализация алгоритмов обработки документа для решения обозначенных проблем;
4. Поиск библиотеки для работы с PDF-документами либо написание собственной;
5. Реализация пользовательского интерфейса для реализованных ранее алгоритмов обработки документа, а также реализация стандартных функций обработки документа;
6. Демонстрация решения инструментом обозначенных проблем.

Ключевым элементом в архитектуре разрабатываемого программного средства являются алгоритмы обработки документа. В формате PDF фрагменты данных не объединены и не упорядочены по смыслу — из этого и вытекают все обозначенные проблемы. Поэтому требуется обрабатывать документ для объединения разрозненных фрагментов в смысловые блоки, с которыми пользователь сможет работать. Также нужно по расположению текста восстанавливать форматирование, и нужно классифицировать тип текста — выделять колонтитулы, сноски, подписи к изображениям. При внесении пользователем изменений в смысловые блоки, должны изменяться соответствующие нижележащие фрагменты данных в PDF-формате.

Для разработки приложения выбран язык Java из-за его кроссплатформенности. Для работы с форматом PDF используется SDK iText7 [4], для создания пользовательского интерфейса — JavaFX [5].

ЛИТЕРАТУРА

1. Chao, H. Layout and content extraction for pdf documents // International Workshop on Document Analysis Systems. — Berlin, Heidelberg : Springer, 2004. — С. 213-224.
2. Gemelli A., Vivoli E., Marinai S. Graph Neural Networks and Representation Embedding for Table Extraction in PDF Documents // 26th International Conference on Pattern Recognition. — IEEE, 2022.

3. Changfeng Yu, Cheng Zhang, Jie Wang Extracting Body Text from Academic PDF Documents for Text Mining // 12th International Conference on Knowledge Discovery and Information Retrieval. - Lowell: University of Massachusett, 2020
4. Lowagie B. iText in Action / B. Lowagie. – 2-е издание. : Manning Publications Co., 2011.
5. JavaFX Documentation: [Электронный ресурс] // Oracle. URL: https://docs.oracle.com/javafx/2/

УДК 004.4

Д. С. Пятов (4 курс бакалавриата),
Т. В. Леонтьева, к.т.н, доцент

РАЗРАБОТКА ПРОГРАММНЫХ СРЕДСТВ ДЛЯ 3D-ПЕЧАТИ НА SLS-ПРИНТЕРАХ

SLS, или селективное лазерное спекание – это метод 3D-печати, разработанный и запатентованный докторами Техасского университета (Остин, США) Карлом Декардом и Джо Биманом в середине 1980-х годов. Первые коммерческие принтеры, основанные на этой технологии, были выпущены в 1992 году. Она заключается в нанесении порошкообразного материала на поверхность подложки, его равномерного распределения, и сканировании поверхности лазером. Под влиянием лазера порошок спекается и таким образом формируется слой изготавливаемого объекта, соответствующий текущему срезу изделия. После сканирования поверхности подложка опускается на толщину слоя изделия, при необходимости добавляется и распределяется порошок, и проводится печать следующего среза. После печати всех слоев готовый объект очищается от неиспользованного порошка и остывает. Преимуществами такой технологии являются отсутствие необходимости в дополнительном опорном материале, возможность одновременной печати нескольких изделий и многообразие возможных материалов для использования [1].

В 2014 году истек срок последнего заявленного патента по технологии селективного лазерного спекания, поэтому с тех пор она является общедоступной [2]. В связи с этим на рынке 3D-принтеров возникла конкуренция, которая вкупе с развитием технологий привела к появлению множества предложений в различных ценовых сегментах. В свою очередь, с возникновением доступности этих инструментов при их удобстве и универсальности, в последние годы возрос и спрос на них как от малого, так и от крупного бизнеса [3].

Большая часть таких принтеров производится зарубежными компаниями, при этом сложность их устройства и коммерческие интересы этих компаний зачастую ведут к тому, что для обслуживания этих инструментов необходимо вызывать их сотрудников. В связи с уходом из России многих зарубежных производителей и действующими санкциями очевидно, что доступность приобретения и обслуживания зарубежных принтеров в нашей стране снизилась. Учитывая это и взятый курс на импортозамещение, остро встает вопрос о производстве отечественных 3D-принтеров и программного обеспечения для них. При этом проблемой является недостаток свободных для использования программных средств такой направленности. Таким образом, целью проводимой работы является упрощение производства 3D-принтеров путем создания необходимого для их работы программного обеспечения с открытым исходным кодом.

Для достижения этой цели необходимо решить следующие задачи:

1. Составление требований к проектируемому программному обеспечению.
2. Обзор существующих программных средств для 3D-печати.
3. Сравнительный анализ найденных средств.
4. Поиск необходимых для реализации инструментов и теоретических сведений.
5. Реализация программного средства.

Предложенное программное обеспечение является модульным, соответственно есть возможность использовать только необходимые его компоненты. Предполагаемыми модулями программы являются:

1. Модуль разделения 3D-фигуры на слои.

2. Модуль нахождения пути лазера на слое.
3. Модуль графического интерфейса
4. Модуль авторизации пользователей.

Модуль разделения фигуры на слои принимает на вход STL модель [4] и толщину слоя. На выходе выдает набор слоев, при этом каждый слой представляет собой набор многоугольников (внешних и внутренних). Для этого необходимо привести входные данные к треугольникам, из которых и состоит 3D модель, выделить треугольники, пересекающие плоскость разрезания, и определить линии пересечения этой плоскости с треугольниками.

Модуль нахождения пути лазера на слое является самым важным, так как производит необходимую для печати информацию. На вход принимает набор многоугольников и толщину лазера, а на выходе выдает набор линий (маршрут лазера) с информацией о его состоянии (включен/выключен). Предполагается представлять многоугольники как список вершин, при том соседние вершины идут друг за другом в этом списке. Первая и последняя вершина в списке также являются соседними. Необходимо найти оптимальный маршрут, на котором время прохода лазера в выключенном состоянии будет минимальным. Один из способов поиска такого маршрута предполагает разделение невыпуклого многоугольника на выпуклые, их штриховку и соединение заштрихованных областей [5]. К тому же, нужно решить проблему поиска срединных линий для участков слоя, ширина которых меньше двух диаметров лазера.

Модуль графического интерфейса позволяет визуализировать маршрут лазера, а также задавать параметры печати. Представляет из себя оконное приложение, в котором можно выбрать STL файл, определить толщину слоя и лазера и увидеть маршрут на каждом из слоев.

Модуль авторизации предназначен для ограничения доступа к управлению принтером неавторизированных пользователей.

Разработка ведется для ОС Linux на языке C++. Для написания кода используется IDE CLion [6]. Для реализации графического интерфейса используется библиотека GTK+ [7].

ЛИТЕРАТУРА

1. Султанова Ф. Р., Нам И. Э., Мирзахакимов С. Б. Технология селективного лазерного спекания (SLS) // Международный научный журнал «Инновационная наука» №10–2/2016 ISSN 2410–6070
2. Carl R. Deckard. Патент US5597589A "Apparatus for producing parts by selective sintering"
3. Кокорев И. В. Анализ развития 3D печати // Научный журнал «E-Scio» №11/2020 ISSN 2658–6924
4. STL (file format). [Электронный ресурс] Режим доступа: https://en.wikipedia.org/wiki/STL_(file_format)
5. G. Gomez, C. Cortes, C. Creus, M. Zelaia, A. Moreno. Generation of continuous hybrid zig-zag and contour paths for 3D printing. [Электронный ресурс] Режим доступа: https://www.researchgate.net/publication/353425686_Preprint_Generation_of_Continuous_Hybrid_Zig-zag_and_Contour_Paths_For_3D_Printing
6. CLion. A cross-platform IDE for C and C++. [Электронный ресурс] Режим доступа: https://www.jetbrains.com/clion/
7. The GTK Project. [Электронный ресурс] Режим доступа: https://www.gtk.org/

УДК 004.91:004.422.8

Д. В. Смелков (1 курс магистратуры),
О. Н. Петров, к.т.н., доцент

ПЛАТФОРМА УПРАВЛЕНИЯ ЗНАНИЯМИ В ОБРАЗОВАТЕЛЬНОМ ПРОЦЕССЕ

Система управления знаниями подразумевает из себя систему, позволяющую хранить, управлять, собирать, структурировать, искать информацию для дальнейшего использования её человеком, а также извлекать новые знания из представленных данных. Платформа, позволяющая организовать систему управления знаниями, даст возможность для человека, желающего получить интересующую информацию в явном виде, собранную в одном месте.

Одним из способов реализации системы управления знаниями является создание платформы, состоящей из рабочего приложения, хранящей информацию базы данных и различных модулей, подключаемых к созданному приложению для расширения функциональности. Модули уменьшают зависимость всей платформы от языка программирования, разработчика, тем самым упрощаются разработка, тестирование и обследование платформы. Появляется возможность настроить под пользователя необходимые части программного обеспечения.

Одним из популярных интерфейсов создания приложений является Windows Forms, предоставляющий возможность эффективного создания классических приложений, простоту разработки с простым развёртыванием, обновлением и удобной работой как в автономном режиме, так и в сети. Данный интерфейс независим от языка разработки, то есть имеется возможность работать с такими языками программирования как C#, C++, так и на VB.Net, J# и др.

Для достижения цели работы были решены следующие задачи:

1. Обзор и анализ существующих языков программирования для работы с Windows Forms.
2. Реализация приложения, способного динамически подключать модули.
3. Реализация библиотеки классов для работы с полученными данными и хранения информации.
4. Реализация сервиса для считывания необходимой информации из источника данных.
5. Реализация первичных модулей для демонстрации работы системы.

Одной из основных частей системы является приложение, особенности разработки и производительность которого существенно зависят от языка программирования. В ходе сравнительного анализа был выбран язык программирования C#.

Все модули подключаются к приложению динамически во время загрузки и представлены в виде .dll файлов. В связи с этом в каждом подмодуле должны быть реализованы методы Create и GetInfoName. Метод Create создает новый экземпляр подмодуля и возвращает его, на вход этот метод получает информацию, необходимую для заполнения модуля информацией. Метод GetInfoName возвращает информационное название подмодуля.

Для реализации сервиса сбора информации был выбран тот же язык программирования, источниками данных являются: сайт университета и документы формата docx, хранящие информацию о дисциплинах.

Для получения информации с сайта университета выбрана библиотека AngleSharp – это библиотека классов, предназначенная для парсинга сайтов. Данная библиотека позволяет собирать информацию со страницы без открытия браузера и затем обрабатывать её, получая DOM структуру в соответствии с спецификациями W3C, что позволяет осуществлять поиск необходимых данных, получать их и (или) изменять. AngleSharp имеет высокую производительность и информативность по сравнению с другими аналогичными библиотеками.

Получение информации из документов реализовано с использованием библиотеки Microsoft.Office.Interop.Word – это библиотека динамических классов для работы с документами word. Данная библиотека позволяет открывать, создавать, изменять документы с расширением .DOC, .DOCX, .DOT и других.

В общем виде архитектура системы управления знаниями представлена на Рис. 1. Модули подразделяются на тематические (реализующие отображение информации на определённую тематику) и функциональные (реализующие функционал обработки и получение информации). Обмен данными с сервером осуществляется посредством TCP – один из основных протоколов передачи данных в интернете. В стеке протоколов TCP/IP выполняет функции транспортного уровня модели OSI.

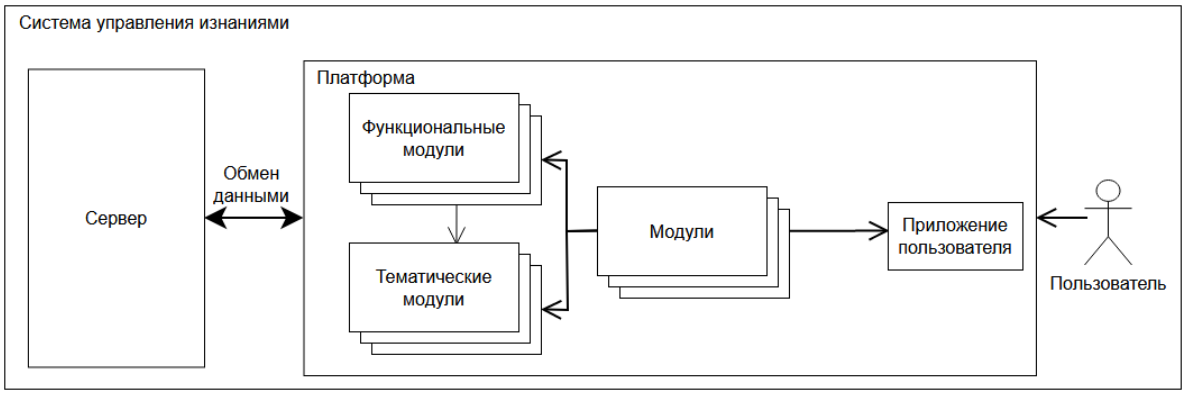


Рисунок 1 – архитектура системы.

Всё взаимодействие с платформой происходит через приложение пользователя путем нажатия на функциональные элементы, каждый из которых имеет в себе некоторый набор вызовов функций системы.

В приложении пользователя загружаются все имеющиеся модули, доступные ему по его типу, после чего он имеет доступ к отображению каждого из модулей. В тематических модулях осуществляется обращение к функциональным модулям, в которых происходит обработка информации и обращения к серверу для получения данных.

ЛИТЕРАТУРА

1. HR-Академия | Система управления знаниями — [URL]: https://hr-academy.ru/hrarticle/sistema-upravleniya-znaniyami.html
2. Троелсен, Э. Язык программирования C# 5.0 и платформа .NET 4.5, 6-е издание / Эндрю Троелсен. — Москва: Издательство «Вильямс», 2013. — 1312 с. — ISBN 978-5-8459-1814-7. — Текст: электронный — [URL]: https://www.goodreads.com/book/show/19048740-c-5-0-net-4-5?rating=4
3. Что такое Windows Forms - Windows Forms .NET | Microsoft Docs — [URL]: https://docs.microsoft.com/ru-ru/dotnet/desktop/winforms/overview/?view=netdesktop-6.0
4. C# и .NET | Введение — [URL]: https://metanit.com/sharp/tutorial/1.1.php
5. Многоуровневые системы клиент-сервер | Сети/Network world | Издательство «Открытые системы» [URL]: https://www.osp.ru/nets/1997/06/142618
6. Просиз, Д. Программирование для Microsoft. / Джеф Просиз — Москва: Русская редакция, 2003. — С. 704. — ISBN 5-7502-0217-8. — Текст: электронный — [URL]: https://knigogid.ru/books/1016619-programmirovanie-dlya-microsoft-net
7. Камер, Д. Сети TCP/IP, том 1. Принципы, протоколы и структура / Дуглас Камер — Москва: «Вильямс», 2003. — С. 880. — ISBN 0-13-018380-6. — Текст: электронный — [URL]: https://www.studmed.ru/kamer-duglas-e-seti-tcpip-principy-protokoly-i-struktura-tom-1_27bc418b53d.html
8. Office - основные сборки взаимодействия - Visual Studio (Windows) | Microsoft Docs — [URL]: https://docs.microsoft.com/ru-ru/visualstudio/vsto/office-primary-interop-assemblies?view=vs-2022

УДК 004.91:004.422.8

Д. В. Смелков (1 курс магистратуры),
О. Н. Петров, к.т.н., доцент

МОДУЛЬ СТУДЕНТА В СИСТЕМЕ УПРАВЛЕНИЯ ЗНАНИЯМИ

Модуль студента в системе управления знаниями предназначен для использования его студентами университета для оценки своей успеваемости, нагрузки на семестр, расписания, просмотра списка литературы и так далее.

Для решения поставленных целей модуля студента пришлось решить следующие задачи:

1. Авторизация студента.

2. Получение информации о студенте.

3. Получение информации о рабочих программах по дисциплинам.

4. Оценка успеваемости студента с учетом оценок и компетенций по дисциплинам.

Необходимую информацию о студенте можно получить с сайта университета, в связи с этим авторизация студента происходит через сайт университета (информационную систему университета СПбГМТУ). Для авторизации создан отдельный модуль системы, который возможно в дальнейшем использовать для других тематических модулей системы. Модульный подход позволяет при необходимости менять способ авторизации (для того же университета или для другого) без изменения как платформы, так и модуля студента.

Получение информации о студенте происходит после удачной авторизации с сайта университета путем парсинга сайта. Для парсинга используется библиотека AngleSharp. Происходит получение такой информации как ФИО, фото, номер группы, успеваемость, программа обучения и расписание студента.

Получение информации о рабочих программах по дисциплинам осуществляется из файлов с рабочими программами. Так как не всегда студент может получить быстрый доступ к необходимым файлам, для работы с ними и своевременного обновления используется специальный файловый сервер для их хранения. Для получения информации из файла используется библиотека Microsoft.Office.Interop. Получение данных о дисциплинах происходит благодаря подключения модуля студента к специальному серверу, обрабатывающему файлы. Из рабочих программ дисциплин получается такая информация как: название дисциплины, компетенции по дисциплинам и список литературы. Связь между модулем студента и сервером происходит с использованием технологии клиент-сервер с использованием протокола TCP/IP.

Оценка успеваемости студента происходит путем сравнения успеваемости студента по пройденным дисциплинам и компетенций по уже пройденным и будущим дисциплинам. По окончании сравнения определяется дисциплина и задачи по компетенциям, которых студент не проходит.

В общей сложности разработано 6 функциональных модулей:

1. Проверка успеваемости – проверяет насколько студент готов к будущим дисциплинам.

2. Литература на семестр – отображает список литературы, которую можно изучать для более полного понимания дисциплин на весь семестр.

3. Успеваемость – отображает таблицу, содержащую успеваемость студента в течении выбранного семестра.

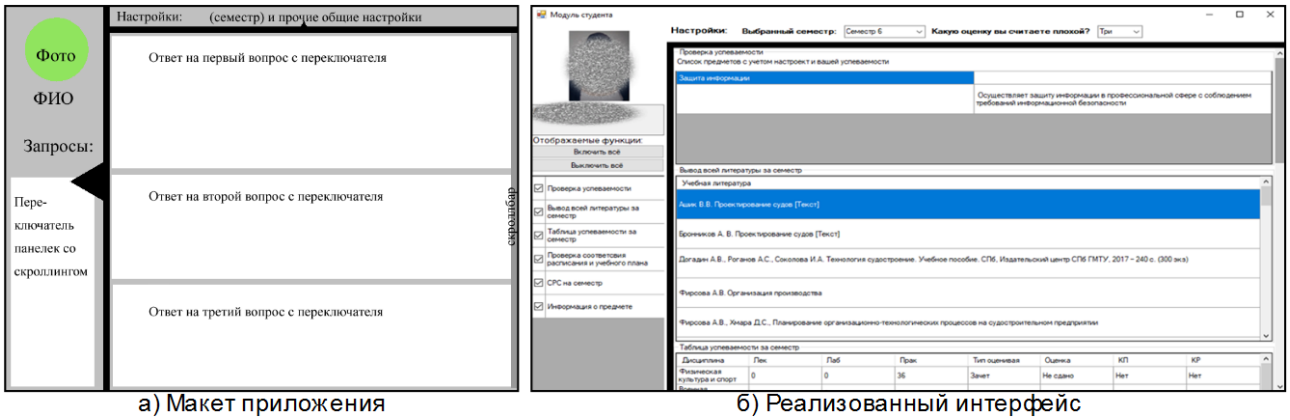
4. Проверка соответствия учебного плана и расписания – проверяет, соответствует ли составленное расписание на семестр поставленному в плане количеству часов.

5. СРС на семестр – отображает в виде гистограммы количество часов самостоятельной работы студента в течении семестра с пометками о наличии курсовых проектов (работ).

6. Информация о дисциплине – отображает о выбранной в списке дисциплине информацию о днях проведения занятия и списке литературы, необходимой для изучения дисциплины.

Для сохранения данных, необходимых для работы модуля студента, разработана специальная библиотека классов, позволяющая хранить следующую информацию: успеваемость, учебный план, дисциплину, занятия, литературу, семестры, преподаватели, расписание, рабочие программы дисциплин. В дальнейшем эту библиотеку можно использовать для других тематических модулей.

Для удобного использования приложения был разработан макет, представленный на Рисунке 1.а. В ходе разработки для создания пользовательского интерфейса использовался интерфейс Windows Forms, так он поддерживает несколько языков программирования, что удобно при использовании модульной системы приложения, так как различные модули, подключаемые к приложению, могут быть написаны на различных языках программирования. Приложение разработано в соответствии с макетом и представлено на Рисунке 1.б.



а) Макет приложения б) Реализованный интерфейс

Рисунок 1 – Внешний вид приложения.

В модуле студента все созданные подмодули динамически загружаются при запуске приложения и пользователь имеет возможность отобразить только те подмодули, что интересуют его в текущий момент времени. Вверху окна имеются базовые настройки, в которых можно изменить семестр, о котором необходимо получить информацию, и так же изменить то, какая оценка считается для пользователя нежелательной для учета в оценке успеваемости.

ЛИТЕРАТУРА

1. Что такое Windows Forms - Windows Forms .NET | Microsoft Docs — [URL]: https://docs.microsoft.com/ru-ru/dotnet/desktop/winforms/overview/?view=netdesktop-6.0
2. C# и .NET | Введение — [URL]: https://metanit.com/sharp/tutorial/1.1.php
3. Многоуровневые системы клиент-сервер | Сети/Network world | Издательство «Открытые системы» [URL]: https://www.osp.ru/nets/1997/06/142618
4. Просиз, Д. Программирование для Microsoft. / Джеф Просиз — Москва: Русская редакция, 2003. — С. 704. — ISBN 5-7502-0217-8. — Текст: электронный — [URL]: https://knigogid.ru/books/1016619-programmirovanie-dlya-microsoft-net
5. Камер, Д. Сети TCP/IP, том 1. Принципы, протоколы и структура / Дуглас Камер — Москва: «Вильямс», 2003. — С. 880. — ISBN 0-13-018380-6. — Текст: электронный — [URL]: https://www.studmed.ru/kamer-duglas-e-seti-tcpip-principy-protokoly-i-struktura-tom-1_27bc418b53d.html
6. Office - основные сборки взаимодействия - Visual Studio (Windows) | Microsoft Docs — [URL]: https://docs.microsoft.com/ru-ru/visualstudio/vsto/office-primary-interop-assemblies?view=vs-2022
7. Пусть за нас все делают боты. Парсинг сайтов на C# и Anglesharp | DFT — [URL]: https://dtf.ru/u/389606-languid-basil/890200-pust-za-nas-vse-delayut-boty-parsing-saytov-na-c-i-anglesharp

УДК 004.773.2

К. Спасоевич (4 курс бакалавриата),
П. Д. Дробинцев, к.т.н., доцент

РЕАЛИЗАЦИЯ СОЦИАЛЬНОЙ СЕТИ ДЛЯ ОБУЧЕНИЯ ИНОСТРАННЫМ ЯЗЫКАМ

В современном мире, изучение иностранных языков является ключевым фактором для развития человека в личной жизни, профессиональной карьере и дает преимущества на международном рынке труда. Современный мир становится боле глобализированным, а владение иностранными языками дает возможности для международного общения и сотрудничества, позволяя лучше понимать и учитывать культуру и менталитет других народов.

Также необходимо отметить, что сегодня социальные сети являются важной частью любого общества и зачастую основным средством получения информации. Осваивание языка

с помощью медиаресурсов довольно захватывающий способ изучения языка т.к. каждый изучающий выбирает свой путь основываясь на своих интересах и желаниях. В добавок это позволяет лучше понять и использовать язык в реальной жизни, услышав или прочитав жаргонную, современную лексику, а в некоторых случаях и лексику конкретной эпохи, если всё смотреть или читать на оригинальном языке, без озвучки или перевода.

Исходя из этого, текущая задача состоит в том, чтобы реализовать платформу для распространения медиаконтента на различных языках и взаимодействия с другими пользователями в реальном времени, с целью изучения иностранного языка, через поп-культуру.

Чтобы выполнить задачу, необходимо реализовать классическое клиент-серверное приложение. Есть несколько способов это сделать, используя язык программирования JavaScript, комбинируя несколько разных технологий, например: базу данных Mongo, Express.js, фреймворк для сервера, который обрабатывает HTTP запросы, Node.js и один фреймворков для JavaScript: React.js или Angular.js. Такие наборы технологий имеют соответсвующее название: MERN [3], если использовать React.js [1] или MEAN [4], если использовать Angular.js [2].

Как вариант, есть ещё и Firebase [5], фреймворк для Node.js, который позволяет создавать веб-приложения и API, Socket.io, библиотеку, которая позволяет создавать веб-приложения в реальном времени используя WebSockets. Данная задача будет реализоваться с использованием MERN стека.

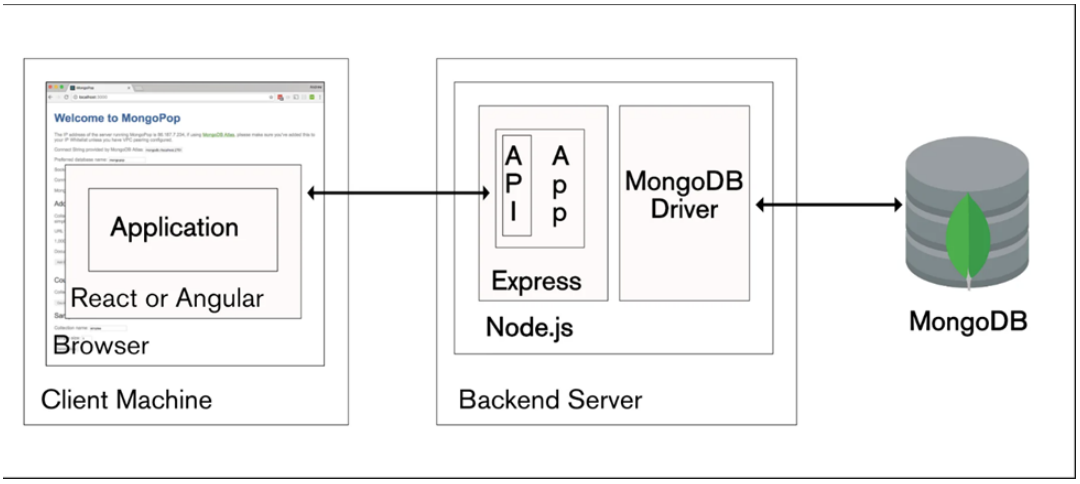


Рисунок 1 – Архитектура клиент-серверного приложения с использованием MERN стека

База данных Mongo [6] является документоориентированной NoSQL базой данных, которая используется для хранения и обработки данных. MongoDB хранит данные в формате JSON, что облегчает работу с ними. В контексте данной задачи, эти данные представляют учетные записи пользователей социальной сети и их публикации.

Как было уже упомянуто, Express.js представляет фреймворк для Node.js, с помощью которого можно создавать веб-приложения и API. Он предоставляет множество встроенных функций, таких как маршрутизация, управление сеансами и обработка ошибок. В контексте задачи, эти функции манипулировали запросами получения и отправки данных в MongoDB, то есть POST и GET запросами. Вместе с Express, на серверной части используется ещё о Node.js. с помощью которого можно запускать код на JavaScript. На клиентской части имеется React.js, с помощью которого создается пользовательский интерфейс, то есть формы для авторизации в социальную сеть, рамки публикации, и другие различные иконки и поля.

Таким образом, финальным результатом будет социальная сеть со всеми функциями, которыми должна обладать такая платформа сегодня: возможность публикации постов и создания комментариев, поиска соответствующих публикации, по ключевым словам, и переписки в чате в реальном времени. Дополнительным плюсом будет то, что данной социальной сетью будут пользоваться только люди, желающие изучать иностранные языки и

другие культуры и тем самым всё содержимое будет ориентированно именно на это, что в случае других классических социальных сетей нельзя сказать, ведь за счет количества их пользователей растёт и количество различных тем встречающихся в их пространстве.

ЛИТЕРАТУРА

1. Официальная документация React.js [Электронный ресурс], режим доступа: https://react.dev/
2. Официальная документация Angular.js [Электронный ресурс], режим доступа: https://angular.io/docs
3. Что такое стек MERN, и как с ним работать? [Электронный ресурс], режим доступа: https://habr.com/ru/company/piter/blog/458096/
4. Стек MEAN. Пример использования [Электронный ресурс], режим доступа: https://habr.com/ru/company/piter/blog/279237/
5. Firebase - App success made simple [Электронный ресурс], режим доступа: https://www.npmjs.com/package/firebase
6. Официальная документация MongoDB [Электронный ресурс], режим доступа: https://www.mongodb.com/

УДК 004.42

Г. В. Смородников (4 курс бакалавриата),
А. В. Самочадин., к.т.н., доцент

РАЗРАБОТКА ПЛАТФОРМЫ ДЛЯ ОНЛАЙН ИГР

С появлением большого количества игр стали появляться онлайн-сервисы для распространения и агрегации компьютерных игр, а с появлением технологии блокчейн (англ. blockchain)[2] в интернете стали появляться и популярные игры, в которых можно играть и зарабатывать деньги.

Такие как Illuvium, где ты исследуешь игровой мир, сражаешься с соперниками и за это получаешь реальные деньги. И как пример агрегатора цифровых проектов – это Steam, собравший на своей платформе более 11 тысяч игр.

Проблематика для разработчиков игр:

1. У игроков отсутствует доверие к незнакомым играм, и они боятся вкладывать реальные деньги;
2. Также для разработчиков Web3[4,6] игр отсутствует программное ядро, некий начальный интерфейс с базовыми игровыми и финансовыми механиками, чтобы упростить разработку собственной игры;
3. Присутствует высокий порог входа на крупные игровые платформы-агрегаторы.

Для игроков:

1. Отсутствие сходства между играми, каждая игра уникальна по своей природе игровых механик и заработка, поэтому многих это пугает и отталкивает;
2. Отсутствие платформы с доверенными Web3[4,6] играми, при выходе нового игрового проекта приходится осуществлять тщательную проверку игры на прозрачность и справедливость внутренних процессов;
3. Существующие решения такие как TON Play и Intella X, которые на основе блокчейнов TON и Polygon соответственно запустили свои платформы для Web3[4,6] игр, объединяя игры внутри платформы единой валютой, но не предоставляющие набор игровых персонажей.

Для решения выделенных проблем были поставлены следующие цели и задачи.

Цель - разработать платформу на основе блокчейн-технологий[2,3,5], которая будет выступать витриной с доверенными играми и предоставлять единую концепцию и набор

игровых персонажей, а также единую внутреннюю валюту и программный интерфейс для создания собственной игры.

Задачи:

1. Необходимо разработать программные интерфейсы для создателей игр и для самой платформы
2. Необходимо разработать пользовательские интерфейсы
3. Модули для взаимодействия с технологией блокчейн[2] и базой данных
4. Развернуть и настроить проект в Облаке.

Общая архитектура проекта выглядит следующим образом:

1. Бэкенд на языке программирования Python и JavaScript
2. Фронтенд самой платформы на JavaScript и Next.js
3. К бэкенду подключаются несколько игр
4. Взаимодействие с блокчейн технологией[5,6]
5. Блокчейн – Ethereum[1,2] и используется язык программирования Solidity[7] для смарт-контрактов
6. Объектное хранилище блокчейн-ориентированных данных Pinata.cloud[8]
7. СУБД PostgreSQL

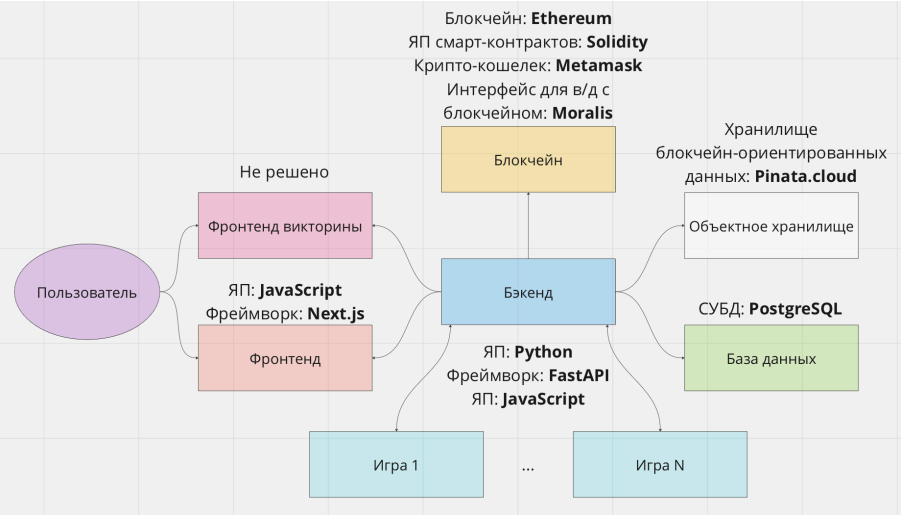


Рисунок 1 – Общая архитектура проекта

Пользовательской сценарий заключается в:

1. Пользователь покупает внутреннюю валюту[6]
2. За внутреннюю валюту покупает персонажей
3. Пользователь играет и зарабатывает (или теряет) внутреннюю валюту
4. Может покупать ещё больше персонажей, чтобы играть в разные игры разными персонажами
5. Персонажи в процессе игры улучшаются: повышается опыт, повышается уровень персонажей, также персонажи имеют разную редкость и способности
6. Пользователь может продать внутреннюю валюту и персонажей
7. Также в будущем пользователь сможет проходить викторину для получения внутр. валюты и персонажей и сможет открывать контейнеры с персонажами.

ЛИТЕРАТУРА

1. Ethereum development documentation. [Электронный ресурс] Режим доступа: https://ethereum.org/en/developers/docs/
2. Блокчейн. [Электронный ресурс] Режим доступа: https://ru.wikipedia.org/wiki/Блокчейн
3. NFT. [Электронный ресурс] Режим доступа: https://ru.wikipedia.org/wiki/NFT
4. Web3.0. [Электронный ресурс] Режим доступа: https://ru.wikipedia.org/wiki/Web_3.0
5. Смарт-контракт. [Электронный ресурс] Режим доступа: https://ru.wikipedia.org/wiki/Смарт-контракт

6. Что такое Web3. [Электронный ресурс] Режим доступа: https://habr.com/ru/post/651077/
7. Solidity programming language. [Электронный ресурс] Режим доступа: https://ru.wikipedia.org/wiki/Solidity
8. Pinata cloud. [Электронный ресурс] Режим доступа: https://www.pinata.cloud

УДК 004.415

Э. О. Стенина(4 курс бакалавриата),
Е. С. Орлов, ст. преподаватель

РАЗРАБОТКА СИСТЕМЫ ЭЛЕКТРОННОГО ГОЛОСОВАНИЯ НА ОСНОВЕ ТЕХНОЛОГИИ BLOCKCHAIN

Информационные технологии со временем все теснее переплетаются с человеческой жизнью. Переход из реального мира в цифровое пространство обычно влечет за собой огромное количество положительных моментов.

Голосование – это основа любой успешной демократии и, поэтому, оно должно быть доступным и безопасным для всех людей. Сегодняшние наиболее распространенные бумажные системы голосования доступные и дешевые, но имеют две основных проблемы. Согласно многим экспертам, такие системы не масштабируемы (поэтому, это приводит к таким основным проблемам, как точность), и они подразумевают “уверенность в процедурной безопасности организаторов, проводящих их правильно и честно”.

Новизна исследования состоит в том, что многие сложные вопросы безопасности, с которыми сегодня сталкиваются электронные системы голосования, могут быть преодолены, если в их разработке применить механизмы репликации, криптографии и верификации, которые использует технология blockchain. Применение данной технологии в области голосований должно положительно сказаться на безопасности и прозрачности таких систем, а, следовательно, и на доверии пользователей к ним.

Цель работы: разработать систему электронного голосования на основе технологии blockchain. Для достижения цели необходимо решить следующие задачи:

1. Проанализировать проблемы существующих решений электронного голосования и изучить теоретическую часть, касающуюся технологии blockchain.
2. Исследовать существующие платформы blockchain и средства разработки децентрализованных приложения.

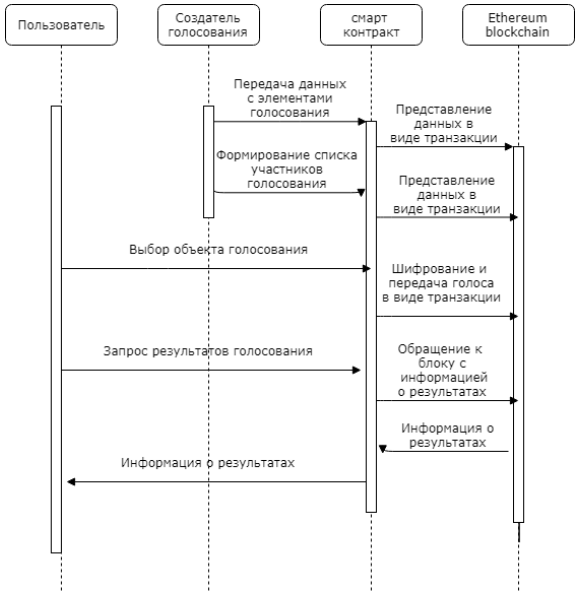


Рисунок 1 – Диаграмма последовательности работы системы электронного голосования

Логическое моделирование представляет собой осуществление проверки функционирования логической схемы.

Основная цель заключается в осуществлении проведения проверки функций проектируемого приложения без полной реализации на данном этапе разработки. Преимущества данной модели заключается в том, что осуществляется проверка, как логических функций разрабатываемого приложения, так и её временные соотношения.

Взаимодействие между объектами и субъектами происходит следующим образом:

1. К разработанному смарт-контракту обращается создатель нового голосования и создает новый опрос.
2. Такие данные, как «Название голосования», «Обсуждаемый вопрос», «Варианты голосования» формируются в блок и отправляются в сеть blockchain.
3. Создатель голосования регистрирует избирателей в систему, путём добавления адресов их аккаунтов в список избирателей, что даёт им право на голосование в этом опросе.
4. Сформированный список избирателей отправляется в blockchain.
5. Пользователь обращается к смарт-контракту и выбирает в опросе объект голосования.
6. Имя объекта голосования, за которого был отдан голос, шифруется и отправляется в blockchain.
7. Пользователь обращается к смарт контракту, для того чтобы узнать результаты голосования.
8. Смарт-контракт обращается к нужному блоку с информацией в блокчейне.
9. Информация о результатах голосования выводится пользователю.

ЛИТЕРАТУРА

1. Прасти Н. Блокчейн. Разработка приложений, // Н. Прасти, В.С., Яценков. — СПб.: БХВ-Петербург, 2018. – 256 с.
2. Равал С. Децентрализованные приложения. Технология Blockchain в действии, // С.Равал. — СПб.: Питер, - 2017. - 192 с.
3. Насколько надежно электронное голосование [Электронный ресурс]. – Режим доступа: https://www.svoboda.org/a/269300.html, свободный.
4. Block The Vote: Could Blockchain Technology Cybersecure Elections? [Электронный ресурс]. – Режим доступа: http://www.forbes.com/sites/realspin/2016/08/30/block-the-votecouldblockchain-technology-cybersecure-elections, свободный.
5. Russian Hackers Acted to Aid Trump in Election, U.S. Says [Электронный ресурс]. – Режим доступа: https://www.nytimes.com/2016/12/09/us/obama-russia-election-hack.html, свободный.
6. Slim. Middeware-slim. [Электронный ресурс]. – Режим доступа: https://www.slimframework.com/docs/v3/concepts/middleware.html, свободный.
7. Тапскотт Д., Тапскотт А. Технология блокчейн - то, что движет финансовой революцией сегодня, // Д. Тапскотт, А. Тапскотт. — М.: Эксмо, 2017. – 448 с.

УДК 004.418

А. А. Тесленко (4 курс бакалавриата),
Н. В. Воинов, к.т.н., доцент

ПОДХОД К ПРОГНОЗИРОВАНИЮ МОЩНОСТИ СЭС

В последние годы активно ведётся разработка систем краткосрочного прогнозирования солнечной инсоляции и генерации солнечных электростанций. Уже существуют готовые решения, например, система от компании ComAp. Однако данная система является коммерческой разработкой иностранной компании, её программное и аппаратное обеспечение закрыты. Известно, что существующие системы испытывают проблемы, связанные с точностью прогноза, в следствие чего исследования на эту тему продолжаются и сегодня.

В работе предлагается подход к прогнозированию, состоящий из нескольких этапов: сегментации облачности, вычисления векторов движения облаков и прогнозирования карты освещенности.

Для сегментации облачности предлагается подход адаптивной коррекции в околосолнечной области. Этот подход не требует составления базы данных референсных изображений чистого неба. Подход позволяет также избежать снижения точности при несовпадении позиции солнца исходного и референсного изображения. Маску солнечного пятна предлагается определять, оценивая область самых ярких пикселей в зеленом канале изображения и в градации серого. Для определения таких областей выполняется сегментация изображений. Порог сегментации вычисляется методом Оцу. Среди полученных контуров выбирается оптимальный исходя из двух критериев: минимальное расстояние от расчетного положения солнечного пятна и максимальная площадь контура. Далее вычисляется композиция контура, полученного в зеленом канале и в градации серого. Данный метод на тестовых изображениях с выделенным вручную контуром солнца показал точность распознавания 92%.

Маска околосолнечной области представляет из себя круговую область с центром, совпадающим с солнечным пятном и линейно уменьшающейся интенсивностью от центра маски к краю. Радиус маски и её максимальная интенсивность являются адаптивными параметрами. Радиус вычисляется на основе производной разности дисперсий представления и размытия по Гауссу. Аргументом такой функции является радиус рассматриваемой круговой области. Оптимальным считается радиус, при котором производная равняется нулю. Такой подход определения радиуса основывается на наблюдаемой размытости преставления в околосолнечной области. Максимальный цвет маски определяется таким, чтобы в области радиусом 10% от края солнечного пятна после применения маски остался определенный процент облачных пикселей. Процент этих пикселей определяется исходя из анализа формы солнечного пятна. А именно: периметра, площади, их соотношения и разницы определённого солнечного пятна в зеленом канале и в градации серого.

Подход, выбранный для построения системы прогнозирования, предполагает, что уровень солнечного излучения изменяется вследствие перекрытия тенью облаков точки, в которой делается прогноз. Следовательно, для того чтобы прогнозировать изменение излучения необходимо прогнозировать движение облаков, а далее преобразовывать эту информацию в прогноз изменения карты теней. Для прогнозирования движения облаков строится вектор движения - CMV. Для упрощения построения прогноза обычно вычисляют глобальный CMV, предполагая, что все облака двигаются в одном направлении с одинаковой скоростью. Основываясь на результатах проведенного сравнения существующих методов построения CMV, для применения в разрабатываемой системе был выбран метод оптического потока [1,2]. Во-первых, потому что реализация данного метода доступна в широко распространённой библиотеке компьютерного зрения OpenCV API [3], которая доступна во многих языках программирования. Это облегчает портирование алгоритма на другие языки при необходимости. Во-вторых, данный метод может работать с изображением в градации серого, это исключает возможность ошибки определения вектора движения, связанной с ошибками сегментации облачности, которые могут по-разному проявляется на паре анализируемых изображений. Необходимость анализа изображений с меньшим временным интервалом, чем у других методов, на данном этапе не представляется критическим недостатком. Так как метод оптического потока предполагает постоянство яркости движущихся объектов, а у облаков, попадающих в околосолнечную область наблюдается повышение яркости, необходимо маскировать околосолнечную область. Здесь в частности будет полезно применение маски солнечного пятна.

В рассматриваемом подходе прогноз карты освещенности строится исходя из прогноза карты теней облаков. Важным здесь является проецирование сегментированного изображения облаков в трехмерную декартову систему координат реального мира с центром, совпадающим с положением камеры. Также необходимо реализовать обратное проецирование из системы

координат реального мира в двухмерную систему координат пикселей изображения, например, для отображения расчетного положения солнца на изображение полного неба. Для получения проекции изображения на горизонтальную плоскость может применятся прямое преобразование. В трехмерной декартовой системе координат строится плоская сетка, на которую необходимо спроецировать изображение. Для каждой точки этой сетки вычисляется соответствующие координаты пикселя изображения через преобразование. После этого изображение проекции может быть получено через переназначение координат пикселей исходного изображения согласно координатам, полученным для сетки. Подобное переназначение в частности можно выполнить при помощи метода remap библиотеки OpenCV.

Карта освещённости является финальным предоставлением прогнозируемого солнечного излучения в рассматриваемом подходе. Она вычисляется преобразованием бинарной карты теней. Для этого применяется метод гистограмм.

Таким образом, в работе описан подход к прогнозированию мощности СЭС, основанный на методе сегментации облачности. Данный метод способен с точностью 90% распознавать солнечное пятно на изображениях. Также рассмотрены основные этапы формирования прогноза мощности СЭС и описана применимость данной особенности на других этапах построения прогноза: определение CMV, построение карты теней.

ЛИТЕРАТУРА

1. Richardson W. et al. A low cost, edge computing, all-sky imager for cloud tracking and intra-hour irradiance forecasting //Sustainability. – 2017. – Т. 9. – №. 4. – С. 482.
2. Schmidt T. et al. Evaluating the spatio-temporal performance of sky-imager-based solar irradiance analysis and forecasts //Atmospheric chemistry and physics. – 2016. – Т. 16. – №. 5. – С. 3399-3412.
3. OpenCV function cv::remaph. // [Электронный ресурс]. URL: ttps://docs.opencv.org/3.4/d1/da0/tutorial_remap.html. (Дата обращения: 17.03.2023).

УДК 004.4

Э.Р. Топузов (аспирант),
Н.Б. Ампилова, к.ф.-м.н., доцент,
И.П. Соловьев, к.ф.-м.н., доцент

РАЗРАБОТКА СИСТЕМЫ ДЛЯ ОБРАБОТКИ ДАННЫХ МЕДИЦИНСКИХ ИССЛЕДОВАНИЙ

Объемы данных медицинских исследований увеличиваются год от года. В настоящее время для их обработки применяются медицинские информационные системы (МИС) [1, 2, 4]. Они, как правило, сосредоточены на узком круге клинических задач [2, 5].

Обзор исследований в этой области показывает, что обычно разработка МИС ограничивается созданием электронных медицинских карт или систем, предполагающих диагноз на основе клинических или лабораторных данных, однако к настоящему моменту не известны системы, которые можно было бы использовать в рамках исследований по государственным заданиям, диссертационным работам, а также для инициативных исследований [3, 5]. Каждая такая медицинская научно-исследовательская работа предусматривает создание индивидуального плана исследования, что включает в себя определение перечня необходимых показателей, группировку пациентов, подбор статистических методов обработки, что невозможно осуществить с использованием доступных МИС.

Врачи-исследователи вынуждены вручную формировать множество таблиц Excel, копируя данные между ними.

Для упрощения процессов нами была разработана веб-ориентированная система, которую можно использовать на различных этапах любых медицинских исследований: от

ввода данных о пациентах до автоматического анализа (включая статистический) показателей и предоставления прогностических данных.

Архитектура системы показана на Рисунке 1 и представляет из себя независимые контейнеры системы управления баз данных PostgreSQL [8] и веб-приложения на основе Ruby on Rails [9]. В качестве веб-сервера используется Puma [6], в качестве обратного прокси-сервера – Nginx [7].

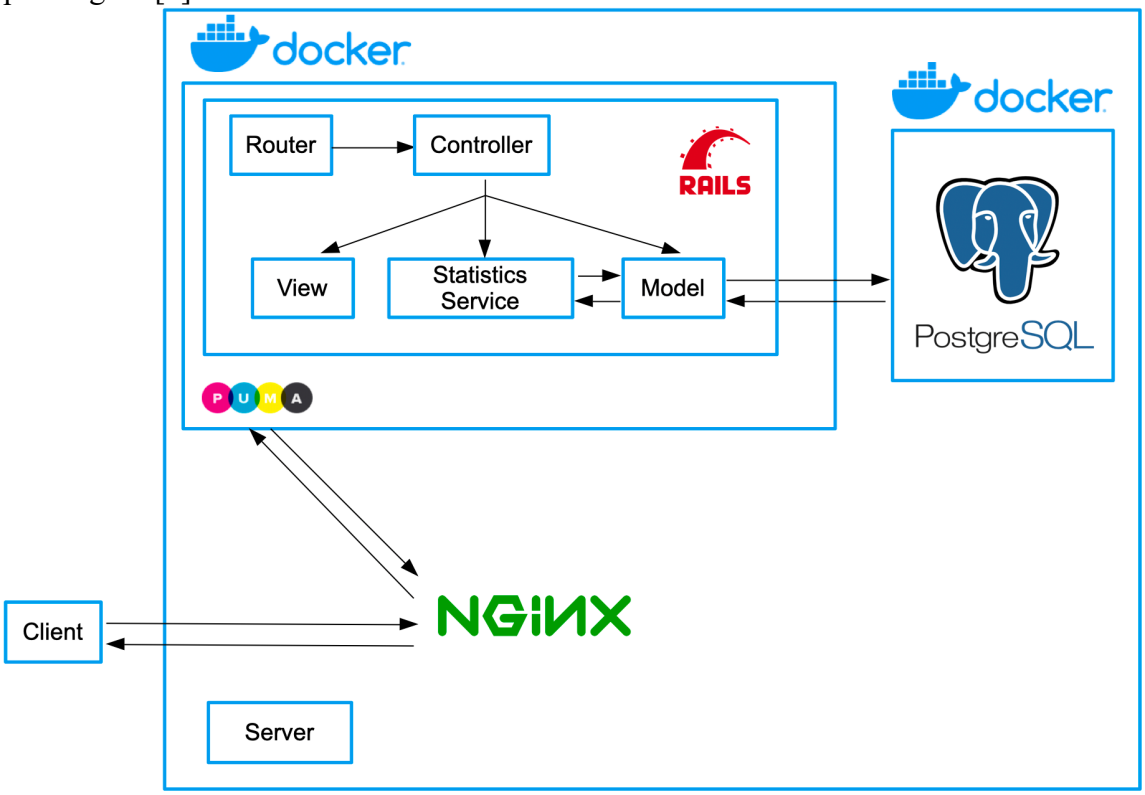


Рисунок 1 – Архитектура системы

В системе реализован функционал, обеспечивающий:
– создание дизайна медицинского исследования;
– внесение данных;
– предоставление разных уровней доступа к системе;
– контроль со стороны главного исследователя изменений в базе данных;
– регулярное резервное копирование базы данных (БД);
– проведение анализа содержимого БД;
– выгрузку внесенных данных в таблицы Excel с проведением статистического анализа.

Система предусматривает дальнейшее развитие: предсказание выгружаемых отчетов, построение и экспорт диаграмм и графиков, расширение списка методов статистической обработки, необходимых для конкретных исследований. Также предполагается расширение функциональных возможностей для поддержки научных исследований. Все это в конечном итоге позволит полностью отказаться от использования дополнительных программных комплексов для статистического анализа.

ЛИТЕРАТУРА

1. Гусев А.В. Перспективы применения больших данных в российском здравоохранении // Московская медицина. – 2022. – № 1(47). – С. 26-30.
2. Луценко Е.В. Развитие медицинский информационных технологий в Российской Федерации // Вятский медицинский вестник. – 2017. – No 2(54). – С.73-77.
3. Шаханов А.С., Ушакова Е.В. Использование современных информационных технологий в государственном управлении // Трансформация бизнеса и общественных институтов в условиях цифровизации экономики. – 2020. – С. 199-211.

4. Auffray C., Balling R., Barroso I., et al. Making sense of big data in health research: towards an EU action plan // Genome medicine. – 2016. – 8(1) – P. 1-13.
5. Magrupov T., Yusupov S., Talatov Y. et al. Intelligent Medical System of Designing Medical Technics and Technology // 2020 International Conference on Information Science and Communications Technologies (ICISCT). – IEEE, 2020. – P. 1-4.
6. A Fast, Concurrent Web Server for Ruby & Rack – Puma. [Электронный ресурс] Режим доступа: https://puma.io
7. nginx. [Электронный ресурс] Режим доступа: https://nginx.org
8. PostgreSQL: The world's most advanced open source database. [Электронный ресурс] Режим доступа: https://www.postgresql.org
9. Ruby on Rails — A web-app framework that includes everything needed to create database-backed web applications according to the Model-View-Controller (MVC) pattern. [Электронный ресурс] Режим доступа: https://rubyonrails.org

УДК 004.453

Ю. Д. Уфимцев (2 курс бакалавриата),
Ю. В. Литвинов, к.т.н., доцент

СОЗДАНИЕ ВЕБ-ПРИЛОЖЕНИЯ ДЛЯ ОБРАБОТКИ ДАННЫХ ПО ПРОХОЖДЕНИЮ ОНЛАЙН-КУРСОВ

Каждую сессию студенты СПбГУ проходят онлайн-курсы с прокторингом. Факультетам приходят отчеты с результатами, содержащие тысячи строк информации. В связи с чем от учебного отдела матмеха СПбГУ требуется:

– отобрать в отчете по баллам, полученным студентами за онлайн-курсы, обучающихся матмеха;

– перевести баллы в оценку ECTS (от A до F);

– свериться с таблицей из прокторинга;

– если студент успешно прошёл прокторинг и получил зачёт, проставить оценку в ведомость.

Это приходится делать для сотен студентов по нескольку раз за сессию, что, естественно, отнимает у сотрудников массу времени и сил, которые могли быть направлены на решение других, более важных задач. Чтобы уменьшить время, затраченное на анализ данных по каждому обучающемуся, прошедшему курс, необходимо было создать приложение, позволяющее сотрудникам учебного отдела загрузить информацию об онлайн-курсах и сразу получить список студентов матмеха с оценками, учитывающий непрошедших прокторинг.

Таким образом, целью данной работы является создание веб-приложения, которое будет принимать таблицу-отчёт по баллам, полученным студентами за онлайн-курс, таблицу-отчёт о прохождении прокторинга и в качестве результата в удобном виде выводить в веб-приложении таблицу, содержащую оценку за курс и статус прокторинга для каждого студента матмеха.

Для достижения этой цели необходимо решение следующих задач:

1. Спроектировать макет приложения
2. Реализовать основную функциональность.
3. Поддержать развёртывание посредством Docker-контейнера.
4. Провести апробацию по методике SUS (System Usability Scale) на представителях учебного отдела. Внедрить технологию в работу учебного отдела.

Платформой разработки веб-приложения была выбрана ASP.NET Core. В качестве модуля для реализации пользовательского веб-интерфейса использовался Blazor [1], а именно модель размещения Blazor WebAssembly, позволяющая создавать клиентское веб-приложение на C#. Для работы с таблицами была выбрана EPPlus [2], самая популярная библиотека

электронных таблиц для .NET Framework/Core по версии создателей, лишенная нескольких недостатков, обнаруженных в Open XML SDK от Microsoft.

Обзор был сфокусирован на изучении систем управления обучением (Learning Managment Systems), необходимом для проектирования интерфейса своего приложения, и обзоре аналогов на Blazor WebAssembly для поиска хороших технических решений. Были рассмотрены три лучшие LMS-системы согласно рейтингу 2022 года от eLearningIndustry [3], составленному на основе пользовательского опыта. В каждой системе были выделены элементы, схожие по функциональности с создаваемым веб-приложением, и подробно изучены. Для поиска хороших технических решений рассматривалось приложение, соответствующее официальной документации Microsoft Blazor [4].

Веб-приложение состоит из главной страницы Index, отвечающей за отрисовку компонент и взаимодействие с пользователем, и нескольких модулей, отвечающих за обработку данных. На Рис. 1 представлена диаграмма классов.

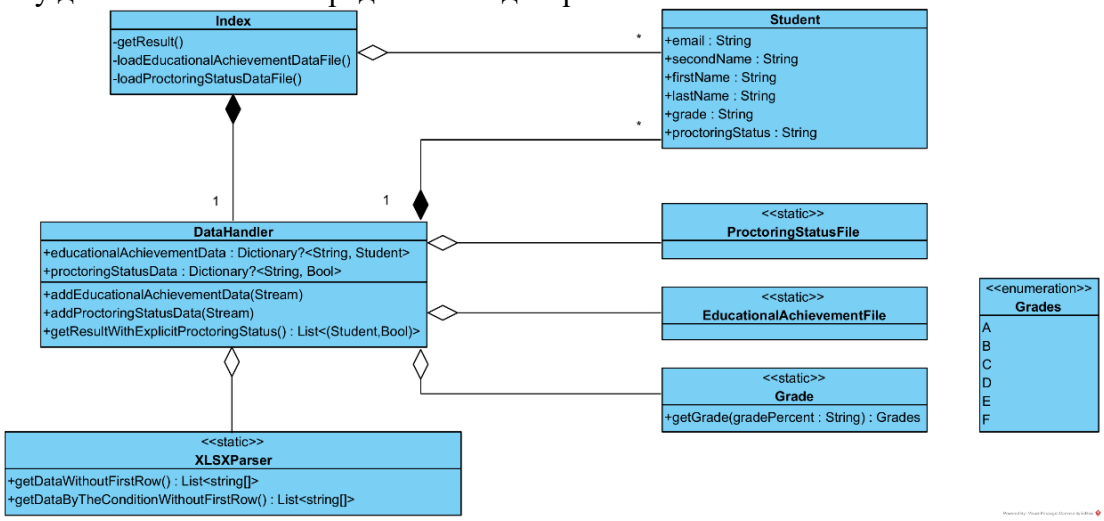


Рисунок 1 – Диаграмма классов веб-приложения

При реализации особое внимание было уделено выбору способа хранения файлов [5], механизму обработки ошибок и тестированию.

Было поддержано развертывание посредством Docker-контейнера [6]. В качестве сервера использовался nginx [7].

Была проведена апробация на представителях учебного отдела матмеха СПбГУ. Результат составил 97,5% по методике SUS, были получены положительные отзывы.

Технология активно применялась для выставления зачетов по онлайн-курсам на зимней сессии. Было проведено пострелизное сопровождение: оперативно устранялись выявленные недочеты, предоставлялись разъяснения особенностей работы.

Внешний вид приложения представлен на Рис. 2. Ссылка на репозиторий проекта: https://github.com/spbu-se/OnlineCoursesAnalyzer

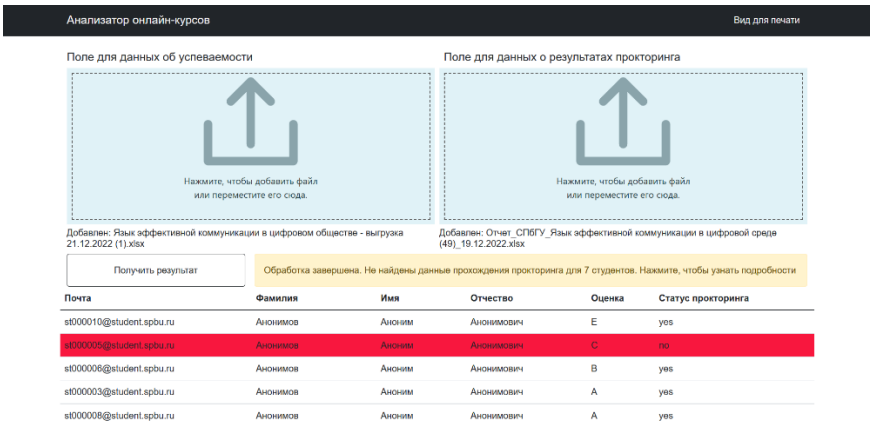


Рисунок 2 – Страница веб-приложения

ЛИТЕРАТУРА

1. Microsoft. Choose an ASP.NET Core web UI. [Электронный ресурс] Режим доступа: https://learn.microsoft.com/en-us/aspnet/core/tutorials/choose-web-ui?view=aspnetcore-6.0
2. EPPlus. [Электронный ресурс] Режим доступа: https://epplussoftware.com/
3. eLearning Industry. The Best Learning Management Systems based on User Experience. [Электронный ресурс] Режим доступа: https://elearningindustry.com/directory/software-categories/learning-management-systems/best/user-experience
4. Latham Luke. Blazor WebAssembly sample. [Электронный ресурс] Режим доступа: https://github.com/dotnet/blazor-samples/tree/main/6.0/BlazorSample_WebAssembly
5. Microsoft. ASP.NET Core Blazor file uploads. [Электронный ресурс] Режим доступа: https://learn.microsoft.com/en-us/aspnet/core/blazor/file-uploads?cid=kerryherger&view=aspnetcore-6.0&pivots=webassembly
6. Docker. [Электронный ресурс] Режим доступа: https://www.docker.com/
7. nginx. [Электронный ресурс] Режим доступа: https://nginx.org/en/

УДК 004.93

С. Э. Федорова (4 курс бакалавриата),
О. В. Прокофьев, ст. преподаватель

РЕАЛИЗАЦИЯ СИСТЕМЫ РАСПОЗНАВАНИЯ БОЛЕЗНЕЙ ПШЕНИЦЫ ЧЕРЕЗ ИНТЕРФЕЙС POSTGRESQL С ИСПОЛЬЗОВАНИЕМ ТЕКСТУРНОГО АНАЛИЗА ИЗОБРАЖЕНИЙ

Болезни культурных растений приводят к серьезным производственным и экономическим потерям в сельском хозяйстве, поскольку площадь поражения быстро разрастается. Одним из актуальных подходов по снижению затрат в этой области сельского хозяйства является мониторинг посевов, который помогает на ранней стадии идентифицировать болезнь и в дальнейшем принять меры к его нераспространению. Высокую эффективность в этой области показали алгоритмы диагностики заболеваний на основе анализа цифровых изображений. Заболевания пшеницы часто вызывают визуальные симптомы на поверхности листьев, такие как пятна, полосы и изменения цвета. Следовательно, становится актуальной задача применения технологий машинного обучения для разработки модели распознавания заболеваний пшеницы по изображениям листьев с помощью методов машинного обучения.

Целью данной работы является разработка программного продукта, позволяющего распознавать заболевания пшеницы на основе методов машинного обучения по текстурным признакам Харалика, вычисляемых на основе матрицы смежности для нормализованных изображений листьев. Предполагается, что пользователь будет иметь возможность взаимодействовать с обученной моделью через интерфейс СУБД *PostgreSQL*.

В работе выполнена реализация системы распознавания двумя способами: на основе нейронной сети прямого распространения с использованием текстурного анализа, а также, для сравнения, на основе сверточной нейронной сети. В обоих методах предварительно применяется нормализация изображения.

Наиболее часто для задач классификации и распознавания объектов на изображении используют сверточные нейронные сети. Основная идея сверточной нейронной сети заключается в чередовании операций свертки и субдискретизации [1]. Одно из преимуществ сверточных нейронных сетей является следующее: благодаря тому, что признаки изображения извлекаются обособленно, эти сети способны находить инварианты в изображении и реагировать главным образом на них, не обращая внимания на прочий шум [2]. Однако у такого подхода есть и недостаток: сверточная нейронная сеть обрабатывает изображение шаг за шагом, поэтому для обеспечения высокой точности необходимо иметь достаточно большой

набор данных. Для снижения негативного влияния малого размера датасета на результат эксперимента применяются техники аугментации данных [3].

Второй метод реализации, на основе сети прямого распространения отличается тем, что в нем реализуется более простая форма нейронной сети, поскольку информация обрабатывается только в одном направлении. Однако сложность такого подхода заключается в том, что для построения и обучения такой нейросети необходимо сформировать набор значимых признаков, позволяющих однозначно идентифицировать изображения.

Для изображений с явно выраженными текстурными характеристиками в работе [4] авторы определили, что текстурные признаки содержат в себе информацию о пространственном распределении тональных вариаций. Концепция признака тона в изображении основана на различных оттенках серого, в то время как текстура связана с пространственным (статистическим) распределением серых тонов. Харалик и другие авторы статьи [4] предложили метод извлечения текстурных свойств в изображении, основанный на вычислении матрицы распределения вероятностей пространственной зависимости оттенков серого для конкретного блока изображения.

Для получения статистических признаков изображение подвергается первичной обработке, во время которой строится матрица GLCM (Gray-Level Co-occurrence Matrix) – полутоновая матрица смежности, представляющая собой гистограмму значений градаций серого с заданным смещением по изображению [5]. Основываясь на нормализованной GLCM матрице, можно посчитать вторичные характеристики текстуры [6], выбранные для данной работы: контраст, корреляция, энергия и однородность.

Предварительное обучение нейронной сети производилось на языке *Python* в среде *Google Colaboratory* с использованием библиотек *Keras*, *Tensorflow* и *Scikit-image*. Для вычисления GLCM-матрицы использована функция *graycomatrix*, для получения свойств из полученной матрицы – *graycoprops* [7]. Архитектура нейронной сети простая – последовательное применение слоев Dense для получения информации со всех узлов предыдущего слоя с функцией активации ReLU и слой Dropout для решения проблемы переобучения.

В поставленной задаче взаимодействие пользователя с обученной моделью предполагается через интерфейс СУБД *PostgreSQL*. Таким образом, обработка входных и выходных параметров искусственной нейронной сети предполагает интеграцию нейросети и СУБД.

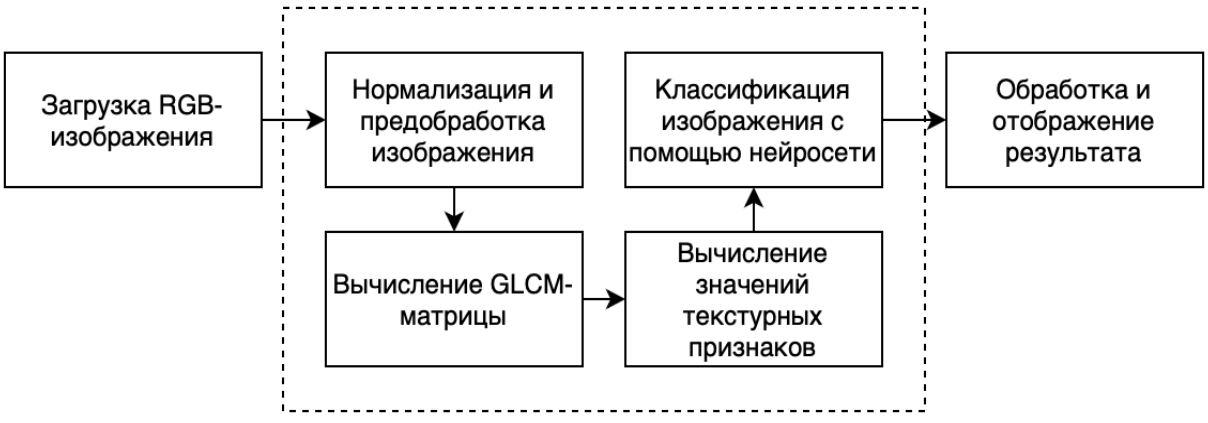


Рисунок 1 – Обобщенная схема распознавания болезни пшеницы

ЛИТЕРАТУРА

1. Шолле, Ф. Глубокое обучение на Python. / Ф. Шолле — СПб.: Питер, 2018. — 400 с.
2. Лагунов Н.А. Применение свёрточных нейронных сетей в задачах распознавания многопараметрических объектов // Пространство и Время, 2013, No3 (13), с.194-197.

3. Афанасьев Д. Ю. Применение аугментации для улучшения качества классификации // StudNet – Москва: Изд-во Электронная наука, 2022 — Т. 5., No 4.
4. Haralick R.M., Shanmugam K., Dinstein I. Textural features for image classification // IEEE Transactions on systems, man and cybernetics. 1973. Vol. SMC 3. No 6. P. 610-621.
5. Лайком Д. Н. Использование алгоритма GLCM для проведения классификации изображений / Д. Н. Лайком, С. В. Аксёнов // Молодежь и современные информационные технологии : сборник трудов XII Всероссийской научно-практической конференции студентов, аспирантов и молодых ученых, г. Томск, 12-14 ноября 2014 г. : в 2 т. — Томск : Изд-во ТПУ, 2014. — Т. 2. — [С. 166-167].
6. Чехина Е. А. Обзор методов текстурного анализа изображений // Евразийское научное объединение, 2020, No 6-2 (14)
7. Scikit-image.org. [Электронный ресурс] Режим доступа: https://scikit-image.org/docs/stable/index.html

УДК 004.453

Д. А. Филимонова (4 курс бакалавриата),
О. В. Прокофьев., ст. преподаватель

АВТОМАТИЗАЦИЯ РАБОТЫ ТЕПЛИЦЫ С ИСПОЛЬЗОВАНИЕМ ТЕХНОЛОГИИ INTERNET OF THINGS (КЛИЕНТСКАЯ ЧАСТЬ)

Теплицы широко распространены в сельском хозяйстве. Это среда, позволяющая воссоздать искусственный микроклимат в зависимости от определенных культур, в том числе тропических. Благодаря теплицам появилась возможность выращивать любые типы растений, которые изначально не предназначались для холодного климата. Также преимуществом теплицы является возможность выращивать культуры в течение всего года, что исключает необходимость запасать продукты на зимнее время. Кроме того, конструкция теплицы защищает почву от вредителей, что уменьшает потребность в обработке химическими средствами.

С помощью автоматизации теплицы можно добиться упрощения человеческого ручного труда. Такая теплица сможет работать автономно, сообщая человеку о текущих характеристиках и требуя минимальных вмешательств, например, при поломке датчиков.

Цель работы состоит в разработке клиентской части веб-приложения для ручного и автоматического удаленного управления теплицей с использованием технологий интернета вещей (Internet of Things — это система взаимосвязанных устройств, которые могут собирать и передавать данные по беспроводной сети без участия человека [1]). Необходимо реализовать удобный, интуитивно понятный интерфейс, который будет корректно взаимодействовать с серверной частью веб-приложения и предоставлять пользователю инструменты контроля теплицей.

Для достижения этой цели необходимо решение следующих *задач*:

1. Изучение способов подключения к датчикам сбора информации и системе контроля.
2. Изучение способов контроля параметров (температура, влажность, освещенность) теплицы для поддержания комфортных условий.
3. Определение алгоритма реагирования устройств контроля на условия.
4. Реализация клиентской части веб-приложения.

В системе задействованы датчики температуры, давления, освещенности и влажности для сбора информации о текущем состоянии, а также используются исполнительные устройства (системы обогрева, полива, проветривания и вентиляции), с помощью которых осуществляется контроль на основе полученных значений. Система поддерживает

определенные комфортные условия для растений, показатели не должны выходить за пределы допустимых значений, которые задаются при настройке.

На данный момент известны несколько решений, позволяющих автоматизировать контроль за условиями теплицы. Они отличаются ценой, сроком эксплуатации, возможностями контроля и наблюдения за состоянием теплицы. Наиболее популярные инструменты при работе с системами «Умного дома» стали инструменты NodeRED [2] и Home Assistant [3], которые часто работают вместе, компенсируя слабые стороны друг друга. В данной работе было принято решение разработать веб-приложение без использования готовых инструментов, так как они не предполагают поддержку для имеющегося аппаратного обеспечения.

Архитектура веб-приложения монолитная, представляющая собой backend и frontend в одном проекте. Такая архитектура удобна для работы небольших групп. Из плюсов монолитной архитектуры можно назвать упрощенную разработку и развертывание, а также лучшую производительность.

Хранение данных осуществляется с помощью СУБД PostgreSQL 11. Данная система является одной из лидирующих среди направления систем управления базами данных. Она предоставляет поддержку JSON, а также возможность удобного хранения различных типов данных, в частности MAC-адресов датчиков, что является преимуществом в данном проекте.

Связь с датчиками будет осуществляться при помощи MQTT протокола с использованием инструмента Eclipse Mosquitto. Eclipse Mosquitto [4] – это брокер сообщений с открытым исходным кодом, реализующий протокол MQTT.

Принятия решений на основе полученных данных из базы происходит при помощи Node.js. Это необходимо, для читаемости кода, так как клиентская часть подразумевает использование JavaScript (JS).

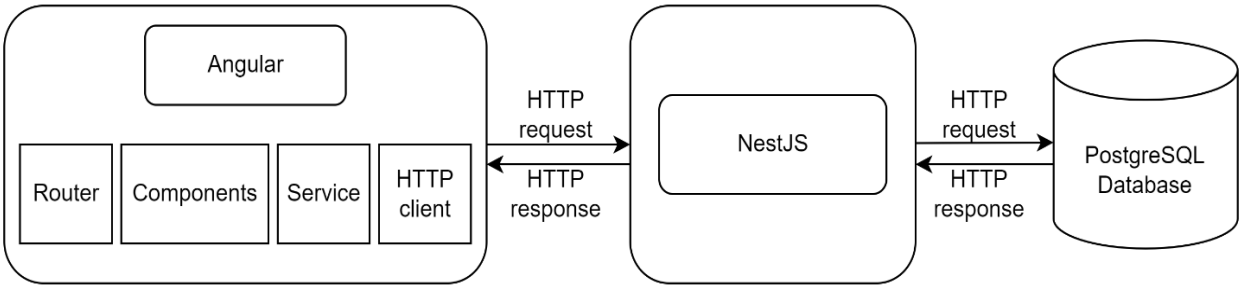


Рисунок 1 – Архитектура системы

Клиентская часть написана на JS с использованием фреймворка Angular. Angular [5] – это фреймворк от компании Google для разработки качественных клиентских веб-приложений. Приложение состоит из компонентов и модулей, каждый из которых получает информацию через http-клиента, что позволяет внедрять зависимости и лучше понимать структуру веб-приложения. Также при разработке использован язык разметки HTML и язык таблиц стилей CSS.

ЛИТЕРАТУРА

1. Что такое интернет вещей? Определение и описание. [Электронный ресурс] Режим доступа: https://www.kaspersky.ru/resource-center/definitions/what-is-iot
2. Node-RED. Low-code programming for event-driven applications. [Электронный ресурс] Режим доступа: https://nodered.org
3. Home Assistant. [Электронный ресурс] Режим доступа: https://www.home-assistant.io/
4. Eclipse Mosquitto. [Электронный ресурс] Режим доступа: https://mosquitto.org/
5. Angular. [Электронный ресурс] Режим доступа https://angular.io/

УДК 004.042

А. В. Филиппов (4 курс бакалавриата),
П. Д. Дробинцев., к.т.н., доцент

ОБРАБОТКА И АНАЛИЗ КАЧЕСТВА ДАННЫХ В АЛГОРИТМИЧЕСКОЙ ТОРГОВЛЕ КРИПТОВАЛЮТАМИ

На сегодняшний день большую роль в принятии решений играют данные, в том числе и в сфере алгоритмического трейдинга, в частности – криптовалюты. Как следует из это задача делится на две части: обработку и анализ. Под обработкой понимается следующее: первое – сбор данных [6]. То есть сбор исторических и реал-тайм данных о криптовалюте и иных показателях, как курс валют, индексы настроения и некоторая «выжимка» из новостей, представленная в численном формате (например, число упоминаний конкретного актива в новостях за прошедший день.) Второе – предобработка данных. Прежде чем заниматься построением моделей на данных, необходимо привести данные к «в порядок». На этом этапе происходит, например, поиск пробелов в данных (с последующим их заполнением из других источников), а также поиск данных, которые явно «выбиваются» из общего датасета. Такими примерами могут служить поиск и замена (или выкидывание) NaN данных. Но при этом нужно отдавать себе отчет, для чего именно будет использоваться полученный набор данных. Для некоторых целей факт наличия NaN’ов будет не менее полезным, чем наличие нормальных данных.

Вторая часть – анализ. Под анализом понимается извлечение признаков, нахождение паттернов и тенденций, которые помогут в прогнозировании поведения криптовалюты. Примером может служить анализ данных, полученных в промежуток времени, равный 10 минутам: на основании их анализа и поиска паттернов мы получаем представление о том, что ожидать в следующие 10 минут (с той или иной долей вероятности).

Обработка данных, не только для алгоритмической торговли, является актуальной темой из-за ряда причин. Во-первых, в современном мире все больше компаний стремятся быть data-driven, то есть принимать решения, основываясь на данных. При этом примерно две трети [1] данных, доступных компаниям, остаются необработанными в меру слишком большого объема этих данных и отсутствия ресурсов на их обработку, что является большой проблемой, которую решают компании привлечением аналитиков и специалистов по обработке данных. Во-вторых, криптовалюта в современном мире приобретает всю большую популярность за счет глобальной доступности, децентрализации и конфиденциальности. И в-третьих, делая вывод из вышесказанного, все больше компаний стремятся получать данные и по криптовалютам, так как основываясь на них у них появляется возможность делать предсказания их поведения, что используется как в алгоритмической торговле, так и для принятия решений на основе полученных результатов.

На сегодняшнем рынке обработки данных не существует единого решения. Разным компаниям нужны разные данные (в зависимости от их целей), в разном формате, вследствие чего возникают различные поставщики данных. У каждого из них составлены уникальные датасеты с уникальным набором данных, требуемых для компаний. Компании выбирают тех, кто, по их мнению, имеет наиболее удобные решения для них самих.

Существует множество концептов, связанных с получением и обработкой данных. Один из основополагающих – понятие ETL [2], которое расшифровывается как extract, transform, load. Это понятие позволяет установить обобщенный подход к обработке данных: они извлекаются, обрабатываются и загружаются в хранилище [3]. Так происходит со всеми данными, которые так или иначе используются для последующего анализа.

Есть несколько типов хранилищ: databases, datalakes, data warehouses [7].

Datalakes – самые вместительные хранилища, которые содержат неупорядоченную информацию. Как правило, для их реализации используется NoSQL [5] (где No расшифровывается как not only). Такая технология является менее гибкой, чем традиционно

известный SQL, но значительно более быстрой при работе с большими объемами данных. Data warehouses – меньше по вместительности, чем datalakes. Их суть в том, что в них находятся предобработанные данные, полученные с помощью вышеописанного алгоритма ETL, взятые из datalakes. Как правило, они тоже неупорядоченные, но «чище», чем в первом типе хранилища. Databases – базы данных, по вместительности наименьшие, но являющиеся наиболее удобными для анализа, так как данные строго упорядочены. Именно они, в конце концов, отдаются аналитикам данных, специалистам по машинному обучению и прочим.

Перетекание данных из различных хранилищ реализуется с помощью пайплайнов – автоматизированных алгоритмов, реализующие ETL.

Изначальное попадание данных в datalakes может быть реализовано множеством способов: web scraping (получение информации путем парсинга веб страниц), путем получения данных от других сервисов (api, специализирующиеся на генерации данных) и прочие.

Языком программирования для обработки данных традиционно является Python [4]. Это высокоуровневый язык, легко читаемый, который прост в использовании и имеет богатую библиотеку (в том числе и для обработки данных). Pandas, NumPy, Matplotlib – лишь некоторые из примеров. А также очень много библиотек для поддержки больших данных. Основной является Spark, но также есть Apache Beam, Kafka, Hadoop и многие другие.

Для хранения данных выбран Postgres, который является наиболее популярной базой данных на текущий момент, поддерживающей все необходимые функции для работы с данными.

ЛИТЕРАТУРА

1. Two thirds of data available to firms goes unused // URL: https://www.frontier-enterprise.com/two-thirds-of-data-available-to-firms-goes-unused/
2. Основные функции ETL-систем // URL: https://habr.com/ru/post/248231/
3. Akidau T., Chernyak S., Lax R. Streaming systems [Электронный ресурс] O'Reilly Media, inc // URL: https://learning.oreilly.com/library/view/streaming-systems/9781491983867/
4 Python documentation // URL: https://www.python.org/doc/
5. Что такое базы данных NoSQL? // URL: https://aws.amazon.com/ru/nosql/
6. Тирни Б., Келлехер Д. Наука о данных. Альпина Паблишер, 2020 175 с // URL: https://mybook.ru/author/brendan-tirni/nauka-o-dannyh/
7. What is a data warehouse? // URL: https://www.oracle.com/ie/database/what-is-a-data-warehouse/
8. What is a data lake? // URL: https://aws.amazon.com/big-data/datalakes-and-analytics/what-is-a-data-lake/

УДК 004.032.26

А. С. Фокин (4 курс бакалавриата),
О. В. Прокофьев, ст. преподаватель

РАЗРАБОТКА ПРИЛОЖЕНИЯ ДЛЯ РАСПОЗНАВАНИЯ БОЛЕЗНЕЙ РАСТЕНИЙ

Вопрос распознавания болезней растений актуален столько, сколько существует сельское хозяйство [1]. Насекомые-вредители и различные заболевания растений [2] — это серьёзная проблема и неотъемлемая часть сельскохозяйственной отрасли. Долгое время за состоянием растений и выявлением болезней следили специально обученные агрономы.

С развитием технологий стало понятно, что следить за состоянием растений вручную неэффективно. Во-первых, при таком подходе присутствует человеческий фактор, поэтому не исключены ошибки в распознавании. Во-вторых, работа экспертов или целых специализированных лабораторий стоит дорого. Также оценка образцов, доставленных в удаленные от полей лаборатории, и вовсе может быть некорректной из-за потери первоначального вида растения в следствие перевозки. В конце концов, засеянные поля в

промышленных масштабах достигают огромных размеров, что также затрудняет классификацию вручную.

Самый эффективный способ автоматизации и упрощения распознавания болезней растений на данный момент – обучить нейронную сеть для классификации этих болезней на примерах уже распознанных болезней [3]. В последующем для очередной проверки растения на наличие болезни достаточно будет передать обученной нейросети изображение пораженной части растения. На основании данных, содержащихся в этой нейросети, она сможет сделать предсказание, с какой вероятностью у растения имеется та или иная болезнь.

Более того, в будущем подход с применением нейросетей для распознавания болезней растений можно расширить и обрабатывать уже не одно изображение за раз, а, используя, например, беспилотные дроны, в автоматическом режиме собирать множество изображений и в реальном времени отслеживать и выявлять болезни намного большего количества растений [4]. При таком подходе влияние человеческого фактора уменьшается, и пропадает необходимость оплаты специализированных экспертов и лабораторий. А главное – решается проблема с распознаванием болезней на больших территориях.

Таким образом, *целью* данной работы является упрощение процесса распознавания и своевременного выявления болезней растений за счёт применения искусственных нейронных сетей. Реализация интерфейса осуществлена при помощи веб технологий.

Для достижения этой цели необходимо решение следующих *задач*:

1. Предложение реализации нейронной сети, а также стека технологий для реализации веб-интерфейса.
2. Реализация данного решения.
3. Демонстрация решения поставленной задачи.

Архитектура решения представлена на Рис. 1. Серверная составляющая работает следующим образом: клиент отправляет изображение для классификации на сервер, nginx перенаправляет запрос на предсказание обратному прокси серверу, в качестве которого выступает Gunicorn. Последний, в свою очередь, направляет его Flask. Он получает картинку в качестве запроса, делает предсказание и возвращает его клиенту [5].

Рисунок 1 – Архитектура решения

Для написания и обучения нейросети используется библиотека TensorFlow в связке с Keras [6] на языке Python.

ЛИТЕРАТУРА

1. Society for General Microbiology. "Combating plant diseases is key for sustainable crops." ScienceDaily. ScienceDaily, 2011. http://www.sciencedaily.com/releases/2011/04/110411194819.htm; 2013.
2. Йорданка Станчева. Атлас болезней сельскохозяйственных культур. Т. 3., Болезни полевых культур. София — Москва, Изд. ПЕНСОФТ, 2003.
3. Jayme Garcia Arnal Barbedo Digital image processing techniques for detecting, quantifying and classifying plant diseases / Barbedo SpringerPlus 2013, 2:660 / http://www.springerplus.com/content/2/1/660/

4. Аунг Ч. Х., Тант З. П., Федоров А. Р., Федоров П. А. Разработка алгоритмов обработки изображений интеллектуальными мобильными роботами на основе нечёткой логики и нейронных сетей // Современные проблемы науки и образования. — 2014. — № 6.
5. Грэм Дамплтон, хостинг веб-приложений Python Sydney PyCon 2011.
6. François Chollet. Deep Learning with Python // Manning Publications Co – 2018. – P. 16–19.

УДК 519.254

П. А. Фомина (4 курс бакалавриата),
А. П. Маслаков, ст. преподаватель

РАЗРАБОТКА УСОВЕРШЕНСТВОВАННОГО МЕТОДА РАЗДЕЛЕНИЯ СМЕСИ СИГНАЛОВ, ОСНОВАННОГО НА ВРЕМЕНИ ПРИХОДА ИМПУЛЬСОВ, С ИСПОЛЬЗОВАНИЕМ АЛГОРИТМОВ SDIF И CDIF

С ростом использования радиооборудования в различных отраслях жизнедеятельности человека проектирование и эксплуатация средств радиотехнической разведки (РТР) стали важными направлениями в радиотехнике. РТР широко применяется для защиты летательных аппаратов, наземных или морских объектов, как гражданских, так и военных.

Одной из основных составляющих РТР являются методы обработки радиолокационных сигналов. Цель обработки — получение разведывательной информации о радиолокационных станциях (РЛС) [1]. Для получения информации о РЛС различных типов система РТР решает задачу селекции сигналов. Станция РТР отделяет последовательность импульсов каждого источника радиоизлучения (ИРИ) от потока импульсов, принятого приемником [2]. Дальнейшая обработка направлена на определение типа РЛС и местоположения ИРИ.

Для определения периода повторения импульсов (ППИ) и последующего разделения импульсных последовательностей используются различные алгоритмы. Некоторые из них основаны на анализе времени прихода импульсов и применении гистограмм накопительных и последовательных разностей — cumulative difference histogram (CDIF) и sequential difference histogram (SDIF) [3]. Эти методы успешно работают со сложными типами сигналов, однако на практике они недостаточно эффективны для разделения смесей сигналов, особенно когда в смеси присутствуют сигналы типов – jitter и stagger-jitter [4].

Таким образом, цель данной работы состоит в разработке усовершенствованного метода разделения смеси сигналов, основанного на времени прихода импульсов, с использованием алгоритмов SDIF и CDIF, который будет эффективен на различных смесях сигналов. Для достижения поставленной цели необходимо выполнить следующие задачи:

– Изучить различные типы сигналов и их особенности

– Изучить существующие алгоритмы SDIF и CDIF

– Проанализировать эффективность существующих алгоритмов на разных типах сигналов и для каждого подобрать наиболее эффективный алгоритм

– Разработать усовершенствованный метод разделения смеси сигналов

– Проанализировать результаты работы разработанного метода

Текущее ПО состоит из двух модулей – сепаратора и классификатора. Исходная импульсная последовательность разделяется на сегменты, таким образом их гораздо проще анализировать. Для каждого типа сигналов в сепараторе есть парные функции, первая определяет возможные ППИ, вторая извлекает импульсные последовательности с найденным ранее периодом. Затем полученные последовательности поступают на вход классификатора. Там выполняется их сравнение и последовательности со схожими параметрами объединяются в сигнал, который принадлежит одному ИРИ. Описанная выше структура существующего ПО представлена в виде блок-схемы на Рисунке 1.

Рисунок 1 – Блок-схема программного обеспечения

Для улучшения процесса разделения сигналов необходимо обновить функции модуля сепаратор, которые определяют возможные ППИ. В текущем ПО во всех функциях строится гистограмма SDIF с фиксированной шириной бина. В ходе проведенного исследования для каждого типа сигналов были подобраны наиболее эффективные гистограммы, так для типов constant и jitter такой гистограммой оказалась SDIF, а для stagger и stagger-jitter – CDIF, они и будут использоваться [5]. Для повышения точности определения ППИ в обновленных методах у гистограмм будет нефиксированная ширина бина, она будет зависеть от периода, который может попасть в этот бин. Также разработанный метод будет использовать «подвижный» уровень, который зависит от количества импульсов. Он позволит отбросить некоторые бины гистограммы, тем самым уменьшив количество вычислений.

В результате разработки усовершенствованного метода селекции, было значительно повышено качество разделения смесей сигналов. Для демонстрации работы алгоритма были смешаны 4 jitter-сигнала с периодами: 280, 700, 1000 и 1500. Результаты работы исходного и разработанного методов представлены на Рисунках 2 и 3. Исходный алгоритм не смог определить ни одного реального ППИ и объединил всю смесь в один сигнал неопределённого типа. В то время как усовершенствованный метод смог достаточно точно найти заданные периоды и корректно определил тип всех сигналов. Фиолетовым цветом выделена группа импульсов, которые алгоритм не смог идентифицировать, но она очень невелика.

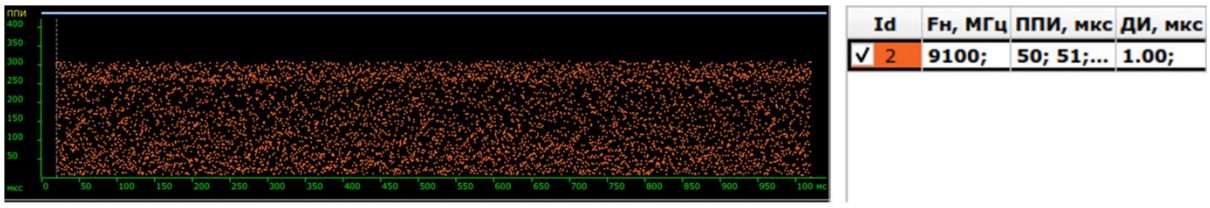


Рисунок 2 – Результаты работы исходного метода разделения сигналов

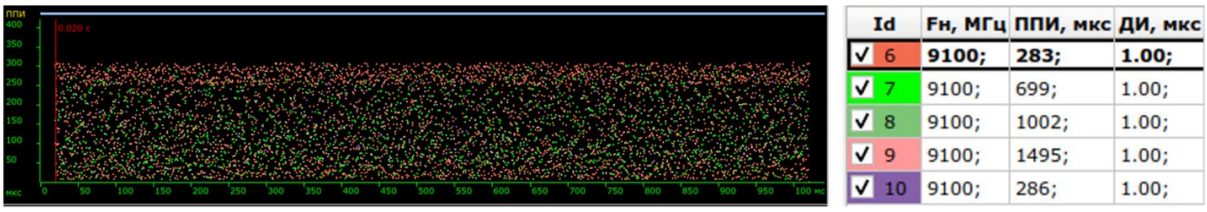


Рисунок 3 – Результаты работы усовершенствованного метода разделения сигналов

В ходе выполнения данной работы было существенно повышено качество селекции сигналов. На данный момент обновленные методы реализованы только для типов constant и jitter. Эффективность этих методов доказывает целесообразность разработки аналогичных методов для типов stagger и stagger-jitter.

ЛИТЕРАТУРА

1. Терентьев А. В., Коротков В. Ф. Радиотехническая разведка: Теория и практика обработки радиолокационных сигналов. – СПб.: Медиапапир, 2021. – 5 с.
2. Jia, J.; Han, Z.; Liu, L.; Xie, H.; Lv, M. Research on the SDIF Failure Principle for RF Stealth Radar Signal Design. Electronics 2022, 11, 1777. https://doi.org/10.3390/ electronics11111777

3. Y. Xi, Y. Wu, X. Wu and K. Jiang, "An improved SDIF algorithm for anti-radiation radar using dynamic sequence search," 2017 36th Chinese Control Conference (CCC), Dalian, China, 2017, pp. 5596-5601, doi: 10.23919/ChiCC.2017.8028245.
4. Korotkov V. F., Zyryanov R. S. Pulse Sequence Division in Mixed Signal Flow. Izvestiya Vysshikh Uchebnykh Zavedenii Rossii. Radioelektronika [Journal of the Russian Universities. Radioelectronics]. 2017, no. 3, pp. 5-10. (In Russian)
5. Nan, H., Peng, S., Yu, J. and Wang, X. (2019), Pulse interference method against PRI sorting. The Journal of Engineering, 2019: 5732-5735. https://doi.org/10.1049/joe.2019.0398

УДК 004.4

Н. А. Чевычелов (4 курс бакалавриата),
И. В. Шошмина, к.т.н, доцент

СОЗДАНИЕ API КИОТ ДЛЯ РАБОТЫ С ПРОФЕССИОНАЛЬНЫМ ПРОФИЛЕМ ОБУЧАЮЩЕГОСЯ

Конструктор индивидуальных образовательных траекторий (КИОТ) – система, предназначенная для построения для каждого обучающегося индивидуальной образовательной траектории на платформе «Moodle».[1] Система предоставляет пользователям набор инструментов для автоматизации подбора элементов дисциплин в соответствии с заданным профессиональным профилем обучающегося. Результатом разработки проекта КИОТ должно быть клиент-серверное приложение, реализованное в формате традиционного веб-приложения.

Цель работы - разработка API, который обеспечит взаимодействие между сущностями и частями конструктора индивидуальных образовательных траекторий.

В рамках данной работы остановимся подробнее на модуле API. Данный модуль представляет из себя протокол взаимодействия между графическим интерфейсом пользователя и ядром системы. В рамках протокола описываются точки взаимодействия между модулями FRONT-END и BACK-END, а также набор принимаемых и пересылаемых данных. [2]

Для описания протокола взаимодействия предлагается использовать REST API. REST API позволяет использовать для общения между модулями протокол HTTP.[3] Для описания REST API предлагается использовать OpenAPI.[6]

OpenAPI - представляет собой формализованную спецификацию и полноценный фреймворк для описания, создания, использования и визуализации веб-сервисов REST.[5]

Для генерации кода описанного посредством OpenAPI предлагается использовать Swagger Codegen. Данное ПО позволяет генерировать код для моделей пересылаемых данных, а также точек доступа для модулей FRONT-END и BACK-END.[4]

В рамках данной работы были выделены следующие задачи:

- Создание API курса и его компонентов
- Создание API для пользователей
- Создание API для платформы
- Реализация пагинации, фильтров и сортировки для курса и его компонентов
- Создание API для получения минимального, профессионального уровня, веса для ЗУНа в рамках индикатора
- Создание API для получения компетенций
- Создание API для получения индикатора компетенций
- Разработка API для абстрактного и профессионального профиля обучающегося
- Релиз прототипа программного продукта КИОТ

В процессе решения данных задач изучил понятие RESTful сервисов и ряд паттернов проектирования API.[6] Кроме того, столкнулся с проблемами выбора формата тела ответа и

тела запроса, сложностью корректной обработки ошибок. После более подробного изучения темы вопроса и обзора различных готовых решений данные проблемы были решены.

На данный момент были реализованы часть этапов из вышеперечисленных в результате которых были:

- Сформированы запросы, достаточные для полной работы (получение, удаление, изменение данных) с курсами, модулями, секциями, тестами и вопросами.
- Сформированы запросы для получения данных о пользователе, таких как его роль, запись на определенный курс, просмотр всех курсов в которых он участвует и другой необходимой информации
- Реализованы запросы необходимые для взаимодействия проекта с другими платформами
- Сформированы запросы, для получения курса или его элементов постранично, с возможностью изменения параметров таких как количество записей на странице. А также возможность получать отсортированные и отфильтрованные данные.

В будущем предстоит реализовать запросы для получения информации о компетенциях, индикаторах и ЗУН (знания умения навыки), а также запросы необходимые для работы с абстрактным профессиональным профилем обучающегося и профессиональным профилем обучающегося.

ЛИТЕРАТУРА

1. Moodle – Open-source learning platform [Электронный ресурс] Режим доступа: https://moodle.org/
2. Проектирование API [Электронный ресурс] Режим доступа: https://systems.education/api-design
3. HTTP-протокол [Электронный ресурс] Режим доступа: https://blog.skillfactory.ru/glossary/http/
4. Swagger [Электронный ресурс] Режим доступа: https://swagger.io/
5. OpenAPI [Электронный ресурс] Режим доступа: https://www.openapis.org/
6. REST API — RESTful веб-сервисы. [Электронный ресурс] Режим доступа: https://habr.com/ru/post/483202/

УДК 004.045

Д. К. Шейко, Д. И. Чирикин (2 курс бакалавриата),
Д. С. Эйзенах (2 курс, аспирант),
А. П. Маслаков, ст. преподаватель

РАЗРАБОТКА СЕРВЕРНОЙ СОСТАВЛЯЮЩЕЙ ДЛЯ СЕРВИСА ПОМОЩИ ПРИЮТАМ И ДОМАШНИМ ЖИВОТНЫМ “ISHELT”

Идея создания приложения основана на анализе потребностей и проблем, с которыми сталкиваются приюты и волонтеры, занимающиеся помощью животным. Недостаток финансирования, трудности в продвижении, ограниченные возможности для коммуникации — это лишь и несколько примеров трудностей, которые могут препятствовать эффективной помощи животным. Целью разработки приложения является устранение этих препятствий и упрощение процесса коммуникации и взаимодействия между приютами и волонтерами. Решение этих проблем в конечном итоге позволит улучшить условия жизни и ухода за животными, повысить эффективность работы приютов и волонтеров, а также улучшить общественное мнение о важности защиты прав животных.

В данной статье будет рассмотрена разработка серверной части приложения “iShelt”. Исходя из цели проекта были сформулированы следующие задачи:

1. Проанализировать и выбрать архитектуру.
2. Спроектировать схему базы данных.
3. Реализовать бизнес-логику приложения.
4. Обеспечить безопасность и защиту данным пользователей.
5. Предоставить доступ через внешнее API.

В качестве архитектурного стиля для взаимодействия компонентов приложения был выбран REST (representational state transfer), который использует протокол HTTP для передачи данных. Данные передаются в формате JSON.

Для реализации приложения требуется веб-фреймворк, было решено использовать Spring [1], благодаря крупному сообществу разработчиков и статической типизации в языке Java. В итоге основной стек технологий для реализации проекта состоит из Java 17 [2], сборщика Gradle и фреймворка Spring Boot версии 3.0.2. Аутентификация пользователей и валидация данных реализованы с помощью компонентов Spring Security [3] и JSON Web Token (JWT). В качестве системы хранения данных была выбрана база данных PostgreSQL.

В качестве шаблона проектирования для разработки выбран MVC (Model-View-Controller). Model отвечает за хранение и обработку данных, View - за отображение информации пользователю, а Controller - за управление взаимодействием между моделью и представлением. В качестве моделей у нас выступают сущности (entity), которые являются представлением полей таблиц из базы данных для приложения. При реализации функций создания, редактирования и получения сущностей возникла необходимость сократить количество запросов к базе данных, увеличить безопасность и разделить интерфейс от реализации. Было принято решение использовать шаблон проектирования DTO (Data Transfer Object). Это объект, который содержит только нужную для представления часть полей сущности и передается в качестве ответа на запрос контроллером.

База данных приложения включает в себя семь сущностей (Рис 1). Для изменений в базе данных используется система контроля версий Liquibase [4], которая позволяет контролировать изменения в структуре базы данных. Для работы с базой данных используется Spring Data JPA [5], который обеспечивает удобный и высокоуровневый интерфейс для работы с базами данных.

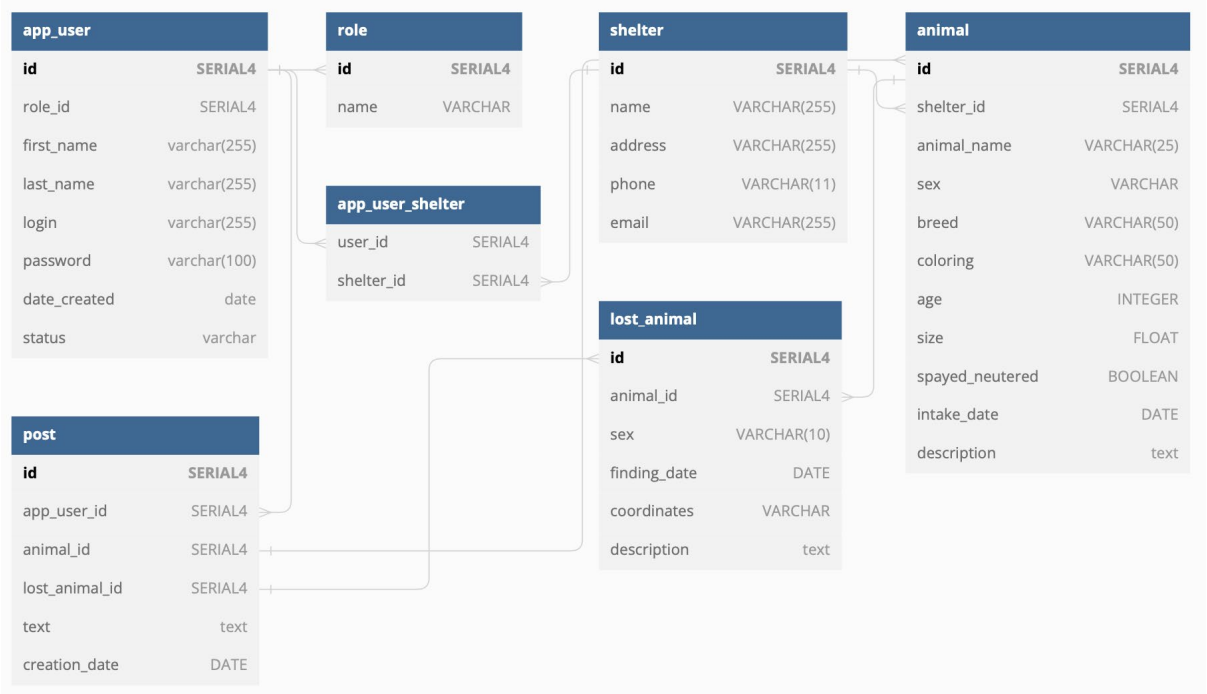


Рисунок 1 – Схема базы данных

При разработке приложения появилась проблема с тем, что нужно проводить валидацию поступающих данных. В связи с этим было решено выбрать валидаторы, предоставляемые фреймворком Hibernate. Этот выбор обоснован простотой использования и широкой функциональностью предоставляемых валидаторов.

В случае возникновения ошибки из базы данных, API либо логики клиенту не было понятно почему ответ на запрос серверу не пришел. Для улучшения этой ситуации были созданы обработчики ошибок для всех типов ошибок, которые возвращают ответ с

информативным сообщением об ошибке, упрощая процесс диагностики и устранения проблем.

Другой проблемой, с которой пришлось столкнуться, было избыточное количество сущностей таблицы, передаваемых в ответ на запрос, что приводило к дополнительным временным затратам. Для решения этой проблемы была внедрена пагинация для функций получения моделей, что позволяет значительно уменьшить количество обрабатываемых данных.

По итогу проведенной работы, вся серверная функциональность приложения была реализована и протестирована. Основная цель создания серверной части приложения была достигнута.

ЛИТЕРАТУРА

1. Craig Walls. Spring in Action. (2013).
2. H. Schildt. Java - The Complete Reference. Eleventh Edition. (2018).
3. Spring Security Reference. https://docs.spring.io/spring-security/reference/
4. Liquibase Documentation. https://docs.liquibase.com/
5. Spring Data JPA Reference. https://docs.spring.io/spring-data/jpa/docs/

УДК 004.42

А. Ю. Шестакова, Д. О. Королев (1 курс магистратуры),
И. Г. Черноруцкий, д.т.н., профессор

РАЗРАБОТКА TELEGRAM-БОТА ДЛЯ ПОИСКА АКЦИОННЫХ ТОВАРОВ В СУПЕРМАРКЕТАХ

На данный момент на территории Российской Федерации существует множество торговых сетей, и поиск акционных товаров, представленных в них, является достаточно частым бытовым занятием для многих людей. Так, для выбора магазина, в который наиболее выгодно пойти для приобретения товаров, покупатель может открыть сайты или приложения интересующих магазинов и посмотреть определенные категории акционных товаров или выполнить поиск по названию, что может быть не так удобно, как если бы товары из разных магазинов были собраны в одном месте.

Таким образом, целью работы является разработка Telegram-бота, отображающего акционные товары в различных торговых сетях и осуществляющего их поиск.

Для разработки проекта требуется выполнить следующие задачи:

– создание Telegram-бота;

– настройка базы данных (создание схемы, содержащей необходимые таблицы);

– разработка программного модуля системы, отвечающего за взаимодействие с базой данных;

– разработка программного модуля системы, отвечающего за обработку команд, выполняемых ботом;

– разработка программного модуля системы, отвечающего за обработку запросов к сайтам сетей магазинов;

– разработка программного модуля системы, отвечающего за взаимодействие с пользователем следующих модулей: модуля взаимодействия с базой данных, модуля обработки команд бота и модуля обработки запросов к сайтам сетей магазинов;

– развертывание системы.

Существуют готовые решения, такие как APiter (сайт) [1] и Edadeal (сайт и мобильное приложение) [2].

На сайте APiter представлены каталоги акций различных магазинов в формате, схожем с печатным вариантом каталога, – при выборе интересующего магазина открываются страницы, содержащие набор товаров, которые можно перелистывать. Такое представление не очень

удобно: нельзя осуществить поиск по интересующим категориям или названиям товаров, также для удобного восприятия информации о товаре требуется устройство с достаточно большим экраном.

На сайте Edadeal можно осуществить поиск как по названиям, так и по категориям товаров из различных сетей магазинов. Каждый товар представлен отдельным блоком, что упрощает просмотр. Существующий недостаток – нельзя выбрать интересующие магазины: можно осуществлять поиск либо по одному магазину, либо по всем представленным на сайте (в приложении).

В отличие от приведенных выше решений в функционале разработанного Telegram-бота присутствует выбор интересующих пользователя категорий товаров и сетей магазинов, а также поиск товаров по названию. Пользователь может выбрать несколько магазинов и просмотреть товары одновременно в нескольких магазинах по нескольким категориям (или по введенному названию товара).

Бот представляет найденные товары в виде последовательности сообщений, где каждое сообщение содержит информацию об одном товаре, что делает информацию более читаемой. Более того, преимуществом разработанного решения является то, что для его использования требуется иметь лишь установленное приложение Telegram (не требуется установки отдельных приложений).

Разработанная система состоит из следующих частей: база данных и приложение-бот, осуществляющее поиск информации по запросам пользователя [3].

Пользователь взаимодействует с ботом посредством Telegram приложения, которое, в свою очередь, принимая пользовательские запросы, передает их на сервер Telegram API Server, который шифрует трафик и обеспечивает связь с реализованным приложением. При этом Telegram использует собственный протокол шифрования MTProto.

При разработке бота был использован язык Java и Telegram Bot API — надстройка над Telegram API, использующая вспомогательный сервер, самостоятельно обрабатывающий все шифрование и связь с Telegram API [4]. Соединение с сервером происходит через HTTPS-интерфейс, который предоставляет простую версию Telegram API [5].

На Рис. 1 представлена архитектура приложения.

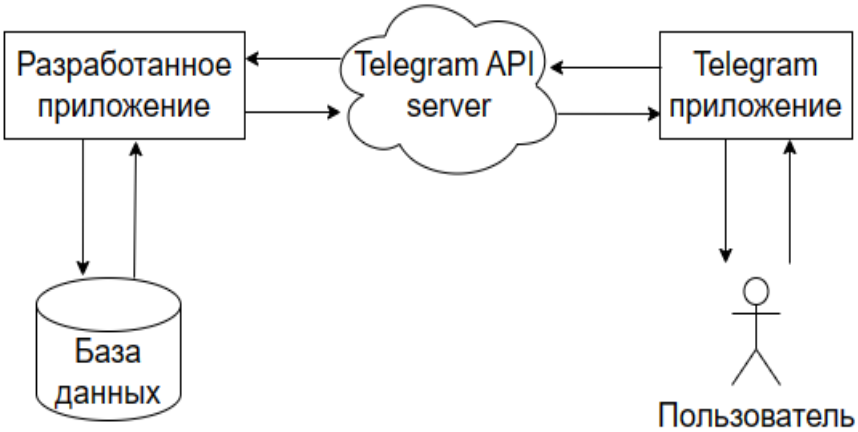


Рисунок 1 – Архитектура приложения Telegram-бота

Для обеспечения функционирования бота на платформе Telegram было произведено развертывание бота на платформу Heroku, на которой также располагалась база данных. Для автоматизации процесса развертывания настроен CI/CD конвейер с использованием Github Actions [6].

ЛИТЕРАТУРА

1. APiter [Электронный ресурс]. Дата обращения: 12.03.2023. http://www.a-piter.ru/.
2. Edadeal [Электронный ресурс]. Дата обращения: 12.03.2023. https://edadeal.ru/.

3. Djoelianto A., Kautsar I., Rosid M. Development of Web Service and Telegram Bot for Location-Based Health Service Information System // Procedia of Engineering and Life Science, 2022 – DOI: 2. 10.21070/pels.v2i2.1280.
4. Telegram APIs [Электронный ресурс]. Дата обращения: 12.03.2023. https://core.telegram.org/#api-methods
5. Modrzyk N. Building Telegram Bots: Develop Bots in 12 Programming Languages using the Telegram Bot API, 2019 – ISBN 978-1-4842-4196-7.
6. Benedetti G., Verderame L., Merlo A. Automatic Security Assessment of GitHub Actions Workflows // CCS '22: 2022 ACM SIGSAC Conference on Computer and Communications, 2022 – P. 37-45 – DOI: 10.1145/3560835.3564554.

УДК 004.056.5

В.А. Шлык (4 курс бакалавриата),
К.В. Пшеничная, к.т.н., доцент

РАЦИОНАЛЬНЫЙ ВЫБОР СРЕДСТВ ЗАЩИТЫ КОМПЬЮТЕРНОЙ СИСТЕМЫ

Известны задачи, решение которых основывается на генерировании возможных вариантов искомых параметров (вариантов решения). Процесс генерирования и модификации вариантов осуществляется с использованием некоторого алгоритма, учитывающего оценку каждого сформированного варианта, с целью нахождения рационального решения. Такой подход может быть применен к актуальной в настоящее время задаче рационального выбора средств защиты компьютерной системы.

Целью работы была разработка программы, которая по заданной матрице «Угрозы-средства защиты» (Табл.1), содержащей значения p_{ij} эффективностей i-го средства защиты (D_1, D_2, …, D_i, …) против j-ой угрозы (T_1, T_2, …, T_j, …), позволяет выбрать рациональный набор средств защиты.

Таблица 1 – Матрица «Угрозы-средства защиты»

Средства защиты	Угрозы			
	T_1	T_2	T_3	…
D_1	p_{11}	p_{12}	p_{13}	p_{ij}
D_2	p_{21}	p_{22}	p_{23}	…
D_3	p_{31}	p_{32}	p_{33}	…
…	…	…	…	…

Помимо матрицы «Угрозы-средства защиты» заданными являются значения стоимостей средств защиты, значения ущерба от реализации угроз, актуальные угрозы и требуемые значения эффективности защиты от них. Минимизируется стоимость набора средств защиты при выполнении ряда ограничений.

Математическая постановка задачи представляет модель условной дискретной оптимизации. Для решения задачи возможны различные подходы: в работе [1] задача решается с использованием методов теории игр, в работе [2] используется метод иерархий Саати. В данной работе поиск рационального набора средств защиты компьютерной системы основывается на применении одного из самых известных эволюционных алгоритмов − генетического алгоритма [3].

Структурная модель системы выбора средства защиты представлена на Рис. 1.

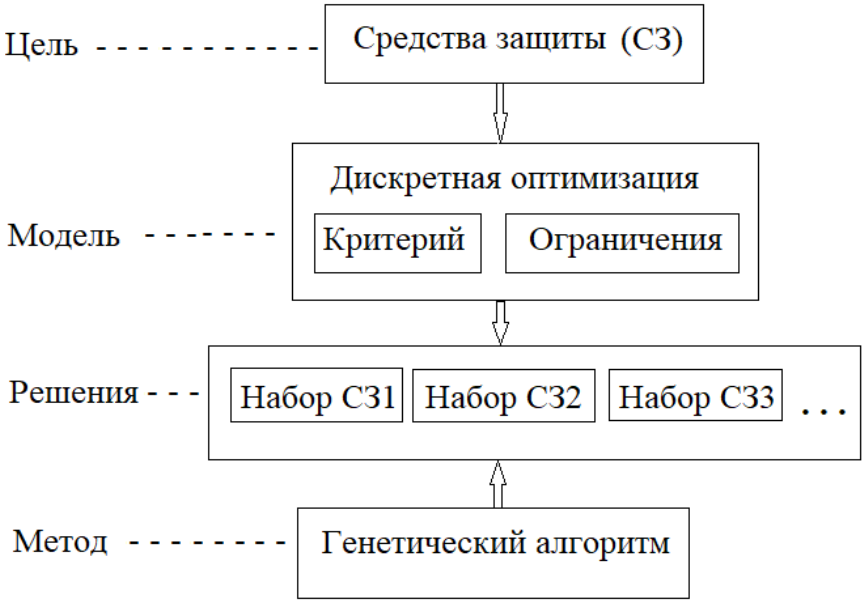


Рисунок 1 – Структурная модель системы выбора средства защиты

В основе генетического алгоритма лежит идея формирования базы знаний (популяции), состоящей из особей. Каждая особь представляет структуру, состоящую из элементов (генов). В рассматриваемой задаче каждый ген равен нулю или единице и указывает на наличие или отсутствие соответствующего средства защиты. Процесс реализации генетического алгоритма предусматривает следующие этапы:

1. Формирование исходной популяции.
2. Вычисление функции приспособленности (критерия качества и ограничений).
3. Проверка условия окончания процесса. При выполнении условия – выход.
4. Формирование новой популяции на основе правил отбора особей и применения генетических операторов.
5. Переход к этапу 2.

Для решения задачи возможно использование современных библиотек, реализующих генетический алгоритм, например[4]. В данной работе было разработано собственное приложение на языке C#, допускающее более гибкую работу с применяемым методом.

ЛИТЕРАТУРА

1. Быков А. Ю., Алтухов Н. О., Сосенко А. С. Задача выбора средств защиты информации в автоматизированных системах на основе модели антагонистической игры // Инженерный вестник МГТУ им. Н.Э. Баумана. Электрон. журн. 2014. №04. Режим доступа http://www.ainjournal.ru/doc/708106.html (дата обращения 17.03.2023).
2. Скоков Н.С., Пшеничная К.В. Выбор средств нейтрализации атак вида «Отказ в обслуживании» // Современные технологии в теории и практике программирования: сборник материалов конференции, 2021 г. – СПб.: ПОЛИТЕХ-ПРЕСС, 2021, с. 58-60.
3. Вирсански Э. Генетические алгоритмы на Python / пер. с англ. А. А. Слинкина. – М.: ДМК Пресс, 2020.
4. DEAP documentation. [Электронный ресурс]. Режим доступа: https://deap.readthedocs.io/en/master/.

УДК 004.415.2

Э.О. Шумковская (2 курс магистратуры),
С.А. Молодяков, д.т.н., доцент

РАЗРАБОТКА ПРОГРАММНЫХ СРЕДСТВ ОБРАБОТКИ ЛОКАЛЬНЫХ ДАВЛЕНИЙ, ИЗМЕРЕННЫХ ДАТЧИКАМИ TEKSCAN

Матричные пленочные датчики Tekscan применяют для исследований значений локальных нагрузок и давлений на части сооружения. Датчики данного типа позволяют узнать размеры и расположение фактически нагруженной поверхности сооружения, а также динамику ее нагружения. В исследованиях локальных ледовых нагрузок [1] и [2] описано использование датчиков Tekscan для сбора информации и методы, применяемые для её дальнейшей обработки.

Оборудование, применяемое в исследованиях - сенсоры, контроллер, подключаемый к компьютеру с ПО I-Scan, из которого данные можно сохранить в текстовом формате [1].

Полученные данные представляют собой ASCII-файл формата csv, содержащий десятки тысяч матриц размерностью 44x44, 44x52 или 100x100 в зависимости от типа датчика. Одна минута записи с частотой 200 Гц дает около 30 Мб данных.

Для обработки данных с датчика Tekscan можно использовать следующее ПО:

– I-Scan — ПО, поставляемое вместе с датчиками Tekscan

– Matlab — платформа для программирования и числовых вычислений, используемая миллионами инженеров и ученых для анализа данных, разработки алгоритмов и создания моделей.

В I-Scan™ Product Selection Guide [4] описаны возможности ПО, поставляемого месте с датчиками. Основные возможности — это визуализация в различных режимах, калибровка, вычисление центра сил, пиковой силы, пикового давления, контактных зон, а также сохранение в формате csv или avi. Возможностей I-Scan недостаточно для исследований, в исследованиях применялись другие инструменты. Поэтому поставляемое ПО используется только для сохранения данных, и данные обрабатываются с помощью других инструментов.

В работе [2] для обработки данных и построения графиков применялся Matlab. Но из-за большого объёма получаемых данных программа на Matlab работает слишком медленно. В сравнении времени вычислений между Matlab и C++ [5] показано, что на задачу, которую код на C++ решает меньше, чем за минуту, код на Matlab тратит от 11 минут до нескольких часов.

Так как функциональность ПО I-Scan недостаточна для исследований, а код на Matlab обрабатывает данные недостаточно быстро, возникает необходимость в создании собственного ПО для обработки данных.

Разрабатываемая программа должна читать и обрабатывать данные с высокой скоростью, так как данных много. Для удобства использования программа должна иметь графический пользовательский интерфейс. Для удобства хранения и поиска данных программа должна иметь подключение к базе данных. Для оперативности работы исследователей программа должна иметь подключение к облачному хранилищу.

Для разработки был выбран язык программирования C++. В исследовании скорости выполнения базовых математических задач [6] показано, что C++ один из самых быстрых языков программирования. Код на C++ выполнил тестовые задачи примерно в 2 раза быстрее, чем код на C#, Python и Java. Также данный язык входит в десятку самых популярных языков программирования по индексу TIOBE.

Для создания интерфейса был выбран .Net Framework с использованием C++/CLI. В IDE Visual Studio есть конструктор форм для данного фреймворка, что сильно упрощает создание пользовательского интерфейса. C++/CLI – расширение, которое связывает код на C++ с фреймворком .NET. Код на C++/CLI является управляемым и может автоматически освобождать память, которая была динамически выделена для хранения данных [7]. Это устраняет источник распространенных собственных ошибок C++ и упрощает разработку.

Для хранения файлов с исходными данными была выбрана СУБД MongoDB [8]. Это документоориентированная NoSQL система управления базами данных. MongoDB позволяет загружать большие файлы благодаря системе FastFS. Также данная СУБД доступна на российских облачных сервисах.

Таким образом, на основе рассмотренных программных средств разработана система для работы с данными, получаемыми с датчиков. Она обеспечивает обработку файлов до 4 Гб. Применение программной системы позволило выполнять анализ нагруженных областей сооружения.

ЛИТЕРАТУРА

1. Экспериментальное изучение локальных ледовых давлений на модель ледостойкого сооружения Звягин П.Н. Гидротехника. 2021. № 4 (65). С. 18-21.
2. Sodhi, D. S. Crushing failure during ice-structure interaction. Engineering Fracture Mechanics, 68 (17–18), 1889–1921. https://doi.org/10.1016/S0013-7944(01)00038-8.
3. Ziemer, G., Evers, K.-U., Jochmann, P., Hoffmann, N. Study of Local Ice Pressure Distribution on a Cylindrical Offshore Structure by Using Tactile Sensors // In: Proceedings of 22nd IAHR International Symposium on Ice, Singapore, August 11–15, 2014, pp. 241–248.
4. I-Scan™ Product Selection Guide [Электронный ресурс]. Режим доступа: https://www.tekscan.com/products-solutions/systems/i-scan-system
5. Giesl P., Hafstein S., Mehrabinezhad I. Computing Contraction Metrics: Comparison of Different Implementations //IFAC-PapersOnLine. – 2021. – Т. 54. – №. 9. – С. 310-316.
6. Исследование скоростей выполнения базовых математических задач популярных языков программирования Коровин И.В. Экономика и качество систем связи. 2019. №3. С. 63-68.
7. Программирование .NET с помощью C++/CLI [Электронный ресурс]. Режим доступа: https://docs.microsoft.com/en-us/cpp/dotnet/dotnet-programming-with-cpp-cli-visual-cpp?view=msvc-170
8. MongoDB [Электронный ресурс]. Режим доступа: https://www.mongodb.com/

УДК 004.455.2

В. И. Шишкина, С. А. Аненко (1 курс, магистратуры),
В. Е. Крейнин,
И. В. Никифоров, к. т. н., доцент

ПРОЦЕСС СИНХРОНИЗАЦИИ ПОЛЬЗОВАТЕЛЕЙ МЕЖДУ СЛУЖБАМИ КАТАЛОГОВ НА ПРИМЕРЕ ИНСТРУМЕНТОВ MS ACTIVE DIRECTORY И FREEIPA

На сегодняшний день в крупных индустриальных компаниях из любой области производства, управления или коммерции используются программные инструменты организации и разграничения прав доступа пользователей к общим информационным, техническим, аппаратным и другим ресурсам. Такие программные средства называются службами каталогов (*directory service*) [1]. К общим ресурсам пользователей можно отнести: папки, файлы, принтеры, доступ в интернет, сетевые ресурсы, административные ресурсы, кадровая информация, бухгалтерия и т. д.

Служба каталогов централизованно хранит всю информацию, требуемую для использования и управления ресурсами, упрощая процесс поиска и конфигурирования для пользователя. Она управляет идентификацией и отношениями между распределенными ресурсами и позволяет им работать вместе.

Доминирующую позицию на рынке служб каталогов занимает продукт компании *Microsoft – MS Active Directory* [2]. Существенным минусом использования этой системы в отечественных компаниях является ее зарубежное происхождение, что влечет за собой непосредственные риски блокировки использовании системы в связи с отзывом лицензий или невозможностью их оплаты и пролонгации.

Поэтому сейчас у отечественных компаний актуальной задачей является переход от многофункциональной, годами развивавшейся инфраструктуры домена на экосистеме зарубежных продуктов к устойчивым к санкционным рискам отечественным решениям.

Для того, чтобы перейти к рассмотрению процессов миграции от существующей службы каталогов к альтернативной, необходимо выбрать новую службу каталогов. В данной работе примем за используемый импортонезависимый ПК *ALDPro* [3] – продукт ГК «Астра» (ООО «РусБИТех-Астра»), разработанный на основе открытого программного продукта *FreeIPA* [4].

Задача ручного перехода и переноса данных – это крайне трудоемкий процесс, который не свободен от ошибок человека по невнимательности [5]. Для компании среднего размера (более 1000 человек) в общем случае трудоемкость ручного процесса переноса данных может оцениваться в 2-3 человеко-месяца. И еще 1-2 человека месяца на тестирование и проверку работоспособности всех информационных систем компании.

Несмотря то, что между схемой *LDAP Active Directory* [6] и *389 Directory Server LDAP*-схемой, используемой *FreeIPA*, существуют значительные различия, существует много атрибутов, которые одинаковы. Эти атрибуты просто синхронизируются между элементами *Active Directory* и *FreeIPA*, без изменений в имени атрибута или в формате значений. Некоторые атрибуты имеют разные названия, но все же их можно ассоциировать между собой.

В данной работе проведен краткий обзор служб каталогов *MS Active Directory* и *ALDPro* и рассмотрен процесс синхронизации существующего «леса» в *MS AD* и нового – во *FreeIPA*.

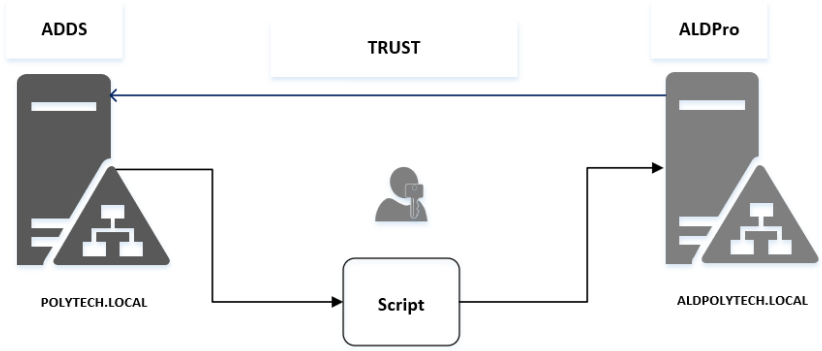


Рисунок 1 – Схема миграции пользователя

Из всего вышесказанного можно сделать вывод, что процесс замены действующей службы каталогов на новую импортонезависимую можно построить следующим образом: необходимо организовать новый домен параллельно с существующим, перенести в него объекты из существующего, а все новые записи вести только в целевой системе.

Глобальной целью исследования является сокращение трудозатрат на работы по миграции данных между разными службами каталогов за счет разработки набора скриптов [7] для копирования объектов и их атрибутов.

Реализовав в программном средстве представленный подход можно ожидать снижения трудоемкости миграции данных в 10-15 раз. При этом трудоемкость тестирования качества миграции данных тоже снизится, но все равно останется существенной.

В ходе реализации данного подхода планируется проведение следующих экспериментов:

– копирование учетных записей пользователей в наполненный сторонними объектами домен FreeIPA;

– копирование учетная записей пользователей и групп доступа, в которые они включены, проверка связей;

– копирование целой ветки объектов, проверка связей.

ЛИТЕРАТУРА

1. Sheresh B., Sheresh D. Understanding Directory Services. System Research Corporation, 2002. ISBN 0-672-32305-2.

2. Коробко И. Каталог Active Directory. Поиск объектов с помощью PowerShell // Системный администратор – 2019. – № 4(197). – С. 36-38.
3. Справочный центр Astra Linux [Электронный ресурс]. URL: https://wiki.astralinux.ru/display/doc/Astra+Linux+Directory (дата обращения: 16.02.2023).
4. Кантер, Л. Построение корпоративной системы управления идентификационной информацией на базе FreeIPA / Л. Кантер // Пятнадцатая конференция разработчиков свободных программ: Тезисы докладов, Калуга, 28–30 сентября 2018 года / Ответственный редактор В.Л. Черный. – Калуга: ООО "МАКС Пресс", 2018. – С. 44-46.
5. Хитев А. Ю. Программа автоматической развертки сервера SAMBA и миграции записей из Active Directory в SAMBA // Бизнес-информатика: состояние, проблемы, перспективы: Сборник материалов Международной школы-семинара, Санкт-Петербург, 19–21 сентября 2013 года. – Санкт-Петербург: Санкт-Петербургский государственный университет, 2013. – С. 179-182.
6. Manav A. Thakur, Rahul Gaikwad. User identity & lifecycle management using LDAP directory server on distributed network. – International Conference on Pervasive Computing (ICPC), 2015.
7. Лавлинская О. Ю., Кузнецов В. И., Петрученко М. К. Скриптовый язык PowerShell как средство автоматизации работы с объектами службы active directory // Информационные технологии в строительных, социальных и экономических системах. – 2019. – № 1(15). – С. 55-58. – EDN CIRQCG.

УДК 004.42

М. А. Шутова (4 курс бакалавриата),
Е. С. Орлов, ст. преподаватель

РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ МГНОВЕННОГО ДОСТУПА К ДОКУМЕНТАМ НА ОСНОВЕ NFC

NFC означает Near Field Communications — «коммуникации ближнего поля» или «ближняя бесконтактная связь». Это технология, которая активно развивается в данный момент, и позволяет передавать данные между двумя устройствами беспроводным способом.

NFC-метка для работы использует принцип электромагнитной индукции: магнитное поле возникает на приемной и передающих антеннах при приближении, и возникающий электроток питает электрочипы метки. Сами же электрочипы выполняют функции шифрования и расшифровки данных, а также хранение и запись информации. Установить соединение с микрочипами помогает металлическая катушка.

У этой технологии есть множество применений.

Область применения NFC	Пример применения NFC
Обмен информацией	▪ обмен электронными визитками; ▪ обмен телефонными номерами; ▪ печать фото с цифровой камеры без подключения к компьютеру;
Электронный доступ	▪ активация двигателя автомобиля без ключа; ▪ предоставление доступа в помещение и контроль прошедших; ▪ включение умной колонки;
Доступ к информации	▪ использование запрограммированной метки в качестве ЭЦП для подписания документов; ▪ считывание навигационных координат; ▪ загрузка расписания поездов;
Финансовые возможности	▪ Оплата покупок; ▪ Запрос денег; ▪ Перевод денег на другой банковский счет; ▪ Использование в качестве электронного билета; ▪ Работа с бесконтактными терминалами продаж;

Но одно из важнейших применений NFC осуществляется через возможность меток предоставлять пользователю электронный доступ к различным местам, функциям и предметам. Примером являются контроль доступа в помещение, открытие автомобиля или настройка умной колонки. По-другому подобные процессы называются микро-автоматизацией. NFC можно использовать также для осуществления контроля доступа к определенным базам данных. Это может помочь автоматизировать доступ к документам.

В рамках данного проекта была поставлена задача создать систему мгновенного доступа к базе данных документов при помощи NFC чипа. Эта система должна включать в себя:

– NFC-носитель, при помощи будет которого осуществляться доступ к удостоверению личности.

– Сканер NFC-меток PN532. Взаимодействие с компьютером должно проходить через плату Arduino Uno.

– Программа, написанная в Arduino IDE, которая считывает записанное на метке сообщение и отправляет его на обработку в приложение.

– Программа, написанная на C#, которая предоставляет пользователю интерфейс, а также занимается обработкой переданного ей с NFC-меткой сообщения и нахождением в базе данных искомых документов.

– База данных, содержащая удостоверение личности, которое содержит как минимум имя, фамилию, отчество, серию и номер паспорта.

Самая главная характеристика данной системы — это безопасность. Главная причина, по которой на NFC-чипе нельзя хранить персональные данные, в том числе документы, заключается в том, что эти чипы подвержены многим видам атак.

Чтобы избежать таких угроз, как модификация данных или перехват данных вместо того, чтобы хранить документ, или ссылку на него, на чипе, документ хранится в базе данных. А на nfc-метке записан ключ, через который идет доступ к нужной записи в такой базе данных. Помимо этого, в систему введена двухфакторная аутентификация: чтобы получить доступ к записи в базе данных, надо не только иметь при себе метку, но также знать дополнительный пароль. Данные изменения делают систему значительно безопаснее, повышая оценку этой характеристики системы до удовлетворительной. Но несмотря на то, что безопасность в данном моменте разработки на удовлетворительном уровне, она может быть усовершенствована в будущем.

Дополнительными характеристиками системы являются скорость и удобство. На данной стадии разработки система не испытывает проблем с задержками, передача данных по порту идет практически моментально, что обеспечено синхронной работой программы в Arduino IDE и графического интерфейса. Удобство системы же оценивается по критерию избыточности интерфейса. Системе не нужен сложный интерфейс, но тем не менее он должен исполнять свою главную функцию: он должен демонстрировать результат поиска конкретной записи пользователю. На данной стадии разработки интерфейс не избыточен. Его функционал, тем не менее, может быть доработан в будущем.

Потенциально, эта система может развиваться. В нее могут быть добавлены новые базы данных, например, для полиса, снилса, документов на питомца, и ко всем ним у человека будет мгновенный доступ, при наличии у него его особой nfc-метки. Эта система может помочь ускорить процесс массовой оцифровки документов, что сделает повседневную жизнь проще и комфортнее.

ЛИТЕРАТУРА

1. Selin Olenik, Hong Seok Lee & Firat Güder// The future of near-field communication-based wireless sensing// 2021 // URL: https://www.nature.com/articles/s41578-021-00299-8 // Дата обращения: 15.03.2023
2. Постников Роман Сергеевич, Сидоров Павел Андреевич// Технология NFC в современном мире// 2020 // URL: https://cyberleninka.ru/article/n/tehnologiya-nfc-v-sovremennom-mire // Дата обращения: 15.03.2023

3. Фетисенко Алёна Константиновна// Криптографическая защита персональных данных при передаче по интерфейсу NFC: дипломный проект// 2014 // URL: https://elib.spbstu.ru/dl/2/6429.pdf/info // Дата обращения: 18.03.2023
4. Макаров Александр Сергеевич// Применение технологии NFC для создания защищенного USB-накопителя: магистерская диссертация// 2016 // URL: https://elib.spbstu.ru/dl/2/v16-1313.pdf/info// Дата обращения: 18.03.2023

УДК 004.42

Ф.А. Фернандез (4 курс бакалавриата),
Т.Н. Самочадина, ст. преподаватель,
А.В. Самочадин, к.т.н., доцент

ПРОТОТИП TELEGRAM-БОТА ДЛЯ ПРЕДОСТАВЛЕНИЯ ИНФОРМАЦИИ

Чат-боты позволяют давать быстрые и четкие ответы на вопросы, информацией по которым они обладают. Это одна из форм обслуживания клиентов [1], которая становится все более популярной. Их функциональность варьируется от сбора, обработки и предоставления информации до психологической помощи. По мере того, как они набирают популярность, растет и количество технологий, которые можно использовать для их разработки. Это привело к разработке low-code/no-code платформ, с помощью которых чат-бот может быть создан без необходимости обладать глубокими техническими знаниями в области информационных технологий [2]. Конечно, у этого типа платформ также есть недостатки, которые необходимо учитывать при выборе способа разработки чат-бота. Например, использование таких платформ приводит к зависимости от этих платформ, что может быть критичным, если платформа станет недоступна или ее качество снизится. Это приведет к необходимости повторной разработке чат-бота на другой платформе. Кроме того, уровень безопасности, предлагаемый этими платформами, может быть не высоким [3].

Целью работы является исследование возможностей Telegram Bot API для предоставления информации о работе посольства иностранного государства и решения других вопросов. Таких, например, как регистрация мигрантов или процесс выдачи документов для оформления справок и паспортов.

Для выполнения проекта был выбран язык программирования Python [4]. Был использован модуль Python-Telegram-Bot. Его ключевая особенность - полная поддержка всего функционала Telegram Bot API, который является интерфейсом, основанным на HTTP протоколе. Для хранения данных была использована система управления базами данных PostgreSQL. Архитектуру проекта было решено разделить на 2 модуля: модуль базы данных, который отвечает за хранение данных; модуль чат бота – отвечает за пользовательский интерфейс и является связующим звеном для всего проекта. На Рис. 1 изображено графическое представление связи модулей в архитектуре проекта [5].

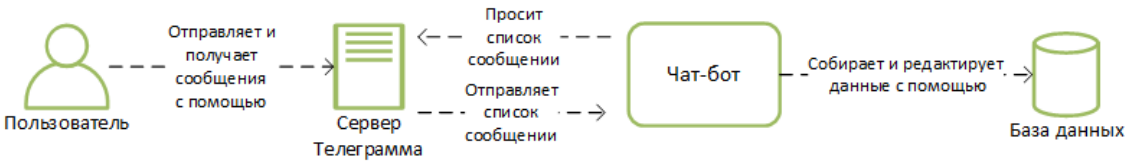


Рисунок 1 – Архитектура чат-бота

База данных состоит из следующих таблиц:

1. Telegram user – содержит информацию о языке, который предпочитает пользователь, id чата и имя/тэг пользователя.
2. Person – данные пользователя, такие как имя, номер телефона, почта, и т.д.
3. Request – запросы на оформление документов.
4. Document – виды документов.

Бот должен обеспечивать реализацию следующих функций (Рис. 2):

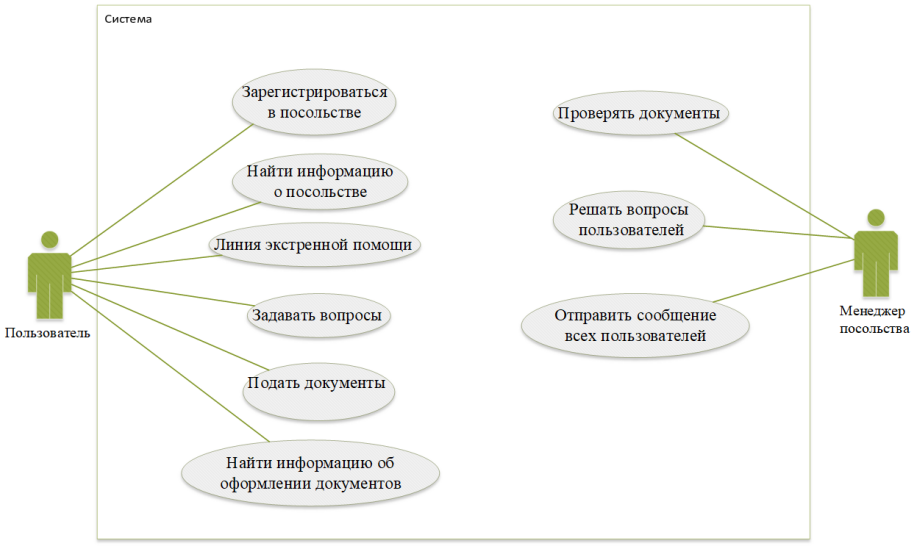


Рисунок 2 – Диаграмма вариантов использования чат-бота

ЛИТЕРАТУРА

1. What is a Chatbot and Why is it Important? (techtarget.com) — Kate Brush, Jesse Scardina [Электронный ресурс] Режим доступа: https://www.techtarget.com/searchcustomerexperience/definition/chatbot
2. What are Low-Code and No-Code Development Platforms? — Mary K. Pratt [Электронный ресурс] Режим доступа: https://www.techtarget.com/searchsoftwarequality/definition/low-code-no-code-development-platform
3. Low-Code as Enabler of Digital Transformation in Manufacturing Industry — Raquel Sanchis 1,*, Óscar García-Perales 2 , Francisco Fraile 3 and Raul Poler [Электронный ресурс] Режим доступа: https://mdpi-res.com/d_attachment/applsci/applsci-10-00012/article_deploy/applsci-10-00012-v2.pdf?version=1577022060
4. Top 6 Programming Languages for Chatbot Development (codecademy.com) — Stephan Miller [Электронный ресурс] Режим доступа: https://www.codecademy.com/resources/blog/top-6-programming-languages-for-chatbot-development/
5. How to make a Telegram Bot in right way | Service Desk — Svyatoslav [Электронный ресурс] Режим доступа:https://www.servicedesk.site/en/2019/08/09/%D1%8F%D0%BA-%D1%81%D1%82%D0%B2%D0%BE%D1%80%D0%B8%D1%82%D0%B8-%D1%82%D0%B5%D0%BB%D0%B5%D0%B3%D1%80%D0%B0%D0%BC-%D0%B1%D0E%D1%82%D0%B0-%D0%BF%D1%80%D0%B0%D0%B2%D0%B8%D0%BB%D1%8C%D0%BD%D0%BE/

Секция «Программная инженерия: инструментальные средства и технологии проектирования и разработки»

УДК 004.415.2

Н.В. Аникин (2 курс магистратуры),
В.В. Амосов, к.т.н., доцент

ПРОГРАММНЫЙ КОМПЛЕКС СБОРА, ХРАНЕНИЯ И АНАЛИЗА СТАТИСТИКИ СЕТЕВОГО ТРАФИКА

Предприятия используют комплексы статистической обработки сетевого трафика для мониторинга загрузки сетевого оборудования и каналов связи, а также для анализа взаимодействия между узлами сети для обеспечения информационной безопасности.

Известные программные средства анализаторов сетевого трафика, такие, как Cisco Stealth Watch, nTop и др., представляют собой мощные и дорогостоящие системы мониторинга, бесплатные версии которых имеют ограничения по производительности [1, 2]. Кроме того, некоторые предприятия, имеющие стратегическое значение в экономике, могут иметь ограничение на использование импортного программного обеспечения. В связи с этим актуальной является задача разработки систем мониторинга и аналитики безопасности сетевой инфраструктуры для отечественных предприятий. В работе рассматривается обновление ранее реализованного комплекса сбора и хранения статистики сетевого трафика и дополнение его модулем анализа, для которого необходимо разработать программные средства сбора и хранения статистики сетевого трафика [3].

Программные средства сбора используют данные телеметрии от маршрутизаторов, коммутаторов, межсетевых экранов и других сетевых инфраструктурных устройств разных производителей. Информация представлена в виде потоковых данных, сформированных по правилам различных протоколов: NetFlow, IPFIX, sFlow, NetStream [4 - 7].

Целью работы является разработка программного комплекса сбора, хранения и анализа статистики сетевого трафика промышленного назначения, обеспечивающего декодирование, нормализацию и анализ на предмет подозрительной статистики пакетов NetFlow, IPFIX, sFlow и NetStream.

Для достижения поставленной цели решены следующие задачи:

1. Разработка программных средств обработки пакетов.
2. Разработка средств тестирования программного комплекса.

Программный комплекс состоит из следующих компонентов:

– Приложение Flow-Collector, выполняющее декодирование приходящих пакетов;

– База данных GeoLite2-City (представлена в виде файла формата .mmdb, содержащего диапазоны IP-адресов и соответствующие им географические расположения), используется для сопоставления IP-адресов, приведённых в каждом пакете, со страной и городом;

– Распределённый программный брокер сообщений Apache Kafka [8];

– Программный комплекс ELK Stack, хранящий обработанные данные [9];

– Приложение Flow-Analytics (Анализатор), выполняющее анализ предварительно обработанных пакетов.

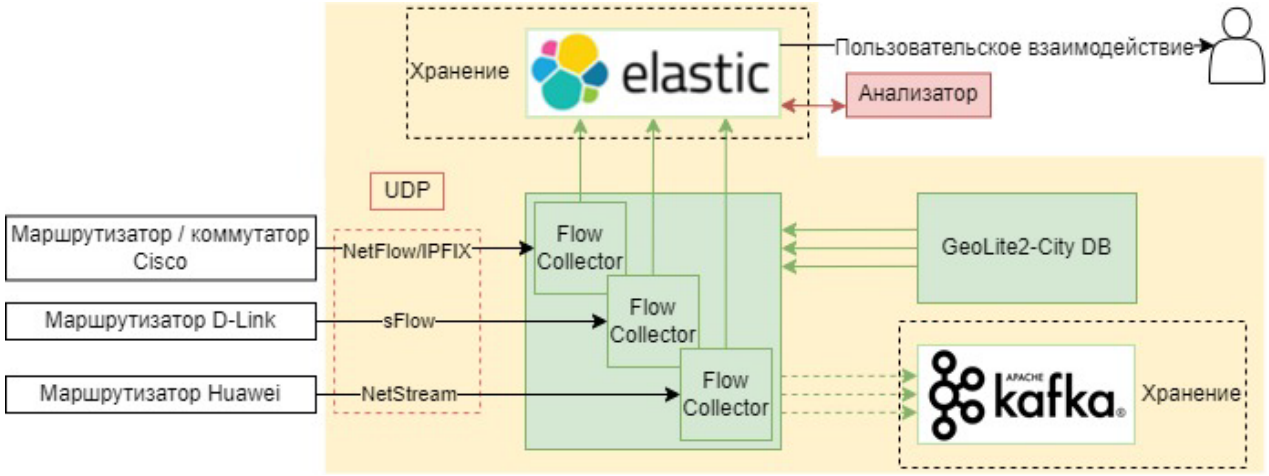


Рисунок 1 – Функциональная схема сбора и хранения статистики сетевого трафика

На Рисунке 1 представлена функциональная схема разрабатываемого комплекса. Маршрутизаторы различных производителей (Cisco, D-Link, Huawei) формируют пакеты, содержащие информацию о сетевом трафике, и отправляют их на обработку в Flow-Collector. Каждый из трех процессов Flow-Collector обрабатывает пакеты только определенного потока (NetFlow/IPFIX, sFlow, NetStream) за счет установки разных UDP портов в настройках маршрутизаторов. Каждый пакет проверяется на соответствие ожидаемому протоколу, данные пакета декодируются и преобразуются к единому формату, а также дополняются (например, сведениями о местоположении сетевого оборудования, полученными из БД GeoLite2-City по IP-адресам). Собранная информация передается на хранение в ELK Stack. Оттуда данные попадают в анализатор. В анализаторе получаемые данные проверяются на предмет наличия статистики о подозрительном поведении в сети (например, сканировании портов, подключении к сайтам из чёрного списка, использовании слабых криптостандартов), информация о потенциальных нарушениях направляется обратно в Elastic. Программные средства Efros Config Inspector обеспечивают интерфейс взаимодействия с пользователем [10]. Распределённый программный брокер сообщений Apache Kafka используется для тестирования и отладки программного обеспечения, а также для последующего расширения функциональности комплекса.

Для проверки пропускной способности был разработан программный генератор тестовых пакетов, отправляющий тестовые пакеты NetFlow/sFlow на UDP порт по заданному IP-адресу. Генератор позволяет задавать задержку между пакетами в диапазоне от 0,4 мс до 5 с и длительность сеанса тестирования (количество тестовых пакетов).

Результаты тестирования показали, что пакеты с информацией о сетевом трафике обрабатываются модулем Flow-Collector в соответствии с требованиями. В дальнейшем инструментарий для тестирования расширится средствами для функционального тестирования модуля анализатора и будет проведено комплексное тестирование системы.

ЛИТЕРАТУРА

1. Cisco Secure Network Analytics (Stealthwatch) // Cisco. – URL: https://www.cisco.com/c/ru_ru/products/security/stealthwatch/index.html (дата обращения: 09.01.2022).
2. ntop – High Performance Network Monitoring Solutions based on Open Source and Commodity Hardware. // ntop. – URL: https://www.ntop.org/ (дата обращения: 09.01.2022).
3. ЭБ СПбПУ - Создание программного комплекса сбора и хранения статистики сетевого трафика промышленного назначения // Н.В. Аникин, СПбПУ. – URL: https://elib.spbstu.ru/dl/3/2021/vr/vr21-793.pdf/info (дата обращения: 21.03.2023).

4. Introduction to Cisco IOS NetFlow - A Technical Overview // Cisco. – URL: https://www.cisco.com/c/en/us/products/collateral/ios-nx-os-software/ios-netflow/prod_white_paper0900aecd80406232.html (дата обращения: 09.01.2022).
5. IP Flow Information Export (IPFIX) Entities // Internet Assigned Numbers Authority – URL: https://www.iana.org/assignments/ipfix/ipfix.xhtml (дата обращения: 09.01.2022).
6. sFlow Version 5 // sFlow – URL: https://sflow.org/sflow_version_5.txt (дата обращения: 09.01.2022).
7. Overview of NetStream // Huawei. – URL: https://support.huawei.com/enterprise/en/doc/EDOC1000178174/986bf11e/overview-of-netstream (дата обращения: 09.01.2022).
8. Documentation // Apache Kafka https://kafka.apache.org/documentation/ (дата обращения: 09.01.2022).
9. What is the ELK Stack? // Elasticsearch B.V. – URL: https://www.elastic.co/what-is/elk-stack (дата обращения: 09.01.2022)
10. Efros Config Inspector // Газинформсервис. – URL: https://www.gaz-is.ru/produkty/zashchita-it-infrastrukturi/efros-config-inspector.html (дата обращения: 09.01.2022).

УДК 004.455.2

А. А. Афонина (1 курс магистратуры),
А. Е. Кузькин,
И. В. Никифоров, к. т. н., доцент

АЛГОРИТМИЗАЦИЯ КОНФИГУРАЦИЙ ОБОРУДОВАНИЯ В ЦОД КАК БАЗИС ДЛЯ АВТОМАТИЗАЦИИ ОБНОВЛЕНИЯ ОБОРУДОВАНИЯ

Превалирующей тенденцией настоящего времени является курс на полную автоматизацию всех технологических процессов предприятия. На фоне этого важным аспектом развития любой технологической области становится своевременное масштабирование ИТ-инфраструктуры, благодаря которой становится возможен стабильный рост организации и удержание позиций на рынке.

Базисом ИТ-инфраструктуры любого учреждения в свою очередь стали центры хранения и обработки данных. Обширный технологический парк, необходимый для бесперебойной работы ЦОД, системы обеспечения безопасности и конечно сами системы хранения данных и серверные установки стали объектами, описываемыми множеством метрик, на основе которых и строятся модели конфигураций ЦОД[1]. Подбор необходимых метрик, их интерпретация и построение взаимосвязей стоят у основания построения системы, которая позволит ускорить и облегчить подбор функционального оборудования для масштабирования ЦОД и оптимизации их использования [2].

Основной интерес в призме современных исследований представляет улучшение показателей систем ЦОД в рамках отдельных метрик. Первоочередными для бизнеса являются TCO (Total Cost of Ownership) и TCS (Total Cost of Services). Для ЦОД основной коррелирующей с TCS метрикой логично становится SCE (Server Compute Efficiency), так как эффективность использования вычислительных ресурсов серверного оборудования напрямую влияет на оправданность закупки новых машин и поддержания работы ЦОД в целом [3]. В целом оптимизация работы ЦОД невозможна без определения и описания характеристик, к достижению которых исследователи стремятся в рамках своих работ. Поэтому важен расчет метрик для эффективного использования ресурсов ЦОД, и особенно калибровка конфигураций на основании данных системы мониторинга [4].

Ансамбль решений по улучшению работы отдельно вычислительного оборудования или систем хранения данных постоянно расширяется, предлагаются новые архитектуры, объединяющие в себе известные инструменты, используемые в новом ключе. Так, например, системы хранения данных с программным обеспечением на основе OAIS (Open Archival

Information System) были стеснены в плане горизонтального масштабирования. Использование HDFS (Hadoop Distributed File System), а также месседж-брокеров, дало возможность в полной мере пользоваться преимуществами распределенной архитектуры [5].

В результате проведенного исследования можно сделать вывод, что несмотря на обширный охват задач по достижению в ЦОД оптимального значения отдельных показателей, верхнеуровневое решение по обновлению отдельных машин и конфигурационных блоков в ЦОД все еще целиком и полностью ложится на эксперта[6]. В данном подходе, традиционном для индустрии, есть большой недостаток – при обширном парке оборудования часть может выпасть из поля зрения из-за отсутствия инцидентов, малой загруженности и просто под воздействием человеческого фактора.

Логичным развитием процессов в крупных компаниях становится автоматизация отдельных задач [7]. В работе предлагается отличительный подход – решение проблемы составления рекомендаций к обновлению конфигураций ЦОД через программную агрегацию всех доступных информационных ресурсов. Предлагается объединить разрозненные данные, уже собираемые и хранимые с помощью различных систем, и свести их в единый отчет, при этом определив «вес» каждой рассматриваемой метрики в контексте ее влияния на ожидаемую бесперебойность работы вычислительного оборудования.

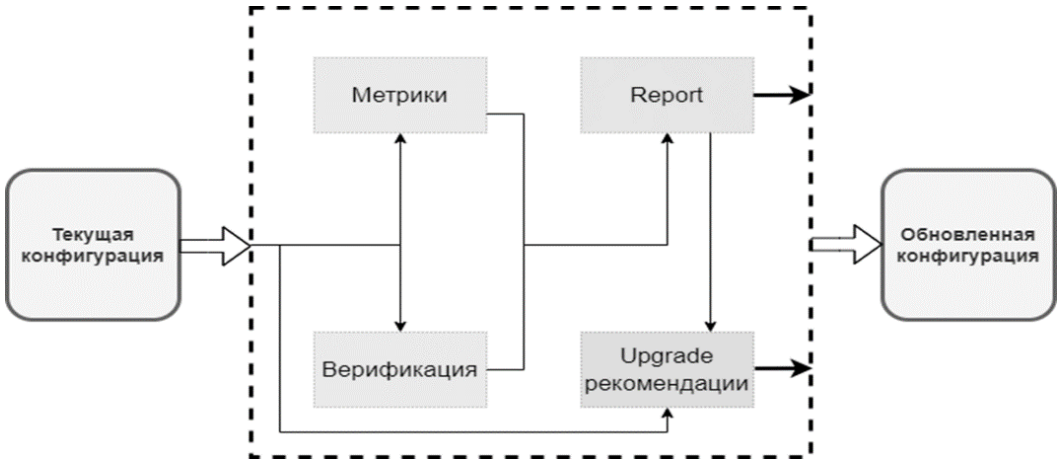


Рисунок 1 – Схема алгоритма по обновлению конфигурации

Для достижения поставленной цели – создания отчета-рекомендации по обновлению конфигурации серверов и систем хранения данных – нужно решить несколько важных задач. Первая – выделить полный список метрик, по которым будут оцениваться единицы оборудования, на основании обзора существующих систем мониторинга и конфигурационных инструментов составить минимальный список метрик, на основании которого разрабатываемый алгоритм сможет работать. Метрики необходимо наделить определенными весами, рассчитать коэффициенты для различных комбинаций метрик. При этом необходимо разделить работу с серверами и системами хранения данных, так как для них определяющее значение имеют разные метрики. Текущую конфигурацию вычислительного оборудования и СХД подается на обработку в качестве входных данных, в результате работы алгоритма выдается карта здоровья компонентов конфигурации, так называемый репорт-файл, и рекомендации по апгрейду системы. Окончательным результатом станет обновленная конфигурация вычислительного оборудования и СХД, учитывающая специфику использования машин в конкретной компании.

На основании предложенного подхода ожидается, что время, затрачиваемое специалистом на составление новой конфигурации, значительно сократится, так как рекомендации к замене оборудования будут собираться по всему объему вычислительных машин и систем хранения данных. Также внедрение алгоритма оптимизирует материальные ресурсы, затрачиваемые на закупку нового оборудования, позволив подбирать успешно решающие задачи компании модели вместо проблемных машин. Это не только даст

возможность говорить о приближении TCO к минимально возможным значениям для серверов и СХД, но и уменьшить TCS за счет высвобождения времени высококлассных специалистов.

ЛИТЕРАТУРА

1. J. Meden, F. Stuardo and B. Barán: Multi-objective Optimization for a Commercial Datacenter in Paraguay, 2019 XLV Latin American Computing Conference (CLEI), pp. 1-11, Panama (2019).
2. G. C. Deka: Cost-Benefit Analysis of Datacenter Consolidation Using Virtualization. in IT Professional, vol. 16, no. 6, pp. 54-62, (2014)
3. Denisenko B., Tyanutov M., Nikiforov I., Ustinov S., "Algorithm for calculating TCO and SCE metrics to assess the efficiency of using a data center," Proc. SPIE 12564, 2nd International Conference on Computer Applications for Management and Sustainable Development of Production and Industry (CMSD-II-2022), 1256403 (5 January 2023); doi:10.1117/12.2669285
4. Лялин Д. С., Мамадалиев Ш. Р., Никифоров И. В., Устинов С. М. Проектирование системы мониторинга и расчета метрик для эффективного использования ресурсов центров обработки данных // Современные технологии в теории и практике программирования : Сборник материалов конференции, Санкт-Петербург, 26 апреля 2022 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2022. – С. 194-197.
5. Voinov N., Drobintsev P., Kotlyarov V., Nikiforov I. Distributed OAIS-Based digital preservation system with HDFS technology // 20th Conference of Open Innovations Association FRUCT: Proceedings, Saint-Petersburg, 03–07 апреля 2017 года / LETI University, St.Petersburg, Russia; S. Balandin, A. Levina, T. Tyutina. – Saint-Petersburg: FRUCT Oy, 2017. – P. 491-497. – DOI 10.23919/FRUCT.2017.8071353.
6. V. Dinesh Reddy, B. Setz, G. Subrahmanya V.R.K. Rao, G. R. Gangadharan and M. Aiello: Best Practices for Sustainable Datacenters". in IT Professional, vol. 20, no. 5, pp. 57-67 (2018)
7. A. A. Khan, M. Zakarya, R. Buyya, R. Khan, M. Khan and O. Rana: An Energy and Performance Aware Consolidation Technique for Containerized Datacenters. In IEEE Transactions on Cloud Computing, vol. 9, no. 4, pp. 1305-1322, (2021).

УДК 004.04

М.И. Бирюкова (4 курс бакалавриата),
А.П. Маслаков, ст. преподаватель

РАЗРАБОТКА ВЕБ-ПРИЛОЖЕНИЯ ДЛЯ АГРЕГИРОВАНИЯ ДАННЫХ О КНИГАХ

В наше время человек ежедневно сталкивается с огромным количеством информации, которую необходимо обработать и систематизировать для нахождения нужных сведений, что требует немалого количества времени и концентрации внимания. В таком случае может сильно помочь автоматизированная агрегация данных и представление в удобном для пользователя виде нужной информации.

Целью работы является разработка веб-приложения с интуитивно понятным пользовательским интерфейсом, которое позволяет по запросу пользователя собирать информацию о книге из нескольких источников и отображать ее на странице браузера.

В разрабатываемом приложении пользователю будет доступна следующая функциональность:

– Просмотр топа лучших книг с возможностью фильтрации по жанрам, что позволит подобрать хорошую книгу исходя из личных предпочтений;
– Поиск по названию и автору (опциональное поле) книги, позволяющее получить с различных источников информацию, такую как рейтинг, цена, аннотация книги и др.

Выбранный тип архитектуры приложения – микросервисный, является альтернативой монолитной архитектуры [1]. Он позволяет разделить выполнение общей функциональности на несколько логических частей, которые не связаны или слабо связаны между собой, что

ускоряет разработку и тестирование. Модель веб-приложения – один веб-сервер, одна модель базы данных. Является наиболее простой и в реальных проектах не используется, однако подходит для учебных проектов [2]. Также в приложении применяется кэширование на уровне сервера.

Для создания прототипа веб-интерфейса использовался графический редактор Figma, содержащий большое количество функций и позволяющий расширить возможности путем использования плагинов.

Пользовательский интерфейс приложения разработан с использованием языка разметки HTML, языка стилей CSS и языка сценариев JavaScript. Также для возможности использования сложных компонентов интерфейса были использованы фреймворк Bootstrap [5] и библиотека JQuery.

Для написания серверной части был выбран язык программирования Java и использовался фреймворк Spring [4], который часто применяется при разработке веб-приложений. Он удобен благодаря модульной структуре, что позволяет подключать только необходимые модули. Для хранения информации в одном из микросервисов используется одна из самых популярных встраиваемых баз данных [3] – H2. Для сбора информации используется фреймворк Selenium. Также в ходе разработки использовались следующие библиотеки:

– Lombok, позволяющая уменьшить количество шаблонного кода;
– Jsoup для парсинга HTML-страниц.

Для парсинга топа книг применяется фреймворк Selenium, позволяющий с помощью веб-драйвера запускать браузер, переходить на нужный url и искать данные на странице по их атрибутам (теги, классы, идентификаторы и другие). Собранная информация сохраняется в базу данных и обновляется по расписанию.

При поиске книги для сбора необходимой информации производятся запросы к выбранным ресурсам (Book24, Bookvoed, Goodreads, Labirint, Livelib) с параметрами, введенными пользователем. Каждый ресурс возвращает в ответ HTML-страницу, путем парсинга которых приложение выделяет характеристики книги. Все полученные данные компонуются в объект и возвращаются сервером в качестве ответа.

Клиент и сервер обмениваются информацией в формате JSON по протоколу HTTP. Такой формат наиболее удобен, он широко распространен и средства для его обработки присутствуют во многих языках программирования.

Весь запланированный функционал приложения реализован и протестирован. Пользователи получили возможность искать интересующую их информацию с разных сайтов в одном месте, затрачивая на это значительно меньше времени и сил.

Функционал приложения может быть расширен путем увеличения количества источников информации, добавления в список собираемых сведений новых характеристик. В данном направлении планируется продолжать работу над программным продуктом.

ЛИТЕРАТУРА

1. Архитектура приложений [Электронный ресурс]. Режим доступа: Архитектура приложений (heaad.ru) (Дата обращения: 27.11.2022)
2. What is Web Application Architecture? Components, Models, and Types [Электронный ресурс]. Режим доступа: What is Web Application Architecture? Components, Models, and Types (hackr.io) (Дата обращения: 27.11.2022)
3. Встраиваемые базы данных в Java [Электронный ресурс]. Режим доступа: Встраиваемые базы данных в Java (mywebcenter.ru) (Дата обращения: 01.12.2022)
4. Spring Framework Documentation [Документация]. Режим доступа: Spring Framework Documentation (Дата обращения: 01.12.2022)
5. Bootstrap Multiselect [Электронный ресурс]. Режим доступа: Bootstrap Multiselect (davidstutz.github.io) (Дата обращения: 29.01.2023)

УДК 004.9

Е. Ю. Буровников (4 курс бакалавриата),
Д.С. Эйзенах (2 курс аспирантуры),
А. П. Маслаков, ст. преподаватель

РАЗРАБОТКА ПОДХОДОВ ДЛЯ АНАЛИЗА КАЧЕСТВА АВТОМАТИЗИРОВАННЫХ ТЕСТОВ ДЛЯ ПРОЕКТА ОДНОКЛАССНИКИ

Качества кода – это одна из важнейших характеристик для любого программного проекта. Однако именно этот показатель нередко страдает при автотестировании. Происходит это по целому ряду причин:

1. Не уделение процессу автотестирования должного внимания. При недостаточной поддержке любые, даже самые хорошие, автотесты, могут стагнировать, особенно при активном изменении проекта.
2. Каждый QA-инженер пишет тесты по-своему. Когда в совместной разработке участвуют много тестировщиков, то каждый может в своем стиле разрабатывать тесты, что приводит не однородности проекта. Разные его части могут сильно отличаться друг от друга, что часто вызывает непонимание или даже вводит в ступор.
3. Огромное количество тестовых сценариев. Когда большой проект развивается без каких-либо правил, то код, написанные в плохом стиле может быстро распространиться, оказывать негативное влияние на другие части проекта, вызывать непонимание у QA-инженеров [1].

Со всеми этими проблемами, работая над проектами mob и web тестов компании «Одноклассники», мне приходится сталкиваться почти каждый день. Обычно самый простой способ борьбы с такими трудностями – это ревью кода [2]. Но он обладает одним огромным минусом, ведь в нем все завязано, в первую очередь, на человеке.

Чтобы минимизировать человеческий фактор, сэкономить время, ресурсы и силы инженеров используются статические анализаторы кода – линтеры, которые содержат собственные проверки и позволяют описывать собственные случаи. Конечно, подобные инструменты не спасут от ревью, но могут помочь значительно упростить его и избежать большого количества ошибок. Статистика доказывает, что на протяжении всего жизненного цикла разработки программного обеспечения от 30% до 70% дефектов проектирования логики кода и кодирования могут быть обнаружены и устранены с помощью статического анализа кода. [3].

Так как наш проект web и mob тестов написан на Java и обладает своей спецификой, то лучшим среди других линтеров был выделен PMD.

PMD – это инструмент статического анализа Java-кода с открытым исходным кодом, опубликованный DARPA на SourceForge. PMD выполняет статические проверки кода Java с помощью своих встроенных правил кодирования, обладающих опцией настройки [4].

Для добавления проект инструментов статического анализа и повышения кодовой базы предварительно были сформулированы и решены следующие задачи:

1. Изучение автоматизации различных видов тестирования.
2. Изучение кодовой базы тестов для web и mob сред.
3. Изучение существующих подходов анализа качества автоматизированных тестов.
4. Формулировка критериев качества для web и mob тестов проекта «Одноклассники» и их воплощение в виде правил для линтера.
5. Настройка работы линтера.
6. Проверка работы линтера и сбор статистики о качестве работы: количество комментариев в ревью, время ревью.

В начале работы, проанализировав все небольшое количество правил, что уже встроены в PMD, было принято решение использовать их, доработать некоторые из них с учетом специфики проекта, а также добавить собственные. Например, линтер позволил мне добавить

требование не формировать локатор в методе класса страницы с явной ссылкой использовать паттерн «обертка» (wrapper) – очень частая ошибка, особенно начинающих инженеров.

Архитектурное устройство правил линтера было сделана таким образом, чтобы все они имели некоторое логическое разделение. Правила были поделены на несколько разделов: предупреждения, правила стиля кода, архитектурные проблемы, инструментальные.

Первые запуски линтера на отдельных частях проекта показали, что он дает много информации о состоянии кодовой базы, однако, формат отчетов был сложен для чтения, а запуск PMD можно было сделать только локально.

Для решения первой проблемы пришлось поработать со свойствами и способами отчетов в PMD, явно указав формат, который был бы нам удобен. Для исправления другой пришлось создать специальные конфигурации (так называемые jobs) в TeamCity [5], позволяющие после сборки проекта запускать статический анализатор для каждого изменения (на каждый commit), при условии наличия файла конфигурации.

В итоге, был получен целый свод правил для PMD, содержащийся в xml файлах, благодаря которым инструмент может формировать удобные даже для начинающего тестировщика отчеты. Результат же работы линтера учитывается все время жизни запроса на добавление изменений в проект без каких-то дополнительных усилий для его запуска. Он отображается в «Stash» [6], в разделе понимания кода, а также запрещает внесение изменений в главную (master) ветку при нарушении правил высшего приоритета. Для того чтобы эффективно использовать практику статического анализа кода для других проектов было принято решение сделать образ со всеми утилитами и настройками. Все это позволило действительно повысить качество кода для тестирования нашего продукта.

ЛИТЕРАТУРА

1. Анализ и использование инструментов статического анализа кода Java. [Электронный ресурс] URL: https://russianblogs.com/article/54361466259 (дата обращения:19.03.2023).
2. Как повысить качество кода в тестовом проекте. [Электронный ресурс] URL: https://habr.com/ru/company/wrike/blog/574320/ (дата обращения:19.03.2023).
3. Анализ и сравнение часто используемых инструментов статического анализа кода Java. [Электронный ресурс] URL: https://russianblogs.com/article/5695807208/ (дата обращения:19.03.2023).
4. PMD. [Электронный ресурс] URL: https://pmd.github.io/ (дата обращения:19.03.2023).
5. Мощная непрерывная интеграция для команд, ориентированных на DevOps. [Электронный ресурс] URl: https://www.jetbrains.com/teamcity/ (дата обращения:19.03.2023).
6. Bitbucket. [Электронный ресурс] URL: https://www.atlassian.com/ru/software/bitbucket (дата обращения:19.03.2023).

УДК 004.45

М. Д. Васильев (2 курс магистратуры),
А. В. Самочадин, к.т.н., доцент

ПОРТИРОВАНИЕ MESA3D НА ЗОСРВ «НЕЙТРИНО»

В настоящее время в каждом компьютере есть графический ускоритель. Данной устройство состоит из множества небольших вычислительных блоков, что позволяет ему очень быстро исполнять многопоточные программы. Для того, чтобы использовать их мощность в пользовательских приложениях существуют такие стандарты как OpenCL [1] и OpenGL [2].

OpenGL - спецификация, определяющая платформо-независимый программный интерфейс для отрисовки двумерной и трёхмерной графики с использованием аппаратного ускорения.

OpenCL – это спецификация для написания программ, связанных с параллельными вычислениями на различных графических и центральных процессорах.

Наиболее распространённой открытой реализацией графического программного интерфейса OpenGL/OpenCL является Mesa3D [3]. В связи с тем, что она разрабатывалась для операционных систем семейства Linux/BSD, её полноценная работа на других операционных системах невозможна. Целью работы является портирование Mesa3D версии 22 на российскую защищённую операционную систему реального времени «Нейтрино» (ЗОСРВ «Нейтрино»).

Для достижения этой цели были поставлены следующие задачи:

1. Анализ зависимостей Mesa3D и подготовка среды сборки
2. Анализ особенностей ЗОСРВ «Нейтрино»
3. Разработка интерфейсов для работы Mesa3D в ЗОСРВ «Нейтрино»
4. Сборка
5. Тестирование

Аналогичный подход портирования использовался в работе [4].

Сборка Mesa3D производилась в docker-контейнере на основании образа Ubuntu 20.04 LTS с установленным комплектом для разработки программного обеспечения для ЗОСРВ «Нейтрино». Преимущества такого подхода состоит в том, что не требуется устанавливать вручную весь необходимый инструментарий и настраивать среду окружения для сборки под ОС, достаточно использовать только образ. Такой же подход к сборке используется в работах [5, 6].

При анализе зависимостей Mesa3D были выделены компоненты, имеющиеся в ОС, но нуждающиеся в обновлении:

– LLVM – это библиотека, предоставляющая аппаратно-независимый оптимизатор кода;
– Clang – это компилятор C/C++, использующий LLVM;
– libclc – это реализация спецификации OpenCL.

Данные компоненты являются частью одного проекта LLVM [7], который был обновлен вместе до 13 версии, которая позволяет в полной мере использовать Mesa3D. Его обновление заключалось в загрузке обновлений из официального репозитория проекта и пересборке.

Mesa3D имеет модульную архитектуру, предоставляющую множеству интерфейсов для создания разных драйверов для видеоускорителей, графических систем, графических программных инструментов и т.д. ЗОСРВ «Нейтрино» имеет отличную от Linux графическую подсистему, поэтому были разработаны модули для работы с ней. Один взаимодействует с графической средой ОС посредствам библиотеки Graphics Framework. Второй соединяет драйвера для видеоустройств и графический интерфейс. Помимо самих модулей потребовалось править внутренний код Mesa3D, чтобы другие компоненты знали, как к ним обращаться, чтобы адаптировать работу под интерфейсы ЗОСРВ «Нейтрино», чтобы она могла работать на таких архитектурах процессоров, работа на которых не предусматривалась разработчиками. Например, «Эльбрус» и PowerPC с обратным порядком байт.

Сборка происходила из нескольких этапов: встраивание Mesa3D в рекурсивную систему сборки ЗОСРВ «Нейтрино», сборка Mesa3D.

Рекурсивная сборочная подсистема ЗОСРВ «Нейтрино» [8] является расширением инструмента GNU Make и позволяет автоматизировать процесс сборки ПО под разные вычислительные платформы. Для интеграции её в проект создаётся дерево каталогов, уровни которого задают разные параметры целевой системы (ОС, архитектура процессора и т.д.). Каждый уровень каталогов содержит Makefile, который содержит правила для определения параметров системы и перехода к следующему уровню. На самом нижнем уровне подключается файл с правилами окончательной конфигурацией и запуска сборки непосредственно проекта.

Сборка Mesa3D происходит с помощью системы Meson. В ней используются meson-скрипты, которые содержат правила для компиляции и компоновки проекта. Сборочные скрипты Mesa3D были дополнены правилами необходимыми для сборки разработанных модулей.

Тестирование OpenGL проводилось на приложения, использующих как 2D, так и 3D графику со сложной геометрией. Тестовые приложения проверяли разные аспекты OpenGL, например, работа с шейдерами, с текстурами, с освещением, с матрицами, проверка работоспособности OpenGL первой версии. Тестирование производилось на нескольких компьютерах с разными архитектурами процессоров (x86, PowerPC, «Эльбрус») и видеоускорителями.

Таким образом, анализ зависимостей, системы сборки, архитектуры и кода, а также изучение особенностей ОС позволили успешно портировать Mesa3D. Тестирование подтвердило ее полную работоспособность.

ЛИТЕРАТУРА

1. Khronos OpenCL Registry. [Электронный ресурс]. Режим доступа: https://registry.khronos.org/OpenCL/
2. Khronos OpenGL Registry. [Электронный ресурс]. Режим доступа: https://registry.khronos.org/OpenGL/index_gl.php
3. Щёкин С. В., Фаттахова М. В. ОСОБЕННОСТИ ЭВОЛЮЦИИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ РАБОТЫ С ТРЕХМЕРНОЙ ГРАФИКОЙ НА ОСНОВЕ ОТКРЫТЫХ ИСХОДНЫХ ТЕКСТОВ //Обработка, передача и защита информации в компьютерных системах'22. – 2022. – С. 127-132.
4. Тачков А.А,. Вуколов А.Ю., Козов А.В. Особенности портирования Robot Operating System на программно-аппаратную платформу семейства «Эльбрус» // Программные продукты и системы. 2019. Т. 32. № 4. С. 655–664. DOI: 10.15827/0236-235X.128.655-664.
5. Маркелов К. Д. СБОРКА МОБИЛЬНЫХ ПРИЛОЖЕНИЙ ДЛЯ ОПЕРАЦИОННОЙ СИСТЕМЫ ANDROID С ПОМОЩЬЮ DOCKER И GRADLE //НАУЧНЫЙ ЛИДЕР Учредители: Общество с ограниченной ответственностью Международный издательский дом" ВОРЛДСАЙПАБЛ". – 2022. – №. 23. – С. 21-23.\
6. Забелин К. В. и др. Автоматизация сборки и развертывания системы анализа данных электронной библиотеки СПБПУ //Современные технологии в теории и практике программирования. – 2021. – С. 207-209.
7. The LLVM Compiler Infrastructure. [Электронный ресурс]. Режим доступа: https://llvm.org/
8. Рекурсивная сборочная подсистема ЗОСРВ "Нейтрино". [Электронный ресурс]. Режим доступа: https://help.kpda.ru/help/topic/ru.kpda.doc.dev_tools_ru/html/environment/mk.html?cp=2_3_1

УДК 004.942

Д. А. Вихляев, А. М. Сабуткевич (2 курс магистратуры),
И. В. Никифоров, к.т.н., доцент,
А. В. Самочадин, к.т.н., доцент

ПРИМЕНЕНИЕ МЕТОДОВ МАШИННОГО ОБУЧЕНИЯ ДЛЯ ПОСТРОЕНИЯ МОДЕЛЕЙ ПОВЕДЕНИЯ МНОГОАГЕНТНЫХ СИСТЕМ И ИХ АНАЛИЗА

Построение моделей имитационного моделирования поведения многоагентных систем [1] - сложный и трудоемкий процесс, требующий от эксперта серьезных когнитивных способностей и обработки больших массивов информации. В дополнении к этому зачастую информация, на основе которой строится модель, является неструктурированной, плохо структурированной, неточной и недостоверной [2], что приводит к неточности модели и огромным накладным затратам по ее актуализации. Автоматизация процесса построения модели поведения и анализа результатов моделирования обеспечивает снижение нагрузки на эксперта, требований к его компетенциям, а в случае моделирования критически-важных процессов [3] позволяет уменьшить время реагирования.

Целью данной работы является рассмотрение возможности применения методов машинного обучения на разных этапах разработки моделей и анализа результатов имитационного моделирования поведения многоагентных систем.

В рамках исследования данной тематики были выделены следующие этапы работы эксперты, которые могут быть частично автоматизированы:

- построение модели поведения;
- актуализация модели поведения;
- анализ результатов имитационного моделирования.

Обозначим структуру модели поведения агентов как математический ориентированный граф, узлами которого являются классы состояния, в которых может находиться агент, а дугами - действия, выполнение агентом которых может осуществить его переход из одного класса состояния в другой. Сам агент характеризуется его текущим классом состояния, действием, которое он выполняет, и набором атрибутов, которые определяют возможность выполнения действий и нахождения агента в определенном классе состояния. В качестве исходной информации, на основе которой строится модель, используются снимки агентов, включающие значения атрибутов агента в определенный момент времени и выполняемое им действие.

Во всех рассматриваемых подходах обработка значений атрибутов агента происходит путем применения ассоциативной классификации [4], на основе которой строятся зависимости и сравниваются атрибуты.

Автоматизацию построения модели поведения агента можно осуществить в 2 шага: выделение классов состояния и построения переходов между ними. Для выделения классов состояния применяется модифицированный алгоритм нечеткой кластеризации c-means [5] на основе значений атрибутов агента. Построение переходов между классами состояния, которые выражаются действиями, происходит путем обработки выполняемых агентом действий. Все действия на основе временных меток разбиваются на две категории: промежуточные и конечные. Действия, отнесенные к категории конечных, не участвуют в построении переходов. Каждое промежуточное действие является переходом между классами состояния и для каждого из них устанавливается исходный класс состояния на основе значений атрибутов агента и конечный класс состояния на основе временных меток «соседних» действий.

В случае, когда существующую модель необходимо актуализировать на основе новых снимков, возможно применение одного из двух подходов: полное перестроение модели и реальное дополнение существующей модели. Для перестроения модели инструментом имитационного моделирования [6] восстанавливаются (в случае, когда модель построена на их основе) или создаются снимки агентов. Полученные данные дополняются новыми снимками, и на их основе происходит описанное выше построение модели. Если же необходимо дополнить модель без ее перестроения, по значениям атрибутов агента из новых снимков определяется класс состояния существующей модели (если ни один из классов состояния не подходит, создается новый) и строятся новые переходы по алгоритму, описанному в подходе автоматизации построения модели поведения агента.

Результаты имитационного моделирования представляют из себя различные траектории поведения агента, которые необходимо правильно проанализировать. Одним из важнейших результатов анализа является определение текущего состояния агента и наиболее вероятного его поведения на основе частичной информации о нем (его атрибутах). Для решения этой задачи можно применить ассоциативную классификацию, которая позволяет определить наиболее вероятное состояние агента и, как следствие, его дальнейшее поведение. Использование на данном этапе методов машинного обучения позволяет сократить время реагирования.

Хоть и описанные подходы позволят только частично автоматизировать процесс построения модели и анализа результатов имитационного моделирования, они многократно

снизят нагрузку на эксперта, повысят точность построенных моделей и сократят время реагирования при моделировании критически-важных систем [3].

ЛИТЕРАТУРА

1. Городецкий В. И. Сети автономных агентов реального времени в среде с противодействием: особенности и компоненты модели / В. И. Городецкий, М. Г. Пантелеев // Материалы конференции «Информационные технологии в управлении» (Санкт-Петербург, 06–08 октября 2020 года) – Санкт-Петербург, 2020. – С.22–31.
2. J. Lu, X. Liu, Y. Feng and X. Lin, "Simulation and Analysis of Community Energy Consumption Based on Multi-agent Modeling," 2020 IEEE 4th Conference on Energy Internet and Energy System Integration (EI2), Wuhan, China, 2020, pp. 1280-1284, doi: 10.1109/EI250167.2020.9346935.
3. Костин В. Н. Методики, модели и методы обоснования и разработки систем физической защиты критически важных объектов: дис. ... д-ра тех. наук. Оренбургский государственный университет, Оренбург, 2021 – 249 с.
4. Городецкий В. И., Тушканова О. Н. Ассоциативная классификация: аналитический обзор. Часть 1 // Труды СПИИРАН. 2015. №1. С. 183–203.
5. A. Heni, I. Jdey and H. Ltifi, "k-means and fuzzy c-means fusion for objectclustering," in 8th InternationalConference on Control, Decision andInformation Technologies, 2022.
6. Сабуткевич А. М., Вихляев Д. А., Никифоров И. В., Самочадин А. В. Имитационное моделирование поведения сложных многоагентных систем с использованием вероятностной модели // Материалы конференции «Современные технологии в теории и практике программирования», (Санкт-Петербург, 26 апреля 2022 года) – Санкт-Петербург, 2022. – С. 98-100.

УДК 004.4'2

С. Б. Габдуллина (4 курс бакалавриата),
И. В. Шошмина, к.т.н., доцент

РАЗРАБОТКА ПЛАГИНА МИГРАЦИИ КОДА МОБИЛЬНЫХ ПРИЛОЖЕНИЙ С SHARED PREFERENCES НА JETPACK DATASTORE

Мобильные приложения сохраняют информацию о пользователях и их действиях. Сохранение может выполняться либо на самом устройстве, либо на внешних серверах. В работе рассматривается хранение данных на мобильном устройстве. Такой способ хранения применяется в следующих случаях:

- приложение не имеет доступа к интернету,
- приложение кэширует данные, чтобы использовать их, когда интернет отсутствует,
- приложение сохраняет настройки, такие как тема, язык, версия.

Для сохранения данных внутри устройства нами рассматривается хранилище настроек. Хранилище настроек представляет собой постоянное хранилище, используемое приложением, для сохранения некоторых флагов, номера сессии, номера последней версии и других его настроек. Данные внутри хранилища записываются как пары ключ-значение в формате XML [1].

Стандартным программный средством взаимодействия с хранилищем настроек является интерфейс Shared Preferences. Он активно используется разработчиками на протяжении последних 15 лет. Однако, за время использования технологии Shared Preferences были выявлены следующие недостатки:

- операции записи выполняются синхронно, в рамках 1-го процесса, что не является безопасным на потоке пользовательского интерфейса и может привести к блокировке взаимодействия пользователя с приложением.
- нет поддержки механизма отлавливания и обработки ошибок, в том числе ошибок во время работы приложения, что также может препятствовать взаимодействию пользователя с приложением.

– не гарантируется консистентность данных пользователя. Консистентность данных подразумевает согласованность данных друг с другом, целостность данных, а также их внутреннюю непротиворечивость. Из-за отсутствия гарантий при чтении данных одновременно из разных частей приложения пользователь может получить разные данные.
– типы данных не закрепляются строго, все данные сохраняются в виде строк в файле XML формата.

Несколько лет назад появилось новое программное решение для сохранения данных на мобильных устройствах – Jetpack DataStore [2]. Применяя библиотеку Jetpack DataStore, разработчики сделают сохранение данных надежным для пользователя и типобезопасным.

На данный момент не создано механизма для автоматической миграции кода с Shared Preferences на Jetpack DataStore, поэтому программистам приходится выполнять такую миграцию каждый раз вручную.

Цель работы – разработать программное средство автоматической миграции кода приложений, сохраняемого на мобильных устройствах платформы Android, чтобы заменить использование технологии Shared Preferences на Jetpack DataStore. Из-за широкой распространенности Shared Preferences программное средство автоматической миграции будет применимо практически во всех приложениях, сэкономит человеческие ресурсы IT компаний, при этом улучшив работу приложений.

Основные задачи работы:

– разработка анализатора для поиска кода с библиотекой Shared Preferences
– генерация файлов с методами чтения и записи с помощью Jetpack DataStore
– разработка алгоритма модификации кода с Shared Preferences на код с Jetpack DataStore
– реализация строгой типизации сохраняемых данных
– тестирование результатов миграции приложений

Работа анализатора строится на последовательном переборе всех файлов, содержащих код на языке Kotlin. Для анализа файла используется PSI (Program Structure Interface) – интерфейс, с помощью которого содержимое файла транслируется в иерархию элементов [3]. После получения иерархии элементов выполняется спуск по этому дереву и поиск выражений, в которых используется методы интерфейса Shared Preferences.

В результате работы анализатора получается список со всеми названиями файлов настроек и хранимыми типами данных. На основе этой информации генерируются файлы с кодом объектов, инкапсулирующих внутри себя взаимодействие с библиотекой Jetpack DataStore. Создание этих файлов выполняется с помощью библиотеки FreeMaker по шаблонам, написанным на языке FreeMarker Template Language (FTL) [4].

После генерации выполняется поиск элемента, который нужно изменить для работы с технологией Jetpack DataStore. После найденный элемент или его части заменяются на новые PSI элементы в соответствии с правилами миграции и информацией, полученной после работы анализатора. Привязка к структуре PSI необходимо для корректного синтаксического распознавания кода средами разработки после миграции.

Для реализации программного средства выбрана среда разработки IntelliJ IDEA и язык программирования Kotlin. Программное средство представляет из себя плагин [5], встраиваемый в среду разработки Android Studio, которая является официальным средством разработки Android приложений. Плагин будет доступен для всех разработчиков под платформу Android. Графический интерфейс плагина сконструирован с использованием библиотеки Swing [6], обладающей богатым и удобным набором интерфейсных примитивов.

ЛИТЕРАТУРА

1. Как хранить данные приложения android локально? [Электронный ресурс] Режим доступа: https://aposnov.ru/как-хранить-данные-приложения-android-локал/

2. Modern data storage on Android: Meet Jetpack DataStore — Part 1/2. [Электронный ресурс] Режим доступа: https://levelup.gitconnected.com/modern-data-storage-on-android-meet-jetpack-datastore-part-1-2-9f314c994fc8
3. PSI Files // IntelliJ Platform Plugin SDK. [Электронный ресурс] Режим доступа: https://plugins.jetbrains.com/docs/intellij/psi-files.html?from=jetbrains.org
4. Template Language Reference. [Электронный ресурс] Режим доступа: https://freemarker.apache.org/docs/ref.html
5. Фантастические плагины, vol. 1. Теория. [Электронный ресурс] Режим доступа: https://habr.com/ru/company/hh/blog/463583/
6. Lesson: Using Swing Components. [Электронный ресурс] Режим доступа: https://docs.oracle.com/javase/tutorial/uiswing/components/index.html

УДК 004.415.2

В. В. Гаврилова (2 курс магистратуры),
Б. М. Медведев, к.т.н., доцент

РАЗРАБОТКА ПРОГРАММНЫХ СРЕДСТВ ТЕСТИРОВАНИЯ ВСТРАИВАЕМЫХ СИСТЕМ ОБРАБОТКИ ИНФОРМАЦИИ

Встраиваемые системы обработки информации становятся все более популярными в самых разных областях: от аэрокосмической и автомобильной промышленности до бытовой электроники и здравоохранения. Эти системы предназначены для сбора и обработки данных, а также для управления устройствами в режиме реального времени [1]. По мере увеличения сложности и функциональности этих систем растет и потребность в надежных и эффективных методах и инструментах тестирования.

Разрабатываемая система обработки информации (СОИ) содержит вычислительные модули (ядра ARM, программируемая логика), DDR память и модули периферийных устройств (АЦП, ЦАП, интерфейсы SATA, PCIE, USB, Ethernet). На этапе изготовления опытных образцов новых устройств СОИ прежде всего необходимо определить работоспособность модулей, так как устройство может содержать ошибки проектирования (например, некорректное соединение сигналов интерфейсов модулей) и ошибки изготовления (например, отсутствие пайки контакта элемента). Тестирование работоспособности модулей разделено на два этапа. На первом этапе проверяется функционирование ядер ARM, памяти DDR и коммуникационных модулей USB и Ethernet путем загрузки bare metal приложений через JTAG интерфейс. На втором этапе тестирования используются: встраиваемое программное обеспечение СОИ, состоящее из ОС Linux с необходимыми драйверами и набора приложений тестирования модулей системы; а также клиентское приложение, которое позволяет управлять процессом тестирования и получать информацию о результатах. Клиентское приложение запускается на компьютере оператора или выполняется в СОИ при возможности подключения к системе монитора и клавиатуры. Такое разделение на два этапа позволяет сократить время тестирования и на втором этапе предоставляет пользователю развитый графический интерфейс со средствами диагностики и формирования протокола выполнения тестов. Для новых создаваемых СОИ со специфическим набором аппаратных модулей необходимо разработать платформенно-независимое клиентское приложение управления тестированием и приложения тестирования модулей [2].

Целью работы является разработка программных средств тестирования встраиваемых систем обработки информации на базе многоядерной ARM архитектуры, обеспечивающих диагностику работоспособности аппаратных модулей системы.

Для достижения поставленной цели необходимо решить следующие задачи:

1. Разработка протокола управления тестированием.
2. Разработка встраиваемых приложений тестирования аппаратных модулей.
3. Разработка платформенно-независимого графического интерфейса пользователя.

Протокол управления тестированием включает в себя план тестирования, тестовые наборы, сценарии тестирования и отчеты о тестировании. Требования к протоколу включают в себя обеспечение режима самотестирования СОИ и удаленного тестирования по сети, определение ошибок выполнения приложений тестирования, последовательное или параллельное выполнение тестов в однократном и циклическом режиме работы, а также обеспечение асинхронного режима опроса состояния выполнения группы тестов.

Приложения тестирования аппаратных модулей определяют работоспособность модулей и диагностику ошибок с использованием драйверов устройств.

Для разработки графического интерфейса используется язык программирования C и библиотека QT[3] с поддержкой несколько платформ, включая Linux, Windows и MacOS. Для удаленного управления тестированием СОИ был выбран коммуникационный протокол Telnet [4, 5], так как для тестирования используется выделенная локальная сеть, в которой не требуется применять шифрование сообщений. Реализация Telnet клиента разработана с использованием библиотеки libcurl [6], которая предоставляет простой в использовании интерфейс для выполнения запросов к сетевым серверам и обработки ответов.

Разработанные программные средства тестирования обеспечивают обработку ошибок на трех уровнях. Первый уровень – обработчик сообщений сервера, который определяет ошибки в формате команд управления и ошибки выполнения приложения тестирования (истечение времени ожидания ответа, отсутствие файла результата). Второй уровень обработки ошибок реализуют приложения тестирования аппаратных модулей. При обнаружении ошибки в работе тестируемого модуля приложение записывает коды ошибок и диагностику неисправности в файл, который отображается в интерфейсе пользователя по запросу. Третий уровень обработки ошибок реализует оператор, анализируя результаты измерений, для которых диапазон значений зависит от сценария тестирования. Так, например, температура микросхем определяется режимом работы модуля, наличием радиатора, скоростью вращения вентилятора (параметры могут меняться в процессе тестирования).

Важным преимуществом разработанной структуры программных средств тестирования является возможность разрабатывать и отлаживать приложения тестирования аппаратных модулей независимо друг от друга и комбинировать их в разных сочетаниях для создаваемых СОИ.

Разработанные программные средства тестирования встраиваемых систем обработки информации за счет автоматизации процесса обеспечивают экономию времени тестирования. На следующем этапе работы средства тестирования работоспособности модулей можно дополнить средствами обнаружения взаимного влияния модулей, а также средствами оценки производительности системы в различных режимах работы [7].

ЛИТЕРАТУРА

1. B. M. Medvedev, S. A. Molodyakov, S. M. Ustinov and S. A. Fyodorov, "Embedded systems software: Trends in industry and education," *2*018 International Symposium on Consumer Technologies (ISCT), St. Petersburg, Russia, 2018, pp. 66-69, doi: 10.1109/ISCE.2018.8408921.
2. Yifeng Zhu. “Embedded Systems with ARM Cortex-M Microcontrollers in Assembly Language and C: Fourth Edition” – 2022. – 730pp.
3. Библиотека Qt. [Электронный ресурс] Режим доступа: https://ru.opensuse.org/Библиотека_Qt
4. Таненбаум Э. Компьютерные сети / Э. Таненбаум, Уэзеролл Д. – 5-е изд. – СПб.: Питер, 2020. – 960 с.
5. Telnet. [Электронный ресурс] Режим доступа: https://www.opennet.ru/docs/RUS/linux_base/node372.html
6. Command line tool and library. [Электронный ресурс] Режим доступа: https://curl.se/
7. Per Erik Strandberg, Eduard Paul Enoiu, Wasif Afzal, Daniel Sundmark, Robert Feldt. “Information Flow in Software Testing – An Interview Study With Embedded Software Engineering Practitioners” – 2019 – 20pp.

УДК 004.93'12

П. А. Гессен (2 курс магистратуры),
С. Э. Сараджишвили, к.т.н, доцент

СИСТЕМА ПОВЫШЕНИЯ ХАРАКТЕРИСТИК АЛГОРИТМОВ АВТОМАТИЧЕСКОГО СОПРОВОЖДЕНИЯ ДЛЯ ОПТИКО-ЭЛЕКТРОННЫХ СИСТЕМ

В настоящее время в робототехнических системах все еще остро стоит вопрос обнаружения и сопровождения недетерминированных объектов. В данной области уже существует большой научный задел и представлено много готовых решений, и с течением времени разрабатываются все более точные и совершенные методы. Однако, основным вектором разработки новых алгоритмов является увеличение точности за счет увеличения затрачиваемых ресурсов [1,2], что влечет за собой повышение вычислительных мощностей для обеспечения работы режима реального времени, а это может быть несовместимо со встраиваемыми оптико-электронными системами в виду накладываемых на них ограничений.

При этом уже существует множество алгоритмов, успешно работающих в таких системах. Они отстают от наиболее современных в плане точности, однако значительно опережают с точки зрения скорости, при этом у них может отсутствовать возможность обнаружения цели интереса после ее потери или у них нет понимания о потере как такового [3]. Такие алгоритмы имеют потенциал для доработки — за счет запаса по скорости работы можно увеличить их точность, тем самым приближая к современным алгоритмам, не затрачивая больших аппаратных и человеческих ресурсов на улучшение. Таким образом создание отдельного модуля, обеспечивающего повышение качественных характеристик уже существующих алгоритмов, добавляющего поддержку переобнаружения цели или проверки правильности переобнаружения в случае его наличия, является актуальной задачей.

Целью данной работы является создание модуля по улучшению существующих алгоритмов на основе машинного обучения и кластеризации, который предоставляет следующий функционал: улучшение точности сопровождения путем корректировки положения описывающего прямоугольника, добавление масштабной и поворотной инвариантности, добавление повторного определения положения цели после ее кратковременной потери.

Для достижения этой цели необходимо решение следующих задач:

1. Обзор существующих подходов реализации алгоритмов сопровождения и обнаружения объектов интереса.
2. Выбор наиболее удачных подходов к решению подзадач внутри существующих подходов.
3. Предложение нового модуля по улучшению характеристик существующих алгоритмов.
4. Реализация данного модуля.
5. Демонстрация качественно-временных характеристик.

В результате проведенной работы отобраны и протестированы методы и подходы, позволяющие, сохраняя вычислительную эффективность, обеспечивать обучение в процессе работы (online learning) без априорного знания о типе объекта слежения и обеспечивать последующую корректировку во время слежения и обнаружение исходного объекта в случае его кратковременной потери.

В число таких способов входит гистограмма направленных градиентов [4] – дескриптор ключевых признаков, основывающийся на анализе распределения градиентов яркости изображения объекта. Его использование позволяет сократить количество используемой информации без потери ключевых данных об объекте и увеличить скорость обработки изображений.

Еще одним подходом является использование одного из алгоритмов классификации, позволяющего решить задачу бинарной классификации – метода опорных векторов. В виду

высокой скорости обработки данных и необходимости небольшого количества исходных обучающих данных для построения разделяющей гиперплоскости, на основе которой и происходит классификация объектов, данный метод выбран как наиболее подходящий для решения поставленной задачи. Для осуществления online-обучения была выбрана модификация метода опорных векторов, реализующая стохастический градиентный спуск на каждом шаге работы алгоритма – Pegasos [5].

Еще одним вспомогательным способом является метод кластеризации ключевых точек – таким образом обеспечивается ускоренный выбор объектов для обучения и классификации.

Тестирование проводилось при помощи полунатурного моделирования с использованием программного комплекса автоматизации тестирования алгоритмов обнаружения и сопровождения, разработанного АО «НПП «АМЭ» [6], и реальных видеозаписей, полученных в различных условиях наблюдения. Для тестирования были размечены видеопоследовательности, содержащие объекты интереса типа «Кунг», «Здание», «Мост» и др. размером от ~64x64 до ~256x256 пикселей. Улучшаемым алгоритмом выступал разработанный в АО «НПП «АМЭ» многоагентный алгоритм автоматического обнаружения и сопровождения недетерминированных объектов [7].

ЛИТЕРАТУРА

1. 1. Zhang Y., Wang L., Qi J., Wang D., Feng M., Lu H. (2018) Structured Siamese Network for Real-Time Visual Tracking. In: Ferrari V., Hebert M., Sminchisescu C., Weiss Y. (eds) Computer Vision – ECCV 2018. ECCV 2018. Lecture Notes in Computer Science, vol 11213. Springer, Cham.
2. 2. Li, D., Yu, Y. & Chen, X. Object tracking framework with Siamese network and re-detection mechanism. J Wireless Com Network, 2019, 261.
3. 3. L. Bertinetto, J. Valmadre, S. Golodetz, O. Miksik and P. H. S. Torr, "Staple: Complementary Learners for Real-Time Tracking," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 1401-1409, doi: 10.1109/CVPR.2016.156.
4. 4. Dalal N., Triggs B., Histograms of oriented gradients for human detection. 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, 2005, pp. 886-893 vol. 1, doi: 10.1109/CVPR.2005.177.
5. 5. Shalev-Shwartz, S., Singer, Y., Srebro, N. et al. “Pegasos: primal estimated sub-gradient solver for SVM”. Math. Program. 127, 2020, 3–30.
6. 6. Бондаренко В.А., Гагарина А.Ю., Павлова В.А., Тупиков В.А. Программный комплекс автоматизации тестирования алгоритмов обнаружения и сопровождения объектов на видеопоследовательностях. Перспективные системы и задачи управления: материалы XVI Всероссийской научно-практической конференции и XII молодежной школы-семинара «Управление и обработка информации в технических системах». – Таганрог: ИП Марук М.Р., 2021. – 355 с.
7. 7. Бондаренко В.А., Ельцова Д.К., Лизин А.И., Павлова В.А., Созинова М.В., Тупиков В.А. Многоагентный алгоритм автоматического обнаружения и сопровождения недетерминированных объектов. Известия ЮФУ. Технические науки, 1 (211), 2020, с. 218-232.

УДК 004.4

К. Р. Головин (4 курс бакалавриата),
О. Г. Малеев, к.т.н, доцент

МОДЕЛЬ МАШИННОГО ОБУЧЕНИЯ ДЛЯ КЛАССИФИКАЦИИ ТЕКСТОВЫХ ДАННЫХ НА НЕСКОЛЬКИХ ЯЗЫКАХ

Обработка естественного языка представляет собой область искусственного интеллекта, направленную на то, чтобы позволить машинам понимать, интерпретировать и генерировать человеческий язык. Он включает в себя такие задачи, как распознавание речи, классификация текста, анализ настроений, машинный перевод и многое другое.

Intent classification (Классификация намерений) — это процесс определения основного намерения или цели данного пользовательского ввода или запроса, как правило, в контексте обработки естественного языка (NLP) и приложений разговорного ИИ. Он включает использование алгоритмов машинного обучения или других методов для анализа и классификации текстовых или речевых данных на основе желаемого действия или цели пользователя.[1]

Для реализации этой задачи необходимо построить Word Embeddings и затем классифицировать текст в векторном пространстве [2].

Word Embeddings (Встраивание слов) — это метод обработки естественного языка, который преобразует слова в числовые векторы в многомерном пространстве, позволяя машинам более эффективно обрабатывать и понимать человеческий язык. Вложения векторов слов обеспечивают более эффективную и действенную языковую обработку, поскольку они фиксируют семантические и синтаксические отношения между словами. Например, слова с похожими значениями представлены в виде векторов, расположенных ближе друг к другу в векторном пространстве.[3]

Различные методы встраивания различаются по своей сложности и возможностям. Например, самая простая форма встраивания слов может быть представлена с помощью кодировок, где каждое слово в наборе некоторого размера V сопоставляется с уникальным индексом в векторе того же размера. Это дает нам вектор из всех нулей, кроме одного элемента, обозначающего слово.[4] Более сложные методы основаны на последних достижениях в области нейронных сетей.[5]

При работе с машинным обучением не менее важной задачей, чем построение модели, является сбор необходимых наборов данных. В данном случае для построения векторных представлений слов необходим набор текстов с параллельным переводом, для того чтобы слова с одинаковой семантикой в векторном представлении имели наибольшую близость.[6] Для классификации намерений необходим набор размеченный набор данных, где предложениям на разных языках сопоставлены соответствующие темы. К примеру запрос: “Будет ли сегодня дождь?” должен быть соотнесен к теме “Погода”

Задача классификации намерений весьма популярна и поэтому имеет множество решений, но эти решения ограничены одним языком, что является проблемой, когда механизм используется в сервисах с многоязычной аудиторией. Именно решение этой проблемы является целью этой работы.

Предлагаемая архитектура решения:

1. Блок построения векторного представления данных
   - Skipgram - метод в котором вектор слова строится на основе случайного соседнего слова. Данный метод, как правило, дает самую слабую точность, но при этом самый ресурсоемкий и быстрый.
   - Cbow ('continuous-bag-of-words') – метод, в котором вектор слова определяется его контекстом. Вектором слова при этом решении является сумма векторов соседних слов в пределах окна фиксированного размера.
   - Bidirectional Encoder Representations from Transformers (BERT) – в этом методе вектор слова определяется на основе всей фразы, за исключением этого слова. Учитывая эту специфику, при помощи технических токенов, можно получить вектор для фразы целиком. Данный подход является самым затратным по памяти и скорости, но дает куда лучший результат.

     Для обеспечивания многоязычности модель обучается строить векторы для базового языка, а потом добивается близости для близких по семантике данных добавочных языков.
2. Блок классификации намерений
   - SVM – метод машинного обучения, при котором разделяющая плоскость между двумя классами строится на основе максимального расстояния между крайними представителями (опорными векторами)

- Feedforward neural networks – многослойная нейронная сеть, которая может выявлять сложные нелинейные зависимости между классами.

В ходе работы рассматриваются различные комбинации блоков, для выявления оптимальной модели при различных условиях производства.

ЛИТЕРАТУРА

1. Qian Chen, Zhu Zhuo, Wen Wang - BERT for Joint Intent Classification and Slot Filling [Электронный ресурс] Режим доступа: https://arxiv.org/pdf/1902.10909.pdf
2. Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Herve´ Je´gou & Tomas Mikolov - FastText.zip: Compressing text classification models [Электронный ресурс] Режим доступа: https://arxiv.org/pdf/1612.03651v1.pdf
3. Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova - BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding [Электронный ресурс] Режим доступа: https://arxiv.org/pdf/1810.04805.pdf
4. Maryam Fallah - An Overview of Different Text Embedding Models [Электронный ресурс] Режим доступа: https://techblog.ezra.com/different-embedding-models-7874197dc410
5. Maryam Fallah - An Overview of Different Transformer-based Language Models [Электронный ресурс] Режим доступа: https://techblog.ezra.com/an-overview-of-different-transformer-based-language-models-c9d3adafead8
6. Goran Glavaš, Robert Litschko, Sebastian Ruder, and Ivan Vulic´ - How to (Properly) Evaluate Cross-Lingual Word Embeddings: On Strong Baselines, Comparative Analyses, and Some Misconceptions [Электронный ресурс] Режим доступа: http://aclanthology.lst.uni-saarland.de/P19-1070.pdf

УДК 004.415

Ф. А. Гырлов (4 курс бакалавриата),
В. Э. Шмаков, к.т.н., доцент,
И. Г. Здобнов

РАЗРАБОТКА АНТИСПАМ ЯДРА ДЛЯ ПОЧТОВЫХ СЕРВИСОВ

Конечной целью разработки является кроссплатформенная динамическая библиотека с одной экспортируемой функцией CheckMessage, которая принимает почтовое сообщение и выдает вердикт после его обработки: спам или нет.

Общие средства разработки - C++ 17 [1] с использованием Cmake для конфигурации сборки, Conan в качестве пакетного менеджера и контроля версий git.

В библиотеке есть 4 основных модуля [2, 3]:

1. MessageParser – отвечает за разбор email-а на составные части [4]: IPs, from, to, attachments, phones, links, plain text и т. п. Модуль работает на основе библиотек Gmime (разбор письма на составные части) и Gumbo (разбор html частей).
2. hecker – набор классов для проверки телефонов, IPs и т. п. по спам базам [5], также присутствует класс для Байесовской фильтрации текста письма [6].
3. YaraRules – основной модуль для написания правил обработки частей письма.
4. DataBase Compiler – отвечает за конвертацию xml базы в бинарный формат и последующая её компрессию при помощи алгоритма LZD.

Рассмотрим подробнее модуль YaraRules. Основой задачей модуля является определение “спамовости” сообщения по бально-рейтинговой системе. Количество баллов, которое набрало сообщение определяется по отработавшим правилам. Все правила можно разделить на три группы:

1. Правила для заголовка письма Рис. 1.

```
rule email_black_list
{
    meta:
        weight = 300
    condition:
        for any mail in mime_message.mails: ( dao.base_contains("email_black_list", mail ))
}
```

Рисунок 1 – пример правила для заголовка письма

2. Правила для текстовой части письма Рис. 2.

```
rule extended_phone_black_list
{
    meta:
        weight = 50
    condition:
        for any phone in mime_message.phones: ( dao.base_contains("extended_phone_black_list", phone ))
}
```

Рисунок 2 – пример правила для текстовой части письма

3. Правила для вложений в письмо Рис. 3.

```
rule virus_attach
{
    meta:
        weight = 300
    condition:
        mime_message.attache_count > 0 and
        mime_message.attache_types[0] == "octet-stream" and
        mime_message.attach_virus

}
```

Рисунок 3 – пример правила для вложений в письмо

Все правила имеют вес (количество баллов), если сумма набранных баллов превышает некоторую границу, то сообщение считается спамом.

Рассмотрим структуру и способ хранения спам базы. Изначально база хранится в формате xml Рис. 4, чтобы было удобно её заполнять по секциям. Впоследствии база конвертируется в бинарный формат и сжимается при помощи алгоритма LZD [7]. База хранится локально у пользователей, это связано с тем, что антиспам может работать на серверах с одним открытым портом на чтение и отправку почты.

```
<base type="patch">
    <forbidden_phone type="string">
        <add_last>7999201459</add_last>
    </forbidden_phone>
    <forbidden_email type="string">
        <add_last>jack@gmail.com</add_last>
    </forbidden_email>
</base>
```

Рисунок 4 – пример xml спам базы

В настоящий момент основные модули антиспам системы реализованы и проходят этап тестирования.

ЛИТЕРАТУРА

1. C++ 17 standard. [Электронный ресурс] Режим доступа: https://en.cppreference.com/w/cpp/17
2. Мартин Фаулер. Шаблоны корпоративных приложений, 2020.

3. Mark Richards, Neal Ford. Fundamentals of Software Architecture, 2021.
4. RFC2822. [Электронный ресурс] Режим доступа: https://www.rfc-editor.org/rfc/rfc2822
5. Самописный антиспам, или как мы боролись за чистоту IP адресов. [Электронный ресурс] Режим доступа: https://habr.com/ru/company/selectel/blog/557400/
6. Jonathan Zdziarski. Ending Spam: Bayesian Content Filtering and the Art of Statistical Language Classification, 2005.
7. Ватолин Д., Ратушняк А., Смирнов М., Юкин В. Методы сжатия данных. Устройство архиваторов, сжатие изображений и видео. - М.: ДИАЛОГ-МИФИ, 2002. - 384 с.

УДК 004.428.4

И. С. Закоулов (2 курс магистратуры),
А. С. Ватьян, к.т.н., доцент

РАЗРАБОТКА ПЛАГИНА КОМПИЛЯТОРА KOTLIN ДЛЯ СКРЫТИЯ ПЕРЕМЕННЫХ В ПАМЯТИ JVM ПРИЛОЖЕНИЙ

В современном цифровом мире разработка программного обеспечения играет важную роль в различных аспектах нашей жизни. Одним из ключевых компонентов разработки программного обеспечения является компилятор [1], который отвечает за перевод исходного кода, написанного на высокоуровневом языке программирования, в машинный код, который может быть выполнен компьютером, либо же другое более низкоуровневое представление, которое далее может исполнено виртуальной машиной Java [2].

Плагины компилятора - это один из способов для разработчиков добавлять новые функции в компилятор без изменения его исходного кода. Плагины встраиваются в процесс компиляции кода и производят различные манипуляции над кодом проекта, позволяя модифицировать существующий код либо сгенерировать новый.

Переменные, используемые во время работы приложения, зачастую хранятся в оперативной памяти устройства для более быстрого доступа. Это допускает возможность модификации данных сторонними приложениями, имеющими права доступа к чужим участкам памяти [3]. Для того чтобы снизить вероятность модификации таких переменных можно воспользоваться различными программными методами сокрытия переменных.

Усложнить поиск и изменение переменных в памяти приложений можно при помощи программной трансформации данных, когда они преобразуются в другой вид. Данные трансформации могут ухудшить читабельность кода разработчиками, а также увеличить время разработки приложения.

Для того, чтобы справиться с данной проблемой, можно прибегнуть к кодогенерации. Кодогенерации можно достичь при помощи использования плагинов компилятора. Разработать плагин компилятора для Kotlin можно с использованием Kotlin IR [4] – абстрактного синтаксического дерева, доступного для модификации.

Главной *целью* работы является разработка плагина компилятора Kotlin [X] с использованием Kotlin IR для сокрытия переменных в памяти приложений.

В связи с этим можно выделить следующие *задачи*:

1. Изучить работу компилятора языка программирования Kotlin с использованием Kotlin IR бекенда.
2. Реализовать несколько способов сокрытия переменных для разных типов данных.
3. Разработать плагин компилятора Kotlin с использованием Kotlin IR для генерации кода, позволяющей скрывать переменные в памяти приложений.
4. Произвести измерение производительности и потребления памяти каждого из способов скрытия переменных.

Было предложено 4 вида скрытия переменных, основанных на исследовании работы приложений для поиска и модификации переменных в памяти приложений:

– строковое представление чисел;

– XOR шифрование;
– шифрование чисел через функцию;
– перемещение значения между ячейками памяти.

Разрабатываемый плагин планируется устанавливать в проект как зависимость, чтобы во время компиляции он выполнился и добавил нужные фрагменты кода. Для того, чтобы у плагина была информация о том, в какие именно места необходимо вставлять сгенерированный код можно воспользоваться аннотациями. Плагин, во время компиляции, найдет все такие поля и сгенерирует код, который скрывает значения в памяти. С примером работы компилятора можно ознакомиться на Рисунке 1.

```
class DataModel {
    @Disguised
    var importantValue = 10;
}
```

```
class DataModel {
    var importantValue = object:
        Disguise<Int>( value: 10, IntPacker) {}
}
```

Рисунок 1 – Пример работы плагина до (слева) и после (справа) кодогенерации

ЛИТЕРАТУРА

1. Habr: Краткий и бодрый обзор архитектуры компиляторов. [Электронный ресурс] Режим доступа: https://habr.com/ru/company/vk/blog/451894/
2. JavaPoint: JVM (Java Virtual Machine) Architecture. [Электронный ресурс] Режим доступа: https://www.javatpoint.com/jvm-java-virtual-machine
3. Habr: GDB Tutorial for Reverse Engineers: Breakpoints, Modifying Memory and Printing its Contents. [Электронный ресурс] Режим доступа: https://habr.com/ru/post/551500/
4. Mobius: Ильмир Усманов: Kotlin IR: прошлое, настоящее и будущее. [Доклад] Режим доступа: https://assets.ctfassets.net/2grufn031spf/55jkcyT6e04UErPjVXwY4T/822a68d2eff68028780306d6020a85c1/Ilmir_Usmanov_Kotlin_IR_proshloye_nastoyashc heye_i_budushcheye_2021_11_23_02_44_36.pdf
5. JetBrains: Kotlin [Электронный ресурс] Режим доступа: https://www.jetbrains.com/opensource/kotlin/

УДК 004.94

Д.В. Доленко (4 курс бакалавриата),
Ю.Б. Сениченков, д.т.н., профессор

СРАВНИТЕЛЬНЫЙ АНАЛИЗ СРЕДЫ МОДЕЛИРОВАНИЯ PTOLEMY II И ЕЁ ПРИМЕНЕНИЕ В УЧЕБНОМ ПРОЦЕССЕ

Цель данной статьи состоит в приведении сравнительной характеристики Ptolemy II и обосновании использования этого инструмента на курсе “Технологии компьютерного моделирования” в Санкт-Петербургском университете Петра Великого.

На указанном курсе, в том числе, рассматриваются среды с различными подходами к моделированию. Можно выделить два подхода:

- *Модельно-ориентированный (МО) подход* концентрируется на процессе разработки (создание и симуляция) и интеграции (генерация исполняемого кода) модели. Данный подход более выражен в коммерческих средах, например, в Simulink [1].

- *Объектно-ориентированный (ОО) подход* задействует принципы создания и наследования классов, инкапсуляции, полиморфизма с целью повторного использования компонентов в новых компонентах и упрощении коллективного проектирования. Представитель с явно выраженным ОО подходом — AnyDynamics [2].

Существует инструмент Ptolemy II [3], хорошо известный по многочисленным публикациям [4], однако мало распространённый в России. В данной среде моделирования основным является *акторно-ориентированный (АО) подход*. В данном подходе акцент

ставится на моделировании систем как набора независимых компонентов, которые взаимодействуют друг с другом путём обмена сообщениями.

Признаки каждого из подходов можно выделить в этих средах моделирования. Каждый из них в зависимости от целей разработчиков реализуется в среде моделирования в той или иной мере. В Таблице 1 представлено сравнение Simulink, AnyDynamics и Ptolemy II по перечисленным подходам.

Таблица 1 – Сравнение различных подходов моделирования в средах Simulink, AnyDynamics и Ptolemy II

Инструмент	**МО аспект**	**ОО аспект**	**АО аспект**
Simulink	*Основной подход.* Создание, симуляция и верификация моделей. Инструменты отладки модели. Генерация кода C, C++, CUDA, PLC, Verilog и VHDL	Представлен в неявном виде: создание новых компонентов как композиции типовых компонентов (экземпляров классов) из обширной библиотеки.	Имеет синтаксически идентичное представление компонентов с Ptolemy II (компоненты с входами-выходами), но фиксированный набор моделей взаимодействия.
AnyDynamics	Создание, симуляция и верификация моделей. Инструменты отладки модели. Нет генерации кода, но есть возможность создания встраиваемой модели (динамическая библиотека DLL) в коммерческой версии среды моделирования (Rand Model Designer Professional).	*Основной подход.* Поддерживается в полной мере, позволяя создавать и наследовать классы, добавлять новые компоненты, переопределять все унаследованные атрибуты, использовать инкапсуляцию, полиморфизм.	Компоненты параллельны и взаимодействуют через порты, но порты не являются ни входами, ни выходами. Вместо этого связи между портами являются ограничениями на переменные. Имеет фиксированный набор моделей взаимодействия.
Ptolemy II	Создание, симуляция и верификация моделей. Инструменты отладки модели. Экспериментальная возможность генерации C и Java кода.	Представлен в упрощённом виде: поддерживается наследование классов, но в классе-наследнике возможно только добавлять новые компоненты и переопределять только параметры.	*Основной подход.* Акторы — это компоненты, которые выполняются одновременно и взаимодействуют посредством сообщений, отправляемых через взаимосвязанные порты. Модель их взаимодействия не является предопределённой, а определяется специальным компонентом.

Ключевой целью проекта Ptolemy является “минимизация случайных различий в синтаксисе, семантике и прагматике между предметными областями и максимизация совместимости конструкций, выраженных в разных предметных областях” [5]. Эта цель, в частности, достигается с помощью абстрактного синтаксиса и абстрактной семантики, позволяющих добавлять в эту программную среду различные модели взаимодействия компонентов (модели вычислений). В то время как в AnyDynamics и Simulink предоставляют фиксированную комбинацию нескольких моделей вычислений (непрерывную и дискретную

модели), в Ptolemy II присутствует множество управляющих компонентов (director, или режисёр) с возможностью их иерархической комбинации.

Исходя из перечисленного, целесообразно использовать Ptolemy II на курсе “Технологии компьютерного моделирования” с целью ознакомления с методами моделирования, предлагаемыми в этой среде. Для этого создаётся методическое пособие, в рамках которого решаются следующие задачи:

1. привести краткое введение по использованию Ptolemy II, его основных особенностях;
2. продемонстрировать набор примеров, основанный на уже существующих примерах из указанного курса, с целью получения навыков работы со средой моделирования.

ЛИТЕРАТУРА

1. Simulink [Электронный ресурс]. Режим доступа: https://www.mathworks.com/products/simulink.html
2. MVSTUDIUM group [Электронный ресурс]. Режим доступа: https://www.mvstudium.com/
3. Ptolemy II [Электронный ресурс]. Режим доступа: https://ptolemy.berkeley.edu/ptolemyII/index.htm
4. The Ptolemy Project publications [Электронный ресурс]. Режим доступа: https://ptolemy.berkeley.edu/publications/index.htm
5. Claudius Ptolemaeus, Editor, *System Design, Modeling, and Simulation Using Ptolemy II*, Ptolemy.org, 2014.

УДК 004.94

З. С. Калачев (2 курс магистратуры),
С. Г. Попов, к.т.н., доцент

ТЕХНОЛОГИЯ ТРЕХМЕРНОЙ ВИЗУАЛИЗАЦИИ СРЕДЫ НА ОСНОВЕ МОРСКИХ КАРТ ФОРМАТА S-57

Развитие автономного судоходства высоких уровней [1] формирует потребность в информационных системах помощи экипажу для повышения осведомленности об окружающей обстановке. К ним относятся навигационные системы, функционирующие в режимах дополненной и виртуальной реальности. Такие системы схожи с системами судовых тренажеров или компьютерных игр с открытым миром, однако к ним предъявляются повышенные требования в части достоверности, точности и надежности представления данных об окружающей обстановке, так как они применяются не в игровых, а реальных условиях и цена ошибки несоизмеримо высока. Главной задачей таких систем является построение и отображение рекомендуемых путей движения судна с учетом окружающей среды и других участников судоходства. При этом официальным форматом представления гидрографических данных является формат S-57 [2], предложенный в начале 1990-х годов, обеспечивший электронное представление двумерных данных. Формат не содержит полного описания рельефа суши, и включает в себя скудное описание естественных или искусственных ориентиров, и лишь в той части, которая обеспечивает навигацию судов, что недостаточно для синтеза реалистичной трехмерной среды.

Исходными данными для создания трехмерной среды является набор высокоточных глубин, высот и реалистичных текстур поверхностей среды для построения и отображения судов и их траекторий, что предполагает объединение и согласование данных из нескольких источников. Основой согласования данных являются географические координаты в единой системе параметров геоида.

Целью работы является формирование исходных данных для визуализации реалистичной трехмерной среды, одновременно содержащей географическую и гидрографическую составляющие.

Для этого требуется осуществить синтез разнородных исходных данных: картографических данных в системе координат WGS84, гидрографических данных из формата S-57, карт ледовой обстановки и реалистических текстур поверхности суши в согласованном формате системы моделирования траекторий и реалистической визуализации судов в условиях ледовой обстановки [3]. Схема формирования исходных данных для среды моделирования представлена на Рисунке 1.

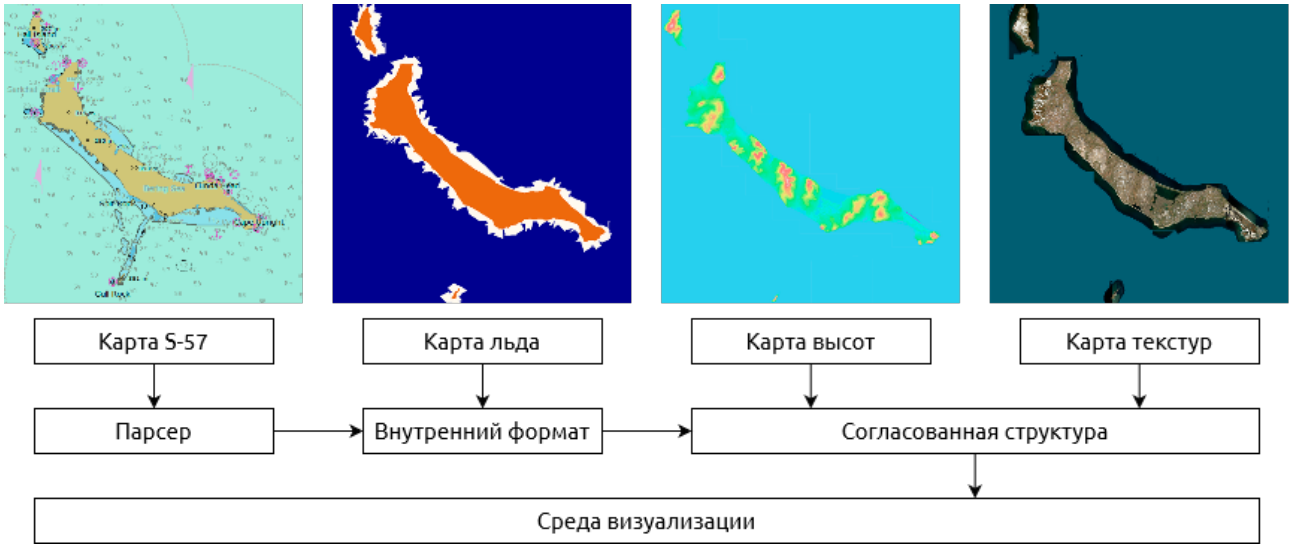


Рисунок 1 – Формирование структуры данных для включения в среду визуализации

Технологически, источниками данных для актуализации слоев геоинформационной системы являются как свободно распространяемые, так и корпоративные информационные системы. Актуализация морских карт осуществляется по подписке карт гидрографической службы или через локальных дистрибьютеров. Ледовая обстановка формируется и актуализируется на основе гидрологических прогнозов и на основе локального анализа результатов ледовой разведки и аэрофотосъемки. Частота обновления данных слоев выполняется, сообразуясь со скоростями обновления исходных данных: от нескольких дней для ледовой обстановки, до часов, для данных о текстурах поверхностей. Реализованная в прототипе модель обеспечивает точность представления исходных данных соразмерно моделируемым судам океанской зоны и составляет 50 метров реального пространства.

Преобразование карты S-57 во внутренний формат системы производилось при помощи библиотеки GDAL [4]. Для этого был реализован парсер с использованием фреймворка Qt 6.2, позволяющий считывать и преобразовывать только необходимые для дальнейшей работы данные, такие как береговая линия и изолинии воды различных глубин. Формирование ледовой обстановки было выполнено при помощи динамической процедурной генерации карты льда на основе выбранных контуров, в том числе и контуров береговой линии. Для добавления рельефа суши использовался подход с использованием карты высот, представляющей из себя двумерное растровое изображение, каждый пиксель которого хранит данные о высоте поверхности. Карты высот были получены из открытого источника [5], использующего базу данных высот TessaDEM. Для повышения реалистичности суши применялась карта текстур, полученная с фотографий высокого разрешения. Подход с использованием карты высот и текстур широко используется в программах для визуализации местности, тренировочных симуляторах и компьютерных играх [6, 7].

Реализованный программный комплекс функционирует на универсальных вычислительных средствах под управлением операционной системы общего назначения. Инструментальные средства комплекса обеспечивают консолидацию данных из различных источников, и обработку полученных данных с высокой производительностью для последующего анализа и формирования трехмерной графической среды.

Программный комплекс функционирует на выделенном аппаратном сервере с процессором Intel 2,5 ГГц, оперативной памятью 16 Гбайт, видеокартой nVidia RTX A3000 с оперативной памятью 16 Гбайт. Для отображения результатов работы сервера визуализации используется монитор диагональю 27 дюймов и видеостена с диагональю 115 дюймов. Аппаратные средства обеспечивают отображение визуализации в разрешении изображения 4К. Объединение программных компонент осуществляется при помощи локальной сети внутрисистемных взаимодействий через сокеты или взаимодействием через Интернет.

Дальнейшим развитием данной работы может служить повышение детализации карты, добавление текстур сезонов года и условий освещенности поверхности.

ЛИТЕРАТУРА

1. Maritime Safety Committee, 100th session. [Электронный ресурс]. Режим доступа: https://www.imo.org/en/MediaCentre/MeetingSummaries/Pages/MSC-100th-session.aspx;
2. IHO Transfer Standard for Digital Hydrographic Data. Publication №57. Edition 3.1 – 2000
3. Забровский В. С., Попов С. Г., Моторин Д. Е., Ямщиков Ю. А. Среда интерактивной визуализации движения судов в условиях ледовой обстановки // Региональная информатика и информационная безопасность. Сборник трудов. Выпуск 11 / СПОИСУ. – СПб., 2022. – с. 257-262.
4. GDAL. [Электронный ресурс]. Режим доступа: https://gdal.org
5. Топографические карты. [Электронный ресурс]. Режим доступа: https://topographic-map.com
6. Dam P., Duarte F., Raposo A. Terrain generation based on real world locations for military and simulation // 2019 18th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames). – IEEE, 2019. – pp. 173-181.
7. Spick R., Walker J. Realistic and textured terrain generation using GANs // CVMP’ 19: European conference on visual media production – 2019. – pp. 1-10.

УДК 004.455.1

К. М. Касимова (4 курс бакалавриата),
Н. В. Воинов, к.т.н., доцент

РАЗРАБОТКА КОНСТРУКТОРА КРОССПЛАТФОРМЕННЫХ ВЕБ-САЙТОВ

Создание сайтов - востребованное направление в разработке программного обеспечения [1-3]. Сфера услуг не стоит на месте, и теперь без опыта разработки и знания языков программирования можно создавать веб-сайты самостоятельно. Такую возможность предоставляют конструкторы сайтов. На сегодняшний день это один из самых быстрых способов разработки.

Конструктор сайтов – это система, позволяющая любому пользователю интернета самостоятельно создать личный или корпоративный сайт, не обладая при этом знаниями и навыками в области программирования, верстки и дизайна. Современные решения предлагают нам систему 3 в 1 – конструктор сайтов, хостинг и доменное имя. Чаще всего данные продукты не требуют установки дополнительных программ, достаточно лишь подключения в Интернет к административной части вашего сайта – редактору. Таким образом, можно войти на свой веб-сайт под индивидуальным логином и паролем, и внести нужную информацию. Функциональность подобных решений очень разнится, но в каждой из них есть свой ряд недостатков. Выделим основные изъяны присущие всем существующим конструкторам:

– платформозависимость;
– невозможность работы с кодом;
– некачественный код на выходе;
– несовместимость версий;
– шаблонизация и невозможность реализовать все свои задумки в дизайне;
– отсутствие бесплатных тарифов.

Таким образом, целью данной работы является разработка конструктора веб-сайтов, искореняющего данные недостатки.

Для достижения этой цели необходимо решение следуюших задач:

– обзор существующих конструкторов и их недостатков;

– формулирование требований к собственному решению;

– выбор технологического стека;

– разработка архитектуры;

– реализация;

– тестирование и вывод в продуктивную среду.

Для данного программного решения используется многоуровневая архитектура, представленная на Рис. 1. Представление контента конечному пользователю через графический интерфейс происходит через уровень клиента. Уровень логики отвечает за обработку запросов, которые поступают от клиента. Далее, в зависимости от поступающей команды, с помощью уровня данных происходит либо извлечение, либо запись на хранение, либо изменение существующих данных.

Рисунок 1 – Архитектура системы

Серверная часть была реализована с помощью Apollo Server [4] – это сервер GraphQL, соответствующий спецификации, который совместим с любым клиентом GraphQL, включая клиент Apollo. В качестве промежуточного обработчика запросов был использован Express [5] для установки CORS политики и отправки токенов JWT [6]. Для создания схем базы данных, сущностей и их обработчиков использовались TypeGraphQL [7] и TypeGoose [8] – менеджеры типов данных для строго типизированного языка TypeScript [9].

Клиентская часть была реализована с помощью React [10] - JavaScript-библиотеки для создания пользовательских интерфейсов. Для стилизации использовалась библиотека компонентов MUI [11]. Управление локальными и удаленными данными происходит с помощью Apollo Client [1] - библиотекой управления состоянием.

ЛИТЕРАТУРА

1. Чавес, К. Разработка javascript-фреймворка для генерации серверных веб-приложений / К. Чавес, Н. В. Воинов, Е. В. Каплан // Современные технологии в теории и практике программирования : Сборник материалов конференции, Санкт-Петербург, 26 апреля 2022 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2022. – С. 108-110. – EDN SHFUQC.
2. Смирнов, П. А. Клиент-серверное приложение для донорской службы / П. А. Смирнов, В. В. Малиновская, Н. В. Воинов // Современные технологии в теории и практике программирования : Сборник материалов конференции, Санкт-Петербург, 26 апреля 2022 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2022. – С. 44-46. – EDN XPPPRC.
3. Smirnov, P. A. A Web Application to Promote Blood Donation in Russia / P. A. Smirnov, V. V. Malinovskaya, N. V. Voinov // Proceedings of the Institute for System Programming of the RAS. – 2022. – Vol. 34, No. 2. – P. 179-190. – DOI 10.15514/ISPRAS-2022-34(2)-14. – EDN SGENXV.
4. Apollo. [Электронный ресурс] Режим доступа: https://www.apollographql.com/docs/
5. Express. [Электронный ресурс] Режим доступа: https://expressjs.com
6. JWT. [Электронный ресурс] Режим доступа: https://jwt.io

7. TypeGraphQL. [Электронный ресурс] Режим доступа: https://typegraphql.com/docs/introduction.html
8. TypeGoose. [Электронный ресурс] Режим доступа: https://typegoose.github.io/typegoose/
9. TypeScript. [Электронный ресурс] Режим доступа: https://www.typescriptlang.org
10. React. [Электронный ресурс] Режим доступа: https://ru.reactjs.org/docs/hello-world.html
11. MUI. [Электронный ресурс] Режим доступа: https://mui.com/material-ui/getting-started/overview/

УДК 004.455.2

А. Г. Коновалова (1 курс магистратуры),
В. И. Маракшин,
И. В. Никифоров, к. т. н., доцент

АРХИТЕКТУРА РЕЗЕРВНОГО КОПИРОВАНИЯ ПО МОДЕЛИ BACKUP-AS-A-SERVICE И КОНСОЛИДАЦИЯ ПОЛУЧЕННЫХ ДАННЫХ

Современному бизнесу требуется инфраструктура информационных технологий доступная 24 часа в сутки, 7 дней в неделю [1]. Компании не могут позволить себе отключение системы продолжительностью более нескольких часов, поскольку это может привести к значительным финансовым потерям. Благодаря резервному копированию можно снизить потери из-за простоя серверов [2].

В свою очередь, система резервного копирования относится к системам обеспечения непрерывной деятельности компании, следовательно, она должна быть независима от внешних факторов. Непрерывность деловой активности организации позволяет обеспечить доступность сервисов, сохранность данных, решить проблемы загруженности и обеспечить безопасность бизнес-критичных сервисов. В зависимости от критичности системы определяются компоненты, обеспечивающие непрерывность процессов: кластерные решения высокой доступности, репликация данных и резервное копирование. Технологии резервного копирования используются согласно метрикам непрерывности услуги для любой системы, не учитывая уровень её критичности. Исходя из значимости решений для бэкапа в корпоративном сегменте, необходим переход на программные продукты отечественных вендеров.

Системы резервного копирования позволяют определять для каких систем и данных создаётся резервная копия, размещение резервных копий на выбранном устройстве хранения и автоматически управлять процессом резервного копирования [3].

Однако готовое решение системы резервного копирования не всегда покрывает потребности компании. Можно выделить несколько факторов, влияющих на это: географическое распределение клиентов, высокая нагрузка на центры обработки данных, отсутствие необходимых функций. Вследствие перечисленных факторов требуется создание дополнительного слоя абстракции над программным продуктом, позволяющим централизованно контролировать статусы регулярных заданий резервного копирования, а также следить за зоной покрытия резервным копированием на уровне группы компаний или большой инфраструктуры.

Целью работы является реализация решения по объединению подсистемы отчётности с нескольких инсталляций ПО системы резервного копирования на основе отечественного программного обеспечения – Кибер Бэкап (ООО КиберПротект) и экспорт полученной информации.

На Рисунке 1 представлена общая архитектура резервного копирования по модели Backup-as-a-Service (BaaS). Модель предоставления услуги резервного копирования как сервиса позволяет контролировать соглашение об уровне обслуживания (SLA), снижать расходы за счёт её централизации и предоставляет возможность бизнесу концентрироваться на основных процессах. Заказчик приобретает измеряемую услугу, а исполнитель, используя

свои инфраструктурные мощности, предоставляет план резервного копирования. Данные хранятся на группе серверов в центре обработки данных.

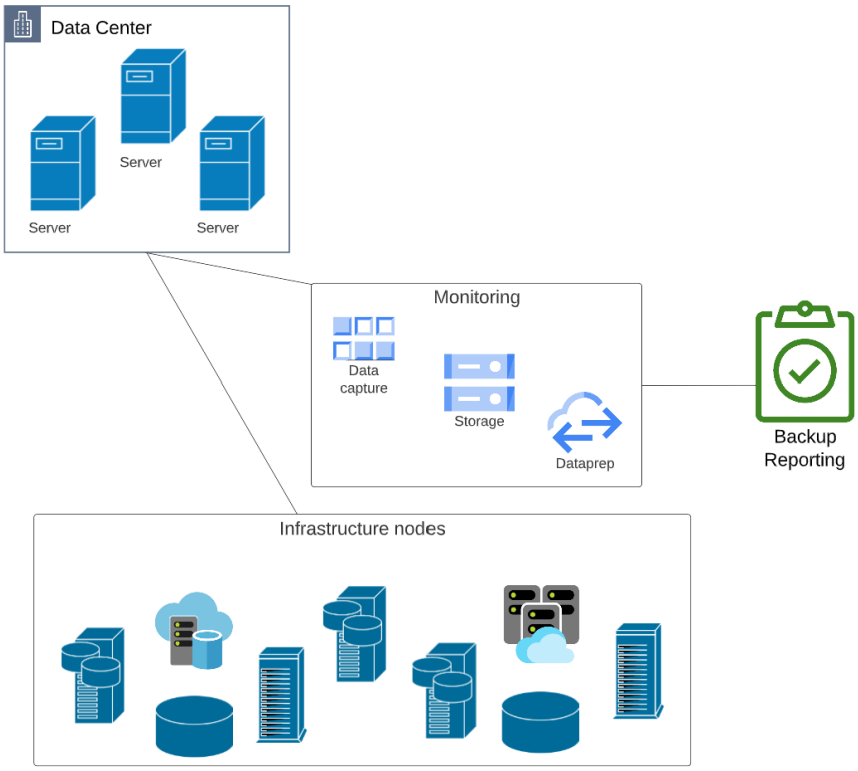


Рисунок 1 – Схема резервного копирования по модели BaaS

При сборке данных выполняется их выборка [4]. Так, для формирования единого отчёта об объёме потребления услуг и дальнейшего выставления счёта заказчикам необходимо наличие информации о статусе выполнения задания резервного копирования. Итоговый отчёт оформляется в виде таблицы.

В будущем планируется проанализировать взаимодействие системы резервного копирования с защищаемыми узлами для дальнейшего построения архитектурного решения и посредством API реализовать цель работы. API является набором программных методов, который позволяет двум приложениям взаимодействовать друг с другом с помощью определённых правил и интегрироваться друг с другом для обмена данными [5]. И в дополнение будет разработан прототип отчёта.

ЛИТЕРАТУРА

1. J. Mendonça, R. Lima, E. Queiroz, E. Andrade and D. S. Kim, "Evaluation of a Backup-as-a-Service Environment for Disaster Recovery," 2019 IEEE Symposium on Computers and Communications (ISCC), Barcelona, Spain, 2019, pp. 1-6, doi: 10.1109/ISCC47284.2019.8969658.
2. Айкашева, Ю. А. Резервное копирование и восстановление данных на предприятиях / Ю. А. Айкашева // Актуальные проблемы авиации и космонавтики. – 2016. – Т. 2, № 12. – С. 7-8. – EDN XRYARX.
3. Питкевич, П. И. Методы резервирования данных для критически важных ит-систем предприятия / П. И. Питкевич – 2021. – № 10-1(91). – С. 25-28. – DOI 10.32743/UniTech.2021.91.10.12405.
4. M. R. Hilmi, M. Sudarma and Linawati, "Virtual Backup Server optimization on Data Centers using Neural Network," 2018 International Conference on Smart Green Technology in Electrical and Information Systems (ICSGTEIS), Bali, Indonesia, 2018, pp. 162-167, doi: 10.1109/ICSGTEIS.2018.8709101.
5. S. Jonnada and J. K. Joy, "Measure your API Complexity and Reliability," 2019 IEEE 17th International Conference on Software Engineering Research, Management and Applications (SERA), Honolulu, HI, USA, 2019, pp. 104-109, doi: 10.1109/SERA.2019.8886790.

УДК 004.9

Ю. С. Кичигин (4 курс бакалавриата),
С. П. Воскобойников, к.ф.м.н., доцент

РАЗРАБОТКА ИНТЕРАКТИВНОГО ИНСТРУМЕНТА ВИЗУАЛИЗАЦИИ СТРУКТУРЫ ПРОГРАММНЫХ СИСТЕМ

Для любого программного проекта, который разрабатывается больше, чем парой человек, встаёт вопрос о создании программной документации, которая включает в себя различные схемы, например, схему потоков данных [1]. Однако для активно разрабатываемых программных систем этот вопрос осложняется постоянными следующими проблемами:

1. Система постоянно модернизируется и изменяется, что приводит к необходимости регулярно обновлять схемы [2].
2. В разработку привлекаются новые люди, которых необходимо максимально быстро ввести в курс дела, что очень удобно делать, когда существует информативная интерактивная схема [3].
3. Разработчикам, которые оперативно устраняют возникающие ошибки, нужно предоставить возможность видеть не только сигналы об ошибках в конкретном модуле, но также и его связи со смежными модулями и их состояния.

Со всеми этими проблемами приходится регулярно сталкиваться разработчикам программных систем и поэтому их решение является актуальной задачей.

Цель работы заключается в создании интерактивного инструмента визуализации структуры программных систем.

В работе проведён анализ доступных решений в области инструментов визуализации структур программных систем. Были выбраны следующие инструменты:

- GIMP для создания рисунков схем как PNG
- Miro [4]
- Graphviz [5]

Каждый инструмент оценивался по ряду критериев, основные из которых приведены в Таблице 1.

Таблица 1 – Сравнение инструментов визуализации схем по основным критериям

Критерии	Рисунки схем как PNG	Miro	Graphviz
Трудоемкость изготовления схемы	Легко рисовать	Очень легко рисовать	Обязательно надо программировать генерацию
Что можно визуализировать	Любая визуализация произвольных объектов и отношений	Визуализация произвольных объектов и отношений ограниченным набором элементов	Визуализация только объектов и отношений, доступных к автоматическому парсингу
Гарантии актуальности схемы	Невозможно гарантировать актуальность схемы	Невозможно гарантировать актуальность схемы	По крайней мере в некоторых случаях можно гарантировать актуальность схемы
Возможность автоматических/массовых исправлений	Исправления – только ручные	Исправления – только ручные	Возможны автоматические обновления
Трудоемкость редактирования схемы	Редактировать умеренно тяжело	Легко редактировать	Редактировать умеренно тяжело
Интерактив при просмотре	Ограниченный интерактив при просмотре	Можно организовать умеренный интерактив	Ограниченный интерактив при просмотре

В связи с тем, что доступные инструменты не удовлетворяют всем критериям даже в минимальном объёме, было выдвинуто предложение о создании инструмента, который смог бы решить указанный круг проблем. По результатам обсуждения были сформулированы следующие требования к такому инструменту:

1. Простые схемы должны создаваться легко либо же генерироваться.
2. Не ставить жесткие ограничения на отображаемые объекты на схеме.
3. Гарантировать актуальность схем.
4. Возможность автоматических обновлений схем.
5. Редактирование схем не должно быть сложным процессом.
6. Просмотр схем должен быть интерактивным.
7. Возможность динамического обогащения схем данными из других источников.

Также была разработана диаграмма потока данных для данного инструмента, Рисунок 1.

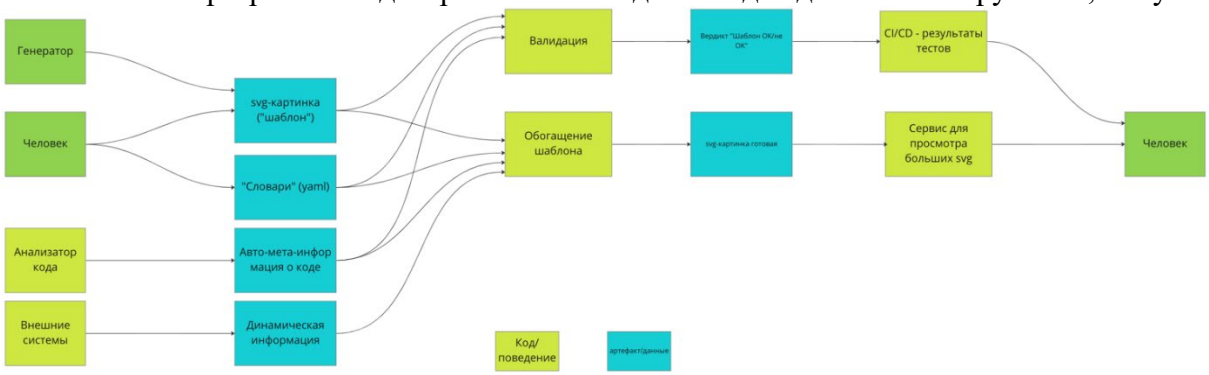


Рисунок 1 – Диаграмма потока данных разрабатываемого инструмента

В ходе работы создана первая версия инструмента, которая будет дорабатываться для дальнейшего улучшения и наиболее полного достижения всех поставленных требований.

Для разработки инструмента использован язык программирования Java и среда IntelliJ IDEA.

ЛИТЕРАТУРА

1. Разработка программной документации прикладного программного обеспечения. [Электронный ресурс] URL: http://www.robotics.by/articles/article_22.html (дата обращения: 20.03.2023)
2. Модернизация программного обеспечения – системная инженерия. Принципы и практика. [Электронный ресурс] URL: https://ozlib.com/823450/informatika/modernizatsiya_programmnogo_obespecheniya (дата обращения: 20.03.2023)
3. Проблемы разработки ПО. [Электронный ресурс] URL: https://studfile.net/preview/1524040/ (дата обращения: 20.03.2023)
4. Onboarding essentials. [Электронный ресурс] URL: https://developers.miro.com/docs (дата обращения: 20.03.2023)
5. Documentation | Graphviz. [Электронный ресурс] URL: https://graphviz.org/documentation/ (дата обращения: 20.03.2023)

УДК 004.453

Д. О. Королев, Г. Н. Толстиков (1 курс магистратуры),
Е. А. Павлов, ассистент

РЕАЛИЗАЦИЯ CI/CD ПРОЦЕССА ДЛЯ TELEGRAM-БОТА ПО ПОИСКУ АКЦИОННЫХ ТОВАРОВ В СУПЕРМАРКЕТАХ СРЕДСТВАМИ GITHUB ACTIONS

Непрерывная интеграция, поставка и развертывание (CI/CD) - основанная на Agile методология DevOps в разработке программного обеспечения, которая помогает повысить качество и эффективность процесса доставки программного обеспечения. Инструменты CI/CD

автоматизируют широкий спектр задач, таких как тестирование, компоновка, обновление зависимостей, создание, развертывание релизов и т.д [1].

Для CI/CD процесса используются различные инструменты, такие как Bitbucket Pipelines, Jenkins, Gitlab CI/CD, Github Actions. В случаях, когда исходный код приложения расположен в системе контроля версий, распространенным подходом организации CI/CD процесса является применение встроенных средств системы контроля версий [2]. Наиболее распространенными системы контроля версий являются Github и Gitlab.

GitLab поддерживает возможность интеграции с такими платформами, как Jira, Microsoft Teams, Slack, Gmail и другими. Однако существенным недостатком является ограничение в 5 пользователей, которые могут одновременно находится в закрытой группе с бесплатным типом подписки. Также Gitlab с бесплатной подпиской предоставляет пользователям всего 400 минут в месяц на выполнение CI/CD процессов в Gitlab CI/CD, что явно недостаточно даже для относительно небольших проектов.

Github предоставляет возможность настройки и оптимизации репозитория под собственный проект на основе множества расширений. Кроме того, в GitHub нет явного ограничения на количество соавторов в репозитории. Также GitHub, в отличие от Gitlab, предоставляет пользователям 2000 минут в месяц для CI/CD процессов в Github Actions.

На основе вышесказанного Github является лучшим выбором в рамках данного проекта. Для организации CI/CD процесса могут быть применены сторонние инструменты, предоставляющие интеграцию с Github, однако они обладают весомым недостатком. Для их использования необходима дополнительная установка на сервер. Это не является проблемой для крупных организаций, однако для небольших команд удобнее использовать встроенный инструмент Github Actions, не требующий дополнительной установки и настройки и поддерживающий множество языков и фреймворков.

В рамках данной работы исходный код проекта на языке Java находится в Github репозитории, и было принято решение использовать Github Actions для организации процесса CI/CD.

Таким образом, целью данной работы является реализация процесса CI/CD с применением Github Actions.

Для достижения этой цели необходимо решение следующих задач:

– Реализация пайплайна для сборки проекта и запуска тестов;

– Реализация пайплайна сборки Docker-образа и загрузки его в Docker Hub [3].

Результатом CI/CD конвейера является сборка исходного кода в Docker-образ и загрузка его в приватный Docker-репозиторий для дальнейшего развертывания в целевой среде. Было создано два пайплайна: первый, запускающий интеграционные тесты при создании пул реквеста; второй, для сборки образа и отправки его в Docker-репозиторий при внесении изменений в главную ветку.

Первый пайплайн содержит следующие шаги:

– Проверка репозитория;

– Установка JDK версии 11;

– Сборка maven проекта;

– Запуск интеграционных тестов.

Для отправки Docker-образа в процессе CI/CD создаем приватный Docker-репозиторий, используя сервис Docker Hub. Также стоит учитывать, что для одной учетной записи с бесплатным доступом можно создать только один приватный репозиторий.

Для интеграции с Docker Hub добавим данные авторизации учетной записи созданного репозитория в Secrets внутри проекта Github. В качестве шагов второго пайплайна выполним:

– Проверка репозитория;

– Вход в Docker Hub;

– Настройка инструмента Docker Buildx;

– Сборка и отправка образа в DockerHub.

Также был создан Dockerfile, который описывает запуск полученного после сборки jar-файла. Для корректной работы в файл конфигурации сборки Maven добавлена зависимость для сборки jar-файла со всеми зависимостями (толстый jar) [4].

После внесения изменений в главную ветку выполняются данные пайплайны, и в Docker-репозитории появляется новый образ, который может быть использован для развертывания приложения в Kubernetes кластере. Для этой цели может применяться множество различных инструментов [5]. Одним из таких является keel, который устанавливается вместе с Kubernetes кластером и обновляет поды с запущенным приложением при изменении его Docker-образа в Docker Hub.

ЛИТЕРАТУРА

1. Rostami Mazrae P., Mens T., Golzadeh M., Decan A. On the usage, co-usage and migration of CI/CD tools: A qualitative analysis // Empirical Software Engineering, 2023. – DOI 10.1007/s10664-022-10285-5.
2. Benedetti G., Verderame L., Merlo A. Automatic Security Assessment of GitHub Actions Workflows // CCS '22: 2022 ACM SIGSAC Conference on Computer and Communications, 2022 – P. 37-45 – DOI: 10.1145/3560835.3564554.
3. Docker Hub [Электронный ресурс]. Режим доступа https://hub.docker.com/
4. Maven Documentation [Электронный ресурс]. Режим доступа https://maven.apache.org/guides/index.html
5. Manish A., Rao, D., Subrahmanyam K. Framework to Deploy Containers using Kubernetes and CI/CD Pipeline // International Journal of Advanced Computer Science and Applications 13 (4), 2022. – P. 522-526. – DOI: 10.14569/IJACSA.2022.0130460.

УДК 004.031.2

Д. О. Королев, А. Ю. Шестакова, А. А. Афанасьев (1 курс магистратуры),
И. В. Никифоров, к.т.н., доцент

РАЗРАБОТКА СИСТЕМЫ МОНИТОРИНГА КОРРЕКТНОСТИ ССЫЛОК НА WEB-СТРАНИЦАХ

Некорректная ссылка (broken link) — ссылка, которая ведет на несуществующую страницу, при переходе по которой пользователь видит сообщение об ошибке. Главная опасность таких ссылок заключается в том, что они затрудняют изучение информации и выполнение целевых действий пользователя, тем самым ухудшая восприятие сайта.

Основными сценариями возникновения некорректных и сломанных ссылок являются отключение сайта и домена, а также проблемы с доступом к конкретному ресурсу, сводящиеся к проблеме доступности информации и возникающие при перемещении или удалении искомого файла или страницы [1].

Существует достаточно большое количество решений, проверяющих битые ссылки: Google Search Console, Яндекс.Вебмастер, Dead Link Checker, Broken Link Checker, Netpeak Spider, Dr Link Check. Главным недостатком этих решений является ограничение количества проверяемых ссылок на странице. Для снятия ограничений в некоторых из перечисленных инструментах предоставлена возможность приобрести платную подписку, что влечет для разработчика сайта ненужные и лишние затраты.

Таким образом, целью данной работы является реализация доступного и простого в использовании сервиса для проверки корректности ссылок на веб-страницах.

Для достижения этой цели необходимо решить следующие задачи, перечисленные ниже.

1. Обзор существующих решений.
2. Проведение сравнительного анализа найденных решений.
3. Реализация серверной части с REST API интерфейсом.
4. Реализация сервиса-анализатора ссылок.
5. Тестирование.

Реализованная система состоит из двух компонент:

– Бэкенд, являющийся входной точкой в систему, взаимодействие с которым осуществляется посредством REST API интерфейса. Данный сервис реализован с применением языка Java и фреймворка Spring.

– Сервис, выполняющий анализ ссылок на веб-страницах, реализованный с применением языка Python и фреймворка Scrapy [2].

Для реализации горизонтального масштабирования [3] сервиса был применен фреймворк Scrapyd [4]. Данный фреймворк представлен в виде сервера, который позволяет параллельно запускать несколько экземпляров сервиса анализа посредством REST API интерфейса. Так, например, для запуска сервиса анализа может быть выполнен следующий запрос:

```
curl http://localhost:6800/schedule.json -d project=myproject -d spider=myspider
```

Архитектура системы приведена на Рис. 1.

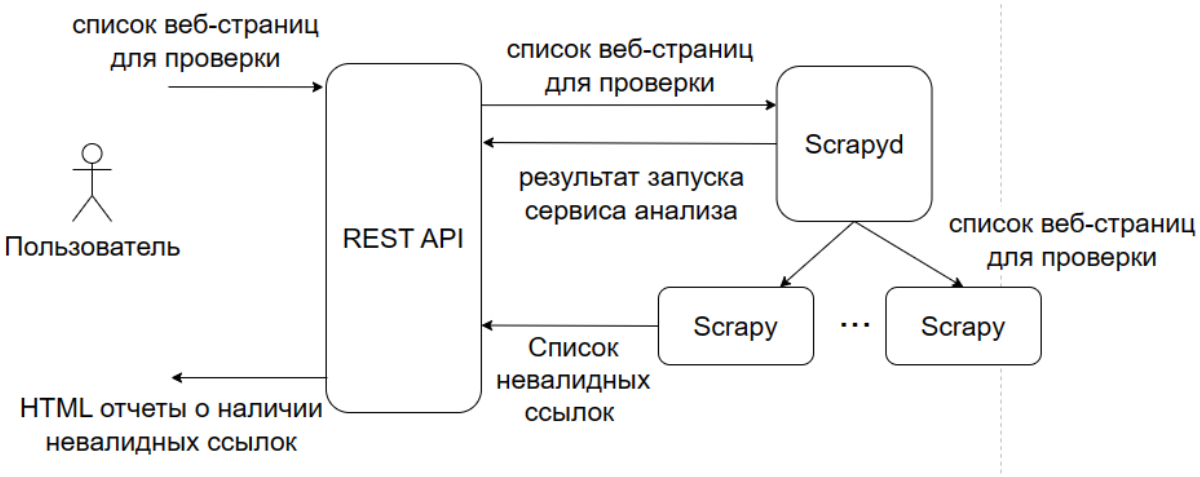


Рисунок 1 – Архитектура системы

Работа с системой осуществляется по следующему сценарию, представленному ниже.

1. Пользователь отправляет long polling [5] запрос на REST API бэкенда, передавая список веб-страниц для проверки, и ожидает ответа от него.

2. Бэкенд отправляет запрос на сервер Scrapyd, который запускает сервис анализа и возвращает ответ с информацией о нем бэкенду.

3. Сервис анализа после завершения своей работы отправляет результат на бэкенд, который, в свою очередь, генерирует и возвращает пользователю отчет в формате HTML [6] о наличии некорретных ссылок на указанных web-страницах, таким образом завершая запрос. HTML отчет содержит таблицу со следующими полями: порядковый номер строки; ссылка, которая ведет на 404 Not Found; страница, на которой присутствует ссылка; текст ссылки.

Для настройки сервиса используется конфигурационный файл в формате YAML, в котором указан список сайтов, которые нужно проанализировать (по умолчанию сайт spbstu.ru и сайты всех институтов Политехнического университета). Также в конфигурационном файле указано расписание, согласно которому сервис автоматически запускается.

ЛИТЕРАТУРА

1. Привалов А. Н., Смирнов В. А., Богатырева Ю. И. К вопросу анализа внешних ссылок и встраиваемых элементов на веб-сайтах // Известия Тульского Государственного Университета. Технические науки 5, 2021. – С. 323-328. – DOI: 10.24412/2071-6168-2021-5-323-329.
2. Yuhao Fan. Design and Implementation of Distributed Crawler System Based on Scrapy // IOP Conference Series Earth and Environmental Science, 2018. – DOI: 10.1088/1755-1315/108/4/042086.
3. Voinov N., Drobintsev P., Kotlyarov V., Nikiforov I. Distributed OAIS-Based digital preservation system with HDFS technology // 20th Conference of Open Innovations Association FRUCT :

Proceedings, Saint-Petersburg, 03–07 апреля 2017 года / LETI University, St.Petersburg, Russia; S. Balandin, A. Levina, T. Tyutina. – Saint-Petersburg: FRUCT Oy, 2017. – P. 491-497. – DOI 10.23919/FRUCT.2017.8071353. – EDN XXPLRR.
4. Eyzenakh D.S., Rameykov A.S., Nikiforov I.V. High performance distributed web-scraper // Proceeding of the Institute for System Programming of the RAS 33 (3), 2021. – P. 87-100. – DOI: 10.15514/ISPRAS-2021-33(3)-7.
5. Cutting D. Evaluation of Long-Held HTTP Polling for PHP/MySQL Architecture, 2015. – DOI: 10.13140/RG.2.2.26264.80648.
6. Kovalev A., Voinov N., Nikiforov I. Using the Doc2Vec Algorithm to Detect Semantically Similar Jira Issues in the Process of Resolving Customer Requests // Studies in Computational Intelligence. – 2020. – Vol. 868. – P. 96-101. – DOI 10.1007/978-3-030-32258-8_11.

УДК 004.942

А. Е. Кущенко (2 курс аспирантуры),
А. В. Самочадин, к.т.н., доцент

ПРОЕКТИРОВАНИЕ МОДУЛЬНОЙ СЕРВИС-ОРИЕНТИРОВАННОЙ СИСТЕМЫ ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ

Проектирование и разработка систем имитационного моделирования является перспективным направлением в области обеспечения принятия решений. Однако большинство популярных систем имитационного моделирования спроектированы без использования современных подходов и парадигм в области проектирования и разработки ПО [1]. Одной из целей данной работы является сравнительный анализ существующего ПО для имитационного моделирования, оценка их функциональности, применённых архитектурных подходов и технологий. Второй целью является описание спроектированной архитектуры и описанных подходов при разработке модульной системы имитационного моделирования.

Сравнительный анализ инструментов имитационного моделирования

Рассмотрим наиболее распространённые инструменты моделирования, представленные в Таблице 1. Следуя современным тенденциям проектирования и разработки ПО, можно выделить ряд востребованных подходов, которые хотелось бы видеть в современной системе. Среди них наличие веб-интерфейса для работы в браузере, сервис-ориентированная архитектура, предполагающая выделение компонентов системы в отдельные сервисы, которые могут быть развёрнуты на выделенных вычислительный ресурсах в сети в т. ч. виртуальных, тем самым реализовывая облачные вычисления. Каждый сервис может быть запущен в изолированной среде (контейнере) со свой ОС и необходимыми зависимостями.

Таблица 1 – Сравнение инструментов имитационного моделирования

Критерий сравнения	OpenModelica [2]	RMD [3]	AnyLogic [4]	Simulink [5]
Веб-интерфейс	-	-	+	-
Сервис-ориентированность	-	-	+	-
Облачные вычисления	-	-	-	-
Не требует опыта в программировании	+	-	+	-
Методы машинного обучения	-	-	-	-
Контейнерная виртуализация	-	-	-	-

Все рассмотренные коммерческие решения, кроме AnyLogic реализованы в виде отдельных десктопных приложений и не требуют доступ к сети. Отсюда следует, что в этих системах отсутствует сервис-ориентированность, контейнеризация и веб-интерфейс. В рассмотренных инструментах не предусмотрено использование методов машинного обучения, все модели описываются пользователем вручную и принятие решений также

остаётся за ним, кроме того, большая часть требует знания программирования для описания моделей.

Проектирование модульной системы имитационного моделирования

В основе проектируемой системы лежит сервис-ориентированная архитектура, предполагающая реализацию частей модульной системы в виде отдельных сервисов, которые взаимодействую между собой. Архитектура проектируемой системы, где отображены разрабатываемые сервисы изображённая на Рисунке 1, далее рассмотрим подробнее каждый из этих сервисов.

Сервисы API механизма симуляции и редактора моделей реализуют REST интерфейс для обмена информацией с клиентским веб-приложением. Взаимодействие реализуется с помощью передачи состояния посредством HTTP-запросов, отправляемых клиентским приложением. Для получения, создания, редактирования и удаления ресурса на сервере используются GET, POST, PUT, DELETE запросы соответственно.

Web-интерфейс позволяет пользователю взаимодействовать с системой моделирования (создавать, редактировать модели, выполнять их симуляцию) в привычном окне браузера. Современные фреймворки для разработки веб-приложений имеют большое число библиотек для наглядного отображения и динамического обновления графов, диаграмм и графиков.

Модели машинного обучения и ИИ также выделены в отдельные сервисы. Первый сервис отвечает за генерацию моделей на основе ассоциативных правил, извлекаемых из исторических данных о событиях. Второй сервис, использует инструмент с искусственным интеллектом для автоматической генерации кода пред/пост условий и выражений, что позволяет работать с инструментом имитационного моделирования пользователю, не имеющему опыта в программировании.

Для хранения моделей, объектов, событий и результатов моделирования используется реляционная база данных.

Все сервисы изолированы в отдельные docker-контейнеры. С помощью инструмента Docker Compose можно развернуть все контейнеры с нашими сервисами одной командой, заранее описав правила маршрутизации между зависимыми контейнерами в конфигурационном YAML-файле.

Данная архитектура позволяет развернуть наши сервисы в облачной среде, например, на распределенном Kubernetes кластере, который будет управлять и поддерживать отказоустойчивую работоспособность сервисов.

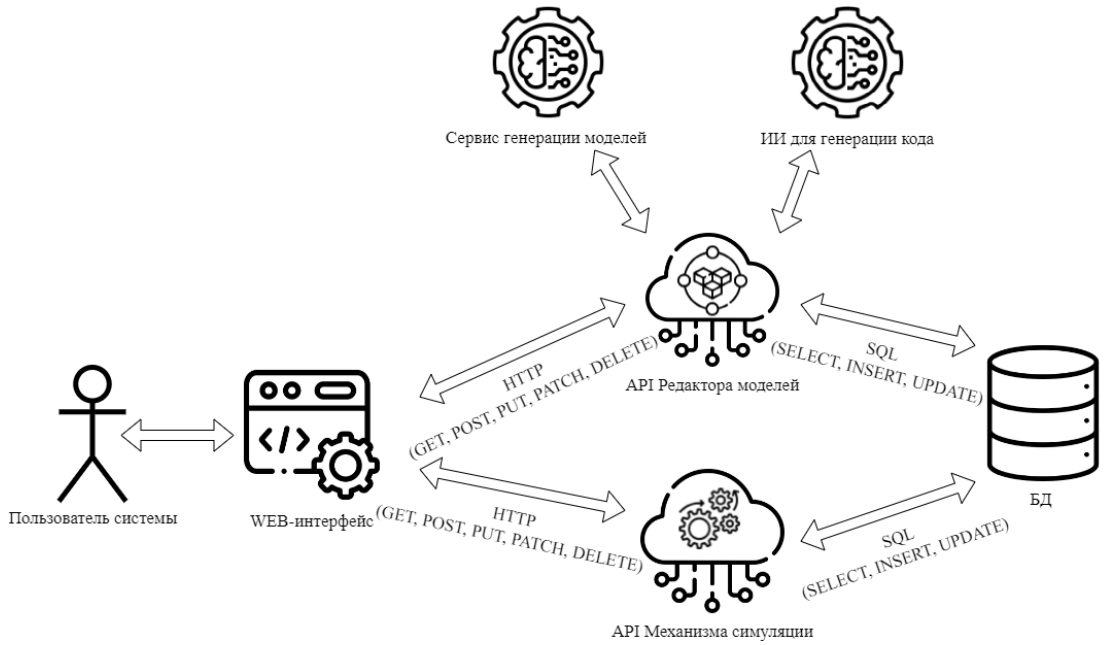


Рисунок 1 – Process Mining и процесс разработки ПО.

Заключение

В этой работе проведён сравнительный анализ наиболее распространённых систем имитационного моделирования на предмет соответствия современным стандартам и подходам к проектированию и разработке ПО. Большая часть из рассмотренных инструментов имеет устаревшую архитектуру, затрудняющую расширение функциональности. В работе представлена сервис-ориентированная архитектура инструмента имитационного моделирования, которая включает в себя веб-приложение, REST-сервер и сервисы машинного обучения и ИИ, которые могут быть развернуты в облачной инфраструктуре.

ЛИТЕРАТУРА

1. Bykov, E. Simulation Modeling Service Design and Implementation / E. Bykov, K. Aksyonov, O. Aksyonova // 2012 UKSim-AMSS 6th European Modelling Symposium. – Malta, 2012. – С. 367-371
2. OpenModelica [Электронный ресурс]. Режим доступа: https://openmodelica.org/
3. RMD [Электронный ресурс]. Режим доступа: https://www.mvstudium.com/
4. AnyLogic [Электронный ресурс]. Режим доступа: https://www.anylogic.ru/
5. SimuLink [Электронный ресурс]. Режим доступа: https://www.mathworks.com/

УДК 681.3.06

Н.А. Кукин, О.С. Реброва (1 курс магистратуры),
В.А. Семенова-Тян-Шанская, к.т.н., доцент

ИСПОЛЬЗОВАНИЕ МЕТОДОВ КЛАСТЕРНОГО АНАЛИЗА ДЛЯ ОЦЕНКИ ЭКОНОМИЧЕСКИХ ПОКАЗАТЕЛЕЙ РЕЙСА СУДНА

Кластерный анализ является важной техникой в быстро развивающейся области, известной как исследовательский анализ данных, и применяется в различных инженерных и научных дисциплинах. Как указано в работе [1] в таких задачах при анализе учитывается большое количество признаков, совокупно определяющих на множестве объектов разбиение, анализ которого помогает аналитику в принятии решений. Хорошо известно, что методы кластеризации можно разделить на иерархические и неиерархические.

В качестве исследуемой предметной области была выбрана «Оценка экономических показателей рейса судна». Экономические показатели морской перевозки, ее рентабельность и целесообразность зависят от типа судна и принимаемого к перевозке груза, а также фрахтовых ставок и конъюнктуры рынка. Для оценки экономических показателей рейса судна были выбраны следующие признаки:

1) Затраты на топливо;
2) Расходы на содержание экипажа;
3) Портовые расходы;
4) Расходы на страхование;
5) Общие расходы на весь рейс;
6) Доход от перевозки груза;
7) Чистый суммарный фрахт;
8) Прибыль, полученная за рейс.

Информация для базы данных собиралась из источника внешней информации, а конкретно из вебсайта. С этой целью был разработан парсер на языке Python, с помощью которого данные извлекались из вебсайтов и сохранялись в базе данных SQLite3 в удобном для дальнейшего использования формате [2].

Схема получения данных и дальнейшая их обработка показана на Рис. 1.

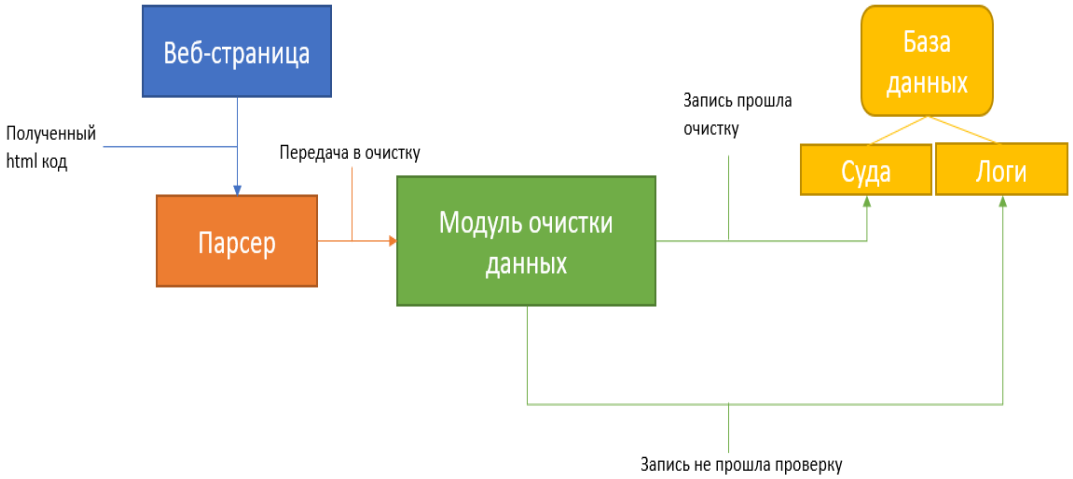


Рисунок 1 – Схема получения данных

После получения данных и формализации их необходимым способом согласно алгоритму из работы [3], может быть выбран один из методов кластерного анализа, используя библиотеку Sklearn [4]. Критерии выбора метода представлены в Таблице 1.

Таблица 1

Алгоритм кластеризации	**Вычислительная сложность**
Иерархический	O(n^2), где n – число точек
K-средних	O($n * k * l$), где n – число точек, k – число кластеров, l – число итераций

Разработанная программа для реализации методов кластерного анализа, использующая языки программирования C#, JavaScript, паттерн MVC, представляет из себя серверную и клиентскую (веб интерфейс) части.

После получения результатов кластерного анализа на серверной части формируется набор данных в JSON формате, который отправляется в пользовательское приложение для отображения его на веб-странице. Для визуализации результатов используются библиотеки Matplotlib и Chart.js [5]. Результат представлен на Рис.2.

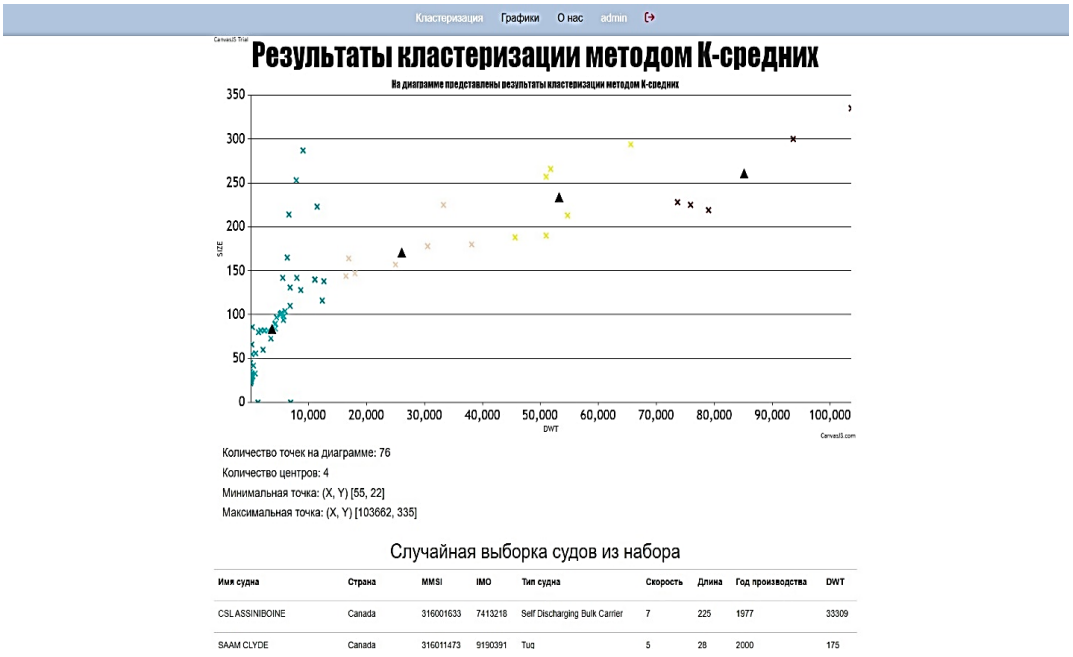


Рисунок 2 – Результаты кластеризации, представленные в веб-приложении

Исходя из графической информации, полученной в результате применения методов кластеризации, аналитик может принимать некоторые решения для оптимизации расходов топлива или улучшения логистики, а также для увеличения прибыли рейса судна.

ЛИТЕРАТУРА

1. Tan P-N., Steinbach M., Kumar V. Introduction to Data Mining (Second Edition) [Book]. – Лондон: Pearson, 2018.
2. Мюллер А., Гвидо С. Введение в машинное обучение с помощью Python. Руководство для специалистов по работе с данными [Книга]. – Москва: Вильямс, 2017.
3. Фролов, В.В. Метод расчета числа кластеров для алгоритма k-means. Экономика. Информатика. (том 47, номер 1) [Текст]/ В.В. Фролов, С.Е. Слипченко, О.Ю. Приходько – Белгород: БГНИУ, 2020. - 213-225 с.
4. Жерон О. Прикладное машинное обучение с помощью Scikit-Learn, Keras и TensorFlow: концепции, инструменты и техники для создания интеллектуальных систем, 2-е издание [Книга]. – Москва: Диалектика-Вильямс, 2020.
5. Бондарев, А.Е. Визуальный анализ кластерных структур в многомерных объемах данных. [Текст]/ А.В. Бондаренко, В.А. Галактионов, Э.С. Клышинский – Москва, 2016 – 24с.

УДК 004.65

Ло Пиньяо (2 курс магистратуры),
И. В. Никифоров, к.т.н., доцент,
А. Д. Ковалев, ассистент

РАЗРАБОТКА АВТОНОМНОГО ХРАНИЛИЩА ДАННЫХ НА ОСНОВЕ APACHE SPARK

В настоящее время предприятия и терминалы производят большое количество производственных данных, социальных данных и других данных каждый день, а традиционные автономные хранилища данных перестают отвечать требованиям к производительности при обработке больших объемов информации. Разработка автономных хранилищ данных на основе современных технологий больших данных может значительно повысить производительность решения задач обработки информации. В крупных компаниях все данные бизнеса хранятся в консолидированном месте, таком, как база данных. Традиционно для хранения используют базы данных Oracle, MySQL и другие СУБД. Но когда речь заходит о, действительно, больших объёмах информации, то данные стараются размещать в NoSQL хранилищах, в том числе для экосистемы Hadoop в таких HDFS и Hive [1]. Использование перечисленных хранилищ позволяет внедрять и применять для решения аналитических задач бизнеса современные вычислительные алгоритмы, такие как MapReduce и его реализацию из проекта Hadoop. Одним из недостатков реализации алгоритма MapReduce в Hadoop является его постоянное взаимодействие с жестким диском для хранения как промежуточных результатов обработки, так и конечных. Этот недостаток устраняется за счет инструмента Apache Spark, который также реализует алгоритм MapReduce, но работает с данными в оперативной памяти. Именно поэтому Apache Spark более эффективен для проведения вычислений над большими данными.

Хранилища данных часто создаются для поддержки Руководства компании и отделов операционного анализа в принятии решений и могут быть созданы для более эффективного управления предприятиями и их бизнесом [2].

Целью данной работы является повышение эффективности обработки и хранения данных в автономных хранилищах данных за счет использования алгоритмов высокопроизводительных распределенных вычислений на основе инструмента Apache Spark.

Система автономного хранилища данных состоит из трех модулей:

– модуль сбора данных;
– модуль автономного хранения данных;

– модуль визуализации.

Общая архитектура системы хранилища данных представлена на Рис. 1. Основные выделяемые уровни программного решения перечислены ниже.

1. Уровень сбора данных в основном собирает поведенческие данные пользователей и данные бизнес-логики в базу данных MySQL или файлы с помощью коллектора Springboot, затем данные в базе данных MySQL инкрементально синхронизируются с Kafka через Maxwell, а данные, хранящиеся в файлах, полностью синхронизируются с Kafka через DataX, где они фильтруются перехватчиком Flume на предмет несоответствующих данных, затем HDFS потребляет данные в Kafka и сохраняет доступные данные в HDFS [3].

2. Уровень автономного хранилища данных в основном создается с использованием теории размерного моделирования и само хранилище данных состоит из следующих пяти модулей [4].

- Модуль ODS (Operation Data Store) используется для хранения информации о необработанных данных в соответствии с источниками данных, которые хранились в HDFS.

- Модуль DIM (Dimission) - это слой размерности, который используется для хранения размерных таблиц, связанных с созданием таблиц фактов.

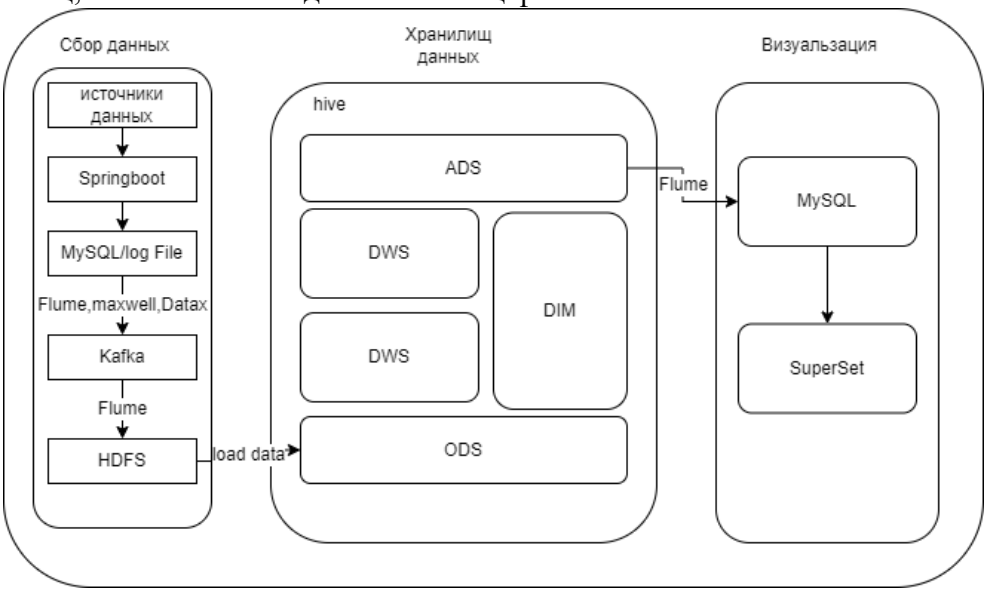


Рисунок 1 – Общая архитектура системы хранилища данных

- Модуль DWD (Data Warehouse Detail) включает подтаблицы для поведенческих данных и информации журнальных данных, в основном, для очистки и преобразования данных.

- Модуль DWS (Data Warehouse Summary) - сводный слой хранилища данных. Является слоем, который отражает активы данных и обычно используется для обобщения данных.

- Модуль ADS (Application Data Store), уровень приложения данных, который представляет собой высоко агрегированные данные и, в основном, используется для представления отчетов.

3. Уровень визуализации данных использует данные, обработанные в хранилище данных, т.е. данные, хранящиеся в модуле ADS. В нем данные хранятся в базе данных MySQL и визуализируются с помощью инструмента SuperSet для принятия управленческих решений.

Предлагаемая программная система автономного хранилища данных реализуется с помощью технологий, перечисленных ниже.

Intellij IDEA – это, в основном, интегрированная среда для языков программирования Java, но также может быть интегрирована с другими языками программирования. Она не только интегрирует Git, Junit, CSV и другой анализ кода, отладочную обработку, чтение данных, обработку, анализ, визуализацию, сохранение, но также имеет отличную среду разработки и интегрирована с различными фреймворками разработки, такими как Springboot и так далее.

Data Grip - это инструмент визуализации SQL, который не только наследует SQL, но и

поддерживает написание различных SQL-запросов, таких как HQL, Clickhouse, MySQL и т.д. Он также интегрируется с технологиями разработки, такими как Git, SVN, Mercurial и т.д. В этом проекте он, в основном, используется для разработки утверждений HQL.

Apache Superset - это современная платформа для исследования и визуализации данных, которая взаимодействует с различными источниками данных, включая многие современные движки для анализа больших данных [5], и имеет богатое графическое представление [6].

Хранилище данных основано на технологии больших данных Hive, где отфильтрованные данные хранятся в HDFS на этапе сбора данных, а затем хранилище данных моделируется на основе Hive. Хранилище данных использует Spark в качестве вычислительного механизма для повышения вычислительной эффективности хранилища данных, поскольку Spark - это вычислительный механизм, производящий вычисления в оперативной памяти [7]. Затем данные визуализируются с помощью SuperSet, который поддерживает различные графики и является визуализацией больших данных.

ЛИТЕРАТУРА

1. Ермаков Н. В., Таленфельд Т. С., Никифоров И. В., Егоров П. Б. Исследование подходов к созданию высокопроизводительного масштабируемого облачного сервиса для дедупликации данных в хранилище // Информатика и кибернетика (ComCon-2016): сборник докладов студенческой научной конференции Института компьютерных наук и технологий, Санкт-Петербург, 04–09 апреля 2016 года, 2016. – С. 144-147.
2. Ло Вэй, Лю ГунЦзун. Текущее состояние исследований в области хранилищ данных на основе больших данных. Новые технологии и новые продукты в Китае, 2020(9):38-39.
3. Voinov N., Drobintsev P., Kotlyarov V., Nikiforov I. Distributed OAIS-Based digital preservation system with HDFS technology // 20th Conference of Open Innovations Association FRUCT: Proceedings, Saint-Petersburg, 03–07 апреля 2017 года / LETI University, St.Petersburg, Russia; S. Balandin, A. Levina, T. Tyutina. – Saint-Petersburg: FRUCT Oy, 2017. – P. 491-497. – DOI 10.23919/FRUCT.2017.8071353.
4. Ли Чонг, Чжан Тунтун, Ду Вэйцзин, Лю Сюэмин. Высокодоступное двухдвигательное хранилище данных на основе Hive. Компьютерные системы и приложения, 2019, 28(9):65-71.
5. Сычев В. К., Ленькова Ю. В., Цветкова Е. А., Никифоров И. В. Анализ данных трендов YouTube // Современные технологии в теории и практике программирования: сборник материалов конференции, Санкт-Петербург, 23 апреля 2020 года / Санкт-Петербургский политехнический университет Петра Великого; Dell Technologies; EPAM Systems. – Санкт-Петербург: ПОЛИТЕХ-ПРЕСС, 2020. – С. 131-132
6. Данилова В.В. Студенческая молодёжь XXI века: наука, творчество, карьера, цифровизация. Сборник материалов Межвузовской научно-практической конференции. Под общей редакцией Е.А. Руднева, под научной редакцией Л.Н. Горбуновой. Москва, 2022. С. 145-150.
7. У Синьдун, Чжи Шэнкай. Сравнение MapReduce и Spark для анализа больших данных. Journal of Software, 2018, 29(6):1770-1791

УДК 004.453

М.С. Лопатин, Н.С. Скоков (1 курс магистратуры),
А.Ю. Гарькушев, к.т.н., доцент

РАЗРАБОТКА КЛИЕНТ-СЕРВЕРНОЙ СИСТЕМЫ ДЛЯ АВТОМАТИЗИРОВАННОГО РАЗВЕРТЫВАНИЯ ИНФОРМАЦИОННОЙ ИНФРАСТРУКТУРЫ ПРЕДПРИЯТИЯ С ПОМОЩЬЮ CLOUD-INIT И ANSIBLE

Работа направлена на удовлетворение потребности в автоматизации развертывания информационной инфраструктуры путем представления разработки клиент-серверной системы, которая автоматизирует процесс развертывания. Основная цель этой системы - исключить ручное вмешательство и повысить эффективность процесса развертывания.

Целью данной работы является снижение трудоёмкости развертывания информационной инфраструктуры проектов или предприятия за счёт создания подхода автоматизированной установки операционной системы и последующей её конфигурации под необходимые задачи с помощью системы управления конфигурациями Ansible и системы инициализации машин Cloud-init.

Ansible — один из наиболее широко используемых инструментов автоматизации для управления конфигурацией и развертывания приложений. Взаимодействие с клиентами основывается на файле inventory — базовом строительном блоке архитектуры Ansible [1]. На сегодняшний день Ansible — самый популярный инструмент автоматизации с открытым исходным кодом на GitHub, который загружают более четверти миллиона раз в месяц [2].

Cloud-init — широко используемое средство для настройки Linux-машины при ее первой загрузке. Часто используется для конфигурирования инфраструктуры частного облачного ресурса и установками на «голом железе» [3].

Модуль клиент-серверной системы для отправки ip-адреса клиента, модификации файлов inventory, запуска выполнения и логирования ansible-плейбуков реализован на языке python [4].

Принцип работы системы следующий:

– Оператор запускает машину, на которой развернута серверная структура системы, после чего запускает клиентские машины.
– Клиенты отправляют DHCP-запрос, получают ip-адрес, загружают образ операционной системы и файлы user-data посредством PXE-загрузки.
– На основе user-data и с помощью cloud-init производится автоматическая установка операционной. В ходе установки создается сервис и скрипт клиентской части системы, который после перезагрузки отправляет на сервер свои данныеданные.
– На сервере скрипт обрабатывает полученные данные, модифицирует inventory-файл, в котором хранится информация о клиентах, после чего запускается выполнение плейбука. Плейбуки для Ansible определяют, как утилита настройки системы Ansible будет работать на управляемых устройствах [5]. Ход выполнения плейбука логируется самим скриптом.
– После успешного выполнения плейбука сервер переходит в режим ожидания, либо обрабатывает запрос следующего клиента.

Архитектура стадий работы системы представлена на Рис. 1-2.

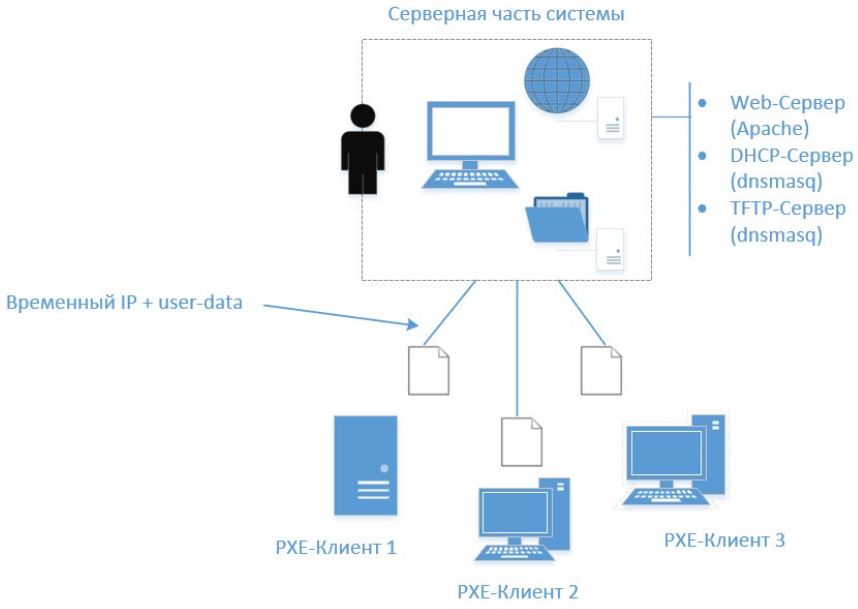


Рисунок 1 – стадия автоматической установки операционной системы

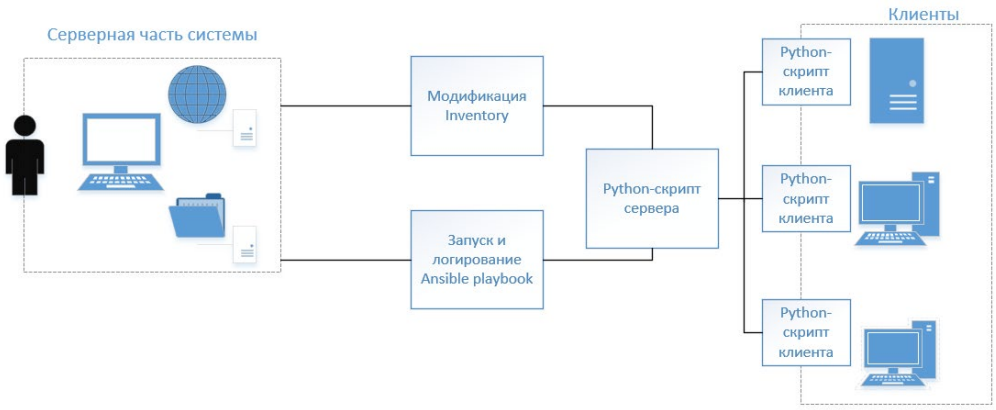


Рисунок 2 – стадия конфигурирования операционной системы

В ходе тестирования работоспособности системы была проведена оценка качества системы. Система проста в эксплуатации, вмешательство оператора практически не требуется. Использование системы значительно снижает трудозатраты сотрудников на развертывание стендов с большим количеством машин, позволяя сконцентрировать трудозатраты на более важных задачах.

Для улучшения работы системы планируется сконцентрировать внимание на корректности написания плейбуков для Ansible, обеспечить обширное покрытие кода тестами для исключения ошибок выполнения сценария. Также для обеспечения кроссплатформенности системы планируется предусмотреть варианты конфигурации cloud-init или иного средства PXE-загрузки под различные операционные системы.

ЛИТЕРАТУРА

1. Keating J. - Mastering Ansible // Packt Publishing - 2015 – 212 pp.
2. Complexity kills productivity. - URL: https://www.ansible.com/overview/it-automation – (дата обращения: 27.01.2023). – Текст: электронный.
3. Cloud-init documentation. [URL]: https://cloudinit.readthedocs.io/en/latest/ (дата обращения: 22.01.2023). – Текст: электронный.
4. Python 3.8 documentation. [URL]: https://www.python.org/ (дата обращения: 10.02.2023). – Текст: электронный.
5. Mallett A. - Working with the Ansible Configuration // Red Hat Certified Engineer (RHCE) Study Guide – 2021 – pp.11-27

УДК 004.42

А. В. Мейник (4 курс бакалавриата),
С. П. Воскобойников, к.ф.м.н., доцент,
Е.Н. Попов, к.ф.м.н., н.с.

РАЗРАБОТКА ПРОГРАММЫ ДЛЯ МОДЕЛИРОВАНИЯ ВЗАИМОДЕЙСТВИЯ ЦЕПОЧКИ ДИПОЛЕЙ

Задача о взаимодействии цепочки диполей в условиях внешнего возбуждения представляет большой интерес для фундаментальной науки, так как в ней моделируется коллективный эффект рассеяния внешнего поля группой из большого количества упорядоченных атомов. Коллективные эффекты позволяют улучшать чувствительность измерительных устройств в прецизионных экспериментах, поэтому данная тема является востребованной в метрологии. Предпосылкой задачи является конструирование зондов точечных электромагнитных излучателей, атомов или молекул, которые должны иметь максимальную квантовую эффективность. Поэтому расположение диполей на поверхности

зонда следует подбирать таким образом, чтобы мощность рассеиваемого поля слабо отличалось от мощности излучаемого поля.

Целью данной работы является разработка программы, позволяющей проводить моделирование взаимодействия диполей с учётом запаздывания и строить зависимость функции угловой поправки интенсивности рассеивания группы из N диполей $\Xi_N(\theta)$ в зависимости от параметров их взаимного расположения, например, дисперсии координат.

Математическая модель взаимодействия описывается системой N линейных дифференциальных уравнений второго порядка с запаздывающим аргументом [2]. На отрезке длины L находятся N точек с координатами x_i :

$$0 \le x_i \le L$$

Координаты x_i являются случайными величинами, которые не зависят друг от друга. Каждый i-тый рассеиватель характеризуется дипольным моментом y_i, который направлен вдоль оси Y. Диполь является индуцированным и подчиняется уравнению:

$$\frac{d^2 y_i}{dt^2} + \gamma \frac{dy_i}{dt} + \omega_0 y_i = \alpha E \sin(\omega t) + \alpha \sum_{\substack{m=1 \\ m \ne i}}^{N} E_{i,m}\left(t - \frac{x_{i,m}}{c}\right), \quad i = 1,2,\ldots,N \tag{1}$$

$$\omega_0 = \omega = \frac{2\pi}{T}, \; x_{i,m} = |x_i - x_m|, \tag{2}$$

где y_i – смещение диполя, γ – скорость затухания диполя рассеивателя, ω_0 – это собственная рассеивателя, ω – частота внешнего поля, α – поляризуемость рассеивателя, E – амплитуда внешнего поля, c – скорость света, $E_{i,m}$ – поле, создаваемое m-тым рассеивателем в точке, где находится i-тый рассеиватель,

$$E_{i,m}(t) = \frac{y_m(t)}{x_{i,m}^3} - \frac{1}{c x_{i,m}^2} \frac{dy_m(t)}{dt}, \tag{3}$$

Начальные условия

$$y_i(t) = 0, \qquad -\frac{x_{i,m}}{c} \le y \le 0, \tag{4}$$

$$\frac{dy_i(t)}{dt} = 0, \qquad -\frac{x_{i,m}}{c} \le y \le 0, \tag{5}$$

В качестве выходной информации в данной работе выбрана функция поправки к рассеиванию поля группой независимых диполей. Она показывает, насколько коллективные эффекты влияют на процесс рассеяния. Обозначим эту функцию символом $\Xi_N(\theta)$ и найдём её угловую зависимость.

$$\Xi_N(\theta) = \lim_{\tau \to \infty} \int_{\tau}^{\tau+T} \left(\sum_{i=1}^{N} \frac{d^2}{dt^2} y_i\left(t - \frac{x_{i,m}}{c}\sin(\theta)\right)\right)^2 dt \tag{6}$$

где угол θ откладывается от первоначального направления распространения луча, падающего на группу диполей-рассеивателей.

Заметим, что постановка задачи (1-5) и целевая функция (6) отличается от постановки задачи в работе [1] учётом запаздывания и формулировкой функции $\Xi_N(\theta)$. Также отметим, что решение задачи имеет сильно осциллирующий характер и система (1-5) не является жёсткой [3].

Так как для вычисления $\Xi_N(\theta)$ требуется знать установившееся решение, то в работе получено полуаналитическое квазистационарное решение системы (1-5), что существенно облегчило вычисление $\Xi_N(\theta)$, которое необходимо сделать для $\theta \in \left[0, \frac{\pi}{2}\right]$.

Для получения сильно осциллирующего решения системы (1-5) на всём временном интервале исследовалось возможность применения программы RETARD [4].

Задача (1-5) решалась при следующих численных значениях параметров:
$\omega_0 = \omega = 10^{15} c^{-1}$, $\gamma = 10^7 c^{-1}$, $c = 3 \cdot 10^{10}$ *см*/*с*, $E = 1$, $\alpha = 1$, $L = 10^{-7}$ *см*, *N=10*.

В работе приводятся результаты применения разработанной программы для вычисления $\Xi_N(\theta)$ для различного расположения диполей на интервале длины *L*.

Разработка программы осуществлена на языке программирования Fortran, входящего в состав MinGW gfortran, который интегрируется с Code::Blocks.

ЛИТЕРАТУРА

1. Шалгуева С.Л., Воскобойников С.П., Попов Е.Н. Разработка программы для моделирования взаимодействия электромагнитных рассеивателей. В сборнике: Современные технологии в теории и практике программирования. Сборник материалов научно-практической конференции. Санкт-Петербург, 2021. С. 181-182.
2. Мышкис А.Д. Линейные дифференциальные уравнения с запаздывающим аргументом. Изд. 3, стереот. URSS. 2014. 360 с.
3. Ракитский Ю.В.,Устинов С.М., Черноруцкий И.Г. Численные методы решения жёстких систем. М., Наука. 1979. 199 с.
4. Хайрер Э.,Нёрсетт С., Ваннер Г. Решение обыкновенных дифференциальных уравнений. Нежёские задачи. М., Мир. 1990. – 512 с.

УДК 004.048

Д.А. Моданов (4 курс бакалавриата),
Т.В. Коликова, ст. преподаватель

РАЗРАБОТКА СИСТЕМЫ ОЦЕНКИ СЛОЖНОСТИ ТЕКСТА НАУЧНЫХ СТАТЕЙ И НОВОСТНЫХ СТАТЕЙ ВУЗОВ РФ

Искусственный интеллект (ИИ) стал незаменимым инструментом для работы с текстом. С развитием алгоритмов обработки естественного языка и машинного обучения теперь можно использовать ИИ для точного измерения сложности текста научных статей и новостных статей высших учебных заведений России. Системы оценки сложности текста научных статей и новостных статей вузов имеют большую актуальность в наше время. С одной стороны, это связано с тем, что в настоящее время количество научных статей и новостных статей растет в геометрической прогрессии. В связи с этим возникает проблема обработки, анализа и понимания всего этого массива информации, особенно для тех людей, у которых ограниченное время и ограниченные ресурсы. Системы оценки сложности текста могут помочь решить эту проблему, позволяя быстро и эффективно определять уровень сложности текста и его подходящую аудиторию.

С другой стороны, системы оценки сложности текста могут помочь улучшить образование и повысить грамотность населения. Разработка таких систем в университетах может помочь студентам и исследователям сократить время, необходимое для чтения и понимания научных статей. Аналогично, создание систем оценки сложности новостных статей может помочь людям, у которых ограниченные навыки чтения и понимания, получать информацию о текущих событиях в мире.

В настоящее время развитие подобных систем происходит в нескольких направлениях.

Во-первых, разработчики и исследователи стараются улучшить алгоритмы и методы оценки сложности текста. В настоящее время, существуют различные методы и подходы для оценки сложности текста, такие как статистические методы, методы машинного обучения и т.д. Исследователи работают над созданием новых алгоритмов, которые могут учитывать различные факторы, такие как длина предложения, сложность лексики, грамматические конструкции и т.д.

Во-вторых, существует растущий интерес к персонализации оценки сложности текста. Это означает, что системы оценки сложности будут учитывать не только уровень сложности текста, но и потребности и возможности конкретной аудитории. Например, система может оценить уровень сложности текста и предложить адаптированный текст для людей с ограниченными навыками чтения и понимания.

В-третьих, системы оценки сложности текста также могут интегрироваться в другие приложения, такие как редакторы текста, электронные книги, онлайн-ресурсы и т.д. Это позволит пользователям быстро оценить уровень сложности текста и выбрать наиболее подходящий текст для чтения или исследования.

Развитие технологий, таких как искусственный интеллект и обработка естественного языка, также будет влиять на развитие систем оценки текста. В будущем, такие системы могут стать еще более точными и универсальными, что позволит людям быстрее и эффективнее обрабатывать и понимать большой объем информации.

Таким образом, системы оценки сложности текста научных статей и новостных статей вузов имеют большую актуальность и могут помочь решить ряд актуальных проблем в области научных исследований, образования и информационной грамотности.

В данный момент существуют сервисы решающие подобные задачи, такие как Text Difficulty Analyzer – сервис, который использует машинное обучение для оценки сложности текста на английском языке. Lexile Analyzer – сервис, который использует метод Lexile, чтобы оценить сложность текста на английском языке, с помощью человека, который проходит тест на понимание прочитанного. LIX Index – сервис, который использует метод LIX, чтобы оценить сложность текста на шведском языке. Главред - сервис, для оценки читаемости текста на русском языке, и аннотирования ошибок. Его основная задача — это придание тексту информационного стиля, данный стиль предназначен для того, чтобы делать сложные тексты простыми и быстро донести суть до читателя. Из-за данного подхода этот сервис не подходит к научному стилю, так как при его использовании можно потерять общий контекст статьи.

Для реализации данного сервиса будут использоваться такие технологии как, scikit-learn, TensorFlow, Keras – это библиотеки машинного обучения, посредством которых будет обучаться классификатор, дающий статье нужную оценку. Библиотеки для обработки текста, а именно его токенизации, лематизации и других операций NLTK, spaCy. Библиотеки для обработки и анализа данных Pandas, NumPY а так же для работы с данными на русском для морфологического анализа pymorphy2, rusenttokenize, rusentlemma.

ЛИТЕРАТУРА

1. Шанский Н.М. Лексикология современного русского языка. - Книжный дом "ЛИБРОКОМ", 2009. - 312 с.
2. Гольдберг Й. Нейросетевые методы в обработке естественного языка. - М.: ДМК Пресс, 2019. - 284 с.
3. Sowmya Vajjala, Bodhisattwa Majumder, Anuj Gupta, Harshit Surana Practical Natural Language Processing. - O'Reilly Media, 2020. - 424 с.
4. Denis Rothman Transformers for Natural Language Processing. - Packt Publishing, 2021. - 384 с.
5. Steven Bird Natural Language Processing with Python. - O'Reilly Media, 2016. - 479 с.

УДК 004.65:004.032.26

Ф. А. Мухин (4 курс бакалавриата),
О. В. Прокофьев, ст. преподаватель

ИСПОЛЬЗОВАНИЕ СУБД POSTGRES ДЛЯ РЕАЛИЗАЦИИ ЗАДАЧ ИСКУССТВЕННОЙ НЕЙРОННОЙ СЕТИ

В настоящее время обучение искусственных нейронных сетей (ИНС) достигло многообещающих результатов в широком спектре прикладных областей, начиная от компьютерного зрения (классификация изображений и обнаружение объектов) и обработки естественного языка (языковое моделирование и машинный перевод) до информационного поиска (система рекомендаций) и многих других [1]. Большие наборы данных неизменно дают более высокую производительность и точность, так как предотвращают переобучение модели и повышают способность сети обобщать эти данные под новые ситуации [2].

Однако, как правило, на обучение тратится больше времени, если данные располагаются не в базе данных, а где-то на локальном компьютере. Научным работникам и разработчикам ИНС требуется перебирать различные конфигурации и тестировать разные модели нейронных сетей, в то время как выделенный канал передачи большого потока информации отсутствует, что занимает много времени на коммуникацию и передачу результатов исследований между сотрудниками.

Одним из способов решения этой проблемы является возможность совместить систему управления базами данных (СУБД) с набирающим популярность машинным обучением, предоставив необходимый функционал (API) – набор функций и методов для удобного взаимодействия между этими технологиями.

Одной из наиболее популярных реляционных СУБД со открытым исходным кодом является PostgreSQL (Postgres) [3], которая позволяет разрабатывать пользовательские функции, расширяющие функциональность базы данных за пределы стандартных команд SQL. Помимо SQL, PostgreSQL также поддерживает несколько процедурных языков, таких как PL/Python, PL/Perl и PL/Java. Это означает, что разработчики могут писать пользовательские функции на этих языках, которые могут выполняться непосредственно в базе данных [6].

Язык Python [7] является одним из наиболее используемых для написания искусственных нейронных сетей и машинного обучения. Существует множество библиотек и расширений, которые делают процесс создания, тренировки и изучения ИНС максимально понятным и эффективным способом. Такие фреймворки как TensorFlow, PyTorch, Theano, Keras и другие дают разработчикам возможность не только настроить и обучить нейронную сеть, но и оперировать специальными структурами – тензорами, воспринимая машинное обучение на более высоком уровне абстракции. Также эти библиотеки предоставляют возможность полноценного ускорения вычислений, с помощью специальных CUDA ядер графических видеокарт NVIDIA, что существенно сокращает время обучения. Язык Python реализуется в PostgreSQL на базе расширения PL/Python как процедурный язык этой базы данных.

Таким образом, *цель* данной работы состоит в повышении эффективности работы с нейронными сетями с точки зрения ускорения создания и сохранения моделей и весов в базу данных, уменьшения времени доступа к обучающим и тестовым данным, а также организации работы с большим количеством нейронных сетей посредством расширения для СУБД Postgres на языке Python.

Для достижения данной цели были поставлены следующие *задачи*:

1. Исследование существующих алгоритмов создания, обучения и сохранения ИНС.
2. Сравнительный анализ найденных алгоритмов с возможностью использования их в базе данных.

3. Предложение собственного API на языке PL/Python в СУБД Postgres для взаимодействия нейронных сетей с базой данных.
4. Разработка данного набора функций и методов.
5. Демонстрация повышения эффективности работы.

Одним из инструментов в создании и обучении нейронных сетей является фреймворк или совокупность связанных и дополняющих друг друга программных платформ. В ходе исследования и сравнительного анализа в качестве таких инструментов был выбран набор фреймворков Tensorflow с Keras [4] как один из наиболее поддерживаемых, удобных и масштабируемых вариантов, обеспечивающих лучшую производительность по сравнению с другими библиотеками [5].

До написания приложения с API процесс создания, сохранения и обучения ИНС занимал много времени, а также требовал перезапуска настройки весов после каждого изменения в архитектуре модели нейросети. Использование базы данных для сохранения весов, моделей и данных обучения позволяет уменьшить время развертывания датасета для обучения, в любое время сохранить и загрузить текущую или другую конфигурацию, быстро запустить уже обученную ИНС с весами из БД.

Функционал (API) состоит из ряда функций, облегчающих изменение наборов данных и осуществляющих обучение нейронных сетей с использованием библиотек TensorFlow и Keras с графическим ускорением (GPU – graphics processing unit). Кроме того, он включает методы сохранения архитектуры модели и весов по времени и по названию, что позволяет быстро развернуть любую из обученных моделей, а также иметь целые группы и массивы моделей ИНС, с возможностью просмотра настроек и сравнения процесса обучения. Помимо этого, в API присутствуют функции настройки конфигурации планируемой нейронной сети, с параметрами и с импортированием из файла или настройка в графическом интерфейсе пользователя (GUI).

Помимо вышеупомянутых функций, еще одним примечательным аспектом данной работы является тестирование и адаптация разработанного API для обучения и хранения нейронной сети прямого распространения, распознающей болезни пшеницы на основе текстурных параметров Харалика. Преимуществом данного подхода является использование компактных данных в виде цифровых образов, что позволяет осуществлять самостоятельную свертку и организовывать более простую нейронную сеть. В результате передача и обработка данных могут быть оптимизированы и ускорены, что приводит к более эффективным и предсказуемым результатам.

ЛИТЕРАТУРА

1. Алемасов Е. П., Зарипова Р. С. Перспективы применения технологий машинного обучения / Информационные технологии в строительных, социальных и экономических системах. 2020. № 2 (20). С. 32-34.
2. Пырнова О. А., Зарипова Р. С. Методы и проблемы переобучения многослойной нейронной сети / Информационные технологии в строительных, социальных и экономических системах. 2020. № 2 (20). С. 101-102.
3. Рогов Е. В. PostgreSQL 14 изнутри. – М.: ДМК Пресс, 2022. – 660 с.
4. Шолле Ф. Глубокое обучение на Python. СПб.: Питер, 2022. – 400 с.
5. Mark Ryan. Deep Learning with Structured Data. Manning Publications Co., 2020. – 264 с.
6. Документация к Postgres Pro Standard 14. [Электронный ресурс] Режим доступа: https://www.postgrespro.ru/docs/postgrespro/14/ (дата обращения: 19.03.2023)
7. Python programming language. [Электронный ресурс] Режим доступа: https://www.python.org (дата обращения: 19.03.2023)

УДК 004.738

А. И. Наумчик (4 курс бакалавриата),
О. В. Прокофьев, ст. преподаватель

СПОСОБЫ УСТАНОВЛЕНИЯ ДВУСТОРОННЕЙ СВЯЗИ СЕРВЕРА И КЛИЕНТА, ИСПОЛЬЗУЮЩИХ NAT

Отсутствие внешнего IP-адреса может стать серьезным препятствием для общения в современном мире. Поэтому целью данной работы является исследование различных методов установления двусторонней связи без внешнего IP при том, что и сервер, и клиент находятся за NAT.

Мы будем исследовать модель TCP/IP [5]. TCP (Transmission Control Protocol) – протокол, работающий на верхнем уровне между двумя системами управления: приёмом и передачей информации (между клиентом и сервером). Он определяет, каким образом информация должна быть разбита на пакеты и отправлена по каналам связи. Каждый информационный пакет содержит IP-адреса (Internet Protocol) компьютера-отправителя и компьютера-получателя.

Актуальность проблемы заключается в том, что организовать работу с сервером, который находится за NAT без доступа к нему достаточно сложно напрямую. В данном случае понадобится использовать сторонние сервера посредники, через которые организуется эта работа.

Мы также будем рассматривать протокол IPv4, который использует 32-битный адрес и технологию NAT (Network Address Translation), которая позволяет всем устройствам из локальной сети выходить в интернет. Хотя NAT обеспечивает множество преимуществ, таких как уменьшение количества требуемых общедоступных IP-адресов, он также может вызывать проблемы при установлении двусторонней связи между сервером и клиентом.

Сервер, находящийся за NAT, может быть недоступен для клиента, не входящего в пределы NAT. Это связано с тем, что устройство NAT не имеет таблицы сопоставления IP-адреса сервера и номера порта. Когда клиент пытается подключить соединение с сервером, устройство NAT не сможет переслать пакет на сервер, так как в его таблице сопоставления нет записи об IP-адресе сервера и номере порта. Таким же образом и сервер, когда пытается отправить ответ обратно клиенту, устройство NAT не сможет переслать пакет, поскольку в таблице сопоставления нет нужных записей. Поэтому пакеты будут отбрасываться устройством NAT, и клиент не сможет установить соединение с сервером. На Рисунке 1 представлено соединение клиента с сервером, в NAT изображена таблица преобразования адреса источника в адрес назначения.

Рисунок 1 – Связь компьютера с внешним сервером

Это позволяет устройству с частным IPv4-адресом получать доступ к ресурсам вне своей частной сети, включая ресурсы в Интернете. Преобразование NAT используется в различных целях, однако основной задачей данного механизма является экономия публичных IPv4-адресов [1].

Также для установления связи сервера и клиента может понадобиться такой метод, как обход NAT. Этот метод позволяет устройствам, находящимся за NAT-маршрутизатором, установить связь друг с другом, используя специальный протокол обхода NAT [2].

Мы можем рассмотреть общий план взаимодействия удаленного сервера, пользователя и интернета (Рисунок 2). Пользователь отправляет запрос на удаленный сервер через интернет. Передача информации между клиентом и сервером осуществляется по определенным правилам – протоколу. Основным протоколом, по которому взаимодействуют компьютеры при запросе и получении веб-страниц в настоящее время является HTTP-протокол [3]. Если у пользователя есть шлюз с транслятором (NAT), то запрос сначала проходит через шлюз, который изменяет адрес и порт отправителя.

Если мы не можем повлиять на шлюз с транслятором (NAT), то мы можем использовать технологию ngrok [4], которая создает общедоступный адрес для нашего локального сервера. Таким образом, запросы будут направляться на ngrok, который перенаправляет их на наш локальный сервер. Ответы также проходят через ngrok и передаются обратно пользователю через интернет.

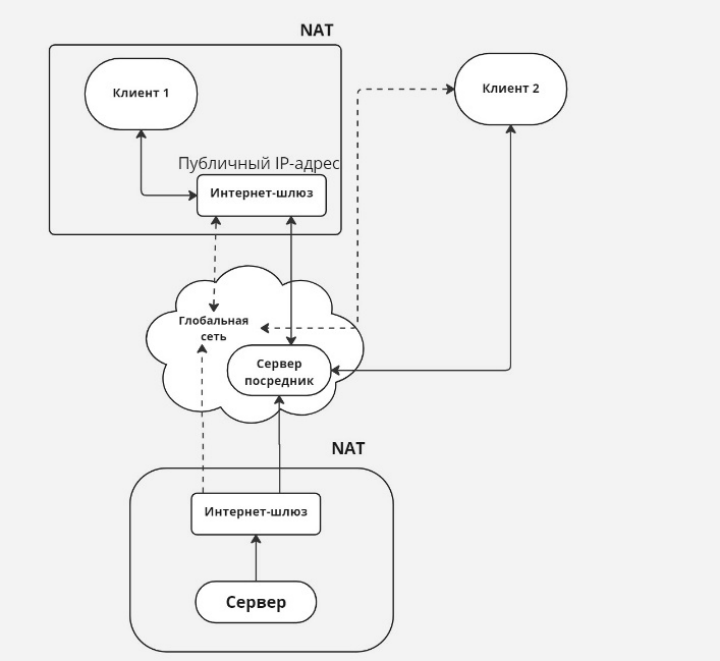


Рисунок 2– Общий план взаимодействия

ЛИТЕРАТУРА

1. Безопасность компьютерных сетей: учеб. пособие / А. С. Коноплев [и др.]. – СПб. : ПОЛИТЕХ-ПРЕСС, 2022. – 90 с.
2. NAT Traversal for Peer-to-Peer Communication: A Comparative Study / A. Razaque, M. Shahid, M. Zia, Journal of Network and Computer Applications, 2017.
3. Методики разработки современных веб-приложений: учеб, пособие / В. М. Тучкевич. - СПб.: Изд-во Политехи, ун-та, 2014. – 135 с.
4. Ngrokking. Организация удаленного доступа без белого IP [Электронный ресурс]. // Сайт: https://habr.com. URL: https://habr.com/ru/post/674070/
5. Компьютерные сети и системы: учеб. пособие / А.Б. Анисифоров. – СПб., 2022. – 98 с.

УДК 32.973

А.Ю Ренжин, С.С. Соболев (4 курс бакалавриата),
Е.А. Муравьев, к.т.н., доцент

ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ НА ОСНОВЕ АВТОМАТИЧЕСКОГО ПОИСКА АНАЛОГОВ

В докладе рассмотрен этап построения семантической модели предметной области (ПрО) в жизненном цикле программного обеспечения (ПО). Наш тестовый класс ПО – программно-геометрические модели в САПР, например КОМПАС. В качестве источников эмпирических знаний о ПрО взят общероссийский классификатор единой системы конструкторской документации (ОК ЕСКД). Синтез семантической модели ПрО рассмотрен на примере группы 7411, сохраняя терминологию ОК – *Детали - не тела вращения плоскостные / Плоскостные с паралл. осн. плоскостями / С плоск. гладкими, без пазов, с контуром прямолин. трех- и четырехуг.* Изделие кодируется 6-ти символьным кодом: класс – 2 цифры, и по одной цифре кода отводится на подкласс, группу, подгруппу, вид.

Классификация представляет дерево с ребрами, помеченными классифицирующими признаками. Применяя алгоритм дихотомии (будет опубликован позже), дерево доводится до диалектического дихотомического дерева (ДД). Фрагмент ДД приведен на Рис.1.

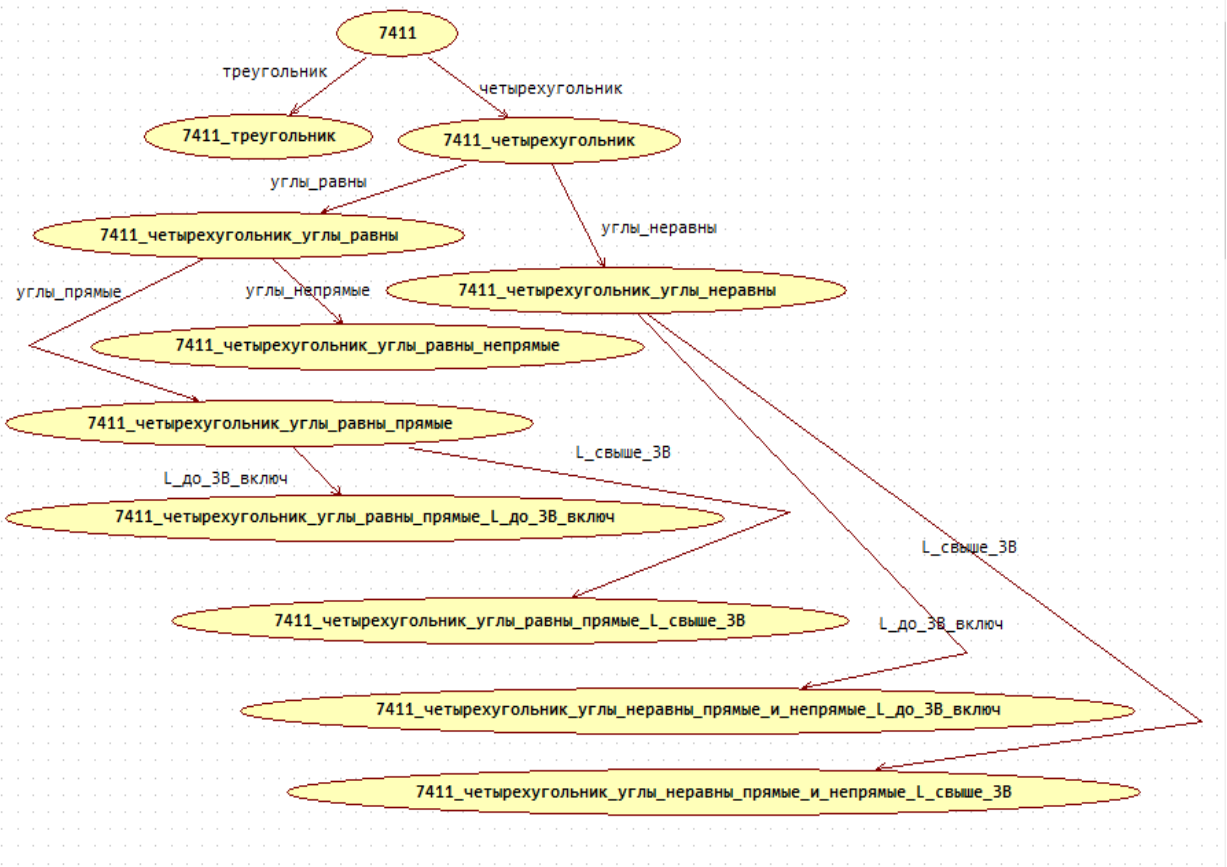


Рисунок 1 – Дихотомическая классификация подгрупп группы 7411 ОК ЕСКД.

ДД индуцирует топологическое пространство[1], на котором могут быть решены задачи проектирования ПО[2 - 4]. В данном докладе решается задача семантического поиска аналога. Постановка задачи: задан список *L* объектов, которым сопоставлены их программные модели, задан объект *запрос*. Найти в *L* и сортировать *n* объектов, семантически ближайших к объекту *запрос*. Объекты списка *L* и запроса представлены иерархическими термами, семантика понятий которых определена ДД, типа представленного выше.

Предварительно выполняется выравнивание по структуре иерархических термов списка *L* и запроса. Затем выполняется отображение выровненных термов в векторы понятий. Алгоритм семантического поиска представлен на Рис.2.

```
Сканирование списка L по i:
{
      Если L[i] не дальше от терма запрос, чем n-й элемент
           текущей выборки, или выборка короче n,
      То поместить L[i] в выборку на соответствующее место
           по удалению от запрос.
      Если при этом выборка становится длиннее n, удалить ее
           последний элемент.
}
```

Рисунок 2 – Алгоритм семантического поиска.

Ключевыми элементами алгоритма являются топологические операции вычисления семантического расстояния объекта *L*[*i*] до терма *запрос* и сортировка элементов выборки по семантической удаленности от запроса.

Приведем пример запроса и выборки. Ответом на запрос: *(деталь, (с_отверстиями, резьба, nil), nil)* , примененный к видам подгруппы *74111* с *n*=3 будет выборка:

- *741114 С отв. круглыми,*
- *741116 С отв. некруглыми,*
- *741118 С отв. кругл. и некругл.*

Из топологических соображений из множества видов подгруппы *74111* отброшено изделие – *741111 Без отв.* Требуется семантическая фильтрация не только по топологическим признакам, но и семантический анализ возможной сочетаемости признаков изделия. Так некруглые отверстия не могут обладать резьбой, поэтому член – *741116 С отв. некруглыми* выборки должен быть отброшен. Резонер онтологической платформы может вывести невозможность *резьбы* у всех видов рода – *некруглое отверстие* при более детализированном запросе аналогов.

Рализация топологических операций выполнена на платформе DotNetRDF[5].

Техника извлечения семантической модели ПрО из классификатора может применяться к другим предметным областям разрабатываемого ПО, в которых действуют соответствующие классификаторы, см. ОКОК – Общероссийский классификатор информации об общероссийских классификаторах.

ЛИТЕРАТУРА

1. Муравьев Е.А. Топологическая модель знаний для работы с предметной семантикой данных и алгоритмов управления в интеллектуальных системах // Труды международных конференций по морской робототехнике для освоения океана. – Санкт-Петербург: СПбГМТУ, 2019, с. 325 – 337.
2. Вайнт К.С., Муравьев Е.А., Абстрагирование UML диаграммы построением полигона вороного. Сборник материалов конференции, 19 апреля 2019 г. – Санкт-Петербург: ПОЛИТЕХ-ПРЕСС, 2019 – стр. 143 – 144.
3. Иванов Г.А., Королев К.И., Семенча В.Е., Муравьев Е.А., Сравнение спецификаций и программ в топологической модели семантики, Сборник материалов конференции, 26 апреля 2021 г. – Санкт-Петербург: ПОЛИТЕХ-ПРЕСС, 2021 – стр. 163 – 164.
4. Кириллов Н.И., Муравьев Е.А., Доменная семантика в задачах технологии программирования, Сборник материалов конференции, 26 апреля 2022 г. – Санкт-Петербург: ПОЛИТЕХ-ПРЕСС, 2022 – стр. 84 – 86.
5. Библиотека dotNetRDF, документация и учебник [Электронный ресурс] Режим доступа: https://dotnetrdf.org/docs/stable/user_guide/index.html

УДК 004.415.2

Л. К. Саськов (4 курс бакалавриата),
Д. А. Щитинин

ПРИМЕНЕНИЕ МЕТОДОВ МАШИННОГО ОБУЧЕНИЯ ДЛЯ РАНЖИРОВАНИЯ ПОИСКОВОЙ ВЫДАЧИ КОНТЕНТА

Целью данной работы является разработка модели машинного обучения для ранжирования поисковой выдачи контента портала “Одноклассники”.

При ранжировании поисковой выдачи учитываются разные факторы, их полное множество защищено NDA, но не будет преступлением рассказать, что основными являются показатели совпадения запросу, свежесть публикации, близость к интересам пользователя. Факторы элемента можно представить как вектор чисел с плавающей точкой (вектор фич), элементы которого показывали бы интенсивность того или иного сигнала. Распространенным решением упорядочивания поисковой выдачи является задание линейной функции, которая бы считала ранг конкретного элемента выдачи на основе его вектора фич и вектора коэффициентов – множителей для каждой фичи. Вектор коэффициентов чаще всего задается вручную экспертами. Такой подход прост и легко интерпретируем, но для разных ситуаций могут быть различные коэффициенты, подбирать их вручную и поддерживать актуальность со временем становится все сложнее и сложнее.

В связи с этим более удачным и перспективным решением являются методы ранжирования, основанные на предпочтениях самих пользователей сервиса. Данную задачу можно рассматривать в качестве задачи машинного обучения с учителем – обучение ранжированию. Задача заключается в автоматическом подборе ранжирующей модели по обучающей выборке, состоящей из множества списков и заданных частичных порядков на элементах внутри каждого списка. Частичный порядок задается путем указания оценки для каждого элемента (например, “релевантен” или “не релевантен”). Цель ранжирующей модели – наилучшим образом приблизить и обобщить способ ранжирования в обучающей выборке на новые данные. Одним из таких методов является построение (обучение) ансамбля деревьев, которые будут формировать ранг. Цель ансамбля – получение рангов элементов любой поисковой выдачи так, чтобы наиболее релевантные элементы (по мнению пользователей) были выше.

В рамках данной работы была введена функция релевантности элемента поисковой выдачи, проведен сбор необходимых данных за заданный период, построен ансамбль деревьев с помощью метода градиентного бустинга, реализованного в библиотеке XGBoost, а также доказана эффективность полученной модели доверительным A/B тестированием.

Введем функцию релевантности конкретного элемента поисковой выдачи как отображение из множества элементов в множество {0, 1}: 0 – клика по элементу не было, 1 – клик был. Обработаем все поисковые выдачи контента за заданный период: возьмем первые 10 элементов выдачи, вычислив для каждого элемента релевантность. При сборе данных исключим выдачи, в которых релевантность для первых 10 элементов была 0. Такие выдачи не повлияют на финальный результат и могут замедлить обучение модели.

На основе полученных данных была обучена модель градиентного бустинга с использованием библиотеки XGBoost. Функция ошибки для обучения – сумма количества инверсий релевантностей в рамках каждой поисковой выдачи. Соответственно, цель модели сделать так, чтобы в как можно большем количестве запросов выше оказывались элементы с большей релевантностью.

Эффективность модели была проверена доверительным A/B тестированием. Тестирование показало наличие роста кликов по контенту в поисковой выдаче, что говорит об улучшении ее релевантности. Данные A/B тестирования с надежностью 0.99 на рост кликов в интервале от 5 до 14 процентов.

ЛИТЕРАТУРА
1. XGBoost documentation. [Электронный ресурс]. Режим доступа: https://xgboost.readthedocs.io/en/stable/
2. D. Turnbull, J. Kowalewski, Berlin Buzzwords conference 17. Elasticsearch Learning to rank. [Электронный ресурс]. Режим доступа: https://www.youtube.com/watch?v=JqqtWfZQUTU&ab_channel=PlainSchwarz
3. F. Casalegno, Learning to rank: a complete guide to ranking using machine learning. [Электронный ресурс]. Режим доступа: https://towardsdatascience.com/learning-to-rank-a-complete-guide-to-ranking-using-machine-learning-4c9688d370d4
4. Elastic search documentation. Learning to rank core concepts. [Электронный ресурс]. Режим доступа: https://elasticsearch-learning-to-rank.readthedocs.io/en/latest/core-concepts.html
5. D. Turnbull, Learning to rank for search. [Электронный ресурс]. Режим доступа: https://opensourceconnections.com/blog/2017/08/03/search-as-machine-learning-prob/

УДК 004.42

В. Д. Рудницкий (4 курс бакалавриата),
Б. М. Медведев, к.т.н., доцент

ОБНАРУЖЕНИЕ НЕИЗВЕСТНЫХ РАДИОСИГНАЛОВ В ЧАСТОТНОМ СПЕКТРЕ

С развитием беспроводных технологий количество доступных радиочастот для передачи данных становится все меньше. Поэтому как-никогда раньше есть необходимость в поиске более эффективных способов использования частот. Одним из таких способов является создание когнитивного радио [1], в котором для увеличения скорости передачи данных при использовании ограниченного радиоспектра применяется обнаружение и идентификация доступных для передачи сигналов частотных диапазонов. Для построения такой системы нужно реализовать алгоритм обнаружения занятых участков спектра путем анализа сигнала в частотном диапазоне. Помимо когнитивного радио, данный алгоритм может использоваться в системах радиоконтроля, а также для автоматизации тестирования работы радиоприемников и радиопередатчиков.

Существует ряд алгоритмов обнаружения сигналов, основанных на знании законов распределения сигналов и шума, которые широко используются в комплексах мониторинга радио обстановки в широком диапазоне частот [2]. Еще одним подходом решения этой задачи является машинное обучение. В качестве примера можно привести инструментарий TorchSig [3], основанный на фреймворке машинного обучения pyTorch [4] языка программирования Python. Данное решение обладает высокой вычислительной сложностью, а эффективность обнаружения можно получить только для широкополосных сигналов [5].

Системы мониторинга радиоспектра в реальном времени накладывают ограничения на время выполнения алгоритма обнаружения, что приводит к ограничению вычислительной сложности в реализации алгоритмов и использованию ограниченных по длительности одиночных кадров радиоспектра. Кроме того, для дальнейшего анализа необходимо оценить параметры обнаруженных сигналов: несущую частоту, ширину полосы, мощность.

Целью работы является разработка программных средств обнаружения неизвестных радиосигналов в частотном спектре, а также оценка параметров обнаруженных сигналов. Для достижения поставленной цели нужно решить следующие *задачи*:

1. Разработка алгоритмов обнаружения неизвестного сигнала и оценки их параметров.
2. Разработка программных средств, включающих реализацию алгоритмов обнаружения и графического интерфейса пользователя для отображения спектра сигналов и результатов измерений.
3. Тестирование и экспериментальное исследование разработанных программных средств.

Разработанный алгоритм обнаружения неизвестного сигнала использует порог обнаружения для отличия сигнала от шума (см. Рис. 1). Порог задается относительно предварительно оцененного шума, измеряемого радиоприемником. Измеряемыми

параметрами являются средняя (несущая) частота, ширина полосы и мощность сигнала. Определение мощности шума может осуществляться калибровкой по спектру, в котором отсутствуют сигналы, или определением по спектрам с сигналами в эфире.

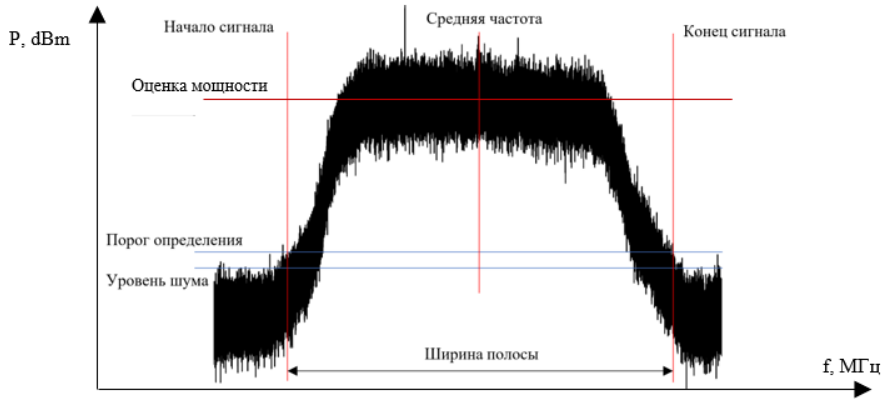


Рисунок 1 – Спектр сигнала с обозначенными параметрами

Была разработана программа на языке C++ в среде разработке Visual Studio 2022 [6] с использованием фреймворка Qt [7]. Она включает в себя модуль обнаружителя и модуль графического интерфейса (GUI). GUI позволяет прочитать файл с записями частотного спектра радиосигналов, представляющий из себя массив чисел формата float – квадрат модуля спектра в логарифмическом масштабе, отобразить спектр на графике и обозначить на нем обнаруженные сигналы. Также GUI позволяет указать частотный диапазон анализа и передать модулю определителя данные с заданными настройками. Модуль определителя обрабатывает полученные данные и возвращает GUI данные об обнаруженных сигналах (несущую частоту, ширину полосы и мощность).

Для тестирования был разработан программный генератор тестовых сигналов в частотном спектре, позволяющий синтезировать сигнал на входе системы обнаружителя сигнала с заданными параметрами: спектральную плотность шума или отношение сигнал шум для группы тестовых сигналов, для каждого сигнала определяется несущая частота, модуляция и мощность. Разработанные программные средства были протестированы на данных, полученных с помощью генератора, а также на экспериментально полученных данных.

Разработанные программные средства обнаружения и оценки параметров сигналов могут быть использованы в системах мониторинга радиоспектра, разрабатываемых в Специальном Технологическом Центре.

ЛИТЕРАТУРА

1. IEEE. 802 LAN/MAN Standards Committee 802.22 WG on WRANs (Wireless Regional Area Networks) / Institute of Electrical and Electronics Engineers. – 2011.
2. Стоянов Дмитрий Драганович Разработка и исследование алгоритмов обнаружения сигналов в когнитивных радиосетях : диссертация на соискание ученой степени кандидата технических наук : 05.12.13 / Ярославский государственный университет имени П. Г. Демидова – Ярославль, 2014.
3. TorchSig. A PyTorch Signal Processing Machine Learning Toolkit [Электронный ресурс] Режим доступа: https://torchsig.com/
4. Pytorch [Электронный ресурс] Режим доступа: https://pytorch.org/
5. Boegner L., Vanhoy G., Vallance P., Gulati M., Feitzinger D., Comar B., Miller R.D. Large Scale Radio Frequency Wideband Signal Detection & Recognition / Peraton Labs, Laboratory for Telecommunication Sciences, Applied Insight // arXiv preprint arXiv:2211.00918v1 [cs.SD]. – 2022.
6. Visual Studio 2022 [Электронный ресурс] Режим доступа: https://visualstudio.microsoft.com/ru/vs/
7. Фреймворк Qt [Электронный ресурс] Режим доступа: https://www.qt.io/product/framework

УДК 004.056

Н.С. Скоков, М.С. Лопатин (1 курс магистратуры),
В.М. Крундышев, к.т.н., доцент

ПРОГНОЗИРОВАНИЕ АНОМАЛЬНОГО ПОВЕДЕНИЯ ПОЛЬЗОВАТЕЛЕЙ В КИБЕРСРЕДЕ С ИСПОЛЬЗОВАНИЕМ БИОМИМЕТИЧЕСКОГО МЕТОДА

Подходы к разработке бизнес-систем с применением алгоритмов машинного обучения (МО) с каждым годом приобретают более значимую роль в задачах кибербезопасности. Необходимость в интеграции искусственного интеллекта сегодня подтверждается ежегодным отчетом российской антивирусной компании «Kaspersky», в журнале «Kaspersky Security Bulletin», о росте количества зафиксированных инцидентов информационной безопасности в правительственных организациях по сравнению с 2021 годом: низкоуровневые в 3 раза, среднеуровневые в 2 раза и высокоуровневые в 1,5 раза [1]. Основные причины для развития внедрения МО в кибербезопасности это:

1. Снижение количества доступных технических специалистов.
2. Проблемы с оборудованием и отказ от западных облаков.
3. Рост количества атак и угроз, в том числе спровоцированных геополитикой.

Целью работы является актуализация подхода на примере разработки системы прогнозирования аномального поведения пользователей сетевой среды с помощью когнитивного алгоритма МО с иерархической временной памятью (Hierarchical Temporal Memory – HTM). В задачи работы входит выбор набора данных для обучения, проведение предварительной обработки признаков аномалий и анализ полученных результатов.

Алгоритм иерархической временной памяти – это технология машинного самообучения, нацеленная на теоретическое описание структурных и алгоритмических свойств коры головного мозга. Сеть HTM представляет из себя иерархическую структуру регионов памяти, обеспечивающих низкоуровневое представление информации об объекте [2].

Для обучения системы прогнозирования аномального трафика имеется ряд доступных публичных наборов данных: UNSW_2018_IoT_Botnet, NSL-KDD, ADFA2013 и др. В рамках данного проекта был выбран набор данных UNSW_2018_IoT_Botnet [3], содержащий 73 миллиона записей. Для обучения модели была сформирована выборка из 9000 записей. Каждая запись представляет собой распакованный с *dump-файла feature-вектор*, который состоит из множества параметров сетевого соединения (27) и метки (нормальный (normal) или аномальный (attack flag) трафик).

Предварительная обработка набора данных сводится к бинарной векторизации *feature-векторов* и определении иерархической архитектуры алгоритма (Рис. 1)

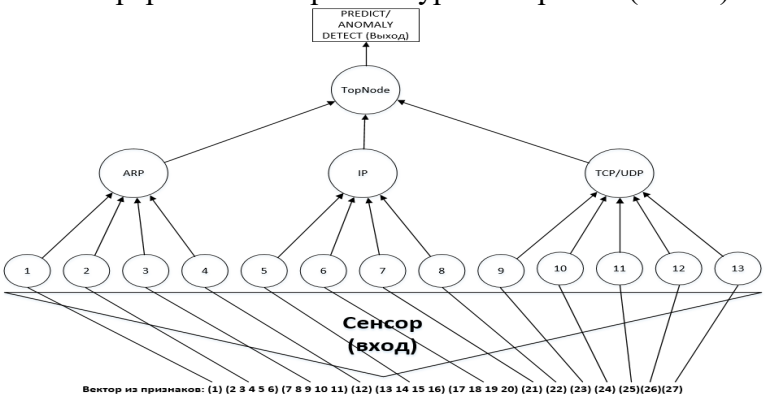


Рисунок 1 – Пример разложения информации о входящем объекте в низкоуровневое иерархическое представление

Каждая запись раскладывается на feature-вектор состоящий из: (1) – порядкового номера записи; (2-6) – mac-адреса источника; (7-11) – mac-адреса получателя; (12) – кода сетевого протокола; (13-16) – IP-адрес источника; (17-20) – IP-адреса получателя; (21) – длины пакета;

(22) – IP-флагов; (23) – протокола передачи данных; (24) – порта источника; (25) – порта получателя; (26) – TCP-флагов; (27) – количества переданных байт.

Преимуществом данного алгоритма является то, что модель может обучаться без учителя (без учета меток сетевого трафика) в следующих режимах: последовательно (с возможностью корректировки процесса обучения) и непрерывно в онлайн-режиме (самообучаясь). Модель обладает памятью, что позволяет ей в процессе обучения делать собственные «умозаключения». Иерархическая структура модели обеспечивает распределенное представление информации об объекте. Благодаря иерархии представлений удается обучать большие и сложные модели невероятно эффективным образом [4, 5]. При этом в процессе обработки информации в модели задействуются лишь определенные группы (клетки) уровней (регионов) иерархии.

В процессе обучения, модель в реальном времени принимает на вход бинарные векторы. Использовался фреймворк Numenta NuPic для языка Python [6]. На раннем этапе обучения алгоритм плохо разбирается в классификации аномалий и порождает много ложных срабатываний (Рис. 2).

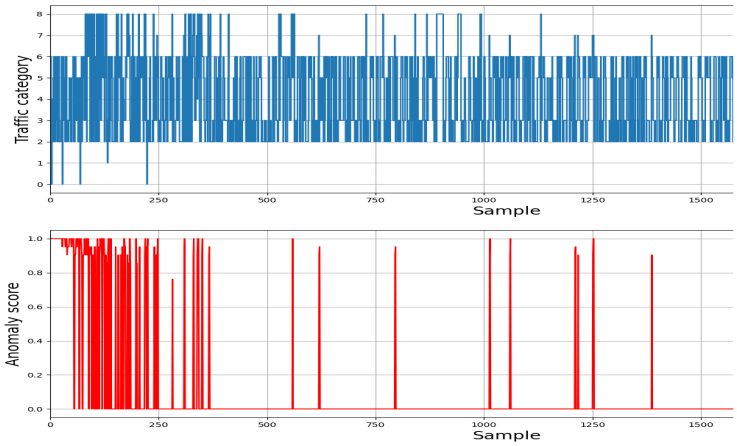


Рисунок 2 – Результаты тестирования модели. График прогнозирования аномалий трафика

Однако, спустя некоторое время, модель учится определять ряд некоторых зависимостей и в результате начинает верно классифицировать трафик (Рис. 3). Следовательно, со временем точность модели возрастает.

В результате исследования алгоритма в задаче прогнозирования аномального поведения сетевого трафика удалось достичь метрики МО accuracy (доля правильных ответов) – 0,87. Правильное внедрение и обучение HTM модели позволит обеспечить более высокую киберустойчивость информационных систем за счет своевременного прогнозирования аномального поведения и вектора реализации деструктивных воздействий или целенаправленных сетевых атак.

ЛИТЕРАТУРА

1. Что ждет российские SOC в 2023 году / Kaspersky Security Bulletin : [сайт]. — URL: https://securelist.ru/soc-socc-predictions-2023/106492/ (дата обращения: 15.03.2023).
2. Иерархическая темпоральная память (HTM) и ее кортикальные алгоритмы обучения / Numenta : [сайт]. — URL: https://www.numenta.com/assets/pdf/whitepapers/hierarchical-temporal-memory-cortical-learning-algorithm-0.2.1-ru.pdf.
3. UNSW_2018_IoT_Botnet : электронная база данных / University of New South Wales. — URL: https://research.unsw.edu.au/projects/bot-iot-dataset (дата обращения: 17.02.2023).
4. Украинцева Д. А. Методы обнаружения кибератак с помощью машинного обучения / Д. А. Украинцева, В. В. Грызунов // Инновационное развитие информационных систем и технологий в гидрометеорологии. — 2022. — С. 92-96.
5. Хокинс Джефф, Джордж Дайлип. Иерархическая темпоральная память // Точки над Ё. — 2013. — Т. №3(8). — С. 62-97.
6. Numenta NuPic: фреймворк для машинного обучения на Python 2 : [сайт]. — URL: https://www.numenta.com/ (дата обращения: 17.02.2023).

УДК 004.42

Н. А. Соколов (4 курс бакалавриата),
Б. М. Медведев, к.т.н., доцент

РАЗРАБОТКА ПРОГРАММНЫХ КОМПОНЕНТОВ ДЛЯ ОТОБРАЖЕНИЯ И АНАЛИЗА КАДРОВОЙ СТРУКТУРЫ БИТОВОГО ПОТОКА

Практически все современные цифровые телекоммуникационные системы используют кадры для передачи данных. Кадр является структурированным повторяющимся временным сегментом архитектуры канала связи, который обеспечивает предсказуемые во времени коммуникационные действия между его началом и концом [1]. Качество работы системы кадровой синхронизации непосредственно влияет на помехоустойчивость и скорость передачи информации [2]. При разработке аппаратно-программных средств систем кадровой синхронизации необходимо иметь средства отладки и тестирования, которые позволяют отображать кадровую структуру данных в различных форматах для последующего анализа.

Отладка и тестирование систем синхронизации занимает много времени, имеет большую трудоемкость, причем большинство сложностей сосредоточены в отладке в реальном времени, в основном при помощи специализированных аппаратных средств [3]. Проектирование и разработка программных решений все еще имеют потенциал для рассмотрения.

Целью работы является разработка программных средств отладки и тестирования систем кадровой синхронизации, которые позволяют отображать и анализировать в реальном времени кадровую структуру битового потока данных.

Для достижения поставленной цели необходимо решение следующих задач:

1. Разработка требований к программным средствам тестирования систем кадровой синхронизации.

2. Разработка программных средств графического интерфейса пользователя, хранилища поступающих данных и анализа кадровой структуры.

3. Экспериментальное исследование созданных программных средств.

Требования к программным средствам тестирования систем кадровой синхронизации включают:

– установку периода обновления изображения (в мс);

– установку периода отображения битового потока;

– задание способа отображения кадра (графический, бинарный, шестнадцатеричный, текстовый с выбором кодировки);

– установку масштаба изображения в графическом режиме (1–32 точки на бит);

– включение/выключение инверсии отображения последовательности бит;

– установку отображения битового потока на паузу и возобновление его получения с выравниванием отображения по границам кадров.

Структура разрабатываемых программных средств приведена на рисунке. Разработанные программные средства состоят из графического интерфейса, класса-хранилища и класса-прокладки.

Класс-хранилище принимает данные из внешнего источника и сохраняет их в кольцевой буфер [4]. Когда буфер заполнен, старые данные удаляются и добавляются новые.

Класс-прокладка нужен для развязки интерфейсов между хранилищем и графической частью. В дальнейшем планируется разделение их на отдельные библиотеки.

Класс-хранилище принимает битовый поток из внешнего источника, которым является демодулятор [5] в формате байтового массива, после чего ожидает запроса на отправку данных от графического интерфейса. Графический интерфейс периодически запрашивает очередную порцию байт у хранилища через класс-прокладку, после чего разделяет их на биты и отображает с сохранением кадровой структуры. Через окно диалога пользователь может установить все необходимые параметры для удобного отображения.

Рисунок 1 – Структура разрабатываемых программных средств

Программное обеспечение разработано на языке программирования C++ в среде разработки Microsoft Visual Studio [6] с использованием фреймворка Qt [7].

Разработанные программные средства будут использованы при разработке систем мониторинга радиоэфира. Разработанные программные средства можно использовать и для решения задачи обнаружения кадров в битовом потоке с неизвестной структурой.

ЛИТЕРАТУРА

1. Doc 9925. Руководство по авиационной подвижной спутниковой (маршрутной) службе. ИКАО, 2011.
2. Ермолаев В. Т., Флаксман А. Г. Адаптивная пространственная обработка сигналов в системах беспроводной связи. Учебно-методический материал по программе повышения квалификации «Современные системы мобильной цифровой связи, проблемы помехозащищенности и защиты информации». Нижний Новгород, 2006, 99с.
3. Masood, S.; Khan, S.A.; Hassan, A.; Khalique, F. A. Knowledge Base Technique for Detecting Multiple High-Speed Serial Interface Synchronization Errors in Multiprocessor-Based Real-Time Embedded Systems. Electronics 2022, 11, 2945. https://doi.org/10.3390.
4. Макс Шлее Qt 4.8 Профессиональное программирование на C++. СПб.: БХВ-Петербург, 2012. — 912 с.
5. Богач Н. В., Гублер Г. Б., Евдокимов В. Е., Куляшова З. В., Перепелица С. А., Хабурзания Т. З. Обработка сигналов в информационных системах : учеб. пособие / Н.В. Богач [и др.]. – СПб.: Изд-во Политехн. ун-та, 2010. – С. 125-129.
6. Кольцевой буфер [Электронный ресурс] URL: https://ru.wikipedia.org/wiki/Кольцевой_буфер
7. Среда разработки Microsoft Visual Studio [Электронный ресурс] URL: https://visualstudio.microsoft.com/ru/vs

УДК 004.932.72'1

А. С. Столяров (4 курс бакалавриата),
С. Э. Сараджишвили, к.т.н., доцент

ОПРЕДЕЛЕНИЕ НА ИЗОБРАЖЕНИИ ТОЧКИ ВПРЫСКА ГЕЛИЯ НА ТОКАМАКЕ

В современном мире появляется все больше способов автоматизации различных процессов, которые до этого человек был вынужден выполнять вручную. В большинстве своем такого рода действия монотонны и не вызывают исследовательского интереса. Также и в данной работе.

При работе с плазмой в токамаке на экспериментальную установку действуют мощные вибрации, которые во время эксперимента смещают камеру, которая снимает внутреннюю часть токамака во время эксперимента. Из-за этого смещения нельзя один раз рассчитать

координаты точки, относительно которой будут вестись вычисления, так как на каждом изображении она смещается. В таком случае из-за неблагоприятных условий на данный момент приходится вручную находить на каждом изображении искомую область, что отнимает значительную часть времени, которое можно было потратить на более важные для исследования вещи. И поэтому было решено произвести попытку автоматизации данного процесса.

Таким образом, целью данной работы является автоматизация нахождения точки впрыска гелия и сокращение затрачиваемого на это времени и усилий.

Для выполнения поставленной цели необходимо решить несколько задач:

– Обзор существующих подходов преобразования изображения для увеличения процента удачных случаев определения объекта.

– Обзор существующих подходов реализации определения объекта на изображении.

– Проведение сравнительного анализа данных подходов.

– Реализация данных подходов

– Определение удовлетворяющего подхода определения объекта на изображении

Для того, чтобы на изображении можно было найти искомый объект, само изображение должно быть достаточно контрастным, незатемненным и с минимальным количеством шумов. Поэтому, прежде чем проводить обработку изображения, нужно произвести работу по улучшению его качества.

Для данной работы я использовал библиотеку на языке программирования Python для работы с изображениями. Из этой библиотеки были данные функции использованы:

– Адаптивная эквализация гистограммы изображения (Adaptive histogram equalization) [1]

– Фильтрация регионального максимума (Filtering regional maxima) [2]

– Черно-белая локальная сегментация (Local thresholding) [3]

Первая функция использовалась для распределения яркости пикселей по изображению. Благодаря ей удалось увеличить контраст на изображении, а также осветлить затемненную картинку. Вторая функция использовалась для создания еще большего контраста между объектом на изображении и фоном, а также избавиться от шумов. И последняя функция локально преобразовывает изображение так, чтобы на выходе в массиве пикселей мы имели только белые и черные пиксели. Глобальную сегментацию не было возможности применить, так как помимо искомого объекта на изображении также имеется область плазмы, которая вместе с точкой впрыска, при глобальной сегментации, воспринимается как один общий объект.

Рисунок 1 – Изображение после преобразования

Далее, после преобразования изображения, необходимо каким-либо способом определять искомую область. Для этого был использован метод, который совмещает в себе корреляцию изображения с шаблоном с помощью преобразований Фурье, и пирамидную свертка изображения [4], которая позволила увеличить точность корреляции. Для данных

преобразований Фурье использовалась библиотека NumPy [5], а для свертки использовалась библиотека Scikit-Image [6].

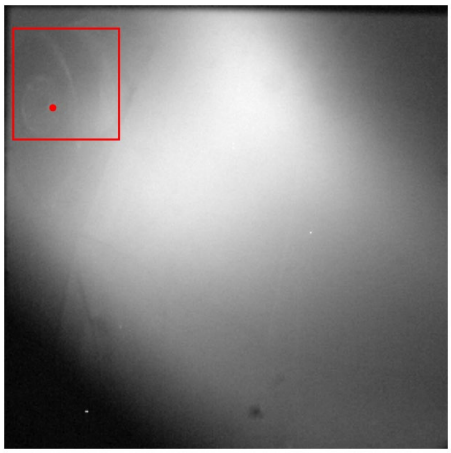


Рисунок 2 – Искомая область и точка впрыска после корреляции

Благодаря выполнению описанных выше шагов удалось успешно автоматически определять искомую точку на изображении. Исключение составили только случаи, когда на изображении в принципе было сложно что-то различить, даже со всевозможными преобразованиями.

ЛИТЕРАТУРА

1. Histogram Equalization [Электронный ресурс] Режим доступа: https://scikit-image.org/docs/dev/auto_examples/color_exposure/plot_equalize.html#sphx-glr-auto-examples-color-exposure-plot-equalize-py
2. Filtering regional maxima [Электронный ресурс] Режим доступа: https://scikit-image.org/docs/dev/auto_examples/color_exposure/plot_regional_maxima.html#sphx-glr-auto-examples-color-exposure-plot-regional-maxima-py
3. Thresholding [Электронный ресурс] Режим доступа: https://scikit-image.org/docs/dev/auto_examples/applications/plot_thresholding_guide.html#local-thresholding
4. Yang Li, Jianli Wang, Kainan Yao, Modified phase correlation algorithm for image registration based on pyramid // Alexandria Engineering Journal – 2022 – Vol. 61 – 709-718
5. NumPy [Электронный ресурс] Режим доступа: https://numpy.org/
6. Build image pyramids [Электронный ресурс] Режим доступа: https://scikit-image.org/docs/dev/auto_examples/transform/plot_pyramid.html#sphx-glr-auto-examples-transform-plot-pyramid-py

УДК 004.651

А.А. Ткаченко (4 курс бакалавриата),
О.В. Прокофьев, ст. преподаватель

РАЗРАБОТКА ВСТРАИВАЕМОЙ ПЕРСИСТЕНТНОЙ ПОДСИСТЕМЫ ХРАНЕНИЯ ДАННЫХ ТИПА «КЛЮЧ-ЗНАЧЕНИЕ», ОПТИМИЗИРОВАННОЙ ДЛЯ ПОСЛЕДОВАТЕЛЬНОГО ЧТЕНИЯ

База данных – это модульная система из составных частей: транспортный уровень, обработчик запросов, подсистема выполнения операций и подсистема хранения. СУБД можно рассматривать как приложение, которое надстроено поверх подсистемы хранения и предлагает некоторую схему данных, язык запросов, индексацию, транзакции и прочее. Подсистема хранения данных же выступает как программный компонент СУБД, отвечающий за хранение, извлечение и управление данными в памяти и на диске, предназначенный для работы с постоянной, долговременной памятью каждого узла [1]. В то время как базы данных могут отвечать на сложные запросы, подсистемы хранения рассматривают данные детально и

предлагают API для манипуляций с данными [10], позволяющий пользователям создавать, обновлять, удалять и извлекать записи [7].

Использование существующих подключаемых подсистем хранения (BerkeleyDB, LevelDB, RocksDB, LMDB, HaloDB и другие), разработанных отдельно [8], позволяет разработчикам СУБД обойти этап их создания (выбор способа хранения, извлечения информации и организации данных) и сосредоточиться на других подсистемах [6]. К тому же это позволяет переключаться между различными подсистемами, потенциально более подходящими для конкретных сценариев использования (например, MySQL – между InnoDB, MyISAM и RocksDB). Это важно, так как не существует подсистемы хранения, идеальным образом подходящей для каждого возможного сценария использования. Поэтому есть множество различных подсистем, использующих разные структуры данных и реализованных на разных языках [11]. Следовательно, выбор нужно делать исходя из рабочих нагрузок и сценариев использования.

Основным мотивом этой работы было достижение быстрого последовательного чтения всех ключей, так как при сравнении этого критерия у популярных хранилищ ключей и значений через YCSB (Yahoo! Cloud Serving Benchmark) [6] – стало ясно, что ни одно из них не работает так же быстро, как простое чтение файла с SSD. Это объяснимо: большинство хранилищ основано на LSM, которое требует сортировки всех данных [12] и кэширования данных файловым кэшем операционной системы [6, 8]. В проекте рассматривается подход, где недавно обновленные ключи располагаются в начале своего файла данных. Таким образом, операционная система может кэшировать только заголовок файла данных и обеспечивать высокую пропускную способность случайного чтения без кэширования всего файла данных. Таким образом, *целью* работы выдвигается разработка встраиваемой персистентной подсистемы хранения типа «ключ–значение», оптимизированной для последовательного чтения. В качестве отдельных *задач* рассмотрены:

– организация высокоскоростного последовательного чтения;

– обеспечение случайной записи на высокой скорости;

– привязка к кластеру недавно обновленных ключей для более высокого случайного I/O на таких ключах;

– возможность восстановления после повреждения файла.

Источники, описывающие архитектуру СУБД [2, 3, 4, 5], определяют компоненты и отношения между ними по–разному. На Рисунке 1 представлены стандартные компоненты. Диспетчер транзакций производит их планировку и гарантирует согласованное состояние базы после выполнения. Диспетчер блокировок блокирует объекты БД для выполняемых транзакций, так физическая целостность не будет нарушена конкурентными операциями. Средства доступа представляют собой структуры для хранения данных, управляя доступом и организацией данных на диске. Диспетчер буферов кэширует страницы данных в памяти. Диспетчер восстановления ведет журнал операций и восстанавливает состояние системы в случае сбоя [6].

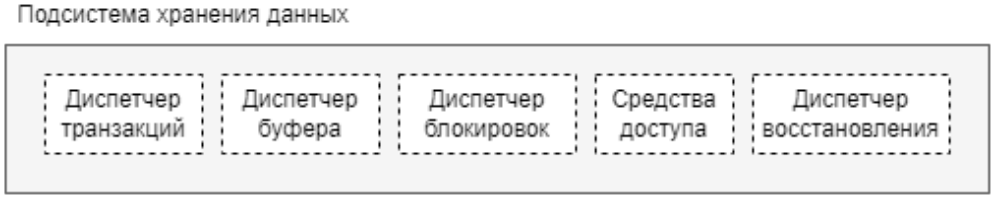


Рисунок 1 – Архитектура ядра СУБД

В проектировании использовался паттерн Builder, чтобы отделить конструирование сложного объекта от его представления [9], что отображено на Рисунке 2.

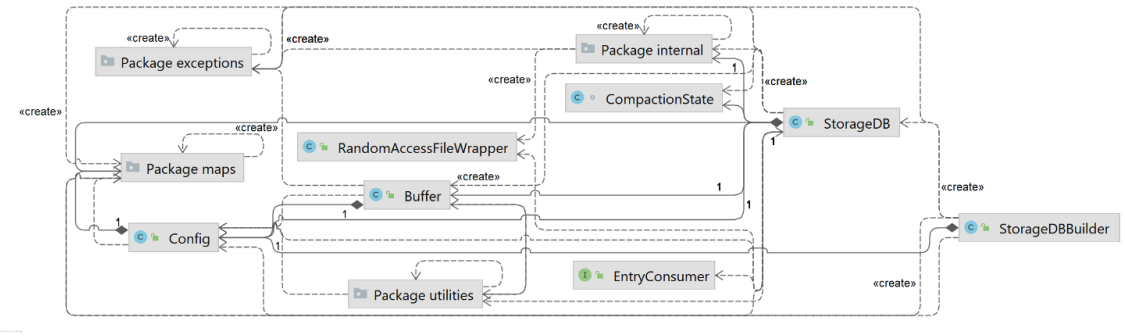


Рисунок 2 – UML–диаграмма классов реализации

Особенности проекта таковы: персистентное OLTP хранилище [13, 14], блоки данных как пара «ключ-значение» задаются сразу и фиксируются (ключи ограничены 4 байтами, т.е. 4,2 млрд), к тому же применяется стратегия смещения (offset) и стратегия сжатия для файла WAL, предотвращающая его чрезмерный рост. Согласованность данных обеспечивается CRC32 и sync marker. Восстановление данных происходит путем последовательного чтения файлов (WAL и данных) с начала и проверки контрольной суммы для каждого блока.

Итоговый проект реализован на языке Java, способном обеспечить высокий уровень абстракции при хорошей производительности и сборку мусора, с использованием фреймворка сборки Maven и библиотек JUnit для тестов и SLF4J для протоколирования.

Конечный результат представляет собой программный модуль, готовый встраиваться для использования в другие. Контроль качества реализации осуществлен через высокое тестовое покрытие. Для получения сравнительных характеристик производительности продукта предполагается использовать бенчмарк Java Microbenchmark Harness (JMH).

ЛИТЕРАТУРА

1. Reed, D. P. Naming and synchronization in a decentralized computer system: technical report. – MIT, 1978.
2. Hellerstein, Joseph M., Michael Stonebraker, and James Hamilton. Architecture of a Database System: foundations and trends in databases 1, no. 2 (february). – MIT, 2007. – 141–259 с. – DOI: https://www.nowpublishers.com/article/Details/DBS-002
3. Weikum, Gerhard, and Gottfried Vossen. Transactional Information Systems: Theory, Algorithms, and the Practice of Concurrency Control and Recovery: монография. – San Francisco: Morgan Kaufmann Publishers Inc, 2001.
4. Elmasri, Ramez and Shamkant Navathe. Fundamentals of Database Systems (6th Ed.) – Boston: Pearson, 2011.
5. Гарсиа-Молина Г., Ульман Дж. Д., Уидом Дж.. Системы баз данных: Полный курс. – СПб.: Вильямс, 2017. - 1088 с.
6. Alex Petrov. Database Internals: A Deep Dive into How Distributed Data Systems Work – Boston: O'Reilly, 2019. – 371 с.
7. Martin, James. Managing the Data–base Environment. – Englewood Cliffs, New Jersey: Prentice–Hall, 1983.
8. Kleppmann, Martin. Designing Data-Intensive Applications. – Boston: O'Reilly, 2022.
9. Gamma, Erich; Helm, Richard; Johnson, Ralph. Design Patterns. – Addison-Wesley, 2022.
10. Безрук П.А. Разработка подсистемы хранения данных для учебной системы управления данными // Актуальные проблемы авиации и космонавтики, Т. 2, 2018. – С. 128-129.
11. Суханов А.А., Маратканов А.С. Анализ способов хранения социальных данных из сети Интернет // International scientific review, 2017. – С. 25-28.
12. Неганов А.М. LSM-индекс с общими компонентами для индексации хронологических данных // Современные информационные технологии и ИТ-образование, Т. 18, № 2, 2022. – С. 339-351. – DOI: 10.25559/SITITO.18.202202.337-352

13. Кузнецов С.Д. В ожидании нативных архитектур СУБД на основе энергонезависимой основной памяти // Труды Института системного программирования РАН, Т. 32, № 1, 2020. – С. 154-180. – DOI: 10.15514/ISPRAS-2020-32(1)-9
14. Белоусова М. Н., Орешина М. Н. Совершенствование информационного обеспечения формирования управленческой отчетности на примере деятельности энергетической компании // Управление, Т. 9, № 3, 2021. – С. 14-26. – DOI: https://doi.org/10.26425/2309-3633-2021-9-3-14-26

УДК 004.92

Н. Л. Терентьев, Р. И. Кильдеев (4 курс бакалавриата),
Т. В. Леонтьева, к.т.н., доцент,
Т. В. Коликова, ст. преподаватель

РАЗРАБОТКА АЛГОРИТМА ЭФФЕКТА СВЕЧЕНИЯ ОБЪЕКТОВ НА ВЫЧИСЛИТЕЛЬНЫХ ШЕЙДЕРАХ ВИДЕОЧИПА

Технологии компьютерной графики постоянно развиваются, и на данный момент позволяют создавать реалистичные и приятные для глаз пользователя визуальные эффекты. Одним из наиболее часто используемых эффектов является эффект свечения, который имитирует светящийся ореол вокруг ярких объектов на сцене.

Целью работы является разработка алгоритма, обрабатывающего входное изображение так, чтобы на выходе получать эффект свечения объектов, присутствующих на нём.

Особенности реализации:

– Разбиение алгоритма на простые подзадачи, которые выполняются на разных ядрах видеочипа, для быстрой обработки одного кадра.

– Кэширование цветов для ускорения работы промежуточного фильтра размытия по Гауссу [1] за счет уменьшения количества образцов текстуры, необходимых для его расчета.

В качестве языка разработки был выбран C++, который необходим для считывания 3D сцены из файла и начальной инициализации графического API Vulkan [2] для вывода изображения и работы с видеокартой. Также был разработан код вычислительных шейдеров алгоритма, подзадачи которого выполняются на разных видео ядрах. Тестирование производилось на видеочипе Nvidia GTX 1070.

Алгоритм состоит из 3 проходов:

1. Итерация фильтрации изображения.

Алгоритм извлекает из исходного изображения фрагменты, уровень яркости которых превышает определенный порог.

2. Проходы размытия по Гауссу.

Целью данного прохода является размытие изображения по Гауссу и предоставление MIP изображения [3] с уровнями детализации.

3. Этап комбинирования.

Алгоритм соединяет полученные MIP изображения с изначальным для получения итогового изображения.

Описание реализации:

Каждый проход алгоритма является отдельным вычислительным шейдером с разным размером рабочих групп [4].

Итерация фильтрации изображения обладает размерностью групп 8x8x1 и разбивается на несколько задач:

1. Считывание исходного изображения из входного потока.

2. Применение билинейной фильтрации на блоки по 4 пикселя к исходному изображению.

3. Выделение ярких частей.

4. Возврат изображения, которое в 4 раза меньше исходного и содержит только яркие части.

Блок схема реализации вычислительного шейдера для проходов размытия по гауссу представлена на Рис. 1.

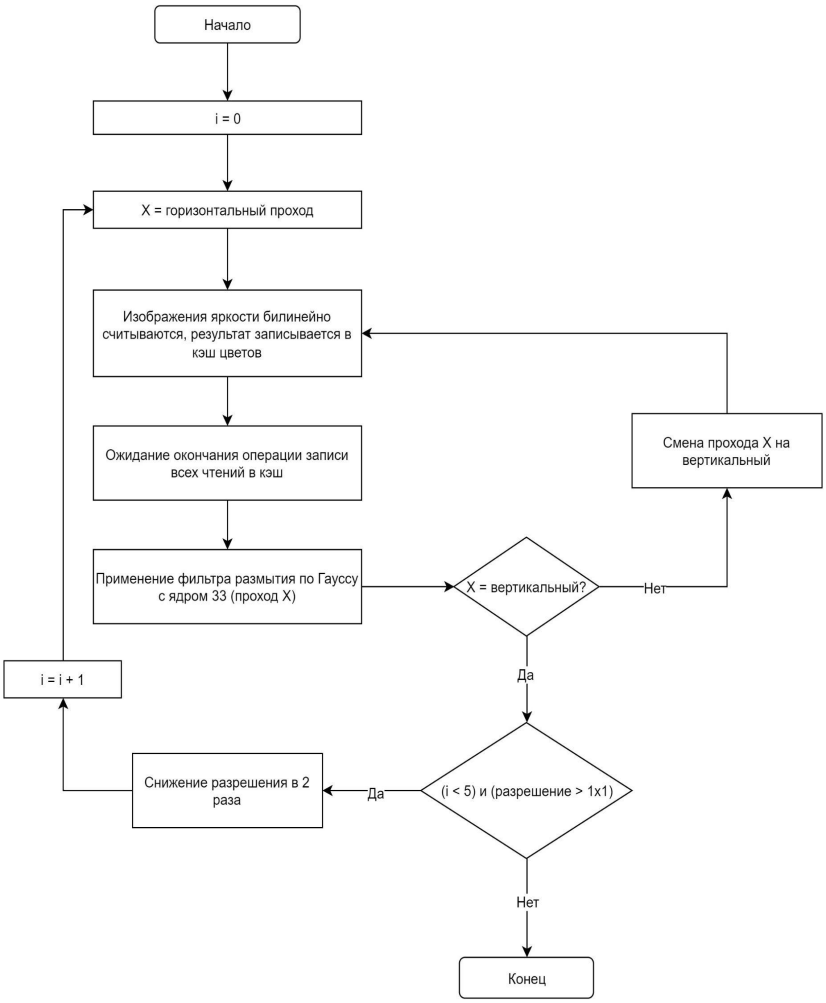


Рисунок 3 – Блок схема шейдера реализации размытия по Гауссу с размером группы 256x1x1

Для реализации *этапа комбинирования* используется шейдер с размерностью 256x256x1, который использует линейную интерполяцию между уровнями детализации MIP при помощи взятия среднего значения между 1 и 2, 3 и 4, 4 и 5 уровнями.

В результате была разработана программа, которая создает эффект свечения объектов на сцене. На Рис. 2 видна разница качества сцены слева без эффекта и справа с эффектом свечения.

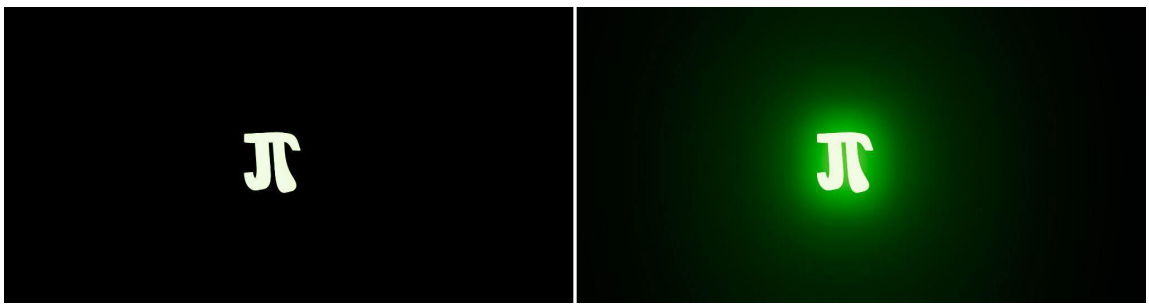


Рисунок 4 – Изображение сцены до и после обработки алгоритмом эффекта свечения

ЛИТЕРАТУРА

1. Efficient Gaussian blur with linear sampling [Электронный ресурс]. URL:https://www.rastergrid.com/blog/2010/09/efficient-gaussian-blur-with-linear-sampling/ (дата обращения: 19.03.2023)

2. Vulkan Documentation and Extensions: Procedures and Conventions [Электронный ресурс]. URL: https://registry.khronos.org/vulkan/specs/1.3/styleguide.html (дата обращения: 19.03.2023)
3. Texture Filtering with MipMaps [Электронный ресурс]. URL: https://learn.microsoft.com/en-us/previous-versions/aa921432(v=msdn.10) (дата обращения: 19.03.2023)
4. Compute Shader [Электронный ресурс]. URL: https://www.khronos.org/opengl/wiki/Compute_Shader (дата обращения: 19.03.2023)

УДК 004.457

А. С. Ткачук (2 курс магистратуры),
Б. М. Медведев, к.т.н., доцент

МОДИФИКАЦИЯ СТЕКА V2X ДЛЯ ОДНОВРЕМЕННОЙ РАБОТЫ С РАДИОРЕЖИМАМИ ITS G5 И LTE-CV2X

За период с 2000 по 20021 год количество зарегистрированных легковых автомобилей выросло приблизительно в 2,5 раза [1]. Несмотря на устойчивый тренд снижения количества ДТП [2], уровень тяжести последствий остаётся достаточно высоким [3]. Таким образом, проблема повышения безопасности движения на дорогах общего пользования остаётся важной задачей. Другой проблемой является управление и мониторинг больших транспортных потоков с целью увеличения пропускной способности транспортной сети.

В ответ на эти вызовы разработана технология V2X (Vehicle to everything) [4]. Технология охватывает различные варианты взаимодействия: V2V (Vehicle to vehicle), V2P (Vehicle to pedestrian), V2I (Vehicle to infrastructure) [5].

Стандарт ETSI [6] «Интеллектуальные транспортные системы» рассматривает следующих участников взаимодействия:

– транспортные средства, на которых установленные бортовые устройства (OBU – on board unit). Пешеходы могут также использовать подобные устройства, они обладают гораздо меньшим функционалом;

– стационарные средства связи, которые расположены на перекрёстках или вдоль дорог (RSU – road side unit). Устройства выполняют роль базовых станций, а также предоставляют информацию от систем умного города.

Беспроводная сеть взаимодействия транспортных средств и стационарных средств связи является ah-hoc сетью. Маршрутизация сообщений осуществляется на основе географического положения устройств (GeoNetworking). Идентификация производиться по MAC адресам сетевых контроллеров устройств.

Быстрое развитие сотовых сетей привело к созданию OBU, которые имеют возможность одновременно использовать технологии Wi-Fi, 5G и 4G (режимы передачи ITS-G5 и LTE-CV2X [9]). При этом для RSU в России существуют решения, осуществляющие передачу данных одновременно только в одном радио режиме [10].

Целью работы является разработка программных средств модифицированных RSU для одновременной работы с радио режимами ITS G5 и LTE-CV2X.

Общая архитектура разрабатываемых программных средств приведена на Рис.1. Она включает в себя сервис GeoNet, содержащий модули маршрутизации и модули, обрабатывающие данные на приём и отправку. Заметим, что сервис приёма может осуществлять пересылку сообщений. Также при приёме сообщений, для определённых типов пакетов (SHB) производится проверка на дубликаты, чтобы осуществлять пересылку единожды. На уровне ITC access работают сервисы, которые отвечают за заполнение параметров перед передачей в радио, а также приём сообщений.

Для достижения поставленной цели требуется разработать программные средства маршрутизации пакетов, учитывающие алгоритмы GeoNetworking [7, 8]:

– SHB (single hop broadcast) пакеты должны пересылаться из одного радио интерфейса в другой. Так как стандарт не предполагает пересылку данного типа пакетов, требуется разработать алгоритм дедубликации SHB пакетов.

– Требуется модифицировать алгоритмы и программные средства, осуществляющие пересылку от устройства с одним типом радио к другому (модернизировать Greedy, CBF, Advanced алгоритмы, входящие в модуль “Алгоритмы определения адреса следующего узла”)

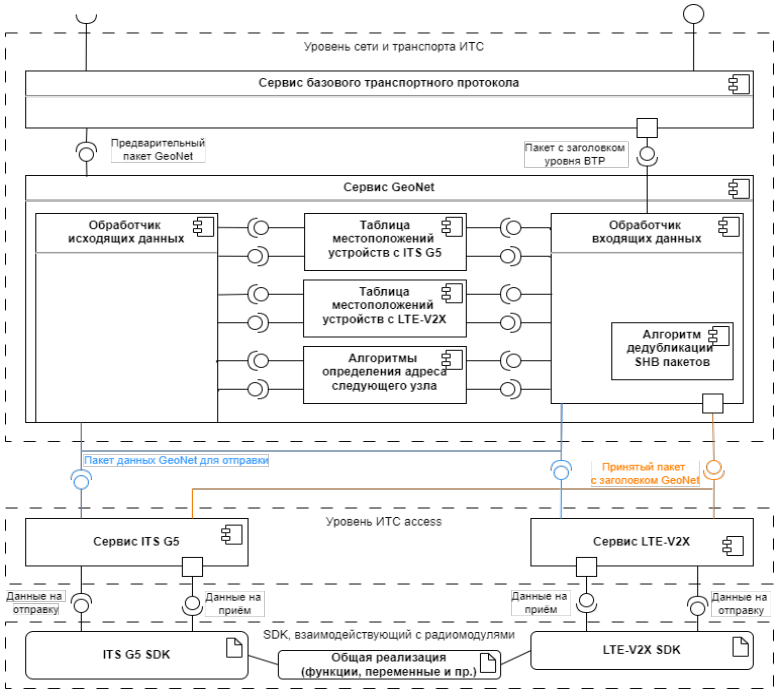


Рисунок 1 – Общая архитектура программных средств

Таким образом, поддержка одновременной работы с радио режимами ITS G5 и LTE-CV2X требует внести изменения в программное обеспечение уровней access ИТС и уровень сети и транспорта ИТС.

Разработка программных средств осуществляется на языке C++для платформы под управлением ОС Debian 9 Linux (версия ядра 4.9.124-imx6-sr).

ЛИТЕРАТУРА

1. Росстат транспорт // Росстат URL: https://rosstat.gov.ru/statistics/transport (дата обращения: 09.03.2023).
2. Строков Д.М., Лыгина Л.А. Территориальный анализ уровня социальных рисков ДТП с пешеходами в Российской Федерации // Проектирование автомобильных дорог. - М.: Общество с ограниченной ответственностью "А-проджект", 2021. - С. 29-41.
3. Киреева Ю.А., Калач Е.В. Состояние дорожно-транспортной аварийности на федеральных автодорогах России // Пожарная безопасность: проблемы и перспективы. - Воронеж: Воронежский институт - филиал ФГБОУ ВО Ивановской пожарно-спасательной академии ГПС МЧС России, г. Воронеж, 2018. - С. 363-366.
4. Сафиуллин Р.Н., Беликова Д.Д., Тянь Хаотянь Концептуальные подходы к построению систем оперативного управления движением транспортных средств при внедрении технологий 5G-V2X // Транспорт: наука, техника, управление, научный информационный сборник. - 2021. - №4. - С. 50–54.
5. Черноярова М.С., Арбузова А.А. Использование технологии V2X в автономных автомобилях // Информационно-вычислительные технологии и их приложения. - Пенза: Пензенский государственный аграрный университет, 2021. - С. 152-156.
6. ETSI E. N. 302 636-2 V1.2.1. Intelligent Transport Systems (ITS); Vehicular Communications; GeoNetworking; Part 3: Network Architecture // Standard. European Telecommunication Standard Institute. – 2014.

7. ETSI E. N. 302 636-2 V1.2.1. Intelligent Transport Systems (ITS); Vehicular Communications; GeoNetworking; Part 2: Scenarios // Standard. European Telecommunication Standard Institute. – 2013.
8. ETSI E. N. 302 636-4-1 V1.4.0. Intelligent Transport Systems (ITS); Vehicular Communications; GeoNetworking; Part 4: Geographical addressing and forwarding for point-to-point and point-to-multipoint communications; Sub-part 1: Media-Independent Functionality // Standard. European Telecommunication Standard Institute. – 2019.
9. C-V2X vs DSRC | Get Your Facts Straight on Cellular V2X, LTE-V & LTE-V2X // Autotalks URL: https://auto-talks.com/technology/dsrc-vs-c-v2x/ (дата обращения: 20.03.2023).
10. Стандарты обеспечения связи «подключенных» автомобилей отработают в НАМИ // Вестник ГЛОНАСС URL: http://vestnik-glonass.ru/news/tech/standarty-obespecheniya-svyazi-podklyuchennykh-avtomobiley-otrabotayut-v-nami/ (дата обращения: 15.03.2023).

УДК 004.021

Д.Э. Шабинский (4 курс бакалавриата),
Т. В. Коликова, ст. преподаватель

СПОСОБЫ УМЕНЬШЕНИЯ РАЗМЕРА МОБИЛЬНОГО ПРИЛОЖЕНИЯ ПРИ РАЗРАБОТКЕ

Разработка мобильных приложений — это процесс создания программных приложений, которые работают на мобильных устройствах, таких как смартфоны и планшеты. Он включает в себя проектирование, создание, тестирование и развертывание мобильных приложений для различных платформ.

Процесс разработки мобильного приложения включает в себя несколько этапов, включая создание идеи, создание прототипа, проектирование, разработку [1], тестирование и развертывание. На протяжении всего процесса разработчики должны убедиться, что приложение удобно для пользователя, визуально привлекательно и функционально. Им также необходимо оптимизировать приложение для повышения производительности, безопасности и совместимости с различными устройствами и операционными системами. Поговорим об одном из таких факторов.

Размер мобильного приложения — важный фактор, который следует учитывать, поскольку он может повлиять на взаимодействие с пользователем и общий успех приложения. Большой размер приложения может привести к замедлению загрузки и установки, повышенному использованию данных и ограниченному объему памяти на устройстве пользователя. Это может вызвать разочарование у пользователей и привести к снижению рейтинга приложения, что может отрицательно сказаться на видимости и популярности приложения.

Существует несколько факторов, которые могут повлиять на размер мобильного приложения, в том числе количество и размер ресурсов, таких как изображения, аудиофайлы и видео, сложность кода, использование сторонних библиотек и фреймворков, а также включение ненужных функций.

Чтобы решить проблему больших размеров приложений, разработчики мобильных приложений могут использовать несколько стратегий. Один из подходов заключается в оптимизации ресурсов приложения, таких как сжатие изображений и уменьшение размера аудио- и видеофайлов. Другая стратегия состоит в том, чтобы разделить код приложения на модули, чтобы в окончательную сборку приложения включались только необходимые компоненты [2]. Разработчики также могут рассмотреть возможность использования альтернативных библиотек и фреймворков, которые являются более легкими, или создания пользовательских решений, чтобы избежать включения ненужных сторонних компонентов.

Таким образом, проблема размера мобильного приложения может повлиять на взаимодействие с пользователем и успех приложения, но разработчики могут использовать различные стратегии для оптимизации и уменьшения размера своих приложений. Поэтому

целью данной работы является рассмотрение всех методик и взвешивание за и против уменьшения размера мобильных приложений.

Для достижения этой цели необходимо решить следующие задачи:

1. Обзор существующих подходов реализации мобильных приложений
2. Проведение сравнительного анализа найденных подходов
3. Предложения по уменьшению размера приложений
4. Реализация данного подхода

Для разработки под ОС Android используется объектно-ориентированный язык Kotlin [3] с Android SDK. [4]. Так же при разработке используются принципы чистой архитектуры [5].

ЛИТЕРАТУРА

1. Высокоуровневое программирование: учебное пособие [/ Уфимск. гос. авиац. техн. ун-т. – Уфа : РИК УГАТУ, 2017.
2. Васильев А. Н. Java. Объектно-ориентированное программирование: учебное пособие. СПб.: Питер, 2014. 400 с.
3. Жемеров Д., Исакова С. Kotlin в действии. / пер. с англ. Киселев А. Н. - М.: ДМК Пресс, 2018. - 402 с.: ил.
4. Documentation for app developers [Электронный ресурс] Режим доступа: https://developer.android.com/docs
5. Мартин Р. М29 Чистая архитектура. Искусство разработки программного обеспечения. — СПб.: Питер, 2018. — 352 с

УДК 004.051

М. П. Токарева (4 курс бакалавриата),
Т. В.Коликова, ст. преподаватель

РАЗРАБОТКА СИСТЕМЫ НА ОСНОВЕ ИНТЕРАКТИВНЫХ ИНФОРМАЦИОННЫХ ПАНЕЛЕЙ ДЛЯ УСОВЕРШЕНСТВОВАНИЯ ПРОЦЕССА ВИЗУАЛИЗАЦИИ ДАННЫХ

Цель этой работы состоит в исследовании существующих установленных алгоритмов конструирования отчетности для анализа, а также в разработке системы для уменьшения времени, затрачиваемого на ее создание пользователем.

Интерес к данной теме возник после знакомства со сферой BI-аналитики и изучения различных методов и инструментов для работы с данными [1]. Множество тонкостей в использовании привычных методов визуализации данных, таких как Microsoft Excel и Power Point, заставили задуматься о возможности улучшения системы подготовки данных для визуализации пользовательской отчетности. Необходимо, чтобы процесс поставки данных со стороны пользователя был максимально упрощен, ускорен и удобен в использовании.

Достижение поставленной цели позволит добиться повышения производительности при работе с аналитическими данными.

Системы мониторинга являются эффективным способом наблюдения за информацией, представляемой в виде определенного набора данных. Очевидно, что всякий бизнес зависит от информации, и чем он серьезнее, чем объемнее массивы данных, тем большее нужно человеческих ресурсов и времени, чтобы информацию обрабатывать [2]. Именно поэтому в мире идет активная дашбордизация [3]. Для понимания упомянутого современного термина, стоит обратиться к следующему определению: “dashboard” (аналитическая панель, информационная панель) – визуальное представление данных, сгруппированных по определенным признакам на одном экране для более облегчённого восприятия информации [4]. Далее в тексте термин будет упоминаться на русском языке, что также считается корректным. Отличия в использовании привычного отчета и современного способа визуализации представлены в Таблице 1.

Таблица 1 – Сравнительный анализ дашборда и отчета

Характеристики	Дашборд	Отчет
Статичность данных	Отсутствует	Присутствует
Обновление данных в реальном времени	Присутствует	Отсутствует
Элементы	Несколько модулей: графики, отчеты и набор данных	Графики, несколько таблиц или диаграмм
Структура	Присутствует возможность отображение данных любым способом	Присутствует определенный стандарт
Отображение разрозненных показателей и их анализ	Присутствует	Отсутствует
Количество данных	Присутствует возможность вывода только ключевых показателей	Объединение всего объема данных (большое количество информации)
Актуальность данных	Актуальны за любой временной период	Актуальны только на момент составления отчета
Возможные ошибки	Присутствуют, но не часто (при корректной настройке системы)	Присутствуют достаточно часто из-за человеческого фактора
Затраты по времени	Единичная большая затрата по времени при настройке системы	Частые затраты по времени на мануальное составление отчетов

При самом распространённом варианте обработки аналитических данных весь процесс производится вручную, работником, что понижает качество работ и повышает количество возможных ошибок в процессе обработки. Подготовка полученной информации для анализа имеет вид отчета, а именно формат PDF-файла, Excel-таблицы или PowerPoint презентации, которые зачастую являются неудобными и малоинформативными для аналитических работ с данными. Исходя из таблицы, представленной выше, очевидно, что привычный и удобный метод составления отчетности проигрывает возможностям визуализации информации через дашборд.

Управление данными следует рассматривать как сервисную функцию, которая будет понятна каждому специалисту, поможет решить задачи, поставленные перед бизнесом и обеспечит эффективную работу в сфере анализа, мониторинга и прогнозирования. Дашборды могут быть самыми разными. Обычно выделяют три категории дашбордов: стратегические, аналитические и оперативные. Каждый из них используется для разных целей, но факт в том, что благодаря современным методикам и программным возможностям визуализировать можно вообще любые данные [5]. Главная задача состоит в том, чтобы сделать это правильно, на максимально выгодных условиях и используя современные технологии.

В представленной работе будут исследованы существующие решения и предложены способы улучшения процесса визуализации данных, в том числе рассмотренные с точки зрения безопасности использования представляемой информации, а также простоты и удобства в использовании.

ЛИТЕРАТУРА

1. M. Tamer Özsu, Perspectives on Business Intelligence Excerpt // SYNTHESIS LECTURES ON DATA MANAGEMENT – 2013. – 39pp. – На англ. яз
2. Что такое Dashboard [Электронный ресурс]: Semantiva – Бизнес Dashboard и аналитика – режим доступа: https://semantica.in/blog/chto-takoe-dashbord.html

3. Как дашборды помогают бизнесу расти и быть эффективнее [Электронный ресурс]: RBK – режим доступа: https://trends.rbc.ru/trends/industry/cmrm/6287acd89a79476c50f27fd9
4. Что такое дашборд. Объясняем простыми словами [Электронный ресурс]: Secretmag – режим доступа: https://secretmag.ru/enciklopediya/chto-takoe-dashbord-obyasnyaem-prostymi-slovami.htm
5. Marchand, K. Types of Dashboards: The Operational Dashboard. [Электронный ресурс]. — 2014. — режим доступа: http://www.dundas.com/blog-post/types-of-dashboards-the-operational-dashboard/ – На англ. яз

УДК 004.934

А.О. Черных (4 курс бакалавриата),
А.В. Петров, ст. преподаватель

РАЗРАБОТКА СИСТЕМЫ ХРАНЕНИЯ И ГЕНЕРАЦИИ АУДИО СИНТЕЗА ТЕКСТА

Голосовые технологии становятся все более востребованными в различных сферах нашей жизни. Использование синтеза речи позволяет создавать голосовые сообщения, аудиокниги, аудиорекламу и другие продукты, которые удобны людям с ограничениями зрения или тем, кто хочет получить информацию, не отвлекаясь на чтение.

Системы синтеза речи становятся все более совершенными и универсальными. Ранее, подобные системы генерировали аудио, звучавшее излишне роботизировано, однако развитие технологий искусственного интеллекта и машинного обучения способствовало генерации более естественных и реалистичных голосов, учитывающих различные языки и диалекты.

Важным аспектом развития систем синтеза речи является возможность синтезировать на основе реальных записей голоса людей. Это позволяет создавать персонализированные голосовые приложения и услуги, которые лучше соответствуют потребностям и предпочтениям конкретных пользователей.

Одним из примеров персонального синтезированного голоса является API Yandex SpeechKit Brand Voice, которая позволяет создавать уникальные голоса для модели синтеза речи на основе обучения на конкретном голосе.

Технологии персонализированного голоса позволяют авторам контента сократить время на добавление своих аудио сообщений или комментариев к тексту, а аудио синтез становится ассоциированным с автором за счет его сходства со звучанием и манерой общения.

Использование синтеза речи предоставляет преимущества авторов контента и их пользователей. Для пользователей, в первую очередь, это возможность получать информацию без необходимости читать текст, что особенно удобно для людей с ограниченными возможностями зрения или тех, кто по тем или иным причинам не может сосредоточиться на чтении текста. Более того, синтез речи, за счет контроля скорости чтения, позволяет сократить время на получение информации.

Существует множество решений аудио синтеза текста, таких как Yandex TTS, MCS TTS, Tinkoff Voice Kit и т.д. Подобные системы предоставляют демо интерфейс для генерации текста, который ограничен количеством символов, либо API, который предоставляет лишь генерацию аудио. Вопрос хранения в таких системах не поднимается, так как они являются специализированными системами синтеза, работающие на основе нейронных сетей.

Таким образом разработка системы хранения и управления аудио файлов является актуальной задачей, позволяющей решить следующие проблемы:

- Снижение затрат на производство аудио контента авторами
- Вовлечение аудитории воспринимающих информацию преимущественно на слух
- Управление сгенерированными аудио файлами
- Организация хранения аудио файлов в паре с текстом

Система будет реализована на основе микросервисной архитектуры (Рисунок 1), базирующейся на Go микросервисах и внешнем S3 хранилище.

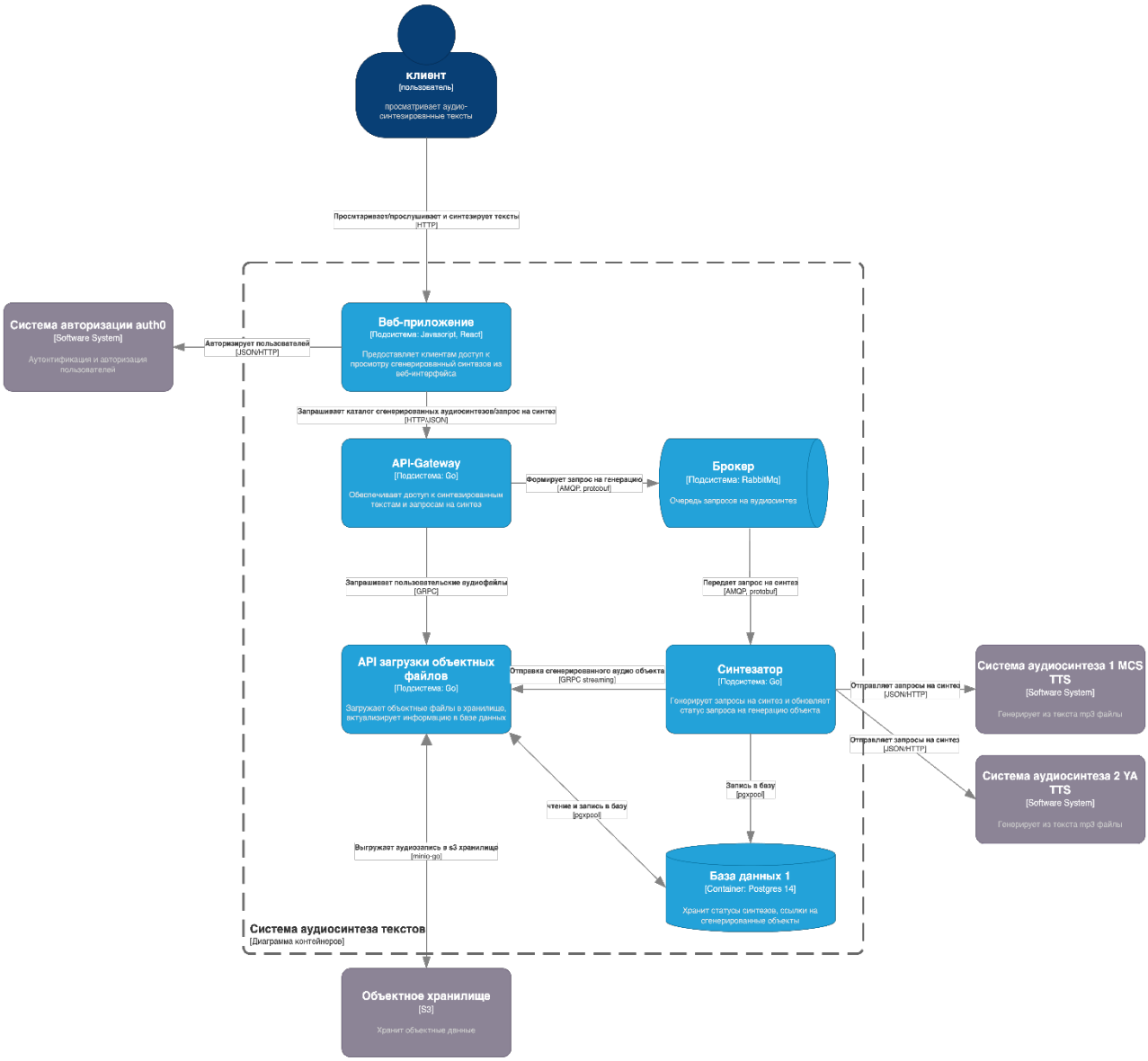


Рисунок 1 – Диаграмма контейнеров

Для реализации функционала системы будут использоваться такие технологии, как S3 объектное хранилище, сервис auth0 для аутентификации пользователей, Postgres для хранения данных пользователей, TTS облачные сервисы для генерации аудио синтеза, RabbitMQ для асинхронного взаимодействия микросервисов.

ЛИТЕРАТУРА

1. Клеппман М. Высоконагруженные приложения. Программирование, масштабирование, поддержка. - СПБ: ПИТЕР, 2022. - 640 с.
2. Цукалос Михалис Golang для профи: работа с сетью, многопоточность, структуры данных и машинное обучение с Go. - 3 изд. - СПБ: Питер, 2022. - 720 с.
3. Yandex SpeechKit // Озвучка текста, распознавание и синтез речи URL: https://cloud.yandex.ru/services/speechkit (дата обращения: 15.02.2023).
4. Что такое объектное хранилище S3 и как его используют // Использование облачного хранилища S3 URL: https://sbercloud.ru/ru/warp/blog/ispolzovanie-oblachnogo-s3-hranilishcha (дата обращения: 15.02.2023).
5. Полное практическое руководство по Docker: с нуля до кластера на AWS // Habr URL: https://habr.com/ru/post/310460/ (дата обращения: 15.02.2023).

УДК 004.051

С. Л. Шалгуева, М. В. Рощин (2 курс магистратуры),
Т. В. Леонтьева, к.т.н., доцент

ИССЛЕДОВАНИЕ ПРИМЕНИМОСТИ БЕССЕРВЕРНЫХ ТЕХНОЛОГИЙ В КАЧЕСТВЕ АЛЬТЕРНАТИВЫ KUBERNETES-ОПЕРАТОРА

Бессерверные вычисления (serverless computing) - новая привлекательная технология для развертывания приложений в облаке [1]. Крупные облачные провайдеры, такие как AWS, Google Cloud и Microsoft Azure, предлагают собственные бессерверные сервисы. Первопроходцем в этой сфере является Amazon Lambda, появившийся в 2014 году. За последние 10 лет технология развивалась и на данный момент представлена тремя направлениями: Function-as-a-Service (FaaS) [2], Container-as-a-Service (CaaS) [3] и с использованием Self-hosted фреймворков [4]. Хотя специфика услуг может отличаться, основная идея, лежащая в основе предлагаемых услуг, практически одинакова: путем перевода вычислений на модель "плати только за время работы", бессерверные вычисления предоставляют возможность автомасштабирования количества экземпляров приложения в зависимости от нагрузки. FaaS и CaaS решения предоставляются облачными провайдерами, но для бизнес-логики использование определённого провайдера означает потерю универсальности решения. Self-hosted бессерверные фреймворки используют другой подход. Это больше не отдельный сервис, предоставляемый облачным провайдером, это дополнительный функционал к существующим PaaS и IaaS решениям [5].

Одним из представителей группы бессерверных фреймворков является Knative. Это решение корпоративного уровня с открытым исходным кодом для создания бессерверных и управляемых событиями приложений. Для работы с бессерверными приложениями был выбран именно этот фреймворк, потому что на данный момент среди всех инструментов он остаётся наиболее популярным и перспективным, его поддерживает активное сообщество разработчиков, и он отлично интегрирован с Kubernetes. Главной особенностью фреймворка является способность автоматически масштабировать контейнеры пропорционально одновременным HTTP запросам. Неиспользуемые сервисы спустя какое-то время масштабируются до нуля, предоставляя масштабирование по требованию и позволяя экономить ресурсы.

Knative состоит из двух компонент: Knative Serving и Knative Eventing. Компоненты Serving предназначены для управления бессерверными приложениями, в том числе процессами деплоя и автомасштабирования. Компоненты Eventing предназначены для разработки и управления системами приложений, построенных в событийно-ориентированной архитектуре.

Kubernetes-оператор [6] — это приложение, которое включает в себя один или несколько контроллеров, обслуживающих некоторый ресурс. Оператор умеет следить за изменением состояния кластера, появлением или удалением новых ресурсов и реагировать на эти события. Однако в системах часто бывает, что такие события появляются редко, а сервис оператора постоянно запущен и потребляет ресурсы RAM и CPU. Возникла идея перевести сервис-оператор в бессерверный режим, чтобы он потреблял ресурсы только во время своей работы.

Работа посвящена исследованию возможности перевода сервиса, являющегося Kubernetes оператором, в бессерверный режим и проверке эффективности такого подхода. За основу был взят существующий сервис, который отслеживает появление ресурсов определённого типа и обновляет сетевой слой на основе считанной конфигурации. Этот сервис был переведён в бессерверный режим на основе Knative Serving, и была настроена event-mesh сеть с помощью Knative Eventing для запуска приложения при появлении нужного события. Итоговый вид системы представлен на Рис. 1.

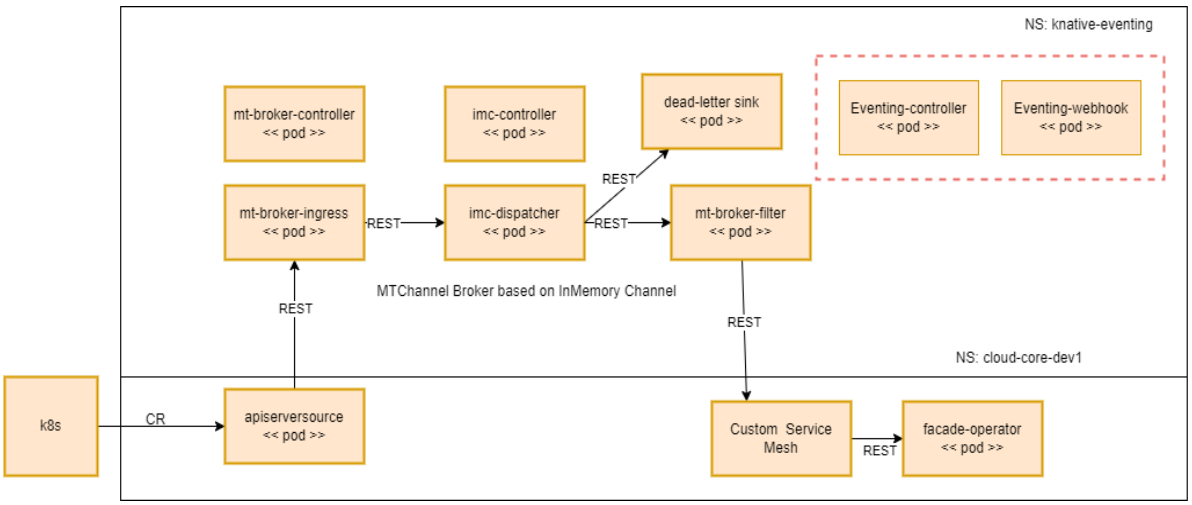


Рисунок 1 – "Схема event-mesh сети"

Разработанная система была протестирована на сохранение функциональности и на отказоустойчивость. Тестирование проводилось в локальном кластере Kubernetes, развернутом с помощью Rancher Desktop. Версия кластера v1.23.10, 12 RAM, 8 CPU. Для тестирования функциональности использовался набор интеграционных тестов, разработанный для исходного микросервиса. Все тесты были пройдены успешно, бессерверное приложение по функциональности не отличается от обычного. Для нагрузочного тестирования был разработан скрипт, который создавал 10 ресурсов в секунду. Бессерверное приложение без ошибок обработало все события, во время нагрузочного тестирования наблюдалась работа балансировщика нагрузки – количество экземпляров сервиса было увеличено.

Описанный выше эксперимент показал, что новая система, хоть и функционирует корректно, но потребляет в разы больше ресурсов. Это связано с тем, что фреймворк сам по себе потребляет ресурсы. Дополнительно необходимо учитывать потребление сервисами разработанной event-mesh сети. Эксперимент был повторён несколько раз для 1, 3, 5 и 7 экземпляров сервисов-операторов, имеющих одинаковую функциональность, но реагирующих на разные события. На основе полученных результатов была построена аппроксимация, которая показала, что использование бессерверной технологии будет выгоднее обычных микросервсов при наличии более 20 операторов. Такой случай возможен на больших корпоративных проектах, где количество операторов может доходить до 50.

Эксперимент показал, что использование Knative как альтернативы Kubernetes-оператору возможно, однако не всегда эффективно.

ЛИТЕРАТУРА

1. Eismann S. et al. A review of serverless use cases and their characteristics //arXiv preprint arXiv:2008.11110. – 2020.
2. Shafiei H., Khonsari A., Mousavi P. Serverless computing: a survey of opportunities, challenges, and applications //ACM Computing Surveys. – 2022. – Т. 54. – №. 11s. – С. 1-32.
3. Hussein M. K., Mousa M. H., Alqarni M. A. A placement architecture for a container as a service (CaaS) in a cloud environment //Journal of Cloud Computing. – 2019. – Т. 8. – №. 1. – С. 1-15.
4. Li J. et al. Analyzing Open-Source Serverless Platforms: Characteristics and Performance //arXiv preprint arXiv:2106.03601. – 2021.
5. Decker J., Kasprzak P., Kunkel J. M. Performance Evaluation of Open-Source Serverless Platforms for Kubernetes //Algorithms. – 2022. – Т. 15. – №. 7. – С. 234.
6. Сафронов Д., Стоноженко К. М., Никифоров И. В. Автоматическая балансировка нагрузки между потоковой обработкой данных и внутренними задачами кластера с использованием Kubernetes // Современные технологии в теории и практике программирования: сборник материалов конференции, Санкт-Петербург, 23 апреля 2020 года / Санкт-Петербургский политехнический университет Петра Великого; Dell Technologies; EPAM Systems. – Санкт-Петербург: ПОЛИТЕХ-ПРЕСС, 2020. – С. 165-167.

УДК 004.4

А.В. Шушарин (4 курс бакалавриата),
А.И. Тышкевич, к.т.н., доцент

КЛИЕНТСКОЕ ПРИЛОЖЕНИЕ НА БАЗЕ ANDROID ДЛЯ ЛИТЕРАТУРНОГО ПОРТАЛА

Идея создания литературного портала возникла при посещении похожих ресурсов, предоставляющих пользователям возможность выкладывать свои книги и читать книги других пользователей. Существует малое количество аналогов, в которых любой пользователь может стать автором и выкладывать свои произведения, которые другие смогут читать по мере его написания.

Приложение будет полезно и удобно многим людям, ведь среди неизвестных авторов и людей, которые хотят заняться написанием книги, может быть много таких, которые имеют большой талант в данной области, и могут быть оценены аудиторией по достоинству, ведь в больших издательствах таким людям в большинстве случаев просто откажут.

Именно поэтому реализованное приложение может помочь людям в начале их авторской карьеры, а также пользователям найти интересующие их произведения в открытом доступе.

Для работы клиенского приложение реализовать удобное и функциональное Android приложение, которое будет приятно в использование и для авторов, и для их читателей

Существует несколько фреймворков для связи с сервером в приложениях на Kotlin для Android. Некоторые из них:

1. Retrofit [1]
2. Volley
3. OkHttp
4. Ktor
5. Fuel

Из данного списка после проведения сравнения и уточнения информации был выбран фреймворк Retrofit по нескольким причинам:

1. Простота использования: Retrofit предоставляет простой и интуитивно понятный API для работы с сервером, что делает его очень удобным для начинающих разработчиков.
2. Высокая производительность: Retrofit использует библиотеку OkHttp для выполнения сетевых запросов, что обеспечивает высокую производительность и эффективность при работе с сервером.
3. Поддержка RESTful API: Retrofit полностью поддерживает RESTful API, что делает его идеальным выбором для работы с большинством современных веб-сервисов.
4. Легко настраиваемый: Retrofit предоставляет множество опций для настройки запросов, включая параметры запроса, заголовки и т. д., что позволяет легко настроить его под свои потребности.
5. Хорошая документация: Retrofit имеет отличную документацию и множество примеров кода, что делает его легко изучаемым и понятным для новых пользователей.

В целом, Retrofit является отличным выбором для связи с сервером на Kotlin Android, благодаря своей простоте использования, высокой производительности и поддержке RESTful API

В системе присутствуют следующие компоненты:

1. MainActivity [2] – Основной модуль, управляющий жизненным циклом приложения, в который по мере использования добавляются или удаляются фрагменты [3], небольшие части интерфейса, в зависимости от потребностей приложения.
2. NavController [4] – Контроллер, с помощью которого осуществляется навигация в приложении, он управляет фрагментами, добавляя их в основной модуль для предоставления их интерфейса пользователю или удаляя для переключения на следующий фрагмент
3. AppBar [5], DrawerLayout [6], AppBarMyLibrary – компоненты, предоставляющие доступ к некоторым фрагментам. AppBar, DrawerLayout добавлены в MainActivity, для перманентного

доступа к фрагментам, за исключение показа фрагмента MyLibrary, AppBar, в котором, заменяется на AppBarMyLibrary

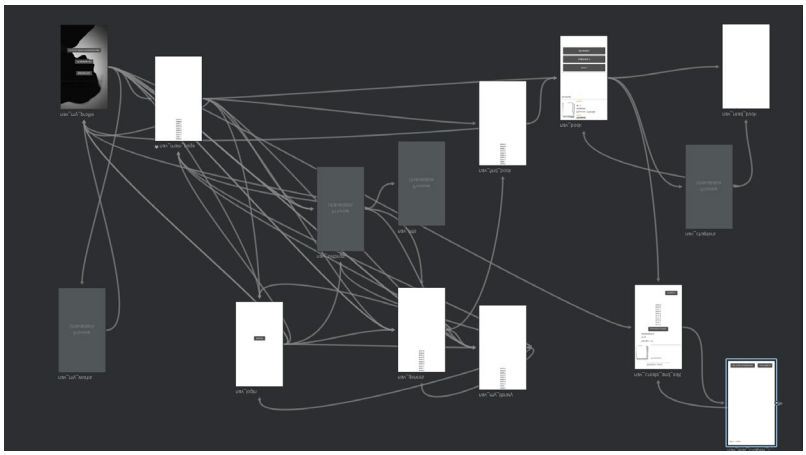


Рисунок 1 – Схема навигации между фрагментами

4. Множество фрагментов – которые переключаются контроллером в соответствии со схемой, навигации между фрагментами использую интерфейс, предоставленный этими фрагментами и AppBar, DrawerLayout, AppBarMyLibrary

5. Множество ViewModel [7] – каждому фрагменту соответствует свой ViewModel, в котором хранятся запросы, которые должны отправиться базе данных от этого фрагмента и ссылки на данные, которые получают запросы

6. Множество запросов – которые связываются с сервером и либо предоставляют данные, либо получают данные от него

ЛИТЕРАТУРА

1. Retrofit [Электронный ресурс]. -Режим доступа: https://square.github.io/retrofit/
2. Activity (Активность, Деятельность) [Электронный ресурс]. -Режим доступа: https://developer.alexanderklimov.ru/android/theory/activity-theory.php
3. Фрагменты [Электронный ресурс]. -Режим доступа: https://developer.alexanderklimov.ru/android/theory/fragments.php
4. NavController [Электронный ресурс]. -Режим доступа: https://developer.android.com/reference/androidx/navigation/NavController
5. Working with the AppBar [Электронный ресурс]. -Режим доступа: https://developer.android.com/guide/fragments/appbar
6. DrawerLayout [Электронный ресурс]. -Режим доступа: https://developer.android.com/reference/androidx/drawerlayout/widget/DrawerLayout
7. ViewModel [Электронный ресурс]. -Режим доступа: https://developer.android.com/topic/libraries/architecture/viewmodel

УДК 004.031.2

А. Ю. Шестакова, Д. О. Королев, А. А. Афанасьев (1 курс магистратуры),
О. А. Юсупова, ассистент

РАЗРАБОТКА СИСТЕМЫ АНАЛИЗА ДАННЫХ БАЗЫ ПУБЛИКАЦИЙ SCOPUS

Scopus - крупнейшая база данных, содержащая аннотации и информацию о цитируемости рецензируемой научной литературы. База содержит более 84 млн записей, более 17.6 млн профилей авторов из 94.8 млн организаций [1].

Цель исследования – анализ публикаций Scopus, определение основных тенденций развития мирового научного сообщества [2].

Для достижения цели были поставлены следующие задачи: выбор источника данных, постановка задач анализа, получение, обработка, анализ и визуализация большого количества данных.

Источником данных был выбран Scopus API. Кроме того, для получения данных, не предоставляемых API, используется Web-парсер.

В представленном решении ставятся следующие аналитические задачи:

– статистика изменения популярности университетов, городов, стран по количеству публикаций по годам;

– статистика изменения среднего размера публикаций по годам;

– статистика популярности месяцев по количеству публикаций;

– статистика популярности областей по количеству публикаций;

– анализ зависимости количества публикаций от нахождения публикации в открытом доступе;

– статистика количества публикаций по типу документов категории Open Access;

– ТОП стран по количеству публикаций;

– ТОП городов по количеству публикаций;

– ТОП университетов по количеству публикаций;

– ТОП авторов статей в СПбПУ.

Архитектура проекта состоит из двух модулей: модуль извлечения, приема, очистки, и подготовки данных, модуль обработки и визуализации данных [3].

Для выполнения аналитических задач с помощью скраппинга было получено и занесено в базу данных 74 млн записей, объем которых составил 200 Гб, а время выгрузки составило около 70 часов.

Для разработки были использованы язык программирования Python и среда разработки PyCharm. Для работы с источником данных использовались такие инструменты, как Postman, Chrome DevTools и Scrapy. Данные хранятся в базе данных PostgreSQL [4, 5], развернутой на облаке Yandex Cloud [6]. В качестве инструмента обработки и визуализации данных была выбрана Grafana.

Горизонтальное масштабирование системы обеспечивается за счет партицирования базы данных и использования фреймворка Scrapyd, позволяющего параллельно запускать несколько экземпляров Scrapy-парсера.

Для результатов представленных аналитических задач были построены графики. На Рис. 1 и Рис. 2 представлены результаты для двух аналитических задач.

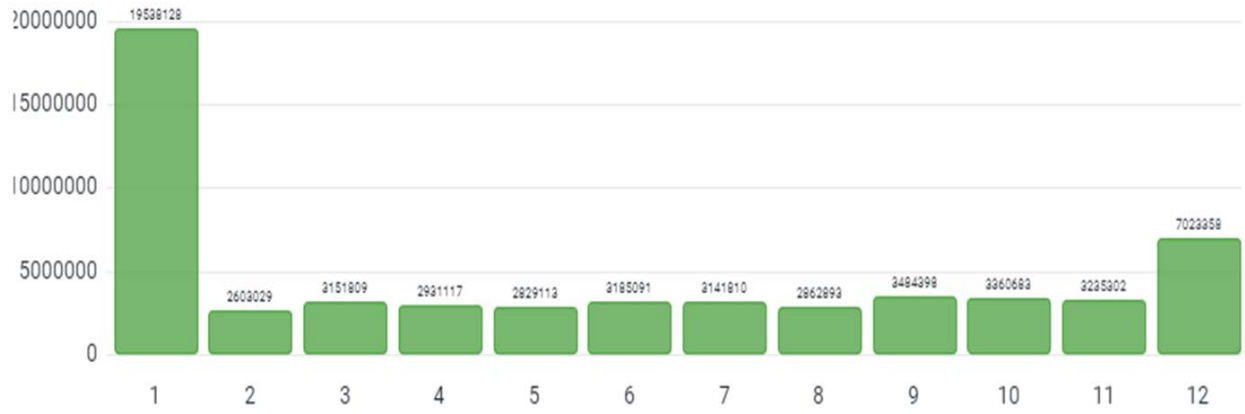


Рисунок 1 – Статистика популярности месяцев по количеству публикаций

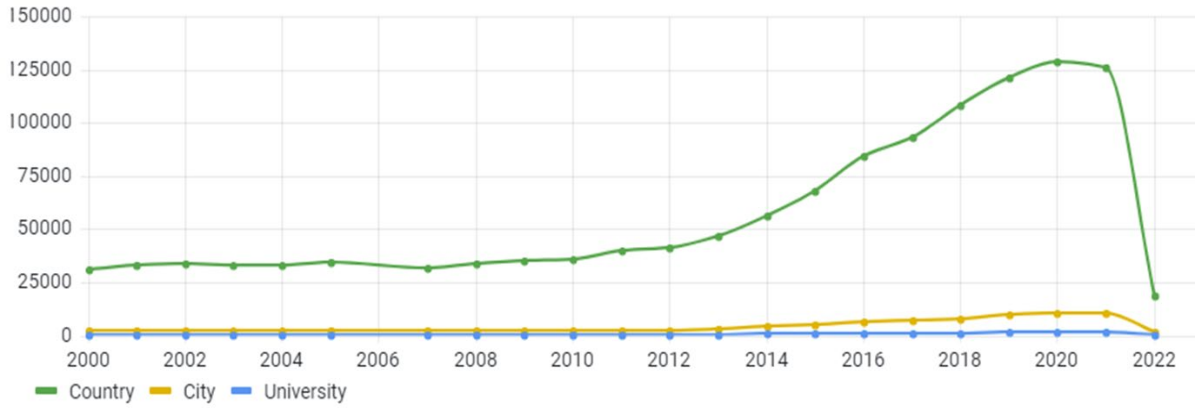


Рисунок 2 – Анализ изменения популярности университетов, городов, стран по количеству публикаций с параметрами “Страна” - Россия, “Город” - Санкт-Петербург, “Университет” - СПбПУ

ЛИТЕРАТУРА

1. Scopus: официальный сайт [Электронный ресурс]. Дата обращения: 12.03.2023. https://www.scopus.com/.
2. Shestakova A. Y., Korolev D. O., Afanasyev A. A., Nikiforov I. V., Yusupova O. A. Scopus publications database analysis using its API // Proc. SPIE 12564, 2nd International Conference on Computer Applications for Management and Sustainable Development of Production and Industry (CMSD-II-2022), 125640O (5 January 2023) – DOI 10.1117/12.2669237.
3. Никифоров И. В., Курсовое проектирование по учебной дисциплине "Наука о данных и аналитика больших объемов информации": Учебное пособие / И. В. Никифоров. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2017. – 62 с. – ISBN 978-5-7422-5638-0.
4. Сычев В. К., Ленькова Ю. В., Цветкова Е. А., Никифоров И. В. Анализ данных трендов YouTube // Современные технологии в теории и практике программирования : сборник материалов конференции, Санкт-Петербург, 23 апреля 2020 года / Санкт-Петербургский политехнический университет Петра Великого; Dell Technologies; EPAM Systems. – Санкт-Петербург: ПОЛИТЕХ-ПРЕСС, 2020. – С. 131-132. – EDN GGZRXB.
5. Voinov N., Rodriguez Garzon K., Nikiforov I., Drobintsev P. Big data processing system for analysis of GitHub events // Proceedings of 2019 22nd International Conference on Soft Computing and Measurements, SCM 2019 : 22, St. Petersburg, 23–25 мая 2019 года. – St. Petersburg, 2019. – P. 187-190. – DOI 10.1109/SCM.2019.8903782.
6. Andreolini, M., Colajanni, M., Pietri, M., Tosi, S. Adaptive, scalable and reliable monitoring of big data on clouds // Journal of Parallel and Distributed Computing 79-80, 2015 – P. 67-79 – DOI: 10.1016/j.jpdc.2014.08.007.

Секция «Программная инженерия: методы и алгоритмы теории программирования»

УДК 004.89

А. А. Алиев (1 курс аспирантуры),
С.А. Молодяков, д.т.н., доцент

АЛГОРИТМ РАБОТЫ СИСТЕМ ВЕРИФИКАЦИИ СПИКЕРОВ ПО ГОЛОСУ

В настоящее время, из-за увеличения количество информационных систем, в которых содержится конфиденциальные данные пользователей, требуются все более надежные системы верификации. Классические методы на основе паролей являются не самыми надежными из-за увеличения количество мошенников и других сопутствующих проблем. В последние годы активно применяются и внедряются методы верификации на основе биометрии [1].

В нашей работе мы рассмотрим методы биометрии по голосу на основе нейронных сетей, так как любой современный и надежный метод биометрии строится именно на основе нейронных сетей [2].

Системы верификации голоса спикера состоят из двух основных компонентов, первый из которых - извлечение признаков, а второй - сопоставление признаков.

Извлечение признаков — это процедура извлечения небольшого объема информации пригодного для последующего сравнения из аудиосигнала сохраняются и

используются в дальнейшем для представления каждого диктора. Так как исходный аудиосигнал является достаточно простым и одновременно достаточно комплексным для использования в качестве исходных данных для задачи верификации, то нам нужно использовать какой-то метод пред. обработки аудиосигнала. Для этого можем воспользоваться быстрым преобразованием Фурье.

Быстрое преобразование Фурье (БПФ) — это алгоритм, который используется для эффективного вычисления дискретного преобразования Фурье (ДПФ) [3]. По сути ДПФ позволяет нам разделить сложный аудиосигнал на составляющие частоты и тем самым сильно упрощает наш анализ и дальнейшие манипуляции с сигналом. Чтобы воспользоваться полученными результатами БПФ в свёрточных нейронных сетях, нам нужно представить их в виде 2-мерной матрицы, иными словами, в виде изображения. Для этого воспользуемся банками фильтров на основе БПФ. Банки фильтров на основе БПФ широко используются в приложениях обработки сигналов, таких как обработка аудио и изображений, где они могут использоваться для извлечения особенностей или проведения анализа в определенных частотных диапазонах. Например, при обработке аудиосигнала банк фильтров БПФ может использоваться для разделения аудиосигнала на отдельные полосы, соответствующие различным частотным диапазонам, таким как низкие, средние и высокие частоты. Это может быть полезно для таких задач, как эквализация или шумоподавление, где важно иметь возможность манипулировать определенными частотными полосами отдельно. Но мы воспользуемся этим инструментом, чтобы представить часть аудиосигнала в виде спектрограммы, который представлен на Рис. 1

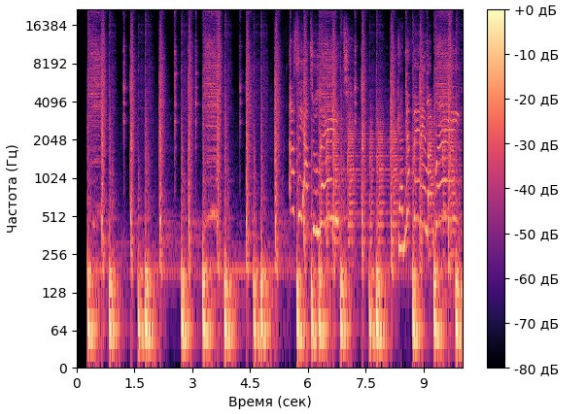


Рисунок – 1. Пример спектрограммы аудиосигнала.

Так как мы превратили звук в изображения, то обработка таких данных находится в области компетенции компьютерного зрения. В компьютерном зрении извлечение признаков — это процесс извлечения значимой информации из изображений или видео, которая может быть использована в качестве входных данных для модели машинного обучения. Целью извлечения объектов является преобразование необработанных данных, таких как изображение, в набор объектов, которые являются более информативными и компактными и которые отражают наиболее важные аспекты данных. Существует несколько методов, используемых для выделения объектов в компьютерном зрении, но мы воспользуемся наиболее эффективным из них, а именно свёрточными нейронными сетями.

Но так как нашей задачей является процесс верификации по голосу, то нам нужно сделать так, чтобы нейронная сеть при подаче на вход спектрограммы из аудиозаписи с голосом одного спикера выдавал максимально схожие результаты, а при разных при подаче спектрограмм голосов разных спикеров результаты нейронной сети максимально отличались. Для решения такого типа задач в нейронных сетях используются сиамские нейронные сети.

Сиамская нейронная сеть [4] — это тип архитектуры нейронной сети, который используется для задач, связанных со сравнением двух или более входных данных. Он состоит из двух или более идентичных нейронных сетей, каждая со своим собственным набором весов, которые совместно обучаются одной и той же задаче. Цель сиамской сети - изучить метрику сходства между входными выборками. Наиболее распространенный способ использования сиамской сети - сравнить два входных изображения и вывести оценку сходства между ними. Две идентичные нейронные сети принимают два разных изображения в качестве входных данных, а выходные данные сравниваются, чтобы определить сходство между ними. Сиамская сеть может быть обучена с использованием различных функций потерь, таких как контрастная потеря или тройная потеря. Цель состоит в том, чтобы минимизировать потери и, следовательно, максимизировать оценку сходства для положительных образцов (образцов одного класса) и максимизировать оценку различия для отрицательных образцов (образцов разных классов). Во время обучения веса обеих сетей обновляются одновременно, так что конечный результат работы сети является точной мерой сходства между двумя входными выборками.

Именно такая схема обучения нейронных сетей используется во всех современных системах биометрии в области человеческой речи и компьютерного зрения.

ЛИТЕРАТУРА

1. Sarkar A., Singh B. K. A review on performance, security and various biometric template protection schemes for biometric authentication systems //Multimedia Tools and Applications. – 2020. – Т. 79. – С. 27721-27776.
2. Snyder D. et al. Deep neural network embeddings for text-independent speaker verification //Interspeech. – 2017. – Т. 2017. – С. 999-1003.

3. Rajaby E., Sayedi S. M. A structured review of sparse fast Fourier transform algorithms //Digital Signal Processing. – 2022. – С. 103403.
4. Zhang Z., Peng H. Deeper and wider siamese networks for real-time visual tracking //Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. – 2019. – С. 4591-4600.

УДК 004.6

Э.Г. Амирасланов (1 курс аспирантуры),
С.Э. Сараджишвили, к.т.н., доцент

ИСПОЛЬЗОВАНИЕ РАСПОЗНАВАНИЯ РЕЧИ В СИСТЕМЕ ГОЛОСОВОГО УПРАВЛЕНИЯ. SPEECH RECOGNITION IN A VOICE CONTROL SYSTEM

Каждый день появляются новые возможности для улучшения жизни человека с помощью новых технологий и систем, которые делают ее более удобной. Одним из актуальных направлений является создание систем распознавания голосовых команд, которые могут быть использованы для управления умными устройствами в доме, мобильными телефонами и для помощи людям с ограниченными возможностями. Это направление активно развивается с каждым годом, и с 2011 года компании Google и Amazon представили свои продукты, такие как Google Home и Alexa, которые являются голосовыми помощниками и могут управлять домашними устройствами через голосовые команды. Системы распознавания речи преобразуют звуковые сигналы в слова или символы. Они используются в медицине, телекоммуникациях, автоматическом письме текстовых документов и управлении домашними устройствами. Процесс распознавания речи сложен и технически непрост, так как звуковой сигнал изменчив и зависит от многих факторов. Наиболее распространенными методами являются скрытые Марковские модели, нейронные сети и алгоритмы динамической трансформации временной шкалы (DTW).

В процессе распознавания речи с использованием нейронных сетей акустический препроцессор обрабатывает входной речевой сигнал и определяет последовательность векторов признаков для каждого семпла. Полученные векторы признаков сравниваются с эталонными векторами, содержащимися в моделях слов, и происходит временное выравнивание последовательностей векторов признаков с эталонными векторами, образующими модели слов. Затем вычисляется мера соответствия для компенсации изменений скорости произнесения и находится максимально соответствующее слово. Нейронные сети позволяют использовать достаточно большое количество слов в словаре, увеличивая только объем обучающего процесса без изменения сложности процесса распознавания. Однако, у этого подхода есть недостатки, такие как отсутствие возможности вносить дополнения к словарю после окончания процесса обучения.

Системы с Марковскими моделями подходят для моделирования спектральных векторных последовательностей. Эти модели основаны на предположении о том, что наблюдаемые данные происходят из некоторой скрытой системы, которая генерирует последовательность состояний. Каждое состояние генерирует наблюдение, и вероятность наблюдения определяется вероятностью перехода между состояниями. Марковские модели позволяют использовать меньшее количество данных для обучения, чем нейронные сети, и могут быть эффективно использованы для распознавания речи на специализированных тематиках.

Распознавание речи может быть реализовано двумя подходами: онлайн распознавание голоса с использованием нейронных сетей и запись ключевых фраз. Первый подход требует постоянного интернет-соединения, в то время как второй более трудоемок. Кроме того, для работы системы распознавания речи необходима система голосового вывода. Используется Kaldi Speech Recognition, который позволяет добавлять новый функционал без потери производительности. Основная задача голосового модуля - понимание естественной речи

пользователя путем реализации понимания естественного языка. Система реализована на языке Python с использованием Alphacep. Постоянное распознавание работает до тех пор, пока произнесенная ключевая фраза не соответствует маркерам в базе данных или пока пользователь не даст команду голосовому помощнику прекратить работу. Архитектура системы распознавания представлена на Рис.1

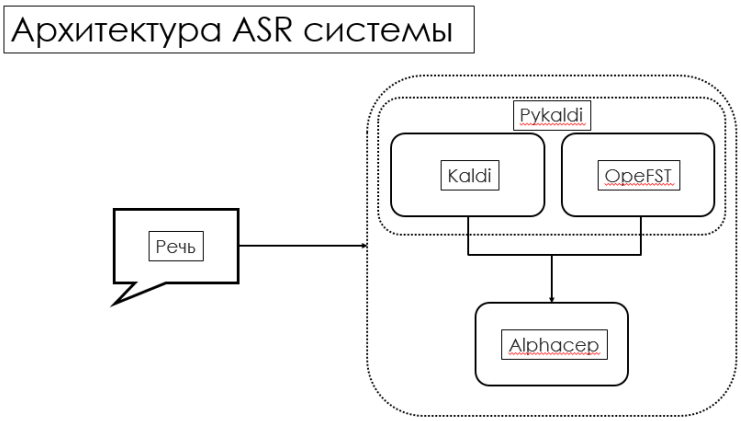


Рисунок 1 – Архитектура автоматизированной системы распознавания речи

В офисном помещении был подключен модуль голосового управления к комплектующим системы «Умный дом». Вся система состоит из подсистемы управления освещением, персональным компьютером и системой видеокамер, которые можно управлять как отдельными командами, так и запускать сценарии. Например, был реализован сценарий настройки рабочей среды, который при произнесении ключевой фразы открывал часто используемые вкладки в браузере, уменьшал яркость экрана, отключал все уведомления и открывал необходимые для работы программы. По точности системы будут сравниваться по наиболее распространенным метрикам: Word Recognition Rate (правильно распознанные слова); Word Error Rate (неправильно распознанные слова); Speed Factor (Скорость распознавания).

Таблица 1

Название технологии	**WER, %**	**WRR, %**	**SF**
Google Speech Recognition	4,3	95,7	0,45
Yandex SpeechKit	8,3	91,7	0,51
My Speech Recognition	6,5	93,5	0,6

Подводя итоги, можно сказать, что все системы показывают результаты примерно на одном уровне (продемонстрированы в Таблице 1), и разработанный голосовой модуль не сильно уступал аналогам из больших компаний.

В ходе данной работы была разработана система голосового управления, которая в настоящее время успешно используется в офисном помещении. В основном упор был сделан на дистанционное управление с помощью диалоговых команд и функцию обратной связи с пользователем, что позволяет выполнить каждый запрос с звуковым подтверждением.

В дальнейшем планируется развитие системы с целью ее усовершенствования. Будет проводится работа по созданию и внедрению пользовательского интерфейса в общее приложение управления системой Умного дома, а также по автоматизации настройки голосового помощника под задачи каждого пользователя.

ЛИТЕРАТУРА

1. Voicehd: Hyperdimensional computing for efficient speech recognition / Imani Mohsen, Kong Dekian, Rosing Tajana // IEEE International Conference on Rebooting Computing. – с : IEEE, 2017. – С. 1–8.

2. Geoffrey Hinton, Li Deng, Dong Yu, George Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara Sainath, and Brian Kingsbury Deep Neural Networks for Acoustic Modeling in Speech Recognition - IEEE, Signal Processing Magazine, 2012
3. An Open Source Machine Learning Framework for Everyone // TensorFlow. - [Электронный ресурс] URL: https://www.tensorflow.org/
4. Graves A., Mohamed A., Hinton G. Speech recognition with deep recurrent neural networks // Proceedings of International Conference on Acoustics, Speech and Signal Processing. Piscataway: IEEE, 2013. P. 6645-6649
5. Беленко М.В., Балакшин П.В. Сравнительный анализ систем распознавания речи с открытым кодом // МНИЖ. 2017. №4-4 (58)
6. Гапочкин, В. А. Нейронные сети в системах распознавания речи. /

В.А.Гапочкин // "Science time". – 2014, № 1. – С. 29-36

7. Алимурадов, А. К. Обзор и классификация методов обработки речевых сигналов в системах распознавания речи. / А.К. Алимурадов и [др]. // Измерение. Мониторинг. Управление. Контроль. - 2015, №2. – С. 27-35
8. Карпов Алексей Анатольевич, Кипяткова Ирина Сергеевна. Методология оценивания работы систем автоматического распознавания речи // Приборостроение. 2012. №11

УДК 004.657

Э.Ф. Вильданов (4 курс бакалавриата),
Д.В. Луцив, к.ф.-м.н., доцент

ОПТИМИЗАЦИЯ РАСПРЕДЕЛЁННЫХ JOIN-ЗАПРОСОВ В КЛАСТЕРЕ TARANTOOL

В современном мире, где часто возникает необходимость хранения, быстрой обработки и доступа к информации, распределённые базы данных стали широко используемой технологией. Они обладают преимуществами отказоустойчивости, более высокой скорости обработки запросов и дешевизны развёртывания.

Однако зачастую за преимущества использования распределённых баз данных приходится платить усложнением процесса исполнения запросов: из-за распределённого характера системы данные оказываются на разных устройствах, и возникает необходимость дорогостоящей передачи данных по сети. Так распределённый JOIN-запрос представляет собой одну из наиболее дорогих распределённых операций.

Библиотека с открытым исходным кодом sbroad, разрабатываемая компанией Пикодата, предоставляет функциональность исполнения распределённых SQL-запросов поверх кластера баз данных Тарантул и сейчас находится на этапе активной разработки. В качестве оптимизации распределённого JOIN-запроса сейчас применяется логика, основывающаяся только на информации об используемых в запросе ключах соединения и распределении данных. Однако никаким образом сейчас не учитывается информация о размерах таблиц, участвующих в соединении – предполагается, что пользователь всегда формирует запрос таким образом, чтобы соединения выполнялись в порядке уменьшения возвращаемых ими строк таблицы. Зачастую такой подход приводит к значительному увеличению времени исполнения запроса [1]. Предполагается, что скорость исполнения запросов можно значительно увеличить, обладая информацией о статистике по таблицам и размерам данных, которые необходимо передавать по сети.

Таким образом, целью данной работы является реализация стоимостной оптимизации распределённых JOIN-запросов. Для её выполнения были поставлены следующие задачи:

1. Провести обзор подходов стоимостной оптимизации, опубликованных в научных статьях и используемых в базах данных с открытым исходным кодом;
2. Разработать архитектуру решения;

3. Реализовать оптимизацию в качестве дополнительного модуля библиотеки sbroad на языке программирования Rust;

4. Провести эксперименты по качеству оптимизации SQL-запросов.

Подход к решению задачи переупорядочивания последовательности JOIN'ов определяется несколькими ключевыми аспектами:

- В каком виде хранится статистика по колонкам;
- Как устроен алгоритм оценки размеров задействованных в запросе таблиц и их результирующих соединений;
- Каким образом работает алгоритм переупорядочивания последовательности JOIN'ов.

Так для хранения статистики зачастую применяют следующие структуры данных: скетчи (структуры данных, основанные на вероятностных алгоритмах), сэмлы (некоторое показательное подмножество записей из таблицы, на основании которого можно делать выводы о характере данных во всей таблице) и гистограммы (структуры данных, хранящие информацию о распределении в виде промежутков фиксированной частоты) [2, 3]. Среди всех них гистограммы являются подходом как наиболее часто встречающимся в базах данных с открытым кодом, так и поддерживающим наибольшее количество операторов фильтрации (использующихся в WHERE и ON выражениях).

Среди алгоритмов переупорядочивания чаще всего используются динамические алгоритмы, рандомизированые и генетические алгоритмы [4], где генетические алгоритмы зачастую показывают себя как оптимальные с точки зрения качества оптимизации запроса и потраченного на неё времени.

В качестве ориентира для разработки оптимизации была выбрана база данных PostgreSQL. Эта база данных использует для оптимизации запросов гистограммы и генетический алгоритм переупорядочивания JOIN'a и написана на языке программирования C, что позволяет по необходимости довольно легко заимствовать логику для реализации оптимизации на языке программирования Rust.

На данный момент реализованы компоненты модуля оптимизации, отвечающие за:

- Трансформацию статистики в ходе разбора дерева плана;
- Подсчёт селективности SELECT'ов и JOIN'ов.

На этапе активной разработки находятся компоненты:

- Генетического алгоритма переупорядочивания JOIN’ов;
- Перестройки дерева изначального запроса в соответствии с логикой алгоритма переупорядочивания.

Исходный код оптимизации можно найти в репозитории библиотеки на GitLab'e (sbroad-core/src/cbo) [5].

ЛИТЕРАТУРА

1. Jaffray Justin. An Introduction to Join Ordering. – URL: https://www.cockroachlabs.com/blog/join-ordering-pt1 .
2. Katsov Ilya. Probabilistic data structures for web analytics and data mining. – URL: https://highlyscalable.wordpress.com/2012/05/01/probabilistic-structures-web-analytics-data-mining.
3. Hai Lan Zhifeng Bao Yuwei Peng. A Survey on Advancing the DBMS Query Optimizer: Cardinality Estimation, Cost Model, and Plan Enumeration. – URL: https://arxiv.org/pdf/2101.01507.pdf .
4. Michael Steinbrunn Guido Moerkott Alfons Kemp. Heuristic and Randomized Optimization for the Join Ordering Problem. – URL: https://www.csd.uoc.gr/~hy460/pdf/HeuristicandRandomizedOptimizationfortheJoinOrdering%20Problem.pdf .
5. Picodata LLC. sbroad library source code. – URL: https://git.picodata.io/picodata/picodata/sbroad.

УДК 004.4

А.А. Гузов (4 курс бакалавриата),
А.И. Тышкевич, к.т.н., доцент

РАСШИРЕНИЕ ОС ДЛЯ ЗАДАЧ РЕАЛЬНОГО ВРЕМЕНИ В СИММЕТРИЧНЫХ МУЛЬТИПРОЦЕССОРНЫХ СИСТЕМАХ

Целью работы является создание на базе мультипроцессорной, симметричной операционной системы программного комплекса, способного решать задачи реального времени.

В разработке систем реального времени возникает задача объединить в себе подпрограммы реального и «мягкого» времени. Например, лабораторное оборудование измеряет данные и передает их в компьютер для обработки. В данном случае требуется, чтобы управление прибором и получением от него данных выполняла подсистема реального времени, потом передавала данные в подсистему «мягкого» времени. А она в свою очередь произведет обработку, сохранение и выдачу этих данных пользователю.

Существует несколько различных решений данной задачи все они имею свою область применения и свой ценовой сегмент. Рассмотрим самые востребованные из них. Система с выделенным сопроцессором, например, am335x+PRU. Такая система имеет симметричную ОС Linux и на выделенном ядре RTOS, например QNX. Такое решение подойдет для управления сложными устройствами и будет давать строгую гарантию реального времени. Однако не имеет смысла использовать ее для простого лабораторного оборудования, к тому же для конечных пользователей это решение не подойдет. Хорошим решением для конечного пользователя Windows можно назвать гипервизор реального времени. Гипервизор – это программа или аппаратная схема, обеспечивающая или позволяющая одновременное, параллельное выполнение нескольких операционных систем на одном и том же хост-компьютере. Недостатков здесь два: система с низкими аппаратными возможностями не сможет поддерживать 2 ОС, а также высокая стоимость лицензии такой программы, которую к тому же придется покупать каждому конечному пользователю.

Область применения представленного решения – несложное лабораторное оборудование, для управления которым экономически невыгодно покупать специальную систему с выделенным сопроцессором реального времени, а легче использовать собственную систему на ОС Windows с выделенным ядром для задач, требующих гарантии реального времени.

В ходе исследования была проверена гипотеза возможности сосуществования Windows и системы реального времени на выделенном ядре процессора. Для этого необходимо было отключить одно из ядер процессора и с помощью драйвера выделить память, необходимую для работы системы, поместить в нее тестовый код, найти неработающее ядро и запустить исполнение кода.

Важным аспектом во взаимодействии двух систем является отправка прерываний, чтобы пробудить что-то в Windows, например, в ходе передачи данных. Для этого было рассмотрено несколько различных вариантов: NMI, MSI, прерывание по номеру вектора. Они могут применяться в разных версиях Windows, поэтому было принято решение использовать NMI прерывания в новых версиях, а прерывания по номеру вектора в Windows XP.

Архитектура проекта представляет собой следующие компоненты, выполняющие различные функции. Драйвер в режиме ядра – управляет подключением устройства к системе Windows, выделяет память для программы реального времени и запускает одно из ядер процессора на исполнение кода реального времени, также драйвер создает интерфейс, с помощью файлового объекта, по которому код реального времени и приложение Windows могут обмениваться данными. Код реального времени – исполняется на выделенном процессоре и управляет периферийным устройством, получает от него данные, обрабатывает и передает их в симметричную ОС по протоколу. Приложение Windows – взаимодействует с

драйвером, получает от него данные устройства, обрабатывает, сохраняет и показывает пользователю.

В качестве проверки на конференции будут представлены осциллограммы, демонстрирующие время задержки, которое дает гарантию реального времени.

ЛИТЕРАТУРА

1. Тышкевич Антон Игоревич, Медведев Борис Моисеевич // Использование эмулятора qemu для разработки программ управления периферийными устройствами: учебное пособие // 2022 // URL: https://elib.spbstu.ru/dl/5/tr/2022/tr22-109.pdf/info // Дата обращения: 25.02.2023
2. Кокаровцев Владимир Павлович, научный руководитель Тышкевич Антон Игоревич // Применение сопроцессоров реального времени процессора am335x arm cortex-a8 sitara для создания нестандартных цифровых интерфейсов// Санкт-Петербург 2020 // Дата обращения: 20.02.2023
3. Intel® 64 and IA-32 ArchitecturesSoftware Developer’s Manual // 2016 // URL: https://pdos.csail.mit.edu/6.828/2018/readings/ia32/IA32-2A.pdf // Дата обращения: 01.02.2023
4. Разработка, тестирование и развертывание драйверов // URL: https://learn.microsoft.com/ru-ru/windows-hardware/drivers/develop // Дата обращения: 05.02.2023

УДК 004.891.2

Н. С. Дергунов (4 курс бакалавриата),
О. Г. Малеев, к.т.н., доцент

РАЗРАБОТКА АЛГОРИТМА АВТОМАТИЧЕСКОЙ ТОРГОВЛИ НА ФОНДОВОМ РЫНКЕ С ИСПОЛЬЗОВАНИЕМ ОБУЧЕНИЯ С ПОДКРЕПЛЕНИЕМ

Фондовые рынки являются важным элементом современной экономики и финансовой системы. Они позволяют компаниям привлекать инвестиции и капитал для развития бизнеса, а инвесторам – получать доход от инвестирования в акции и другие финансовые инструменты. Фондовые рынки также являются индикатором экономического состояния страны и мировой экономики в целом [1]. Они отражают изменения в экономической политике, тенденции в мировой торговле и другие факторы, которые могут повлиять на бизнес и инвестиции.

Алгоритмическая торговля акциями стала основным элементом современного финансового рынка, большинство сделок сейчас полностью автоматизированы. Агенты глубокого обучения с подкреплением доказали свою состоятельность во многих сложных задачах, таких как управление автомобилем, игры в шахматы и го [2]. Мы можем рассматривать исторические ценовые ряды и движения фондового рынка как сложную среду с неполной информацией, в которой мы пытаемся максимизировать прибыль и минимизировать риск.

Разработка алгоритма автоматической торговли на фондовой бирже является актуальной задачей в современном мире. Это связано с увеличением объемов и скорости совершения сделок на фондовых биржах, что приводит к необходимости быстрого и эффективного принятия решений на основе больших объемов данных.

Известен алгоритм торговли на бирже с использованием нейронной сети долгой краткосрочной памяти (Long Short-Term Memory, LSTM), основанный на анализе временных рядов ценовых данных и прогнозировании будущих цен [3]. Алгоритм показал неплохие результаты в прогнозе цены акций, но в работе [3] осталась неосвещенной реализация непосредственно стратегии покупок-продаж.

Цель данной работы улучшить результаты работы алгоритма, внедрив стратегию автоматической торговли, которую будет реализовывать агент, построенный на базе нейронной сети и обученный с подкреплением [4].

Для достижения этой цели необходимо решить следующие задачи:

1. Сбор и обработка данных: необходимо собрать и обработать данные о ценах активов, объемах торгов и других факторах, влияющих на рынок [5].
2. Выбор и обучение модели:

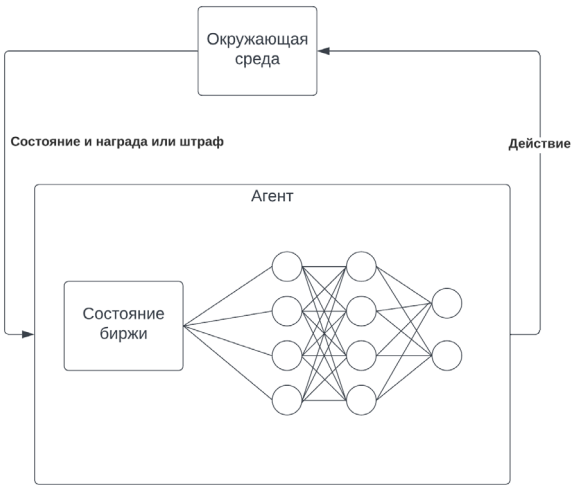


Рисунок 1 – Архитектура системы

3. Определение стратегии торговли: на основе предсказаний модели нужно разработать стратегию торговли, которая будет учитывать риски и максимизировать прибыль.

На настоящий момент нами выбрана стратегия, при которой у агента есть три действия в каждый момент времени: держать актив, продавать его или покупать. Итоговой наградой является прибыль или убыток за период t. В зависимости от удачности действия мы будем получать вознаграждения или штраф, в случае с продажей эквивалентный разнице закрытия сделок, для покупки введен небольшой штраф (комиссия) за долгое удержание финансового инструмента, что будет понуждать агента к совершению сделок. Так же введем коэффициент обесценения вознаграждения со временем, который будет определять важность будущих вознаграждений.

Реализована данная архитектура и стратегия на языке программирования Python с использованием дополнительных пакетов. Применение нейронных сетей и обучения с подкреплением продемонстрировало огромный потенциал применимости, позволяющий построить алгоритм успешной автоматической торговли на бирже. Программное обеспечение позволяет тестировать алгоритм на исторических данных и следить за его работой в реальном масштабе времени. Важно также проводить регулярное обновление и подстройку алгоритма для поддержания его эффективности в изменяющихся рыночных условиях.

ЛИТЕРАТУРА

1. Ли Япэй, Л. А. Гузикова. Мировые фондовые индексы: бакалаврская работа// Санкт-Петербургский политехнический университет Петра Великого, Институт промышленного менеджмента, экономики и торговли, 2016 г. – СПб. [Электронный ресурс] URL: https://elib.spbstu.ru/dl/2/v16-805.pdf/view (дата обращения: 20.03.2023).
2. Tidor-Vlad Pricope. Deep Reinforcement Learning in Quantitative Algorithmic Trading: A Review [Электронный ресурс] URL: https://arxiv.org/abs/2106.00123 (дата обращения: 20.03.2023).
3. Чигасова М. А., Малеев О. Г. Прогнозирование котировок акций с использованием нейронной сети // Современные технологии в теории и практике программирования. Сборник материалов конференции. Санкт-Петербург, 2022. С. 180–182.
4. Chloe Wang. A Look Into Neural Networks and Deep Reinforcement Learning. 31 октября 2021г. [Электронный ресурс] URL: https://chloeewang.medium.com/a-look-into-neural-networks-and-deep-reinforcement-learning-2d5a9baef3e3 (дата обращения: 20.03.2023).
5. Preparing Your Dataset for Machine Learning: 10 Basic Techniques That Make Your Data Better, 19 марта 2021г. [Электронный ресурс] URL: https://www.altexsoft.com/blog/datascience/preparing-your-dataset-for-machine-learning-8-basic-techniques-that-make-your-data-better/ (дата обращения: 20.03.2023).

УДК 004.021

Е. Н. Кемпи (4 курс бакалавриата),
И. Г. Черноруцкий, д.т.н., профессор

ИССЛЕДОВАНИЕ АЛГОРИТМА РАСПОЗНАВАНИЯ РУКОПИСНОГО ТЕКСТА С ИСПОЛЬЗОВАНИЕМ СВЕРТОЧНОЙ РЕКУРРЕНТНОЙ НЕЙРОННОЙ СЕТИ В АВТОНОМНОМ РЕЖИМЕ

Целью работы является исследование алгоритма распознавания рукописного текста с использованием сверточной рекуррентной нейронной сети в автономном режиме.

Распознавание рукописного текста в автономном режиме — это задача распознавания последовательности изображений в рамках компьютерного зрения. Традиционные подходы основаны на лексической сегментации, сложных методах извлечения признаков и значительных знаниях в области лингвистики. Именно по этой причине предлагается подход распознавания рукописного текста с использованием сверточной рекуррентной нейронной сети в сочетании с коннекционистской временной классификацией. Преимуществом данной сети является отсутствие зависимости от лексической сегментации, а также от ручного извлечения признаков.

Алгоритм, рассматриваемый в данной работе, может быть полезен для применения в учреждениях, хранящих по сей день большой объем рукописных записей. Такими учреждениями являются больницы, в которых до сих пор присутствует ручное заполнение медицинских карт, бланков, рецептов на лекарственные препараты, а также учебные заведения, такие как школы или университеты, в которых ученики/студенты записывают лекционный материал от руки.

Архитектура данной сверточной рекуррентной нейронной сети представляет из себя последовательность из 3 блоков:

– «Сверточный блок» для извлечения признаков изображения;

– «Рекуррентный блок» для изучения последовательности;

– «Блок маркировки» с коннекционистской временной классификацией.

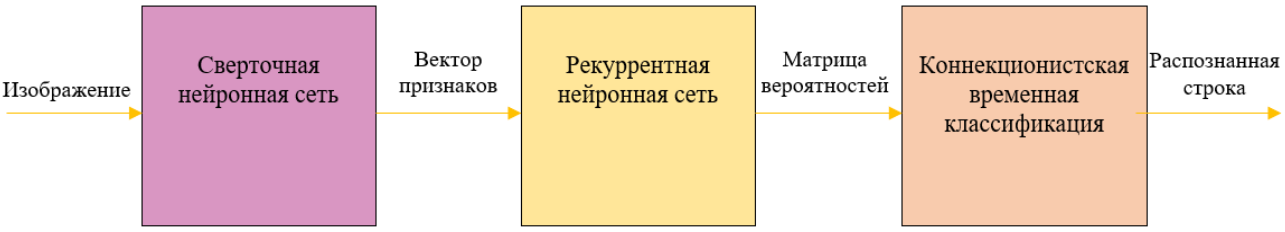


Рисунок 1 – Схема алгоритма распознавания рукописного текста с использованием сверточной рекуррентной нейронной сети

В процессе работы алгоритма происходит следующее:

– Поступает входное изображение, которое уменьшается и передается на сверточную нейронную сеть;

– Сверточная нейронная сеть формирует вектор признаков, который позже передает рекуррентной нейронной сети;

– Выходные данные рекуррентной нейронной сети вместе с последовательностью меток передаются, проходя через модуль коннекционистской временной классификации;

– В коннекционистской временной классификации происходит вычисление потерь, обновление весов и создание прогнозов;

– Результатом является распознанная строка.

Для обучения сети применяется подход стохастического градиентного спуска (SGD). В SGD случайным образом выбирается только один обучающий пример для вычисления градиентов функции стоимости на каждой итерации, что дает оценку истинного градиента. Это решение приводит к довольно шумному пути сходимости, но в итоге он сходится намного

быстрее, чем обычный градиентный спуск. Такой шум является также улучшающим фактором для прогнозов. Функция стоимости для нейронной сети обычно является многомерной, с несколькими локальными минимумами, различающимися по глубине. Шаг градиента с шумом может позволить модели обнаружить новый более глубокий локальный минимум. По этим причинам стохастический градиентный спуск предпочтительнее обычного градиентного спуска.

В предлагаемой системе применяется метод оптимизации Адама, представляющий собой алгоритм оптимизации стохастического градиентного спуска первого порядка. Он прост в реализации, эффективен в вычислительном отношении и требует меньше памяти, чем большинство его аналогов.

В качестве средств для реализации модели и ее последующего тестирования был выбран язык программирования Python и среда разработки PyCharm Community.

Данный алгоритм был протестирован путем использования нескольких вариаций изображений с разными почерками и разными фразами на английском языке.

Алгоритм показывает хорошие результаты для данной области. В будущем можно рассмотреть различные способы модернизации системы для повышения процента распознаваемости текста.

ЛИТЕРАТУРА

1. И. Г. Черноруцкий, Н. В. Воинов, Л. П. Котлярова // Методы оптимизации. Учебное пособие // Санкт-Петербург 2020
2. Интуитивное объяснение коннекционистской временной классификации. [Электронный ресурс] Режим доступа: https://machinelearningmastery.ru/intuitively-understanding-connectionist-temporal-classification-3797e43a86c
3. Offline handwritten text recognition using convolutional recurrent neural network. [Электронный ресурс] Режим доступа: https://www.researchgate.net/publication/340118194_Offline_Handwritten_Text_Recognition_using_Convolutional_Recurrent_Neural_Network

УДК 004.6

Р. А. Долматов (1 курс аспирантуры),
С. Э. Сараджишвили, к.т.н., доцент

ИССЛЕДОВАНИЕ ПОДХОДОВ ДЕТЕКЦИИ ОБЪЕКТОВ С ПОМОЩЬЮ СИМУЛЯТОРА СБОРА ДАННЫХ

Автопилотирование, автомашинист, беспилотное управление — это все разная терминология в определении современных систем автоматизированного управления движением поездов, предназначенного для качественного обеспечения перевозок из одного пункта. В нынешнюю эпоху в основном существуют две категории обнаружения объектов: подходы обнаружения «снизу вверх» и подходы «сверху вниз». Люди считали, что подходы «снизу вверх» могут занимать много времени и приводить к большему количеству ложных срабатываний, в то время как подходы «сверху вниз» постепенно превратились в основные подходы из-за их эффективности на практике. Все подходы «сверху вниз» моделируют каждый объект как предшествующую точку или предопределенную якорную рамку, а затем предсказывают соответствующие смещения к ограничивающей рамке. Им нравится способность воспринимать объекты как единое целое, что упрощает пост-процесс создания ограничивающих рамок. Однако они обычно испытывают трудности с восприятием объектов необычной формы.

Движок для симулятора был выбран, исходя из начальных условий. Использовался движок Unity 3D, было решено использовать этот же движок. Для написания клиента был выбран язык программирования Python, так как он очень удобен и легок в применении для

данной задачи. Для передачи данных от сервера клиенту был выбран протокол передачи данных protobuf [4]. В процессе создания симулятора осталось объединить наработки двух компаний, добавить необходимых актеров, управляющие воздействие, создать клиент-серверную архитектуру для приема данных и управления симуляцией. Сервер симулирует мир с актерами, в нем также можно менять погодные условия и время суток. Он также симулирует посылку данных от сенсоров: камер, лидаров, радаров и тепловизора. Также реализована передача истинных значений семантической сегментации и карты глубины. Симулятор включает в себя 3D-модели (пешеходы, животные, велосипедисты, деревья и др.) и сценарий поведения этих объектов. Есть возможность моделировать поведение людей и животных и тестировать реагирование алгоритмов на появление препятствий в разных зонах. Клиент же посылает серверу сигналы управляющего воздействия: включение/выключение фар, подача гудка, экстренное торможение. Физика разгона и торможения состава тоже вынесена в клиентскую часть, данные берутся из протоколов испытаний, а на сервере скорость задается константой.

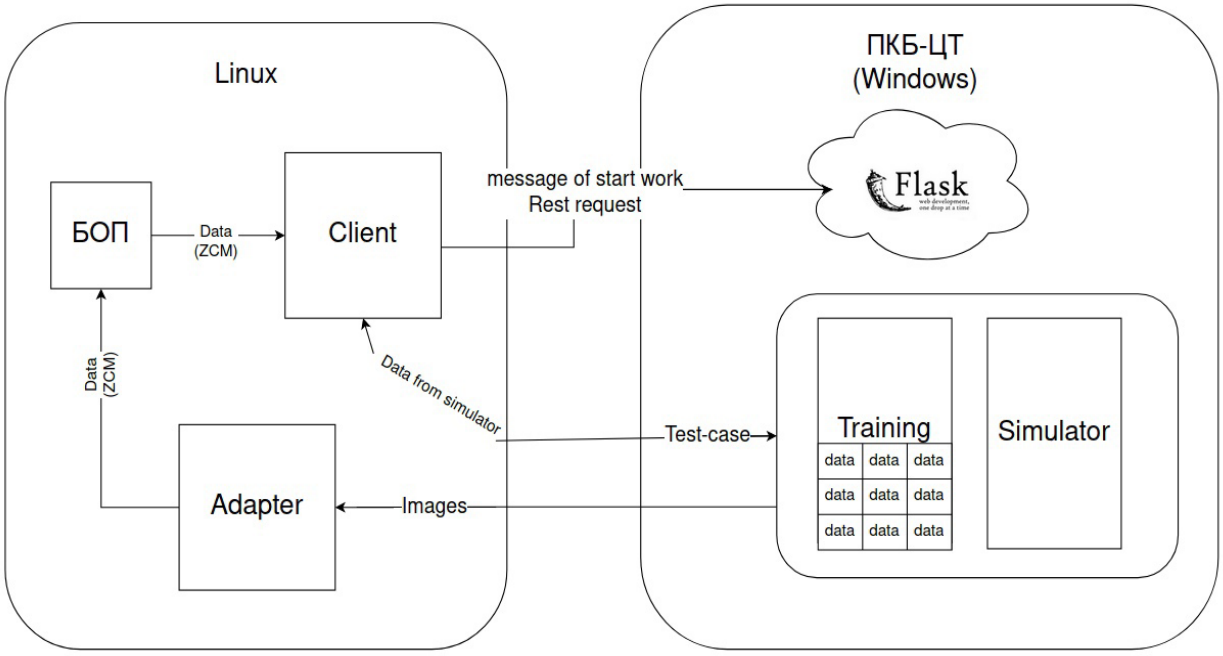


Рисунок 1 – Архитектура системы

Исследование детекции объектов происходит различными способами. Идея в том, чтобы иметь три хитмапы(тепловая карта). Хитмапы мультиклассовые, для каждого класса своя хитмапа. Чтобы у каждого пикселя в хитмапах был контекст, делается центр и угол.

Для достижения этой цели необходимо решение следующих *задач*:

1. Центр - для каждого пикселя предсказываем локацию центра объекта, для возвращения
2. Локации в оригинальном изображении так как хитмапа имеет меньшее разрешение.
3. Для каждого пикселя предсказываем локацию верхнего левого угла и вектор эмбеддинга для сопоставления.
4. Для каждого пикселя предсказываем локацию нижнего правого оффсет, вектор эмбеддинга для сопоставления.

За счет разнообразия датасета OMNI3D модель Cube R-CNN демонстрирует сильное обобщение (generalization) и показывает лучше, чем предыдущие работы, результаты, работая как внутри помещений, так и снаружи. За счет того, что модель работает с виртуальной шириной, а не метрическая глубина, решается проблема того, что изображения, снятые камерами с разными фокусными расстояниями, преувеличивают неоднозначность масштаба глубины. Глубина объекта преобразовывается в виртуальной глубине так, что

эффективный размер изображения и фокусное расстояние одинаковые среди всех изображений. И так получается, что как будто изображение снято с одной и той же виртуальной камеры и все изображения как будто сняты с одинаковыми значениями. Модель Cube R-CNN — это развитие Faster R-CNN с дополнительной головой для предсказания объектов кубоидов по каждому обнаруженному 2D объекту.

Реализована модель Cube R-CNN, используя Detectron2 и PyTorch3D. Используется DLA-34, переобученный на ImageNet, с FPN. Обучение проходило в 128 эпох в 192 изображениях, распределенных среди 48 GPU V100s. Использовался SGD с темпом обучения в 0.12, который снижался в 10 раз после 60% и 80% обучения. Во время обучения использовалось аугментацию с горизонтальным отражением и измерением размера изображения $\in$ [0.50, 1.25]. Виртуальные параметры fv = Hv = 512.

Главной метрикой для 3D детекции используется метрика average-precision (AP). Количественный анализ показал, что 3D детекция - не простая задача, особенно для дальних объектов.Модель Cube R-CNN лучше, чем M3D-RPN, которая на KITTI в среднем делает вывод 191 миллисекунд и лучше, чем GUPNet, которая на KITTI в среднем делает inference 66ms. Собраны несколько открытых 3D датасетов (KITTI, SUN RGB-D, nuScenes, Objectron, ARKitScenes и Hypersim) в один Omni3D, где получается 234 тысяч размеченных изображений, 3 млн 3D объектов в 97 категориях (которые включают в себя различные атрибуты). Датасет разделен на 175 тысяч изображений для обучения, 19 тысяч для validation и 39 тысяч для теста.

Для более быстрой оценки по такому большому датасету был добавлен новый, быстрый, batched алгоритм для расчета IoU для 3D, который в 450 раз быстрее предыдущих решений. Алгоритм заключается в том, что выполняется поиск формы пересечения путем представления кубоида и происходит поиск пересечений граней (faces). Алгоритм самодостаточен и написан на C++ и CUDA. Реализация в 90 раз быстрее, чем на C++ и в 450 раз быстрее на CUDA.

ЛИТЕРАТУРА

1. Ивлев В. А., Никифоров И. В., Леонтьева Т. В. Обработка данных в геоинформационных системах для выбора местоположения рекламы // Современные технологии в теории и практике программирования: сборник материалов конференции, Санкт-Петербург, 19 апреля 2019 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2019. – С. 27-30.
2. James Turnbull. “The Docker Book” Version v17.03.0 – 2017. – 400pp.
3. Shemyakinskaya A. S., Nikiforov I. V. Hard drives monitoring automation approach for Kubernetes container orchestration system // Proceedings of the Institute for System Programming of the RAS. – 2020. – Vol. 32. – No 2. – P. 99-106. – DOI 10.15514/ISPRAS- 2020-32(2)-8.
4. J. Banks; J. Carson; B. Nelson; D. Nicol (2001). Discrete-Event System Simulation. Prentice Hall. p. 3. ISBN 978-0-13-088702. [Электронный ресурс]
5. Rong, Guodong and Shin, Byung Hyun and Tabatabaee, Hadi and Lu, Qiang and Lemke, «LGSVL Simulator: A High Fidelity Simulator for Autonomous Driving», arXiv preprint, 2020;Ermakov N. V., Molodyakov S. A. A caching model for a quick file access system // Journal of Physics: Conference Series. IOP Publishing, 2021. Т. 1864. №. 1programming language. [Электронный ресурс] Режим доступа: https://go.dev/
6. vSphere. Xiang Li, Wenhai Wang, Xiaolin Hu, Jun Li, Jinhui Tang, and Jian Yang. Generalized focal loss v2: Learning reliable localization quality estimation for dense object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 11632–11641, 2021

УДК 004.89:007.3

В. А. Ивлев, Ф. Ю. Чемашкин (2 курс аспирантуры),
И. В. Никифоров, к.т.н., доцент,
А. Д. Ковалев, ассистент

ИССЛЕДОВАНИЕ СИСТЕМ АВТОМАТИЗАЦИИ НАСТРОЙКИ ИНФРАСТРУКТУРЫ ИТ-ПРОЕКТА

Сегодня программное обеспечение (далее ПО) является неотъемлемой частью многих аспектов повседневной деятельности человека. С ростом сложности приложений и инфраструктуры появляются новые вызовы в управлении ИТ-систем и настройке инфраструктуры для проектов. Это может приводить к увеличению затрат времени и ресурсов на создание ПО в большом количестве, а также к увеличению зависимости от квалификации ИТ-специалистов [1]. Таким образом автоматизация процессов настройки ИТ-инфраструктуры становится все более важной, чтобы упростить и ускорить процесс развертывания и поддержки ИТ-проектов.

Исходя из этого внедрение систем автоматизации - один из подходов для улучшения процесса разработки ПО, который позволяет оптимизировать этапы разработки [2]. Несмотря на это, внедрение в полноценный ИТ-проект такой системы является сложной задачей, т.к. сам проект может имеет множество уникальных факторов, которые в свою очередь могут увеличить сложность и время выполнения работ настройки его инфраструктуры [3]. Большинство доступных на рынке систем являются либо платными и требуют опытных специалистов, либо не могут решить поставленные задачи автоматизации инфраструктуры на проекте, так или иначе, не удовлетворяя требования к автоматизации инфраструктуры.

Таким образом, опираясь на анализ актуальности автоматизации настройки инфраструктуры ИТ-проекта [4] и ряд проблем, с которыми потенциально сталкиваются ИТ предприятия при автоматизации инфраструктуры проектов были выдвинуты ряд требований к системам, а именно: управление конфигурацией, декларативное программирование, скриптование, интеграция с облачными сервисами, открытый исходный код, расширяемость, простота использования, гибкость настройки, удобный интерфейс.

На основе поставленных требований было проведено исследование следующих систем автоматизации настройки инфраструктуры: SaltStack, Terraform, 1С Предприятие, Chef, Ansible, JetBrains Space, Puppet.

После получения результатов первого этапа исследования по сопоставлению систем и выставленных к ним критериям, описанных выше, только три системы имеют возможность декларативного программирования [5] – Puppet [6], Terraform [7], JetBrains Space [8]. Это в свою очередь является важным пунктом при автоматизации инфраструктуры ИТ-проекта, потому что для большинства ИТ-проектов подходящей системой для автоматизации поставленных ими задач в большинстве случаев будет является система, принимающая на вход описание задачи и на его основе реализующая автоматическую настройку инфраструктуры. Декларативный стиль программирования в этом случае позволяет описывать, какие условия должны быть выполнены для достижения желаемого результата, а не как нужно его достигнуть.

Поэтому системы, имеющие возможность декларативного стиля программирования, требуют дополнительного анализа возможностей, в том числе и на предмет их функционала в качестве систем автоматизации настройки инфраструктуры ИТ-проекта с возможностью преобразования описанных задачи в конфигурационный файл [9] с набором инструкций с его дальнейшим выполнением.

Таким образом можно сформулировать следующие критерии для систем с возможностью поддержки декларативного программирования: декларативное описание инфраструктуры, отчетность, оперативное управление инфраструктурой, разработка в команде, модульность, кроссплатформенность, работа с инфраструктурой как с кодом,

развертывание инфраструктуры, поддержка преобразования описанной задачи в конфигурационный файл с его дальнейшим выполнением.

По результатам второго этапа исследования можно сделать вывод, что ни одна из систем не поддерживает преобразование описанной задачи в конфигурационный файл с его дальнейшим выполнением в виде инструмента для автоматизации настройки инфраструктуры ИТ-проекта. Таким образом, была выявлена область, которая требует более детального анализа и потенциально является актуальной для решения проблем сегодняшней автоматизации инфраструктуры ИТ-проекта. Реализация системы с поддержкой данного требования в виде метода с использованием Natural Language Processing (NLP) [10] может являться важным шагом в развитии систем автоматизации настройки инфраструктуры ИТ-проектов.

В дальнейшем, авторами планируется на основе проведенного исследования предложить реализацию прототипа системы, в основе которого будет лежать поддержка преобразования человекочитаемого описания задач в набор команд и их дальнейшее выполнение с целью автоматизации настройки инфраструктуры ИТ-проекта.

ЛИТЕРАТУРА

1. Chougule S. et al. Knowledge Acquisition for Automation in IT Infrastructure Support //Mining Intelligence and Knowledge Exploration: Second International Conference, MIKE 2014, Cork, Ireland, December 10-12, 2014. Proceedings. – Springer International Publishing, 2014. – С. 351-360.
2. Al-Ali A. R., Al-Rousan M. Java-based home automation system //IEEE Transactions on Consumer Electronics. – 2004. – Т. 50. – №. 2. – С. 498-504.
3. Ивлев, В. А. Обработка данных в геоинформационных системах для выбора местоположения рекламы / В. А. Ивлев, И. В. Никифоров, Т. В. Леонтьева // Современные технологии в теории и практике программирования : сборник материалов конференции, Санкт-Петербург, 19 апреля 2019 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2019. – С. 27-30. – EDN DNQPMC.
4. Ивлев В. А., Никифоров И. В. Актуальность автоматизированной настройки инфраструктуры ит-проекта // Современные технологии в теории и практике программирования : Сборник материалов конференции, Санкт-Петербург, 26 апреля 2022 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2022. – С. 79-80.
5. Дудин, А. А. Применение языков декларативного программирования в автоматизированных обучающих системах : специальность 05.13.11 "Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей" : диссертация на соискание ученой степени кандидата технических наук / Дудин Андрей Александрович. – Москва, 2003. – 173 с. – EDN QDVYJL.
6. Kumar M. et al. Infrastructure as Code (IaC): Insights on Various Platforms //Sentiment Analysis and Deep Learning: Proceedings of ICSADL 2022. – Singapore : Springer Nature Singapore, 2023. – С. 439-449.
7. Brikman Y. Why we use terraform and not chef, puppet, ansible, saltstack, or cloudformation //Retrieved April. – 2016. – Т. 24. – С. 2020.
8. Jabrayilzade E. et al. Bus factor in practice //Proceedings of the 44th International Conference on Software Engineering: Software Engineering in Practice. – 2022. – С. 97-106.
9. Сысоев И. М., Никифоров И. В., Каплан Е. В. Создание подхода автоматизации обновления конфигурационных файлов программного обеспечения // Современные технологии в теории и практике программирования: сборник материалов конференции, Санкт-Петербург, 19 апреля 2019 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2019. – С. 126-127.
10. Kovalev A., Voinov N., Nikiforov I. Using the Doc2Vec Algorithm to Detect Semantically Similar Jira Issues in the Process of Resolving Customer Requests // Studies in Computational Intelligence. – 2020. – Vol. 868. – P. 96-101. – DOI 10.1007/978-3-030-32258-8_11.

УДК 004.932.72

А. Н. Кулачкова (4 курс бакалавриата),
В. С. Тутыгин, к.т.н., доцент

СИСТЕМА ДИАГНОСТИКИ БОЛЕЗНЕЙ РАСТЕНИЙ ПО ИЗОБРАЖЕНИЯМ ЛИСТЬЕВ С ПРИМЕНЕНИЕМ НЕЙРОННОЙ СЕТИ ПРИ МНОГОЦВЕТНОЙ КЛАСТЕРИЗАЦИИ ИЗОБРАЖЕНИЙ ЛИСТЬЕВ

Пшеница – ценная зерновая культура, как для России, так и для всего мира. Она выращивается на всех континентах, площадь посевов, занимает наибольшую территорию среди всех сельскохозяйственных культур. Важно оберегать эту культуру от различных болезней, которые могут уничтожить все посевы при широком распространении.

Правильная диагностика болезни пшеницы – один из важнейших этапов для успешного проведения защитных мер. Агрономы могут распознать и определить ее, однако иногда могут возникнуть сложности из-за нескольких факторов: ослабление компетентности со временем, долгое и дорогое обучение специалистов, при неопределенных ситуациях может потребоваться рекомендация от другого агронома. В качестве эксперта предлагается использовать компьютерную программу с обученной нейронной сетью. Основная проблема заключается в том, что количество изображений, классифицированных экспертами, недостаточно для обучения нейронной сети, необходимо его увеличить.

На данную тему было предложено несколько решений. В статье «Новый метод диагностики болезней растений на основе цифрового описания изображений листьев и нейронной сети прямого распространения» Тутыгин В. С., Прокофьев О. В. [1] предложили использовать 6 цветовых компонент и 4 параметра Харалика для описания изображений листьев, в результате для 6 болезней количество правильных результатов диагностики составило 97%.

Таким образом, *целью* данной работы является исследование возможности улучшения качества диагностики за счет использования многоцветной кластеризации изображений и разработать на этой основе систему диагностики болезней растений.

Для достижения цели необходимо решение следующих *задач*:

1. Обзор существующих подходов для диагностики болезней растений.
2. Исследование возможности улучшения качества диагностики за счет использования многоцветной кластеризации изображений.
3. Реализация данного подхода на основе системы диагностики болезней растений.
4. Демонстрация улучшения качества диагностики болезней растений.

Точное описание цветов имеет большое значение. Нужна система цветов, поскольку просто названий недостаточно, потому что люди по-разному трактуют их. Исходные изображения, поделенные на 8 классов, имеют RGB формат, который состоит из 3 основных цветов: красный, зеленый и синий. Цветовая модель HSV лучше, чем RGB [2]. Это связано с тем, что HSV может выражать оттенок, насыщенность и яркость. В данной работе исходные изображения из модели RGB переводятся в HSV, чтобы использовать только одну переменную для выполнения операций, связанных с цветом. Значение цветового оттенка лежат в интервале от 0° до 360°.

Для выделения компонентов, связанных с цветом, на каждое изображение накладывается маска от одного определенного значения оттенка до другого. Количество компонентов и значения оттенков можно варьировать. Было выбрано 16 компонентов с диапазоном оттенка от 0° до 150°.

Роберт М. Харалик [3] предложил способ описания текстурных особенностей изображения, который основан на предположении, что текстурная информация на изображении содержится в общей или "средней" пространственной связи, которую серые тона в изображении имеют друг к другу. Из полученных компонент исходных изображений вычисляются матрицы с 8 уровнями серого. Для каждой рассчитываются Gray Level Co-

Occurrence Matrix (GLCM) – матрица совпадений уровней серого. GLCM представляет собой матрицу, которая представляет отношения соседства между пикселями в изображении под разными углами и определенными расстояниями [4]. Для получения GLCM матрицы рассчитываются 4 GLCM матрицы с расстоянием между пикселями, равным 1, и с угловыми направлениями 0°, 45°, 90°, 135°, после данные матрицы нормируются. Итоговая GLCM матрица получается путем нахождения их среднего значения.

Роберт М. Харалик предложил 14 параметров [3], которые можно найти с помощью GLCM матриц. Нами установлено, что для надёжного распознавания болезней растений достаточно использовать 6 их них: контрастность, корреляция, неоднородность, второй угловой момент и энергия.

При помощи генерации случайных чисел с нормальным распределением с использованием усредненных значений параметров Харалика для 80 реальных изображений каждой цветовой компоненты в каждом классе было создано 1000 цифровых описаний изображений для тестирования, 200 для валидации и 200 для тестирования для каждого класса.

Сеть прямого распространения является одним из типов искусственных нейронных сетей. В этой сети информация перемещается только в одном направлении вперед от входных узлов, через скрытые узлы и к выходным узлам [5]. Именно ее используем ее для обучения на сгенерированных цифровых описаниях изображений.

Диапазон разброса значений параметров цифровых описаний с использованием рандомизации на фазе обучения нейронной сети, необходимо взять таким, чтобы вероятность правильного распознавания болезни пшеницы была не менее 97%. Если доверительные интервалы цифровых описаний реальных изображений больше этого значения, необходимо предъявлять нейронной сети цифровые описания на основе усреднённых значений параметров Харалика для нескольких анализируемых изображений листьев. Если доверительный интервал разброса значений какого-либо параметра цифровых описаний реальных изображений без учёта усреднения больше доверительного интервала для заданного уровня доверительной вероятности, то в предположении, что закон распределения значений этого параметра близок к нормальному, необходимое количество усреднений может быть вычислено на основе выражения для доверительного интервала математического ожидания.

Подход *был реализован* на языке программирования Python [6]. В результате обучения количество правильно опознанных изображений из обучающей выборки составило 99% при делении на 8 классов. Нейронная сеть была протестирована на реальных изображениях, не использованных при обучении. Для каждого класса отбиралось 5-10 изображений, параметры которых усреднялись по нескольким изображениям для тестирования, при таком подходе была достигнута высокая вероятность правильного распознавания класса болезни.

ЛИТЕРАТУРА

1. Тутыгин В. С., Прокофьев О. В. Новый метод диагностики болезней растений на основе цифрового описания изображений листьев и нейронной сети прямого распространения // Современная наука: актуальные проблемы теории и практики. Серия: Естественные и Технические Науки. – 2022. – №10. – С. 135-143 DOI 10.37882/2223-2966.2022.10.33
2. Ningsih L., Cholidhazia, P. Classification Of Tomato Maturity Levels Based on RGB And HSV Colors Using KNN Algorithm. // Journal of Artificial Intelligence and Digital Business. – 2022. – Vol. 1. – No 1. – P. 25-30.
3. Haralick R. M., Shanmugam K., Distein I. Textural features for image classification. // IEEE Transactions on systems, man and cybernetics. – 1973. – Vol. SMC-3. – No 6. – P. 610-621.
4. Lesiangi F. S., Mauko A., Djahi B. S. Feature extraction Hue, Saturation, Value (HSV) and Gray Level Cooccurrence Matrix (GLCM) for identification of woven fabric motifs in South Central Timor Regency. // Journal of Physics: Conference Series. – 2021. – Vol. 2017. – No 1. – P. 012010.
5. Львов Ф. М. Типы и применение нейронных сетей. // Кронос: естественные и технические науки. – 2019. – № 2. – С. 25-31
6. Python. [Электронный ресурс] Режим доступа: https://www.python.org/

УДК 004.453

А. А. Лейнсоо (4 курс бакалавриата),
П. Д. Дробинцев., к.т.н., доцент,

РАЗРАБОТКА КОРПОРАТИВНЫХ ПРИЛОЖЕНИЙ С ИСПОЛЬЗОВАНИЕМ ПЛАТФОРМЫ БЫСТРОЙ РАЗРАБОТКИ JMIX

В 2022 году на Российском рынке разработки корпоративного программного обеспечения возникла необходимость разработки большого количества различных компонентов корпоративных информационных систем. Это связано с тем, что крупные поставщики (вендоры) корпоративных решений, такие как SAP, Oracle, Microsoft, AspenTech, и многие другие отказались от продажи и поддержки решений на территории России [1], [2]. Использование программного обеспечения из недружественных стран несет за собой неоправданно большие санкционные и технологические риски, недопустимые для крупных технологических и производственных компаний, включая уничтожение данных, блокировку сервисам, отключение оборудования. Кроме того, согласно Указу Президента России №166 с 01 января 2025 года вводится запрет на использование иностранных решений органами государственной власти и государственными заказчиками.

Соответственно, большинство крупных компании столкнулись с необходимостью замены решений зарубежных поставщиков. Однако на Российском рынке отсутствуют готовые решений соответствующих корпоративных масштабов, простая замена одной «коробки» на другую не представляется возможной, т.к. значительные ресурсы вложены в кастомизацию решений поставщика под конкретные бизнес-процессы в компании.

В такой ситуации наиболее обоснованным решением является использование инструментов для быстрой разработки с поддержкой всех уровней разработки (full-stack framework), которые могли бы обеспечить реализацию компонентов корпоративных систем, отличающихся наличием большой модели данных, сложной бизнес-логикой, большим количеством и разнообразием форм пользовательского интерфейса, сотнями или тысячами конкурентных пользователей.

Целью работы является оценка эффективности использования инструментов быстрой разработки на примере реализации функционального модуля в действующей корпоративной информационной системе группы компаний Bacora.PGS (ООО «Бакора ПГС»).

Для достижения этой цели необходимо решение следующих задач:

1. Обзор существующих инструментов быстрой разработки (включая «LowCode» и «NoCode» технологий).
2. Проведение сравнительного анализа выбранных инструментов.
3. Выбор инструмента.
4. Реализация данного реализация функционального модуля с использованием выбранного инструмента.
5. Оценка эффективности применения инструмента (оценка снижения трудоемкости).

Одним из наиболее функциональных, надежных и зарекомендовавших себя инструментов быстрой разработки является платформа JMIX (ранее CUBA Platform, [3]).

Jmix основан на Spring Boot [4], что является фактически стандартом для разработки корпоративных веб-приложений на Java. Преимуществами JMIX являются:

– лицензия только на разработчиков, количество пользователей не ограничено;

– современный стек технологий на основе решений с открытым исходным кодом (Java/Kotlin);

– наличие платформенной среды разработки на основе IntelliJ IDEA и инструменты для ускорения разработки (Jmix Studio);

– масштабируемость, модульность и большое количество готовых дополнения для решения различных задач и построения пользовательского интерфейса;

– технологическая независимость за счет открытого исходного кода, наличия поддержки Российской компании, входит в реестр Российского программного обеспечения;

– возможность независимого создания расширений основного продукта для адаптации под конкретного заказчика

В качестве компонента для реализации и последующей оценки эффективности сотрудниками компании-заказчика был предложен модуль агрегации ценовой информации. Модуль должен быть реализован в контексте общей корпоративной системы управления проектами, которая находится в фазе внедрения и активной доработки. Для реализации необходимо будет использовать средства построения модели данных, получение информации из корпоративного хранилища, описания экранов, загрузку данных из внешних источников, формирования ролевого и ресурсного доступа.

По результатам разработки будет произведена оценка эффективности освоения и применения стека инструментов JMIX.

ЛИТЕРАТУРА

1. Уход иностранных IT-вендоров оставил больше 50% российских компаний без техподдержки. [Электронный ресурс] Режим доступа: https://www.forbes.ru/tekhnologii/479623-uhod-inostrannyh-it-vendorov-ostavil-bol-se-50-rossijskoj-kompanij-bez-tehpodderzki
2. 90 компаний из IT сферы которые ушли с рынка России. [Электронный ресурс] Режим доступа: https://bool.dev/news/detail/n-kompaniy-kotorye-ushli-s-rossii
3. Технологическая open-source платформа быстрой разработки бизнес-приложений. [Электронный ресурс] Режим доступа: https://www.jmix.ru/
4. Spring Boot, [Электронный ресурс] Режим доступа: https://spring.io/projects/spring-boot

УДК 004.453

Г. В. Мироненков, В. А. Ивлев (2 курс аспирантуры),
Н. В. Воинов, к.т.н., доцент

РЕШЕНИЕ ЗАДАЧИ N ТЕЛ С ИСПОЛЬЗОВАНИЕМ ИМПУЛЬСНОЙ НЕЙРОННОЙ СЕТИ

Задача N тел — это классическая задача в физике, которая связана с предсказанием поведения группы небесных объектов, таких как планеты или звезды, которые взаимодействуют друг с другом посредством гравитации. Несмотря на то, что это хорошо известная проблема, найти общее аналитическое решение задачи N тел чрезвычайно сложно. Поэтому для решения этой задачи широко используются численные методы и методы машинного обучения [1-3].

Целью данного исследования являлось сравнение производительности численных методов и нейронных сетей при решении задачи N тел. Для достижения этой цели необходимо решить следующие задачи:

– Внедрить численные методы и нейронные сети для решения задачи N тел.

– Оценить точность и скорость вычислений данных методов.

– Сравнить их эффективность.

Задача N тел состоит в нахождении позиций и скоростей N небесных объектов, которые взаимодействуют друг с другом посредством гравитации. Существуют два разных подхода к решению этой проблемы: численные методы и нейронные сети. Численные методы включают аппроксимацию задачи N тел путем дискретизации времени и вычисления динамики небесных объектов с использованием численного интегрирования. Одним из наиболее часто используемых численных методов является алгоритм моделирования N тел, который

использует метод Стёрмера для вычисления траекторий объектов. С другой стороны, нейронные сети предлагают альтернативный подход к решению проблемы N тел. В этом подходе обучается нейронная сеть на наборе смоделированных данных N тел, чтобы изучить основную динамику системы. Затем обученную сеть можно использовать для прогнозирования положения и скорости объектов с течением времени.

Производительность метода Стёрмера N тел была сравнена с импульсной нейронной сетью на смоделированном наборе данных N тел. Была оценена точность и скорость вычислений обоих методов на ряде тестовых наборов данных. Результаты показывают, что алгоритм моделирования N тел имеет сопоставимую с нейросетевым подходом точность при более высокой скорости вычислений. Производительность двух методов была проверена на большом наборе от 1000 до 10000 объектов. При проведении опытов была обнаружено, что при росте числа объектов метод Стрёмера начинает проигрывать импульсной нейронной сети в скорости вычисления при сопоставимой точности.

В заключение были сравнены два разных подхода к решению задачи N тел: численные методы и нейронные сети (Рис.1). Обнаружено, что при сопоставимой точности метод Стрёмера выигрывает на малом числе объектов. Однако, при большем их числе импульсная нейронная сеть показывает лучший результат. Полученные данные свидетельствуют о том, что выбор метода решения задачи N тел зависит от конкретной рассматриваемой задачи. Для небольших наборов данных численные методы могут обеспечить наилучшую точность. Однако для больших наборов данных нейронные сети могут предложить более эффективное и точное решение.

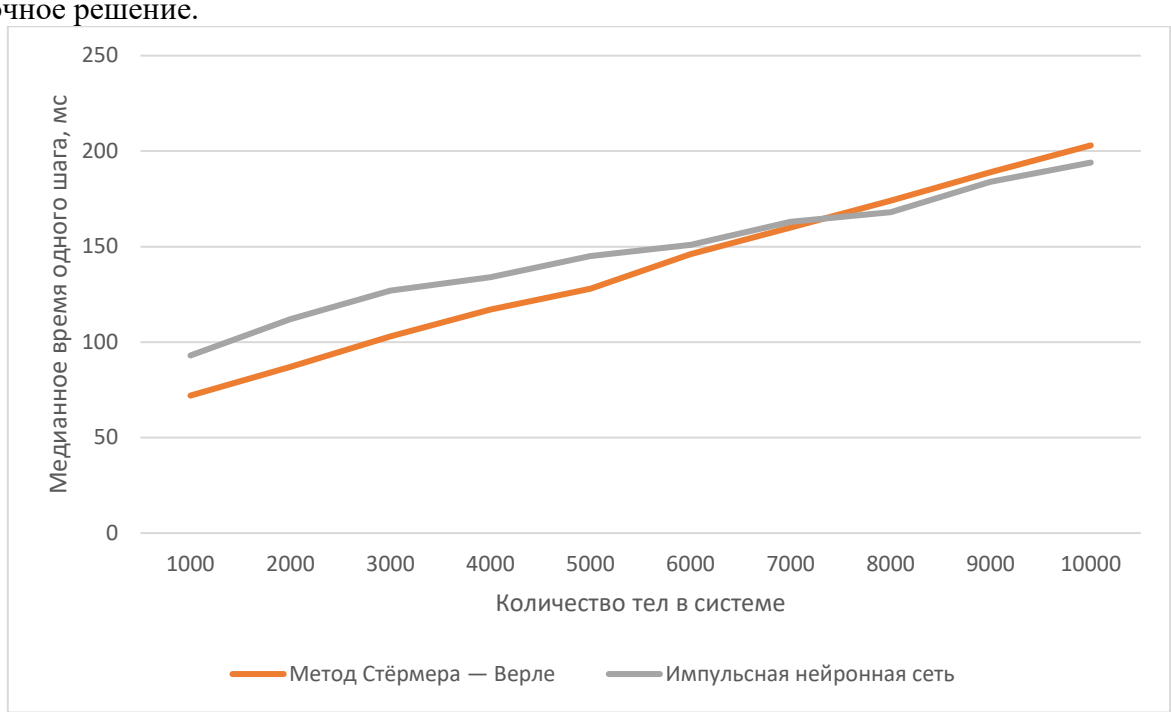


Рисунок 1 – Сравнение времени работы алгоритмов решения задачи N тел

ЛИТЕРАТУРА

1. V. S. Andreev, S. V. Goryainov and A. V. Krasilnikov, "N-body problem solution with composition numerical integration methods," *2016 XIX IEEE International Conference on Soft Computing and Measurements (SCM)*, St. Petersburg, Russia, 2016, pp. 196-198, doi: 10.1109/SCM.2016.7519726.
2. S. Aluru, G. M. Prabhu and J. Gustafson, "Truly distribution-independent algorithms for the N-body problem," Supercomputing '94:Proceedings of the 1994 ACM/IEEE Conference on Supercomputing, Washington, DC, USA, 1994, pp. 420-428, doi: 10.1109/SUPERC.1994.344305.
3. E. Del Sozzo, L. Di Tucci and M. D. Santambrogio, "A highly scalable and efficient parallel design of N-body simulation on FPGA," 2017 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), Lake Buena Vista, FL, USA, 2017, pp. 241-246, doi: 10.1109/IPDPSW.2017.43.

УДК 004.021

Ц. Мэн (4 курс бакалавриата),
И. Г. Черноруцкий, д.т.н, профессор

КЛАССИФИКАТОР БЛОКОВ КОДИРОВАНИЯ
CODING UNIT CLASSIFIER

Машинное обучение является ветвью искусственного интеллекта и в настоящее время широко используется в таких областях, как поиск данных, компьютерное зрение и обработка естественного языка [1]. Его теория сосредоточена на разработке и анализе алгоритмов, которые позволяют компьютерам "обучаться" автоматически. Алгоритмы машинного обучения — это класс алгоритмов, которые автоматически анализируют данные для получения закономерностей и используют эти закономерности для составления прогнозов относительно неизвестных данных. В области разработки алгоритмов теория машинного обучения фокусируется на достижимых и эффективных алгоритмах обучения [2].

Для решения задач классификационного прогнозирования в машинном обучении было предложено множество алгоритмов, дающих хорошие результаты. Примеры включают: kNN, деревья решений, классификаторы логистической регрессии, SVM и т. д [2].

В данной работе представлен новый алгоритм (Coding unit classification algorithm, CUCL) для решения проблемы предсказания классификации в машинном обучении. CUCL вдохновлен блоком дерева кодирования в высокоэффективное кодирование видеоизображений HEVC [3]. Используя идею единиц кодирования в дереве кодирования, мы разработали классификатор блоков кодирования (Coding unit classifier, CUC).

CUC снижает зависимость от производительности компьютера и повышает производительность и точность при обработке наборов данных с вложенными структурами. Конструкция разумно сочетает преимущества kNN и SVM и использует блоки кодирования, генерируемые во время обучения, для снижения вычислительной нагрузки на процессор, при этом CUC имеет хорошую поддержку многопоточности.

Для реализации CUC необходимо выполнить следующую работу.

1. представить обзор алгоритма CUCL и проанализировать его осуществимость
2. реализация классификатора кодированных блоков CUC
3. тестирование и анализ на наборе данных

Классификатор кодовых блоков состоит из двух основных этапов: "сегментация блоков кодирования" и "заражение блоков кодирования", при этом этап сегментации подразделяется на этап "предварительной сегментации" и этап "уточняющей сегментации".

Рисунок 1 – Основные этапы создания CUC

Эти шаги кратко описаны в следующем примере:

1. Предварительная сегментация: принцип предварительной сегментации блока кодирования на этом этапе заключается в том, что если текущий блок содержит частицы с более чем одним уникальным целевым значением, то этот блок кодирования следует продолжать сегментировать на более мелкие блоки до тех пор, пока каждый блок кодирования не будет содержать популяцию частиц с одним уникальным целевым значением.

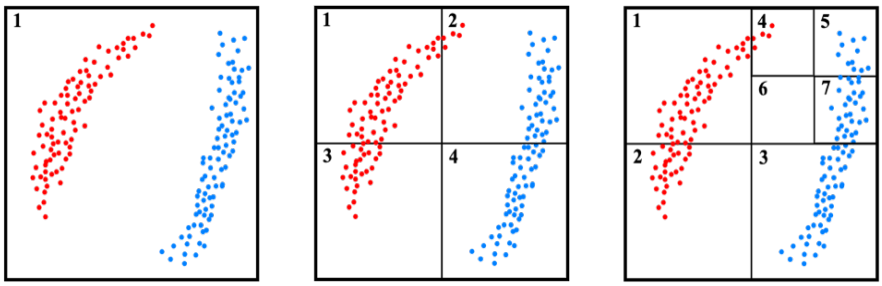


Рисунок 2 –. Этап 1 - Предварительная сегментация

2. Уточняющая сегментация выполняется поверх предварительной сегментации, чтобы повысить точность последующих предсказаний и избежать недооценки (underfitting), вызванной "заражением" слишком больших блоков кодирования на более позднем этапе. Принцип заключается в том, что каждый блок разбивается на части определенное количество раз и, в конце концов, каждый блок, содержащий частицу, помечается как целевое значение, к которому принадлежит его частица. На рисунке ниже показан блок, подвергшийся одному уточнению сегментации.

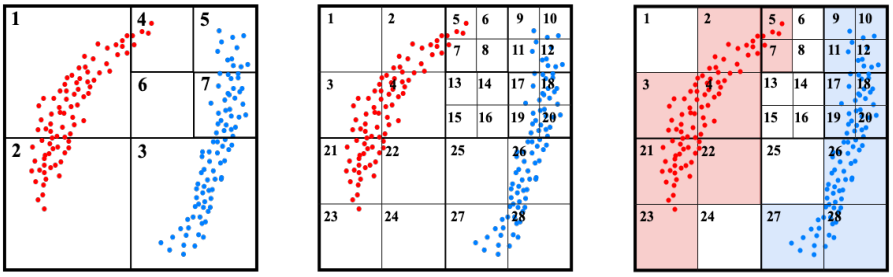


Рисунок 3 – Этап 2 - Уточняющая сегментация

3. Принцип этапа "заражения" основан на SIR-модели [4] передачи вирусов и эпидемиологии [5]. Блок кодирования с наибольшей заражающей способностью должен заразить соседние пустые блоки кодирования, цель заражения - пометить эти пустые блоки

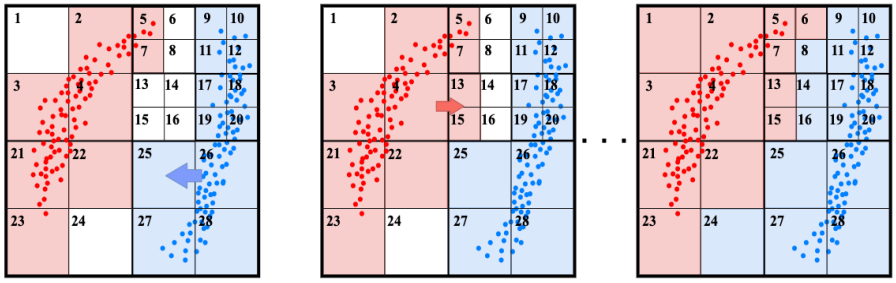


Рисунок 4 – Этап 3 - Заражения

кодирования одним из целевых значений. Этап "заражения" показан на графике ниже.

После создания CUC новая частица во входном классификаторе может быть предсказана как целевое значение на основе типа блока кодирования, к которому она принадлежит.

ЛИТЕРАТУРА

1. Binshan Lin, 2023. Machine Learning with Applications. [Электронный ресурс] Режим доступа: https://www.sciencedirect.com/journal/machine-learning-with-applications
2. Machine learning, 2023. Режим доступа: https://en.wikipedia.org/wiki/Machine_learning
3. Gary J. Sullivan, Woo-Jin Han, Thomas Wiegand, 2012. Overview of the High Efficiency Video Coding (HEVC) Standard, IEEE Transactions on circuits and systems for video technology, VOL.22, NO.12
4. Virus transmission and epidemiology, 2023. Viruses, Understanding to Investigation, Pages 59-71
5. Hao Hu, Karima Nigmatulina, Philip Eckhoff, 2013. The scaling of contact rates with population density for the infectious disease models. Mathematical Biosciences Volume 244, Issue 2, Pages 125-134

УДК 004.942

А. М. Сабуткевич, Д. А. Вихляев (2 курс магистратуры),
И. В. Никифоров, к.т.н., доцент,
А. В. Самочадин, к.т.н., доцент

РЕШЕНИЕ ПРОБЛЕМЫ КОМБИНАТОРНОГО ВЗРЫВА ПРИ ГЕНЕРАЦИИ ТРАЕКТОРИЙ ПОВЕДЕНИЯ АГЕНТОВ В ПРОЦЕССЕ ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ

Проблема комбинаторного взрыва характеризуется резким ростом временной сложности алгоритма при относительно невысоком росте размерности исходных данных [1]. Данная проблема наблюдается в алгоритмах, используемых в различных прикладных областях, в том числе и в области имитационного моделирования [2].

Целью данной работы является рассмотрение подхода к решению проблемы комбинаторного взрыва при генерации траекторий поведения агентов посредством введения критериев валидности траекторий, проверка которых реализуется при помощи верификации *LTL* формул с использованием алгоритма *Model Checking* [3].

В рамках исследования данной темы был рассмотрен ряд ключевых подходов к решению проблемы комбинаторного взрыва:

- повышение производительность за счет распределенных вычислений [1][4];
- введение правил для исключения не валидных вариантов [5];
- использование эвристических функций [6].

В качестве наиболее предпочтительного был выбран поход, основывающийся на введение правил для исключения не валидных вариантов. Это обуславливается тем, что другие подходы сохраняют более сильную зависимость от размерности системы или обладают меньшей точностью [5].

Генерация траекторий поведения агента осуществляется на основе изначально заданного графа возможного поведения агента. Граф возможного поведения агента представляет собой ориентированный математический граф, узлами которого являются состояния агента, а дугами – действия, которые провоцируют смену состояний агента. Каждый из агентов характеризуется набором свойств. Состояние агента – множество допустимых значений свойств агента. Если текущие значения свойств агента удовлетворяют допустимым значениям рассматриваемого состояния, то агент может осуществить переход в данное состояние. Изменение значений свойств агент может происходить как посредством действий, так и на основе событий внешнего мира, нацеленных на актуализацию общего состояния модели в процессе симуляции [7]. Каждая из сформированных траекторий поведения агента сопровождается аудитом свойств агента, которая последовательно описывает изменения значений свойств агента при его перемещении по соответствующей траектории поведения.

В случае, если граф поведения агента содержит состояния, которые на уровне формального представления принадлежат циклической траектории и имеют полустепень исхода вершины [8] большую единицы, то возможно возникновение эффекта комбинаторного взрыва при генерации возможных траекторий поведений. Это обуславливается тем, что при наличии у агента возможности перехода в несколько последующих состояний на уровне модели создаются двойники исходного агента [9], каждый из которых переходит в одно из доступных состояний. Тем самым количество траекторий поведения агента эквивалентно количеству его двойников с учетом исходного агента.

Одним из возможных подходов к исключению эффекта комбинаторного взрыва при генерации траекторий поведения агентов является введение дополнительных критериев корректности генерируемых траекторий поведения относительно соответствующих аудитов свойств. Данный подход позволит своевременно исключать из рассмотрения неудовлетворяющие заданным критериям траектории.

Для реализации описанного подхода может быть использован алгоритм *Model Checking*. Аудит свойств агента, соответствующий проверяемой траектории поведения, может быть представлен в виде структуры Крипке [3], а критерии заданы при помощи формул *LTL*.

Предварительно аудит свойств агента необходимо формально представить в виде ориентированного математического графа, расширенного добавлением петли [8] для конечного представления аудита [10]. С точки зрения реального мира введение петли для конечного представления обозначает, что в будущем значения свойств агента будут неизменными по достижению конечно состояния данной траектории. Каждая вершина полученного графа должна быть помечена соответствующими свойствами и их значениями. Результирующий граф может быть представлен структурой Крипке, в которой состояния и переходы между ними однозначно сопоставляются вершинам и дугам сформированного графа соответственно, а атомарными высказываниями выступают значения свойств для каждого из состояний.

Критерии, используемые для исключения неудовлетворяющих траекторий поведения, описываются с использованием формул *LTL*, которые задаются относительно значений свойств агентов.

Для прикладного применения, описанного выше подхода может быть использован верификатор *Spin* [10]. Аудит свойств агента может быть ретранслирован в модель на языке *Promela* [10], которая будет состоять из одного процесса, в рамках которого последовательно будут сменяться значения свойств агента в соответствии с аудитом. Установка новых значений свойств из очередного представления аудита должна происходить с использованием блока *atomic*. Операции по установке значений свойств, соответствующих конечному представлению аудита, должны повторяться в бесконечном цикле.

Описанный подход должен позволить исключить ошибку комбинаторного взрыва при генерации траекторий поведения агентов в процессе имитационного моделирования. Однако важным условием при этом является корректность выбора и задания критерия, нацеленного на исключение неудовлетворяющих ему траекторий.

ЛИТЕРАТУРА

1. Шипов А. А. Метод распределенного анализа свойств верификации моделей / А. А. Шипов // Программные продукты и системы / РГСУ. – Москва, 2016. – №2 (114). – С. 53–61.
2. Тюгашев А. А. Логическое исчисление управляющих алгоритмов / А. А. Тюгашев, А. Ю. Богатов // НиКа. / СГАУ. – Самара, 2013.
3. Карпов Ю. Г. Model Checking. Верификация параллельных и распределенных программных систем. СПб: БХВ Петербург, 2010. 552 с.
4. Allal L. Towards Distributed Solution to the State Explosion Problem / L. Allal, G. Belalem, P. Dhaussy.—India, 2016.
5. Carvalho M. A solution to the state space explosion problem in declarative business process modeling / R. M. de Carvalho, N. C. Silva, C. A. L. Oliveira, R. M. F. Lima // Proceedings of the International Conference on Software Engineering and Knowledge Engineering, SEKE. 2013. – C. 26-29.
6. Никифоров И. В., Петров А. В., Котляров В. П. Статический метод отладки тестовых сценариев, сгенерированных с использованием эвристик // Научно-технические ведомости Санкт-Петербургского государственного политехнического университета. Информатика. Телекоммуникации. Управление. – 2012. – № 4(152). – С. 114-119.
7. Городецкий В. И. Принципы автономного группового управления // Интегрированные модели и мягкие вычисления в искусственном интеллекте. Сборник научных трудов X-й Международной научно-технической конференции (ИММВ-2021, Коломна, 17–20 мая 2021 года). В 2-х томах. Т1. – Смоленск: Универсум, 2021. – 449 с.
8. Никифоров И. В., Петров А. В., Юсупов Ю. В., Котляров В. П. Применение методик формализации для построения верификационных моделей систем по UCM-спецификациям // Научно-технические ведомости Санкт-Петербургского государственного политехнического университета. Информатика. Телекоммуникации. Управление. – 2011. – № 3(126). – С. 180-184.

9. Городецкий В. И. Сети автономных агентов реального времени в среде с противодействи-ем: особенности и компоненты модели / В. И. Городецкий, М. Г. Пантелеев // Материалы конференции «Информационные технологии в управлении» (Санкт-Петербург, 06–08 ок-тября 2020 года) – Санкт-Петербург, 2020. – С.22–31.
10. Карпов Ю. Г. Верификация распределенных систем: учеб. пособие / Ю. Г. Карпов, И. В. Шошмина.—СПб. : Изд-во Политехн. ун-та, 2011.—212 с.

УДК 519.6

А. В. Назаренко (4 курс бакалавриата),
И. Г. Черноруцкий, д.т.н., профессор

ИССЛЕДОВАНИЕ АЛГОРИТМА БАКТЕРИАЛЬНОЙ ОПТИМИЗАЦИИ

Цель работы заключается в рассмотрении алгоритма бактериальной оптимизации, который позволяет находить глобальный оптимум и предназначен для решения задач непрерывной оптимизации.

На создание бактериального алгоритма К. М. Пассино вдохновил способ поиска пищи двумя бактериями: кишечной палочкой и сальмонеллой. Для передвижения в среде они используют вращающиеся жгутики. В случае, если жгутики вращаются против часовой стрелки, то бактерия двигается вперёд. Иначе, движение происходит в случайном направлении. Комбинируя случайное движение и поступательное, бактерии ищут пищу наиболее эффективным образом.

Рассмотрим алгоритм бактериальной оптимизации. Он состоит из трёх этапов: хемотаксиса, размножения и элиминации-рассеивания. Начальная популяция генерируется случайным образом.

На этапе *хемотаксиса* определяется направление движения бактерии. Если её перемещение сопровождается улучшением значения целевого функционала, то движение вперёд продолжится. Такой процесс называется плаванием. В противном случае вектор движения меняется, т. е. происходит кувырок. Положение бактерии на каждой стадии хемотаксиса обновляется в соответствии с формулами:

$$\theta^i(j+1,k,l) = \theta^i(j,k,l) + C(i) * \varphi(i), \qquad (1)$$

$$\varphi(i) = \frac{V_i}{\|V_i\|_E}, \qquad (2)$$

где θ^i(j,k,l) - положение i-й бактерии в популяции, а j, k и l - индексы стадии хемотаксиса, стадии размножения и элиминации-рассеивания соответственно. C(i) - размер шага хемотаксиса, V_i – текущий направляющий вектор шага хемотаксиса бактерии. При плавании направляющий вектор остаётся неизменным, а при кувырке состоит из случайных значений, определённых в интервале [-1;1].

После хемотаксиса начинается фаза *размножения*: бактерии сортируются по состоянию здоровья (сумме значений целевого функционала во всех точках её траектории движения от первой до текущей t-й итерации):

$$h_i = \sum_{\tau=1}^{t} F_i(\tau), \qquad (3)$$

Затем половина бактерий с наихудшими показателями удаляются, а каждая из оставшихся дублируется. За счёт этого численность популяции бактерий не меняется. Благодаря размножению область поиска сужается, что приводит к ускорению сходимости алгоритма.

Наконец, наступает стадия *элиминации-рассеивания*, когда с некоторой заданной вероятностью выбираются и уничтожаются n бактерий. Вместо уничтоженных случайным образом создаются новые бактерии с таким же порядковым номером. Т. е. на данном этапе происходит перенос бактерий, который является способом покинуть локальные оптимальные

точки. Благодаря этому снижается вероятность стагнации (зацикливания в локальном оптимуме).

На Рисунке 1 представлена блок-схема описанного алгоритма.

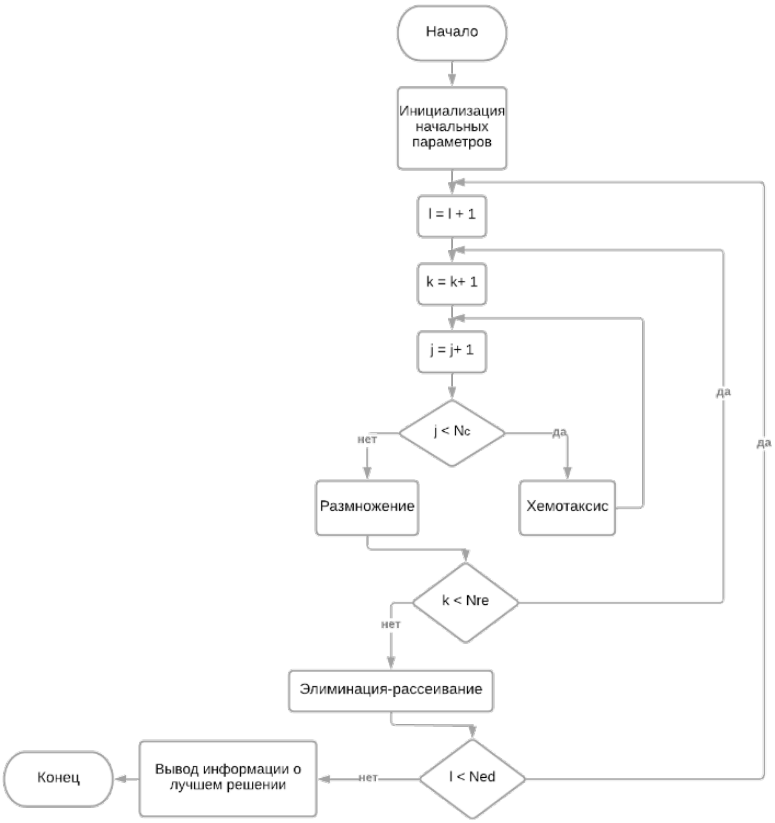


Рисунок 1 – Блок-схема алгоритма

Алгоритм бактериальной оптимизации реализован на языке C++ в среде выполнения Visual Studio 2019. Тестирование проводилось на невыпуклых функциях Растригина и Розенброка.

Результат работы программы для функции Растригина изображен на Рисунке 2:

```
Best solution BFO: 6.93295e-05 pos: (0.000579806 ; -0.000115248)
77
```

Рисунок 2 – Результат работы алгоритма для функции Растригина

Полученное значение глобального минимума достаточно близкое к действительному. При этом время выполнения программы составляет всего 77 мс.

Результат работы программы для функции Розенброка изображен на Рисунке 3:

```
Best solution BFO: 1.17182e-05 pos: (1.00262 ; 1.00546)
100
```

Рисунок 3 – Результат работы алгоритма для функции Розенброка

Глобальный минимум вновь найден верно, но немного увеличилось время выполнения(100 мс).

Вывод. Реализованный алгоритм бактериальной оптимизации позволяет получить достаточно точные значения оптимумов, погрешность при этом минимальна. Также, стоит отметить, что время выполнения программы небольшое. Значит, алгоритм является эффективным и может применяться для оптимизации невыпуклых функций.

ЛИТЕРАТУРА

1. Rahkar Farshi, T., Orujpour, M. A (2021) multi-modal bacterial foraging optimization algorithm. J Ambient Intell Human Comput 12, 10035–10049.

2. Chen H, Zhang Q, Luo J, Xu Y, Zhang X (2020) An enhanced bacterial foraging optimization and its application for training kernel extreme learning machine. Appl Soft Comput 86:105884.
3. Zhang Q, Wang R, Yang J, Ding K, Li Y, Hu J (2017) Collective decision optimization algo-rithm: a new heuristic optimization method. Neurocomputing 221:123–137.
4. Xing, T., Wan, M., Wen, S., Chen, L., Wang, H. (2021). An Improved Bacterial Foraging Optimization for Global Optimization. In: Tan, Y., Shi, Y., Zomaya, A., Yan, H., Cai, J. (eds) Data Mining and Big Data. DMBD 2021. Communications in Computer and Information Science, vol 1454. Springer, Singapore.
5. Passino KM (2002) Biomimicry of bacterial foraging for distributed optimization and control. IEEE Control Syst Mag 22:52–67.

УДК 004.6

Д. Е. Печенев (2 курс бакалавриата),
Я. А. Кириленко, ст. преподаватель

ФРЕЙМВОРК ДЛЯ СБОРА И АНАЛИЗА ДАННЫХ ОБ ИСПОЛЬЗОВАНИИ МАШИННЫХ ИНСТРУКЦИЙ

При миграции программного обеспечения на процессоры новой открытой архитектуры RISC-V остро стоит вопрос оптимизации программ именно под эту архитектуру. Разработчикам компиляторов или специалистам, оптимизирующим отдельные участки машинного кода нетривиальным образом вручную, было бы полезно понимать, в каких приложениях и утилитах в популярных дистрибутивах GNU/Linux на платформе x86-64 используются, например, векторные расширения или инструкции для ускорения шифрования. Эти знания помогли бы понять, как можно улучшить компилятор или в каких программах есть участки машинного кода, которые необходимо оптимизировать вручную для архитектуры RISC-V.

С такой задачей столкнулся коллектив Лаборатории технологий программирования инфраструктурных решений СПбГУ. Однако это далеко не единственный случай, когда был бы полезен статистический анализ данных о появлении различных инструкций (или групп инструкций) в машинном коде программ. Другим примером является ситуация, когда разработчику компилятора необходимо узнать, как в целом поменялся сгенерированный машинный код программ после изменений в компиляторе. Эта методика, в частности, планируется к использованию для оценки качества оптимизации прошивок встраиваемых систем, например, маршрутизаторов и хранилищ данных.

С целью быстрого нахождения ответов на такого рода вопросы создается фреймворк, автоматизирующий сбор данных об использовании машинных инструкций и предоставляющий инструментарий для их статистического анализа и визуализации. Ему и посвящена эта работа. Код фреймворка открытый и выложен в репозитории на GitHub [1].

С одной стороны, фреймворк предоставляет возможность автоматизированного и единовременного сбора данных с разных дистрибутивов GNU/Linux и является легко расширяемым — добавить новый дистрибутив не составит труда. С другой — он содержит в себе широкий инструментарий для анализа и визуализации полученных данных, который, расширяя стандартные библиотечные функции для работы с таблицами, позволяет в несколько строк кода просматривать статистику частоты появления инструкций, строить интерактивные гистограммы, отвечать на такие вопросы, как «используют ли конкретные пакеты в дистрибутивах Manjaro или Ubuntu векторные инструкции?», узнавать самые популярные и редкие инструкции и многое другое. Отметим, что была также добавлена функциональность разделения инструкций на категории и группы, что значительно повышает информативность и ясность анализа данных.

Сбор данных выполняется для различных дистрибутивов GNU/Linux вне зависимости от того, какая операционная система установлена на машине, производящей запуск. Для этого соответствующий скрипт запускается в Docker-контейнерах дистрибутивов. Такой подход позволяет достигнуть требуемой расширяемости.

Локальный сбор данных может занимать весьма продолжительное время. Кроме того, разделение сбора и анализа данных во времени гораздо удобнее для использования, поскольку заранее подготовленный набор данных может быть независимо использован для разных подзадач. Для этого сбор данных запускается автоматически на серверах. На данный момент использована возможность запуска через GitHub Actions. Собранные данные выгружаются во временное хранилище в облаке, откуда могут быть взяты для статистического анализа.

Для сбора данных был написал Python-скрипт, который предоставляет множество возможностей для конфигурации под нужды конкретных пользователей. Главной его функциональностью является подсчет количества инструкций во всех программах в системе. Скрипт проходит по всем файлам, содержащимся в определенной директории (мы будем начинать с корневой папки) и ее поддиректориях и пробует получить ассемблерный листинг файла. Если попытка удачна, то есть файл содержит код, путь до файла и количество всех инструкций, встречающихся в нем, записываются в csv-таблицу, которая и является результатом работы скрипта.

Для упрощения типовых задач анализа данных реализована библиотека предметно-ориентированных вспомогательных функций, которая позволяет, например, в интерактивной среде Jupyter Notebook в несколько строк кода получать ответы на вопросы предметной области. Пример такого анализа с демонстрацией некоторых возможностей инструмента представлен в репозитории. Благодаря тщательному документированию всех функций анализа и визуализации разобраться в них не составит проблем.

Одна из самых значительных трудностей, с которой приходится сталкиваться при анализе использования машинных инструкций — это их большое количество. Так, многие инструкции, являясь различными, делают принципиально одно и то же действие. Эта проблема является нетривиальной, поскольку множество источников, такие как [2], предлагают описание инструкций без какого-либо деления их на более крупные единицы. Другие же (например, [3]), предлагая разделение инструкций на группы, не включают некоторые важные расширения.

Для решения обозначенной выше проблемы был написан Python-скрипт, собирающий информацию с сайта [4], содержащего достаточное количество инструкций. Мы будем называть категорией инструкции тот раздел сайта слева, куда она включена, а группой — ее подраздел в нем. Таким образом, скрипт собирает для каждой инструкции ее описание, категорию и группу и сохраняет результат в json-файле.

На данный момент представленный фреймворк позволяет в несколько строк кода воспроизвести основные результаты статьи [5], в которой рассматривается статистика использования инструкций в C/C++ приложениях в дистрибутиве Ubuntu 16.04 на платформе x86-64, и отличается от него следующими преимуществами.

1. Возможность обновлять данные (в то время как результаты статьи не обновлялись уже практически три года).
2. Возможность легко варьировать приложения для анализа, конфигурируя сборку образов дистрибутивов.
3. Возможность собрать статистику с произвольного дистрибутива GNU/Linux.
4. Двухуровневая кластеризация инструкций.
5. Возможность гибкого изменения фильтров данных и средств визуализации.

Говоря о перспективах проекта, отметим, что в самое ближайшее время планируется добавить возможность сбора данных на процессорах других архитектур, таких как, например, RISC-V.

ЛИТЕРАТУРА

1. GitHub-репозиторий [Электронный ресурс]. Режим доступа: https://github.com/Danila-Pechenev/InstructionAnalysis. [Дата обращения: 20.03.2023].
2. x86 and amd64 instruction reference [Электронный ресурс]. Режим доступа: https://www.felixcloutier.com/x86/. [Дата обращения: 20.03.2023].
3. X86 Opcode and Instruction Reference 1.12 [Электронный ресурс]. Режим доступа: http://ref.x86asm.net/geek.html. [Дата обращения: 20.03.2023].
4. x86-64 Instructions Set [Электронный ресурс]. Режим доступа: https://linasm.sourceforge.net/docs/instructions/index.php. [Дата обращения: 20.03.2023].
5. Amogh Akshintala, Bhushan Jain, Chia-Che Tsai, Michael Ferdman and Donald Porter. x86-64 Instruction Usage among C/C++ Applications // The 12th ACM International Conference. – 2019. – P. 68-79. – DOI 10.1145/3319647.3325833.

УДК 004.023

К. А. Пятышев (4 курс бакалавриата),
И.Г. Черноруцкий, д.т.н., профессор

ГИБРИДНЫЙ МЕТОД СОВМЕСТНЫЙ С ПОЛЕТАМИ ЛЕВИ ДЛЯ ПЛАНИРОВАНИЯ ЗАРЯДКИ ЭЛЕКТРОМОБИЛЕЙ

Электротранспортные средства (ЭС) стали популярными благодаря своей экологичности и экономичности. Таким образом, проблема планирования зарядки электромобилей (ПЗЭМ), которая пытается определить график зарядки для каждого электромобиля и является NP-трудной, стала серьезной исследовательской проблемой. Поскольку задача ПЗЭМ является NP-трудной, не существует известных точных алгоритмов для нахождения оптимальных решений за полиномиальное время.

Целью работы является планирование зарядки электромобилей (ПЗЭМ) с помощью гибридного алгоритма оптимизации, который сочетает в себе алгоритмы роя частиц (PSO) и Светлячка (FFA), а также стратегию полетов Леви.

Данный метод реализован путем интеграции трех компонентов:

– различные возможности разведки и эксплуатации PSO и FFA
– сотрудничество между ними в форме общих глобальных оптимумов;
– использование стратегии поиска полетов Леви для обхода локальных минимумов.

В этой работе также исследуется масштабируемость различных решений ПЗЭМ. Кроме того, в этой работе были оценены различные варианты гибридных методов PSO и FFA, чтобы найти наиболее эффективный гибридный вариант. Также сравним алгоритмы в трех аспектах: качество решения средних решений в двух транспортных сетях; кривые сходимости алгоритмов; и преимущество расширенного подхода с точки зрения сходимости и качества.

В процессе работы алгоритма происходит следующее:

– сначала происходит этап инициализации, на котором генерируется случайная совокупность, т. е. набор решений или графиков зарядки, состоящий из набора решений FFA и другого набора решений PSO;

– после получения начальных решений для алгоритма выполняется оценка этих решений с учетом ограничений для проверки нарушений ограничений;

– далее исходные решения сортируются по показателям приспособленности для получения личных лучших решений FFA и PSO;

– затем решения ранжируются с использованием оценок пригодности, и получается глобальное лучшее решение. После g_{max} числа итераций глобальный лучший решение возвращается как глобальный лучший результате алгоритм заканчивает свою работу.

В качестве средств для реализации алгоритма и его последующего тестирования был выбран язык программирования MATLAB и среда разработки MATLAB R2021b.

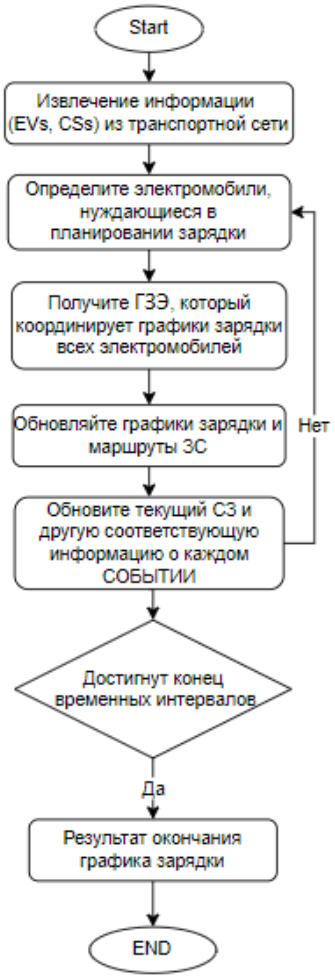


Рисунок 1 – Структура управления системой ПЗЭМ

В качестве проверки работоспособности алгоритма было выбрано две транспортные сети. Первая представляет собой синтетическую сеть из 24 вершин и 76 ребер, представляющую Су-Фолс, Южная Дакота, США. Вторая — реальная сеть, представляющая Гуанчжоу, Китай. Была произведена оценка производительности алгоритма в трех аспектах: качество решения средних решений в двух транспортных сетях; кривые сходимости алгоритмов; и преимущество расширенного подхода с точки зрения сходимости и качества. Также алгоритм сравнен с другими эволюционными алгоритмами с точки зрения масштабируемости и времени выполнения.

Гибридный Алгоритм показывает достойные результаты и производительность, о чем сообщают нам результаты сравнения. Мы определили лучшие характеристики сходимости алгоритма. Кроме того, стратегия поиска рейсов Леви в HCL помогает избежать локальных минимумов, что помогает данному алгоритму иметь большую эффективность относительно других алгоритмов.

ЛИТЕРАТУРА

1. Arun Kumar K, Shrisha Rao // A Hybrid Cooperative Method With Lévy Flights for Electric Vehicle Charge Scheduling // 2022 // URL: https://www.researchgate.net/publication/355874238_A_Hybrid_Cooperative_Method_With_Levy_Flights_for_Electric_Vehicle_Charge_Schedulingity_allocation_Hybrid_quantum-classical_optimization // Дата обращения: 25.11.2022
2. И. Г. Черноруцкий, Н. В. Воинов, Л. П. Котлярова // Методы оптимизации. Учебное пособие // Санкт-Петербург 2020 // Дата обращения: 29.11.2022

УДК 004.021

И. Н. Уткин (4 курс, бакалавр),
Ю. В. Литвинов, к. т. н., доцент,
М. С. Осечкина, инженер-программист

ДЕРЕВЬЯ РЕШЕНИЙ ДЛЯ НАСТРОЙКИ ГИПЕРПАРАМЕТРОВ АЛГОРИТМОВ ADAS

Усовершенствованная система помощи водителю (Advanced Driver Assistance System, ADAS) – класс систем дополненной реальности для помощи водителю на дороге. ADAS использует сенсоры для моделирования окружающего мира, в частности камеры и лидары, чтобы избежать столкновения, предупреждать водителя об опасностях или даже взять контроль над транспортным средством. Для обработки сигналов с датчиков ADAS использует различные алгоритмы и, так как ситуация на дороге может меняться очень часто, то используемые алгоритмы должны работать быстро даже на сравнительно слабых устройствах. Поэтому в ADAS, наряду с современными и сложными нейронными сетями, до сих пор используются классические методы без обучения [1].

Одним из таких алгоритмов является Canny [2]. Он позволяет находить контуры объектов на фотографии, что, например, помогает ориентироваться в дорожной разметке. В классической реализации Canny имеет следующие гиперепараметры: два пороговых значения для удаления шумных линий и степень сглаживания. К сожалению, результат алгоритма сильно зависит от этих параметров. Так, например, при малых пороговых значениях выделится много “шумных” контуров, которые будут затруднять распознавание, а при высоких порогах могут не выделиться контуры, необходимые для разработчика. Часто такие гиперпараметры подбираются вручную, однако это долгий и неточный процесс, так как оптимальные гиперпараметры будут сильно зависеть от конкретной дорожной сцены. Поэтому имеется потребность в методе, который смог бы подбирать оптимальные параметры автоматически в зависимости от признаков изображения.

Также, помимо точности в ADAS ценится объяснимость каждого решения системы. В качестве объяснимого метода для предсказания гиперпараметров может выступить дерево решений – структура данных, которую можно представить как дерево условных операторов. При получении вектора чисел в дереве происходит последовательность покомпонентных сравнений и выдается ответ.

Таким образом целью данной работы является изучение модели дерева решений в качестве предсказателя гиперпараметров для алгоритмов ADAS.

Для достижения этой цели были поставлены следующие задачи:

1. Провести обзор существующих методов для подбора гиперпараметров.
2. Предложить несколько моделей-оптимизаторов на основе деревьев решений.
3. Протестировать предложенные модели.

Системы, которые подбирают гиперпараметры для других алгоритмов, уже давно существуют. Однако они либо слишком простые и не обобщаются на новые дорожные сцены (такие как случайный поиск и байесовская оптимизация), либо предназначены только для нейросетевых подходов [3], и не позволяют конвертировать модель в дерево решений.

Вернемся к примеру, с алгоритмом Canny. Для обучения дерева-предсказателя необходим набор признаков изображений (например, гистограммы серого) и набор оптимальных гиперпараметров. Найти примеры дорожных сцен не трудно, однако найти оптимальные параметры алгоритма Canny в открытом доступе довольно сложно, а для произвольного или собственного алгоритма это будет невозможно. Чтобы найти оптимальные гиперпараметры можно поставить следующую задачу оптимизации: для каждого изображения надо найти такой набор гиперпараметров, с которым алгоритм Canny выделит наиболее верные по некоторой метрике контуры. В такой постановке больше не нужен набор оптимальных гиперпараметров, а только набор признаков изображения, правильно подобранные контуры, метрика для оценки и метод для оптимизации.

В рамках данной работы в качестве изображений и контуров были взяты наборы данных CityScapes [4] и BDD100k [5], а в качестве метрики – Normalized Figure of Merit [6].

В качестве метода оптимизации для максимизации вышеописанной функции из вектора гиперпараметров в некоторое значение метрики были рассмотрены два метода: генетический алгоритм и алгоритм машинного обучения с подкреплением REINFORCE [7]. Сравнение методов на разных наборах данных представлено на Табл. 1.

Таблица 1 – Результаты эксперимента.

Датасет \ Метод оптимизации	Константные параметры	Генетический алгоритм	REINFORCE
BDD10k	46.89	48.72	**49.19**
CityScapes	43.65	58.31	**60.35**

Как можно заметить, методы разметки предоставили хорошие данные и дерево решений смогло обобщить их на новую выборку данных, которую она еще не видела. Лучше всего получилось обобщить метод REINFORCE.

Описанный выше метод можно обобщить на любой алгоритм нахождения контуров на изображении без замены данных и метрики. При обобщении на алгоритмы вне поля компьютерного зрения достаточно лишь заменить наборы данных и метрику для оценивания, так как методы оптимизаций и дерево решений абстрактны от типа алгоритма.

ЛИТЕРАТУРА

1. He L. Research on lane detection algorithm of urban road //Sixth International Conference on Traffic Engineering and Transportation System (ICTETS 2022). – SPIE, 2023. – Т. 12591. – С. 752-757.
2. Canny J. A computational approach to edge detection //IEEE Transactions on pattern analysis and machine intelligence. – 1986. – №. 6. – С. 679-698.
3. Chen Y. et al. Learning to learn without gradient descent by gradient descent //International Conference on Machine Learning. – PMLR, 2017. – С. 748-756.
4. Cordts M. et al. The cityscapes dataset //CVPR Workshop on the Future of Datasets in Vision. – sn, 2015. – Т. 2.
5. Yu F. et al. Bdd100k: A diverse driving dataset for heterogeneous multitask learning //Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. – 2020. – С. 2636-2645.
6. Magnier B. Edge detection evaluation: A new normalized figure of merit //ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). – IEEE, 2019. – С. 2407-2411.
7. Williams R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning //Reinforcement learning. – 1992. – С. 5-32.

УДК 004.023

Е.Р. Селищев (4 курс бакалавриата),
И.Г. Черноруцкий, д.т.н., профессор

ИССЛЕДОВАНИЕ АЛГОРИТМА РАСПРЕДЕЛЕНИЯ ТРАНСПОРТНЫХ СРЕДСТВ: ГИБРИДНАЯ КВАНТОВО-КЛАССИЧЕСКАЯ ОПТИМИЗАЦИЯ

Целью работы является исследование алгоритма оптимизации, основанного на анализе транспортных систем, который называется квантово-гибридным решателем.

В растущем обществе системы общественного транспорта широко используются для обслуживания больших групп населения с учетом пространственно-временного спроса на

поездки. Чем успешнее система общественного транспорта, тем больше положительное влияние на население, которое ею пользуется. Успех транспортной системы требует баланса между доступностью и эффективностью системы. Исторически у некоторых транспортных систем были проблемы с достижением этого баланса.

Доступность относится к тому, насколько легко узлы спроса (например, городские районы) могут получить доступ к транзитным объектам. Как правило, чем ближе объект к узлу спроса, тем более доступным он становится. Таким образом, большее количество объектов вдоль данного маршрута создаст большую доступность для транзитных пассажиров. Эффективность определяется тем, какое расстояние транзитный маршрут может преодолеть в конкретном временном окне. Эффективность транзитного маршрута зависит от нескольких факторов, таких как количество объектов на маршруте, транспортный поток, время пребывания на объектах и количество пассажиров, садящихся и выходящих на каждом транзитном объекте. В идеале транзитный маршрут очень эффективен, чтобы удовлетворить пассажиров и сократить расходы во времени в пути. Однако предпочтительнее найти баланс между доступностью и эффективностью.

В процессе работы алгоритма происходит следующее:

Гибридный решатель пытается решить всю проблему сразу, одновременно решая несколько дочерних подзадач исходной задачи. Объединяются классические эвристические методы табу-поиска и имитации отжига с квантовым отжигом в единую систему. И табу-поиск, и имитация отжига работают со всей проблемой, тогда как квантовый отжиг работает с несколькими дочерними подзадачами меньшего размера, где оптимизируются только переменные воздействия высокой энергии.

Можно задаться вопросом, почему бы не положиться полностью на квантовые вычисления, которые могут выполнять как восхождение, так и туннелирование. Ответ довольно нюансный. Нынешнее поколение квантового оборудования страдает от шума и ошибок, возникающих во время квантовых вычислений. Без исправления ошибок их накопление может значительно ухудшить качество результатов оптимизации и даже исказить квантовую реализацию задачи. Поэтому полагаться исключительно на квантовые вычисления все же не совсем осуществимо. С другой стороны, гибридный подход берет лучшее из обоих миров, где параллельные ветви выполнения создают необходимую избыточность. Если одна система выйдет из строя из-за ошибок или попадет в ловушку, другая ветвь вступит во владение и предоставит свое решение для следующей итерации.

Реализация алгоритма предоставлена на языке Python 3.7 с использование фрэймворка dwave-hybrid с открытым исходным кодом. Упрощенный пример итерации алгоритма показан на Рисунке 1:

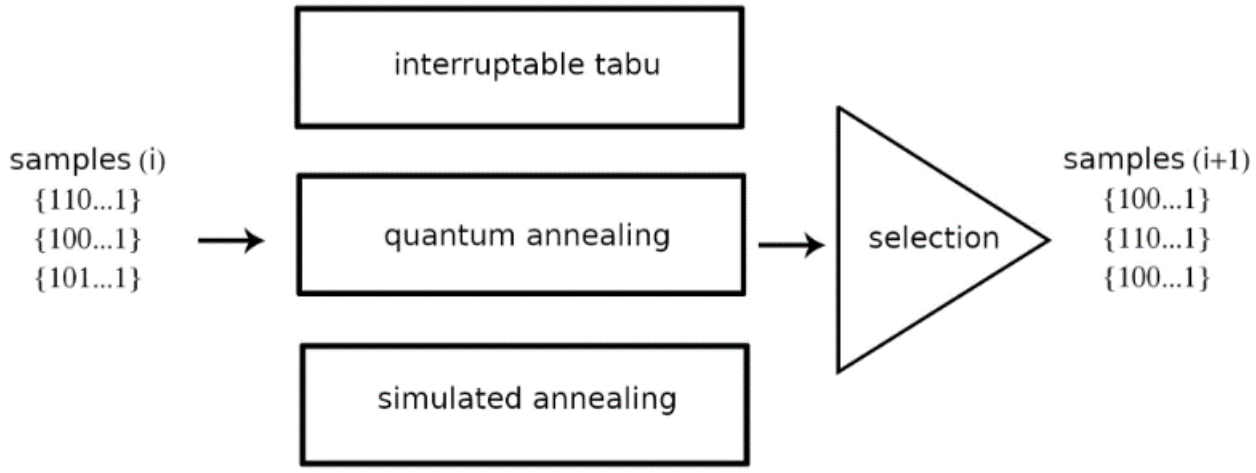


Рисунок 1 – Упрощенная схема единого итерационного цикла трех ветвей решателей.

В качестве объекта тестирования рассматривалась транспортная система одного из самых густонаселенных городов Британской Колумбии, Ванкувера.

Алгоритм показывает достойные результаты, следовательно, он может быть использован в будущем для решения нескольких видов задач оптимизации в реальном мире, а также возможна модернизация алгоритма.

ЛИТЕРАТУРА

1. Einar Gabbassov // Transit facility allocation: Hybrid quantum-classical optimization // 2022 // URL: https://www.researchgate.net/publication/363521591_Transit_facility_allocation_Hybrid_quantum-classical_optimization // Дата обращения: 25.11.2022
2. И. Г. Черноруцкий, Н. В. Воинов, Л. П. Котлярова // Методы оптимизации. Учебное пособие // Санкт-Петербург 2020 // Дата обращения: 26.11.2022
3. D-Wave. D-Wave Hybrid // 2022 // URL: https://github.com/dwavesystems/dwave-hybrid // Дата обращения: 25.12.2022

УДК 004.021

Е.О. Симонова (4 курс бакалавриата),
И.Г.Черноруцкий, д.т.н., профессор

ИССЛЕДОВАНИЕ НОВОГО АЛГОРИТМА ЭВОЛЮЦИОННОЙ ОПТИМИЗАЦИИ, ОСНОВАННОГО НА ЖИЗНЕННОМ ЦИКЛЕ РЕПЛИЦКАЦИИ КОРОНАВИРУСНОЙ БОЛЕЗНИ

Целью работы является исследование нового алгоритма эволюционной оптимизации, основанный на жизненном цикле репликации коронавирусной болезни.

Исследование имитирует механизм коронавируса при захвате человеческих клеток. Алгоритм основан на технологии смены кадров, используемой коронавирусом для репликации. Вирус использует сдвиг кадра для создания большего количества своих копий, что приводит к крупномасштабным изменениям длины полипептида и химического состава. Применение метода сдвига кадров в предлагаемом алгоритме помогает обновлять решения таким образом, чтобы решения в каждом поколении не становились слишком похожими, что позволило бы алгоритму сходиться к глобальному минимуму.

Процесс работы алгоритма можно кратко представить в следующих шагах:

– Популяция решений инициализируется случайным образом, стоимость рассчитывается для каждого решения. После чего решения упорядочиваются по возрастанию в соответствии с функцией пригодности, и первое решение считается лучшим.

– Фаза репликации вируса с помощью метода сдвига рамки. Для каждого решения в популяции родительский элемент выбирается с помощью выбора колеса рулетки, затем метод сдвига рамки применяется для получения нескольких белков из выбранного исходного материала соответствующим образом.

– Оператор мутации применяется к решению, созданному на предыдущем шаге, для того чтобы сгенерировать новое измененное решение.

– Целевая функция оценивается для нового решения, и совокупность обновляется для следующей генерации (решения с наибольшей пригодностью остаются, а остальные удаляются).

– Происходит повтор предыдущих шагов для новой совокупности до тех пор, пока не будут достигнуты выбранные критерии остановки работы алгоритма.

В качестве средств для реализации алгоритма был выбран язык программирования MATLAB и среда разработки MATLAB R2021b. Алгоритм был реализован в соответствии с блок-схемой:

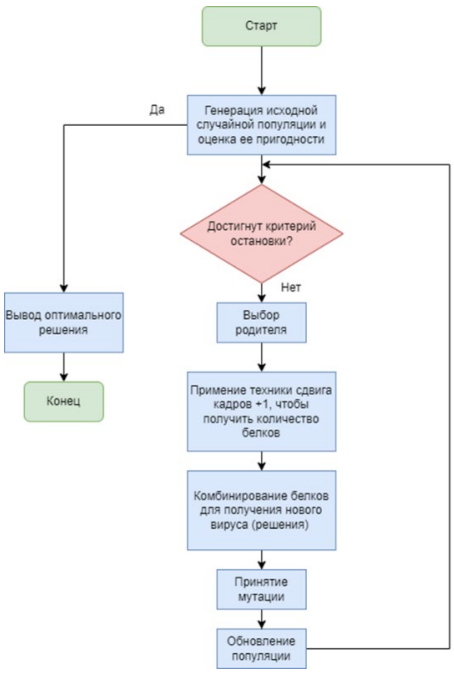


Рисунок 1 – блок-схема алгоритма эволюционной оптимизации, основанный на жизненном цикле репликации коронавирусной болезни.

В качестве проверки работоспособности алгоритма и его тестирования были выбраны несколько реальных тестовых функций CEC, а также несколько различных сценариев использования путем изменения входных параметров.

В ходе тестирования алгоритм показал достойные результаты, значит, он может быть использован для решения нескольких видов задач оптимизации и в будущем может быть внедрен в решение крупномасштабных проблем в различных областях.

ЛИТЕРАТУРА

1. Kaveh A, Talatahari S (2010) A novel heuristic optimization method: charged system search. Acta Mech 213(3):267–289
2. Hsiao YT, Chuang CL, Jiang JA, Chien CC (2005) A novel optimization algorithm: space gravitational optimization. In: 2005 IEEE international conference on systems, man and cybernetics, vol 3, pp 2323–2328
3. Тестовые функции для оптимизации. [Электронный ресурс] Википедия. URL: https://ru.wikipedia.org/wiki/Тестовые_функции_для_оптимизации (дата обращения 20.11.2022)
4. Holland JH (1992) Genetic algorithms. Sci Am 267:66–72
5. И. Г. Черноруцкий, Н. В. Воинов, Л. П. Котлярова // Методы оптимизации. Учебное пособие // Санкт-Петербург 2020 // Дата обращения: 21.10.2022

УДК 004.657

И. А. Шаповалова (2 курс магистратуры),
Т. В. Леонтьева, к.т.н., доцент,
О. В. Прокофьев, ст. преподаватель

РЕШЕНИЕ ПРОБЛЕМЫ ВЫБОРА ИНДЕКСА НА ОСНОВЕ ОБУЧЕНИЯ С ПОДКРЕПЛЕНИЕМ ДЛЯ POSTGRESQL

Хорошо подобранные индексы очень важны для производительности реляционных баз данных. Проблема выбора индекса заключается в определении оптимального набора индексов для рабочей нагрузки при заданных ограничениях, например, при небольшом объеме хранилища. Сложность в решении проблемы возникает из-за большого пространства решений,

которое растет при увеличении количества атрибутов в схеме, атрибутов в индексе и типов индекса, а также во взаимодействии индексов между собой.

Проблема выбора оптимального набора индексов изучается давно и на эту тему есть много исследований [1, 4]. В последнее время в качестве альтернативы существующим эвристическим алгоритмам, основанным на правилах или статистике, изучается подход к решению проблемы на основе обучения с подкреплением (Reinforcement Learning – RL) [2], которое охватывает группу алгоритмов для работы с задачами принятия решений. Алгоритм состоит из агента и среды, с которой он взаимодействует. На каждом шаге взаимодействия t агент видит состояние среды s_t и затем совершает действие a_t, переводя среду в состояние s_{t+1}. Агент также получает вознаграждение r_t, которое показывает, насколько хорошим было действие. Цель агента – максимизировать совокупное вознаграждение. В рамках данной работы предлагается пользовать модификацию алгоритма Proximal Policy Optimization (PPO) [3], которая нацелена на изучение политики $\pi_\theta(a|s)$, где θ – параметры нейронной сети.

В рамках задачи выбора индекса в качестве среды выступает база данных, над которой можно выполнять следующие действия:

– создать индекс(i_h), $i_h \epsilon [I_h]$, где I_h – множество гипотетических индексов;

– удалить индекс(i_e), $i_e \epsilon [I_e]$, где I_e – множество реально существующих индексов.

Награда зависит от оценки планов запросов рабочей нагрузки до и после совершенного действия.

Архитектура предлагаемой системы показана на Рис. 1.

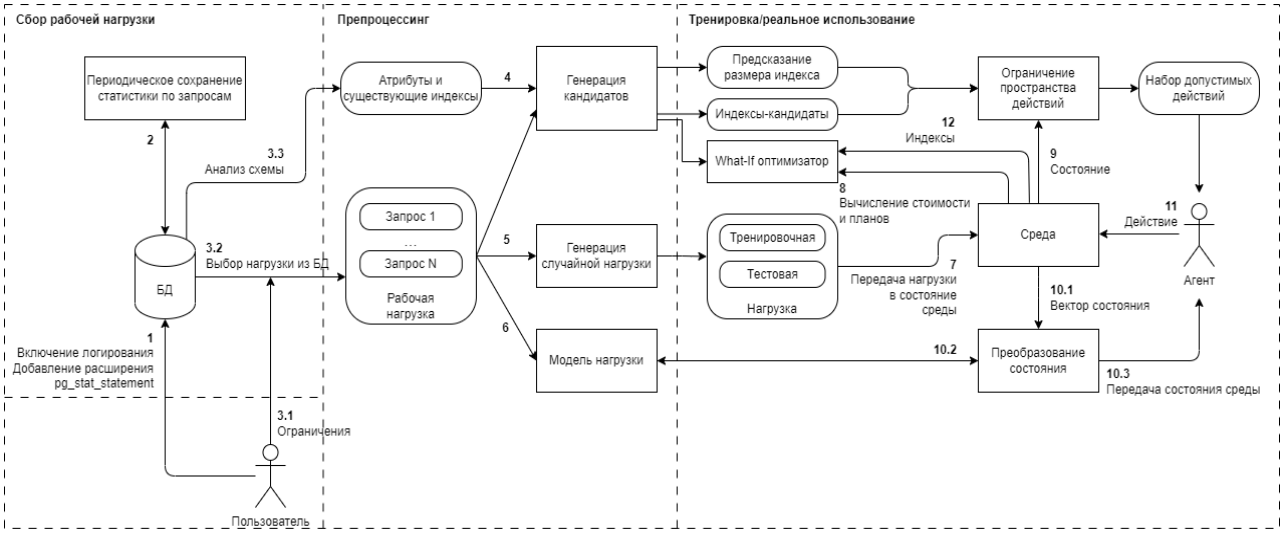


Рисунок 1 – Архитектура системы для выбора индексов на основе обучения с подкреплением.

Сбор рабочей нагрузки предлагается сделать автоматизированным с помощью сбора статистики из расширения pg_stat_statement и логов (1–2). После этого пользователь может запустить процесс обучения, выбрав временной промежуток с нагрузкой и задать ограничения (3.1).

На основе атрибутов схемы и рабочей нагрузки формируется список кандидатов на индексирование (4). На основе реальной нагрузки формируется случайная нагрузка (5), которая состоит из подмножества репрезентативных запросов со случайной частотой. Это необходимо для того, чтобы агент мог адаптироваться к разной нагрузке во время реальной работы. Нагрузка разбивается на тренировочную и тестовую. Так как в основе похода PPO используется нейронная сеть, то она принимает на вход числовые характеристики. Для перевода запросов в числовое представление используется модуль «Модель нагрузки» (6).

Обучение. На вход среды в каждом тренировочном эпизоде поступает новая нагрузка (7). Далее для текущей конфигурации и нагрузки вычисляются планы выполнения с помощью

what-if оптимизатора (8). На основе состояния среды множество действий агента ограничивается (9).

Состояние среды (бюджет, стоимость планов запросов, активные индексы) переводится в числовое представление для передачи агенту (10.2). Состояние среды, и, если есть, награда, передаются агенту (10.3), который в ответ совершает действие (11). Далее процесс повторяется с шага 8 до момента, пока не останется доступных действий.

Практическое использование. После обучения модели на вход поступает реальная нагрузка. Агент постепенно выбирает действия a_t с наилучшей наградой для состояния s_t, пока не закончится пространство выбора.

Преимущество решения состоит в автоматизации сбора рабочей нагрузки и быстрой работе при реальном использовании, так как используется обученная нейронная сеть и нет необходимости проверять гипотезы.

Для реализации алгоритма предлагается использовать язык программирования Python3, в качестве СУБД – PostgreSQL, what-if оптимизатор – HypoPG. Для реализации RL алгоритма (PPO) можно использовать библиотеку Stable-Baselines3 [5], основанную на фреймворке машинного обучения PyTorch. Среда базы данных в данном случае задается в соответствии с OpenAPI gym интерфейсом.

ЛИТЕРАТУРА

1. Kossmann, Jan & Kastius, Alexander & Schlosser, Rainer. (2022). SWIRL: Selection of Workload-aware Indexes using Reinforcement Learning.
2. Машинное обучение // Home page of Lev V. Utkin URL: https://levutkin.github.io/teaching/machine-learning (дата обращения: 18.03.2023).
3. John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. CoRR abs/1707.06347 (2017)
4. Kossmann, Jan & Halfpap, Stefan & Jankrift, Marcel & Schlosser, Rainer. (2020). Magic mirror in my hand, which is the best in the land? An Experimental Evaluation of Index Selection Algorithms. Proceedings of the VLDB Endowment. 13. 10.14778/3407790.3407832.
5. Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, & Noah Dormann (2021). Stable-Baselines3: Reliable Reinforcement Learning Implementations. Journal of Machine Learning Research, 22(268), 1-8.

УДК 004.942

А. В. Шевцов (4 курс бакалавриата),
Ю. Б. Сениченков, д.т.н., профессор

РЕАЛИЗАЦИЯ МОДЕЛИ ПРОГНОЗИРОВАНИЯ COVID-19 НА ОСНОВЕ НЕЙРОННЫХ СЕТЕЙ И СРАВНИТЕЛЬНЫЙ АНАЛИЗ ПОЛУЧЕННОГО РЕШЕНИЯ

Моделирование инфекционных заболеваний является инструментом в изучении механизмов их распространения и средств борьбы с вероятными эпидемиологическими обстановками, а также прогнозирования будущих возможных вспышек. Одной из задач также можно назвать прогнозирование поведения вируса, а, следовательно, и число заболевших, что может быть использовано при принятии решений по введению противоэпидемиологических мер.

Существуют различные подходы к моделированию распространения заболеваний [1]:

- Модели, основанные на анализе временных рядов
- Модели, основанные на дифференциальных уравнениях
- Агентно-ориентированные модели
- Имитационные модели

Вспышка COVID-19 в 2019 году вновь подняла интерес к моделированию распространения вирусных заболеваний. Для этих целей использовались различные виды моделей, перечисленные выше, но многие из них всё ещё требуют участия человека для настройки параметров модели, в частности модели SEIR и SEIR-HCD требуют решения обратной задачи для нахождения параметров [2]. С другой стороны, в наше время всё большую популярность набирают нейронные сети. Они показывают превосходную точность прогнозирования для множества прикладных процессов. В том числе и в задачах моделирования. Нейронные сети обладают возможностью саморегуляции своих параметров, что исключает вовлеченность человека в эту деятельность.

Таким образом *целью* данной работы ставиться разработка новой модели прогнозирования COVID-19 на основе нейронных сетей и проведение сравнительного анализа полученной модели. Для достижения этой цели требуют решения следующие задачи:

- Рассмотреть существующие модели прогнозирования COVID-19 на основе нейронных сетей и их результаты
- Рассмотреть способы улучшения показателей моделей на основе нейронных сетей с помощью методов предобработки данных
- Предложить свою реализацию модели прогнозирования COVID-19 на основе нейронных сетей
- Реализовать предложенную модели
- Провести сравнительный анализ результатов работы своей модели с существующими моделями, рассмотренными ранее

Основой нашей модели является искусственная нейронная сеть (ИНС) – математическая модель, а также её программное или аппаратное воплощение, построенная по принципу организации и функционирования биологических нейронных сетей — сетей нервных клеток живого организма. ИНС представляет собой систему соединённых и взаимодействующих между собой простых процессоров (искусственных нейронов). Важным аспектом является то, что нейронные сети не программируют в привычном нам понимании, а обучают, то есть находят оптимальные связи между нейронами сети для решения конкретной поставленной задачи.

Для описания структуры нейронной сети, а также последующего обучения и визуализации результатов предполагается использование следующих инструментов:

- Python 3
- Tensorflow
- NumPy
- Matplotlib

Язык программирования Python [3] – это высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью. Tensorflow [4] – это открытая программная библиотека для машинного обучения, разработанная компанией Google для решения задач построения и тренировки нейронной сети с целью автоматического нахождения и классификации образов, достигая качества человеческого восприятия. Основной API для работы реализован на Python. NumPy [5] – это библиотека с открытым исходным кодом для языка программирования Python. Имеет поддержку многомерных массивов, высокоуровневых математических функций, предназначенных для работы с многомерными массивами. Matplotlib [6] – это библиотека на языке программирования Python для визуализации данных двумерной и трёхмерной графикой.

Предполагаемый подход подразумевает использование предобработки выходных данных, то есть сглаживание временного ряда, а также использование структуры данных Matrix Profile [7] для большей точности предсказания.

Планируемые результаты:

- Обучение нейронной сети на наборах данных приведенных в рассмотренных исследованиях

– Получение долгосрочного прогноза распространения COVID-19
– Сравнительный анализ полученного прогноза с прогнозами из рассмотренных статей

ЛИТЕРАТУРА

1. Криворотько О. И., Кабанихин С. И. Математические модели распространения COVID-19 – Новосибирск, 2022
2. Криворотько О. И., Кабанихин С. И., Зятьков Н. Ю., Приходько А. Ю., Прохошин Н. М., Шишленин М. А. Математическое моделирование и прогнозирование COVID-10 в Москве и Новосибирской области – Новосибирск, 2022
3. Welcome to Python.org [Электронный ресурс]. Режим доступа: https://www.python.org/
4. TensorFlow. [Электронный ресурс]. Режим доступа: https://www.tensorflow.org/
5. NumPy. [Электронный ресурс]. Режим доступа: https://numpy.org/
6. Matplotlib: Visualization with Python. [Электронный ресурс]. Режим доступа: https://matplotlib.org/
7. Chin-Chia Michael Yeh, Yan Zhu, Liudmila Ulanova, Nurjahan Begum, Yifei Ding, Hoang Anh Dau, Diego Furtado Silva, Abdullah Mueen, Eamonn Keogh. Matrix Profile I: All Pairs Similarity Join for Time Series: A Unifying View that Includes Motifs, Discords and Shapelets – 2016

Секция «Подходы к разработке программного обеспечения на основе технологий группы компаний YADRO»

УДК 004.052.3

Д. А. Варламов (2 курс магистратуры),
А. Д. Ковалев, ассистент,
С. М. Устинов, д.т.н., профессор

РЕАЛИЗАЦИЯ ПРОГРАММНОГО СРЕДСТВА МОНИТОРИНГА СТАБИЛЬНОСТИ ИТ-ИНФРАСТРУКТУРЫ

Информационно-технологическая стабильность сервисов компаний крайне важна. Связанные со стабильностью проблемы приводят к потерям ресурсов и простоям в работе [1]. Анализ предыдущих сбоев нацелен на поиск наиболее критичных и уязвимых компонентов ИТ-инфраструктуры, склонных к поломке [2]. Это позволяет устранять проблемы, связанные со стабильностью работы ИТ-инфраструктуры, чтобы избегать потерь в ресурсах и человеческом времени в будущем. Поэтому существует необходимость в создании и улучшении алгоритмов повышения стабильности ИТ-инфраструктуры [3].

Цель работы – разработка и реализация в программном средстве алгоритма, позволяющего повысить стабильность ИТ-инфраструктуры, в основе которого лежит анализ предыдущих сбоев с целью выявления уязвимых компонентов.

Для определения того, насколько компонент ИТ-инфраструктуры уязвим к поломкам, было принято решение вычислять для него метрику нестабильности. Обнаружение уязвимых компонентов сводится к вычислению для ИТ-инфраструктуры целиком и ее компонентов в частности этой метрики с последующим поиском высоких значений. После анализа подходов к вычислению метрик стабильности [4, 5] было принято решение использовать комбинацию из двух метрик и выделить три возможных состояния ИТ-инфраструктуры: корректная работа, деградация (сервис способен выполнять свой основной функционал, но наблюдаются проблемы со скоростью выполнения и/или дополнительным функционалом) и отключение (сервис не может выполнять свой базовый функционал). Предлагается использовать два показателя нестабильности: хрупкость и отсутствие.

Хрупкость вычисляется по следующей формуле:

$$Fragility = 100 * (1 - (AST-OT-DT)/AST)$$

где AST (Agreed Service Time) – согласованное время обслуживания, OT (Outage Time) – время отключения, а DT (Degradation Time) – время деградации.

Отсутствие вычисляется по следующей формуле:

$$Absence = 100 * (1 - (AST-OT)/AST)$$

где AST (Agreed Service Time) – согласованное время обслуживания, а OT (Outage Time) – время отключения.

На Рисунке 1 представлена общая схема алгоритма. Он состоит из пяти этапов и заключается в сборе информации о состоянии всех компонентов ИТ-инфраструктуры и конвейерной обработке этих данных в нескольких независимо работающих модулях. На первом этапе при помощи модулей сбора данных мониторинга и логов [6] мы получаем данные о текущих состояниях всех компонентов ИТ-инфраструктуры. Одновременно при помощи модуля хранения конфигураций мы получаем их иерархию. На втором этапе все собранные данные объединяются в модуле, хранящем слепок всего состояния ИТ-инфраструктуры в текущий момент. На третьем этапе слепки текущих состояний отправляются в модуль хранения исторических данных, где будут храниться с целью

последующей обработки и анализа. На четвертом этапе исторические данные запрашиваются модулем анализа данных, которые подсчитывает показатели нестабильности для всех компонентов ИТ-инфраструктуры. На последнем пятом этапе полученные показатели представляются в виде дашбордов, облегчая и делая наглядным поиск уязвимых компонентов.

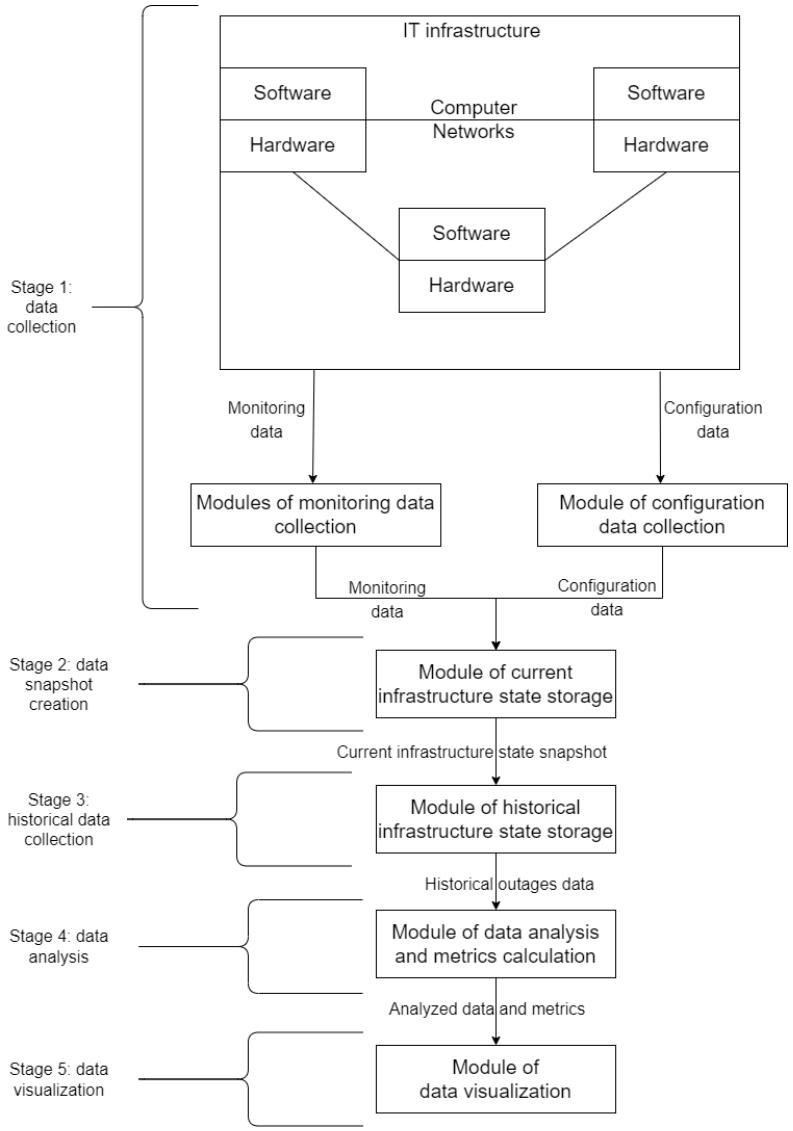


Рисунок 1 – Общая схема алгоритма.

Предложенный алгоритм был применен на практике в производственной компании. Было создано программное средство, реализующее алгоритм, которое выявило уязвимость в файловой системе одного из сервисов. После анализа выяснилось, что сетевая файловая система не справлялась с нагрузкой, вызванной одновременными бэкапами узлов, из-за чего наблюдались перебои в работе сервиса. Был создан внутренний запрос (задача) в системе трекина ошибок JIRA [7]. Проблема была решена настройкой бэкапов на каждом узле сервиса в разное время. В результате среднемесячный показатель хрупкости снизился с 7.42 до 5.57, а показатель отсутствия с 1.50 до 1.31.

Таким образом, применение алгоритма помогло снизить хрупкость практически на 25% и отсутствие практически на 13%.

ЛИТЕРАТУРА

1. P. D. Drobintsev, L. P. Kotlyarova, N. V. Voinov [et al.], Automating preparation of small-scale production for reliable net-centric IoT workshop // CEUR Workshop Proceedings: APSSE 2019 - Proceedings of the 6th International Conference Actual Problems of System and Software Engineering, Moscow, 12–14 ноября 2019 года. – Moscow: Без издательства, 2019. – P. 75-85.

2. Лазарева, Н. Б. Организация отказоустойчивого (на) кластера / Н. Б. Лазарева // E-Scio. – 2021. – № 2(53). – С. 61-66.
3. Унагаев, С. Организация ИТ-инфраструктуры компании // Системный администратор. – 2013. – № 4(125). – С. 40-41.
4. Wang G., Zhang W., Huang C., Chen Z. Service Availability Monitoring and Measurement Based on Customer Perception. 2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS), 2018, pp. 328-331, doi: 10.1109/ICSESS.2018.8663889.
5. Pan C., Xu H., Li W., Tu Z., Xu X., Wang Z. Quality Monitoring and Measuring for Internet of Services. 2021, 107-114. 10.1109/ICSS53362.2021.00025.
6. А. Д. Ковалев, И. В. Никифоров, В. П. Котляров, Разработка распределенной системы архивации данных на основе стандарта OAIS с использованием технологии Apache Hadoop // Информатика и кибернетика (ComCon-2016): сборник докладов студенческой научной конференции Института компьютерных наук и технологий, Санкт-Петербург, 04–09 апреля 2016 года / Министерство образования и науки РФ; Санкт-Петербургский политехнический университет Петра Великого. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2016. – С. 250-252.
7. Kovalev A., Voinov N., Nikiforov I. Using the Doc2Vec Algorithm to Detect Semantically Similar Jira Issues in the Process of Resolving Customer Requests // Studies in Computational Intelligence. – 2020. – Vol. 868. – P. 96-101. – DOI 10.1007/978-3-030-32258-8_11

УДК 004.457

Е. В. Гераськин (2 курс магистратуры),
Н. В. Воинов., к.т.н., доцент

РАЗРАБОТКА ПОДХОДА К СКВОЗНОМУ ТЕСТИРОВАНИЮ ПРИЛОЖЕНИЯ ДЛЯ УПРАВЛЕНИЯ КОНФИГУРАЦИЕЙ ВИРТУАЛЬНОЙ ИНФРАСТРУКТУРЫ ПРЕДПРИЯТИЯ

Настоящая работа посвящена разработке подхода к сквозному (E2E) тестированию [1] приложения для управления конфигурацией виртуальной инфраструктуры предприятия с реализацией фреймворка на языке Python для осуществления этого тестирования. Дано краткое описание предметной области и проблем, связанных с ней. Главный интерес в данной статье представляет подход, основанный на методологии BDD (Behaviour Driven Development) [2], реализованный в рамках вышеуказанного инструмента.

Как и процесс разработки, процесс последующего сквозного тестирования программного обеспечения также следует определенной методологии. Под методологией в данном случае мы понимаем разнообразные комбинации принципов, идей, методов и концептов, к которым инженеры прибегают во время работы над проектом. В настоящее время существует довольно большое количество разнообразных подходов к сквозному тестированию, каждый со своими отправными точками, продолжительностью выполнения и методами, используемыми на каждом этапе. И выбор того или иного из них может быть нетривиальной задачей.

ПО (программное обеспечение) для управления конфигурацией виртуальной инфраструктуры предприятия состоит из трёх основных компонентов:

1. Клиентская часть;
2. Модуль скриптов управления конфигурацией (Ansible [3]);
3. Целевая система инфраструктуры (сервер, компьютер, виртуальная машина и т. д.).

Множество существующих подходов к сквозному тестированию предлагают тестировать подобное ПО в три этапа, то есть тестировать каждый компонент по отдельности. И зачастую это выполняется разными инженерами с разным набором знаний, навыков и

компетенций. При таком раскладе критичным является простой по времени между каждым из этапов сквозного тестирования. Этот простой может занимать как несколько часов, так и несколько дней.

Такие подходы являются очень затратными не только по времени, но и по денежным средствам. Поэтому для сквозного тестирования ПО для управления конфигурацией виртуальной инфраструктуры предприятия необходим подход, который не будет «дробить» систему на отдельные компоненты, а будет рассматривать всю систему как единое целое.

При сквозном тестировании как отдельных компонентов, так и системы в целом используется методология BDD. BDD - это разработка, основанная на описании поведения. Инженер пишет описания вида “я как пользователь хочу, чтобы при нажатии кнопки «Пуск» показывалось меню”. А после описания сценария следует разработка и написание тестов в классическом понимании. BDD предполагает описание инженером пользовательских сценариев на естественном языке. То есть на выходе мы будем иметь не только сквозные тестовые сценарии и результаты их выполнения, но и описание на естественном языке, которое может быть использовано в качестве документации или отчётности. Причём разобраться в результатах сможет человек, не посвящённый в техническую реализацию системы, так как все результаты будут описаны на естественном языке.

Главная проблема применения BDD в сквозном тестировании ПО для управления конфигурацией виртуальной инфраструктуры предприятия – наличие таких компонентов как модуль скриптов управления конфигурацией и целевой системы инфраструктуры.

В классическом варианте применения BDD предлагается покрывать сквозными тестами только клиентскую часть, которая пишется на общеизвестных языках программирования (java, C#, python т.п.). Но непонятно, как быть с инструментами управления конфигурацией (Ansible) и тем более не понятно, как применять BDD к сквозному тестированию состояния сервера или виртуальной машины.

Для решения различных задач разрабатываются специализированные фреймворки (в том числе для тестирования) [4]. Поскольку автоматизированное тестирование клиентской части реализовано с помощью pytest [5], было решено разработать программное средство, которое представляет собой фреймворк для языка Python, и позволяет применить BDD для нетипичных для данного подхода компонентов.

Разработанное для данного подхода программное решение написано на языке python, в качестве пользовательского интерфейса используется специально сконфигурированная Jenkins job [6], за сквозное тестирование целой системы отвечают инструменты pytest и molecule, на основе которых удалось создать модуль для применения подхода BDD для сквозного тестирования приложения для управления конфигурацией виртуальной инфраструктуры предприятия. За генерацию тестового отчёта на естественном языке отвечает инструмент Allure [7].

Программное решение является фреймворком для языка python, который можно импортировать в любой проект с автоматизированными тестами. Взаимосвязь всех компонентов итоговой системы для сквозного тестирования можно описать UML диаграммой последовательностей (Рис. 1):

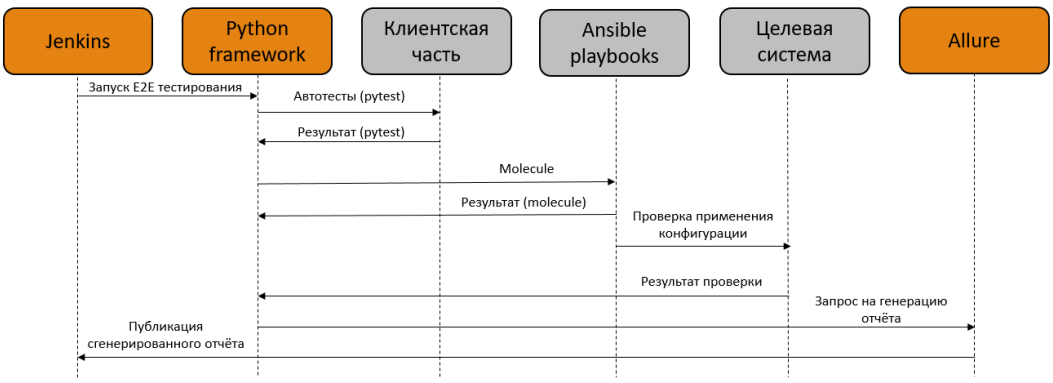


Рисунок 1 - UML диаграмма последовательностей программного решения

На рисунке 1 серым цветом выделены основные модули тестируемого приложения, а оранжевым – модули разработанного программного средства. Jenkins реализует пользовательский интерфейс, через который инженер конфигурирует и запускает процесс сквозного тестирования. Далее Python фреймворк выполняет тестовые сценарии последовательно для каждого модуля: клиентской части (с помощью pytest), Ansible скриптов (с помощью molecule) и целевой системы (проверяя начальное и конечное её состояния). Затем результаты собираются в отчёт, который Allure формирует на естественном языке. Итоговый отчёт публикуется в Jenkins.

ЛИТЕРАТУРА

1. Tsai W. T. End-to-end integration testing design // Annual International Computer Software and Applications Conference. – 2001. – Vol. 25. – DOI 10.1109/CMPSAC.2001.960613.
2. Smart J. F. BDD in Action: Behavior-driven development for the whole software lifecycle. – 2014.
3. Ansible документация. [Электронный ресурс] Режим доступа: https://www.ansible.com/
4. Huang Y. W. A testing framework for Web application security assessment // Computer Networks. – 2005. – Vol. 48. – p. 739 – 761.
5. Pytest документация [Электронный ресурс] Режим доступа: https://docs.pytest.org/en/7.2.x/
6. Jenkins документация. [Электронный ресурс] Режим доступа: https://www.jenkins.io/
7. Allure документация. [Электронный ресурс] Режим доступа: https://docs.qameta.io/allure-report/

УДК 004.42

И. Д. Горшечников (4 курс бакалавриата),
И. В. Никифоров, к.т.н., доцент,
Л. П. Котлярова, ст. преподаватель

ВЫЧИСЛЕНИЕ МЕТРИКИ DCCE ДЛЯ ОЦЕНКИ КАЧЕСТВА ИСПОЛЬЗОВАНИЯ АППАРАТНЫХ РЕСУРСОВ ЦОД

С ростом популярности ИТ-услуг потребность в центрах обработки данных (ЦОД) стала значительно выше [1]. Однако, с увеличением их числа, требуется большее потребление различных ресурсов, включая электроэнергию [2]. Эффективное использование ресурсов становится важной задачей, и одним из предлагаемых решений является внедрение показателей, отражающих корректное применение ресурсов ЦОД [3].

В этой статье предлагается алгоритм к оценке качества использования вычислительных ресурсов центра обработки данных с использованием метрики DCcE (Data Center compute Efficiency). Метрика DCcE относится к классу метрик оценки производительности. Вычисление этой метрики облегчает оценку загрузки компьютера для оптимизации использования ЦП и ОЗУ для улучшения эффективности сервера.

Инструменты для сбора и расчета метрик незаменимы при проектировании центров обработки данных, а также при их последующем обслуживании.

Вычислительная эффективность центра обработки данных (DCcE) используется для оценки эффективности вычислительных ресурсов и выражается как отношение общей вычислительной эффективности сервера к количеству серверов. Она обеспечивает меру эффективности всех вычислительных ресурсов.

DCcE использует оценку эффективности на уровне отдельного сервера, известную как эффективность вычислений сервера (SCE), которая позволяет определить производительность сервера в процентах по нагрузке на ЦП или ОЗУ. Формула SCE приведена ниже, где n – количество замеров, p_i – i-ый замер нагрузки сервера.

$$SCE = \frac{\sum_{i=1}^{n} p_i}{n} \cdot 100$$

Вычислив значения метрики SCE для каждого сервера центра обработки данных, мы можем вычислить DCcE, объединив значения SCE всех серверов. Формула DCcE

представлена ниже, где m – количество серверов ЦОД, SCE_i – значение метрики SCE i-ого сервера.

$$DCcE = \frac{\sum_{i=1}^{m} SCE_i}{m} \cdot 100$$

На Рисунке 1 представлена архитектура программного инструмента для вычисления метрики DCcE. Она состоит из трех основных модулей: визуализации, сбора и обработки данных.

Пользователь взаимодействует только с модулем визуализации, в роли которого выступает веб-приложение. Задумано, что для упрощения работы с центром обработки данных, модуль визуализации может работать с подключением не к одному серверу, а вести работу параллельно с несколькими.

При помощи модуля сбора данных, расположенном на сервере, вычисляются значения нагрузки ОЗУ и ЦП сервера. Эти данные снимаются с интервалом, выбранным пользователем, и добавляются в базу данных PostgreSQL.

Модуль обработки данных получает от модуля визуализации информацию для каких серверов ЦОД необходимо вычислить значение метрики. В результате вычисляется метрика SCE для каждого сервера, по формулам, указанным выше. Далее вычисляется по этим значениям метрика DCcE.

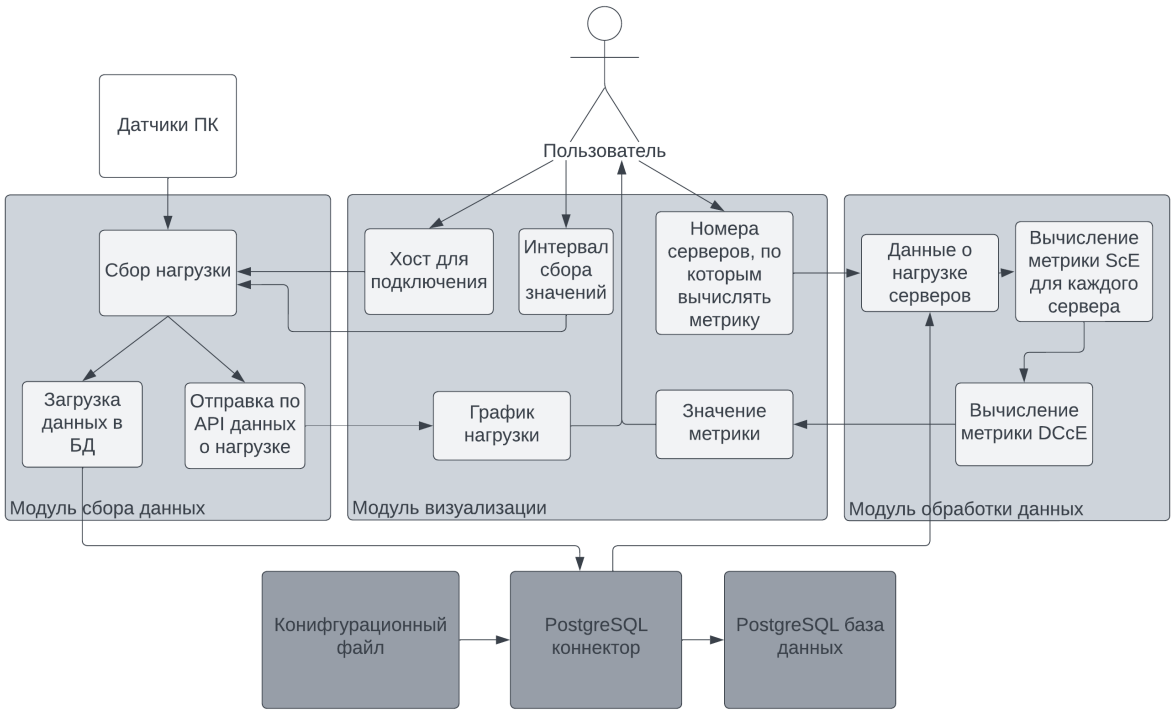


Рисунок 1 – Архитектура инструмента вычисления метрики

Данное приложение предлагается реализовывать, как RESTful-приложение. Модуль сбора и обработки данных был реализован на Java при помощи фреймворка Spring для реализации REST API [4]. В качестве, коннектора к базе данных PostgreSQL выступает JDBC-драйвер. Модуль визуализации реализован на языке JavaScript на фреймворке React.

Работа выполнялась при поддержке компании Dell Technologies.

ЛИТЕРАТУРА

1. Денисенко Б. А., Тянутов М. В., Кубов Н. А., Никифоров И.В Анализ открытых источников данных по нагрузкам на ЦОД // Современные технологии в теории и практике программирования : Сборник материалов конференции, Санкт-Петербург, 26 апреля 2022 года. – Санкт-Петербург:

Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2022. – С. 191-193.

2. Denisenko B., Tyanutov M., Nikiforov I., Ustinov S. "Algorithm for calculating TCO and SCE metrics to assess the efficiency of using a data center," Proc. SPIE 12564, 2nd International Conference on Computer Applications for Management and Sustainable Development of Production and Industry (CMSD-II-2022), 1256403 (5 January 2023); doi:10.1117/12.2669285.
3. Лялин Д. С., Мамадалиев Ш. Р., Никифоров И. В., Устинов С. М. Проектирование системы мониторинга и расчета метрик для эффективного использования ресурсов центров обработки данных // Современные технологии в теории и практике программирования : Сборник материалов конференции, Санкт-Петербург, 26 апреля 2022 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2022. – С. 194-197. – EDN KDNTKM.
4. Shemyakinskaya A. S., Nikiforov I. V. Hard drives monitoring automation approach for Kubernetes container orchestration system // Proceedings of the Institute for System Programming of the RAS. – 2020. – Vol. 32, No. 2. – P. 99-106. – DOI 10.15514/ISPRAS-2020-32(2)-8.

УДК 004.657

М. К. Григорьева (4 курс бакалавриата),
Т. В. Коликова, ст. преподаватель

ПОВЫШЕНИЕ ПРОИЗВОДИТЕЛЬНОСТИ ВЫСОКОНАГРУЖЕННЫХ СИСТЕМ ПУТЁМ ИЗМЕНЕНИЯ ПОДХОДА В РАБОТЕ С СИСТЕМАМИ ХРАНЕНИЯ ДАННЫХ

В настоящее время производительность работы высоконагруженных систем является одним из ключевых факторов, влияющих на успешность бизнеса. С увеличением объема данных и ростом числа пользователей, возникает необходимость в более эффективном использовании ресурсов и повышении производительности систем.

Повышение производительности высоконагруженных систем позволяет значительно снизить время обработки запросов и ускорить выполнение операций, что в свою очередь повышает удобство использования системы для конечных пользователей. Кроме того, улучшение производительности помогает снизить нагрузку на серверы, что способствует повышению надежности и стабильности работы системы.

Важно отметить, что полноценное повышение производительности высоконагруженных систем требует комплексного подхода, который включает в себя оптимизацию аппаратного обеспечения, программного обеспечения и архитектуры системы. Это также может включать использование новых технологий, таких как облачные вычисления и машинное обучение.

В своей работе я буду рассматривать в качестве системы для анализа систему Solar [1], являющуюся разработкой компании Solanteq, обеспечивающую ведение банковского процессинга. Система Solar является высоконагруженной системой, предназначенной для обработки банковских операций. Она обеспечивает быстрое и эффективное выполнение операций, связанных с финансовыми транзакциями, такими как переводы, платежи, работа с программами лояльности и так далее.

Операции, выполняемые в системе Solar, подвергаются строгой проверке на соответствие правилам безопасности и соответствия законодательству. Система обеспечивает надежность и стабильность работы, что особенно важно при обработке крупных объемов данных и высокой нагрузке.

Однако, поскольку количество пользователей, взаимодействующих с системой Solar, каждый день неуклонно растёт, появляется желание развивать систему, чтобы её производительность не снижалась. В случае, если возникает ситуация, в которой произвести обработку полученных банковских данных вовремя не представляется возможным, то есть не

хватает производительности системы для обеспечения целостности данных, последствия могут быть серьёзными и даже необратимыми:

1. Задержка в обработке транзакций, что может привести к недостаточности средств на счетах клиентов и задержке в проведении платежей.
2. Нарушение графика выплат и начислений, что может привести к неудовлетворенности клиентов и потере доверия к банку.
3. Нарушение законодательных требований, связанных с обработкой банковских данных, что может привести к штрафам и убыткам для банка.
4. Потеря конкурентоспособности банка на рынке.
5. Угроза безопасности финансовой системы в целом.

Существует множество способов работать с производительностью и отказоустойчивостью системы, однако мною был выбран путь повышения показателей системы путём изменения правил в работе с системами хранения данных. Все проводимые далее нагрузочные и симуляционные испытания выполнены для реляционной базы данных PostgreSQL, так как именно её чаще всего подключают и используют клиенты Компании.

После проведенного анализа текущих настроек системы, а также изученной по данной теме литературы, было выявлено несколько основных «узких» мест, которые могут повлиять на корректность работы системы:

– Количество коннектов к базе данных: если указать минимальное количество коннектов, связь системы и базы данных может быть разорвана. Однако, при указании излишнего количества коннектов, ресурсы системы будут потрачены в пустую, что также не является оптимальным решением [2].

– Качество запросов, отправляемых в базу данных: любой запрос возможно оптимизировать, чтобы его работа с хранимыми данными была наиболее эффективна и минимально ресурснозатратна [3].

– Высокое количество запросов: важно постоянно проводить проверки актуальности запросов к БД, поскольку система является динамической, то есть подвержена изменениями, вследствие чего часть запросов могут устаревать, вследствие чего перегружать систему.

Проверка теоретических изысканий производилась с помощью программы AnyLogic [4], где были заданы параметры тестируемой системы, после чего была произведена симуляция нагрузки в режиме 1 секунда – 1 запрос. По результатам отклика системы на тот или иной объём передаваемых запросов, было получено понимание о необходимом количестве коннектов к базе данных, а также о количестве отправляемых запросов.

После тщательного анализа полученных ранее результатов испытания были перенесены на реальные банковские стенды, где появилась возможность проверять не только количество коннектов и запросов, но и качество данных запросов, а также их актуальность для конкретного банковского клиента. Нагрузочное тестирование [5] было проведено с помощью внутреннего инструмента компании Solanteq – Testing Toolkit, параметры в рамках которого были откалиброваны и выставлены в соответствии с техническими возможностями Клиента.

После проведения всех перечисленных ранее испытаний рекомендации по повышению производительности системы были переданы Клиенту. В результате чего было сокращено количество запросов, отправляемых Клиентом в отдел поддержки, что свидетельствует об увеличении производительности и повышении отказоустойчивости системы, что благотворно скажется на дальнейшей работе и репутации Банка – Клиента. Дополнительным плюсом выполнения переданных рекомендаций стала экономическая выгода со стороны Клиента, поскольку на каждого Клиента в Компании закладывается определенный бюджет поддержки, ресурсы которого теперь могут быть направлены на другие важные проблемы.

ЛИТЕРАТУРА

1. Главная страница продукта Solar компании Solanteq [Электронный ресурс]. – Режим доступа: https://solanteq.ru/products-and-solutions/
2. Сиднев, Александр Георгиевич (1950-). Системный анализ и принятие решений. Массовое обслуживание для исследования и оптимизации систем: учебное пособие / А. Г. Сиднев, В. Н.

Цыган; Санкт-Петербургский политехнический университет Петра Великого. Санкт-Петербург: Изд-во Политехн. ун-та, 2017 (Санкт-Петербург, 2021) [Электронный ресурс]. – Режим доступа: https://doi.org/10.18720/SPBPU/2/si21-330

3. High Performance MySQL Optimization, Backups, and Replication by Baron Schwartz (Author), Peter Zaitsev (Author), Vadim Tkachenko (Author) [Электронный ресурс]. – Режим доступа: https://www.academia.edu/27289779/High_Performance_MySQL_Optimization_Backups_and_Replication
4. Официальный руководство пользователя по работе с системой имитационного моделирования AnyLogic [Электронный ресурс]. – Режим доступа: https://anylogic.help/ru/
5. Бородин, Антон Александрович. Исследование нагрузочных способностей компьютерных систем = Research of computer systems load capability / А. А. Бородин: схемы, табл. // Научно-технические ведомости Санкт-Петербургского государственного политехнического университета. Сер.: Информатика. Телекоммуникации. Управление = St. Petersburg state polytechnical university journal. Computer science. Telecommunications and control systems : научное издание. Санкт-Петербург. 2014. № 3 (198). С. 19-27. ISSN 2304-9766.

УДК 004.65

А. А. Дутова (4 курс бакалавриата),
И. В. Никифоров, к.т.н., доцент,
Л. П. Котлярова, ст. преподаватель

МЕТОДИКА ПРОЕКТИРОВАНИЯ ОТКАЗОУСТОЙЧИВЫХ ХРАНИЛИЩ ДАННЫХ В ВЫСОКОНАГРУЖЕННЫХ СИСТЕМАХ

Развитие всемирной сети Интернет диктует необходимость в обслуживании как можно большего количества пользователей за короткое время. Такое требование вынуждает разработчиков программного обеспечения обращаться к микросервисной архитектуре [1], основной подход которой состоит в построении единого приложения как набора небольших сервисов. Разбиение на несколько взаимосвязанных компонентов повышает бесперебойность и отказоустойчивость всей системы [2].

Зачастую описанный выше стиль проектирования архитектуры не привязан к определенным аппаратным особенностям системы, поэтому для создания дополнительной изоляции пользовательского пространства от операционной системы используется метод контейнеризации - программные сервисы оборачиваются в контейнеры - легковесные виртуальные наборы изолированных ресурсов, управление которыми доверяется системам оркестрации [3]. Оркестратор позволяет быстро и эффективно разворачивать необходимую инфраструктуру с выбранным заранее количеством создаваемых экземпляров компонентов системы.

Однако, каждому такому компоненту необходим доступ к постоянному хранилищу данных. Чем больше реплик компонентов запущено, тем быстрее пользователь получит результат и тем выше нагрузка на хранилище [4]. Но, к сожалению, база данных не начнет работать быстрее, если к ней на обработку будет приходить больше запросов. Таким образом *целью работы* является изучение технических особенностей построения отказоустойчивых хранилищ данных в масштабируемых системах, анализ методов обеспечения консистентности данных и составление в дальнейшем методик проектирования архитектуры, удовлетворяющей поставленным требованиям.

Для достижения этой цели необходимо решение *задач*, перечисленных ниже.

1. Обзор существующих подходов к построению отказоустойчивых систем в среде оркестрации контейнеров – Kubernetes.
2. Изучение подходов к масштабированию СУБД.
3. Проведение сравнительного анализа найденных методов.

4. Предложение обобщенного вида архитектуры приложения, удовлетворяющего требованиям отказоустойчивости.

5. Аккумулирование полученных знаний с целью написания методики создания консистентных хранилищ данных в высоконагруженных системах.

В качестве основной СУБД для исследования была выбрана свободно распространяемая объектно-реляционная система управления базами данных – PostgreSQL [5]. Основной язык вспомогательных систем необходимых для демонстрации работоспособности - Golang. Для конфигурации и реализации взаимодействия между репликами баз данных использовался скриптовый язык bash.

Архитектура системы, предложенная для рассмотрения, приведена на Рис. 1. На ней в качестве основного механизма обеспечения консистентности данных используется метод репликации.

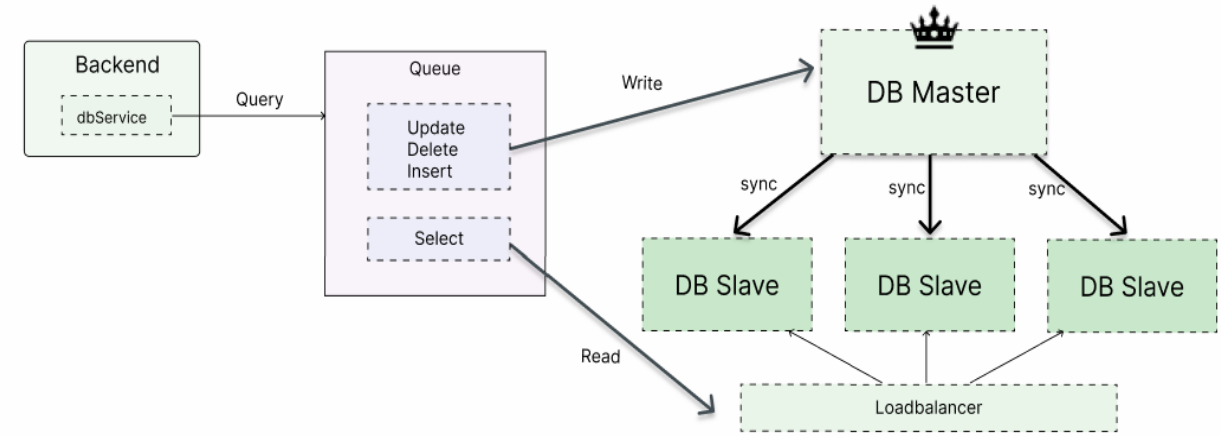


Рисунок 1 – Архитектура системы

На данный момент система находится на этапе разработки: был реализован механизм разделения запросов и синхронизации реплик баз данных, все микросервисы были запущены внутри системы оркестрации и предложенное решение было протестировано с помощью нагрузочных тестов на основе модульного пакета bombardier.

ЛИТЕРАТУРА

1 Барсуков Н. Д., Посметныйс Д., Никифоров И. В. Распределенная микросервисная архитектура системы анализа логов платформы «Открытое образование» // Современные технологии в теории и практике программирования: Сборник материалов научно-практической конференции, Санкт-Петербург, 22 апреля 2021 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2021. – С. 83-85.

2. Betsy Beyer, Chris Jones, Jennifer Petoff, Niall Richard Murphy - Site Reliability Engineering. How Google runs production systems —— СПб.: Питер, 2019. — 592 с.

3. Shemyakinskaya A. S., Nikiforov I. V. Hard drives monitoring automation approach for Kubernetes container orchestration system // Proceedings of the Institute for System Programming of the RAS. – 2020. – Vol. 32, No. 2. – P. 99-106. – DOI 10.15514/ISPRAS-2020-32(2)-8.

4. Voinov N., Drobintsev P., Kotlyarov V., Nikiforov I. Distributed OAIS-Based digital preservation system with HDFS technology // 20th Conference of Open Innovations Association FRUCT: Proceedings, Saint-Petersburg, 03–07 апреля 2017 года / LETI University, St.Petersburg, Russia; S. Balandin, A. Levina, T. Tyutina. – Saint-Petersburg: FRUCT Oy, 2017. – P. 491-497. – DOI 10.23919/FRUCT.2017.8071353.

5. Баданина О. В., Гиндин С. И., Хомоненко А. Д. Оценка оперативности передачи больших данных на примере базы данных PostgreSQL, платформы Hadoop и системы Sqoop. – 2020. – № 2(22). – С. 18-27.

УДК 004.75

Б.А. Денисенко (4 курс бакалавриата),
И.В. Никифоров, к.т.н., доцент,
Л. П. Котлярова, ст. преподаватель

МНОГОПОТОЧНЫЙ ETL-ПРОЦЕСС ДЛЯ СОЗДАНИЯ ЛОКАЛЬНОЙ КОПИИ ИНФОРМАЦИИ ПОСРЕДСТВОМ REST-ИНТЕРФЕЙСОВ

Объем информации в Интернет растет с каждым годом, и, по прогнозам компании IDC, к 2025-ому году он составит 175 зеттабайт [1]. Такой рост обусловлен тем, что данные становятся самым ценным активом для любой организации [2], и потребность в них возникает во многих сферах: бизнес-аналитика, маркетинг, продажи, разработка инновационных продуктов и решений. Для предоставления возможности обмена данными web-ресурсы используют различные интерфейсы. Одним из популярных интерфейсов является REST API [3], в котором для обмена данными используется формат JSON.

Программная обработка информации для получения выводов и умозаключений - трудоемкий по времени процесс. А если ресурс, предоставляющий данные, накладывает ограничения на количество обращений к нему за данными в определенный промежуток времени, как, например, это делает Jira [4], тогда продолжительность обработки увеличивается в разы. Одним из способов ускорения обработки информации является создание локальной копии исходных данных для того, чтобы постоянно и неограниченно иметь возможность к ней обращаться. Для реализации данного решения можно воспользоваться концепцией ETL-процесса (сокращение от Extract, Transform, Load), которая была разработана в 1970-х годах [5]. Хотя локальное хранение данных позволяет избежать многократного ожидания загрузки необходимых данных из Интернет-ресурса каждый раз, когда это понадобится, все равно придется хотя бы один раз произвести процесс загрузки данных. Поэтому при разработке ETL-процесса важно позаботиться о его производительности, которая может достигаться путем распараллеливания процесса.

Целью работы является разработка многопоточного и производительного ETL-процесса для создания локальной копии информации посредством REST-интерфейсов.

ETL-процесс реализован внутри микросервиса “JiraConnector”, который является составной частью приложения “Jira analyzer”, позволяющего проводить аналитику по проектам из Jira. Архитектура коннектора представлена на Рисунке 1.

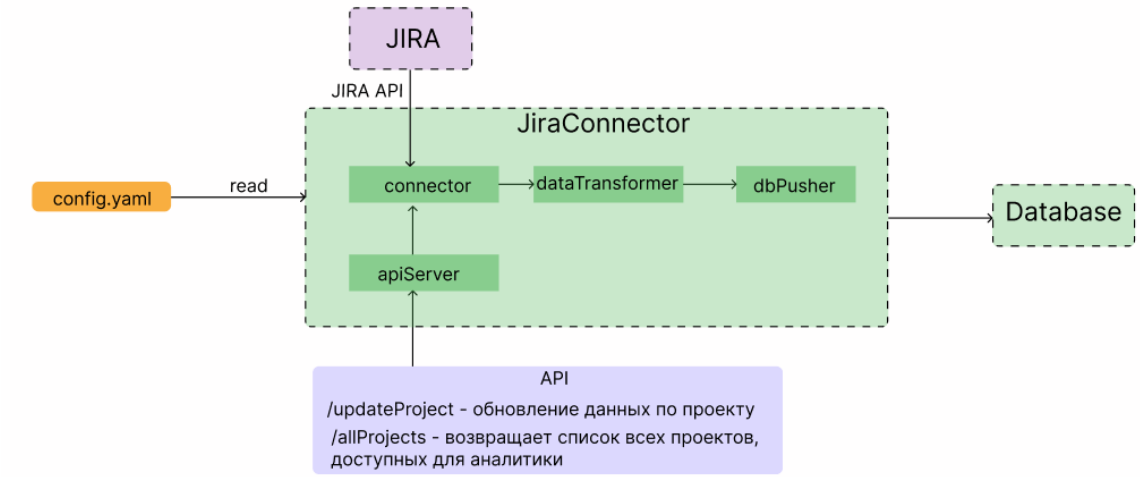


Рисунок 1 - Архитектура микросервиса JiraConnector

Подмодуль connector отвечает за процесс извлечения данных при помощи REST API, предоставляемого Jira. Далее происходит трансформация данных в подмодуле dataTransformer и загрузка этих данных в БД PostregreSQL при помощи подмодуля dbPusher. Модуль

JiraConnector предоставляет API интерфейс для возможности взаимодействия с другими микросервисами разрабатываемого приложения.

Параметры ETL-процесса задаются внутри файла config.yaml. Внутри него можно указать следующие настройки:

- j iraUrl – URL ресурса, предоставляющего Jira REST API
- issueInOneRequest – количество задач, которое будет загружено за один запрос
- threadCount – количество горутин (функций языка Go), в которых будет происходить загрузка задач одного проекта
- maxTimeSleep - максимальное время ожидания перед повторной попыткой запроса
- minTimeSleep – начальное время ожидания перед повторной попыткой запроса
- parallelProjectsDownload - максимальное количество параллельно загружаемых проектов

Модуль JiraConnector был разработан на языке программирования Go с учетом особенностей и ограничений платформы Jira. Важным преимуществом этого модуля является его адаптивность - скорость загрузки данных подстраивается под ограничения Jira. Это означает, что модуль работает эффективно и загружает данные с такой скоростью, которую допускает Jira. При разработке программного решения использовалась система контроля версий Git [6].

Загрузка данных происходит многопоточно. Это значит, что данные по проекту будут загружаться в нескольких горутинах параллельно, что значительно сокращает время загрузки. Для улучшения производительности при помощи персонализации модуль JiraConnector имеет возможность настройки многопоточности с помощью параметров threadCount и parallelProjectsDownload в config.yaml.

Процесс загрузки задач проекта во множестве горутин происходит как атомарная операция. Это означает, что, если в одной из горутин произойдет ошибка, все остальные горутины будут завершены как можно быстрее, и данные, загружаемые в них, не будут отправлены в БД. Это гарантирует, что данные будут сохранены только в том случае, если загрузка во всех горутинах прошла успешно.

Для установки значений по умолчанию параметров issueInOneRequest и threadCount был проведен и описан ряд тестов [7], позволяющих определить их оптимальные значения для эффективной загрузки.

Таким образом, в работе был разработан многопоточный и производительный ETL-процесс для создания локальной копии информации из REST-интерфейсов. Используется этот процесс в микросервисе “JiraConnector” приложения “JiraAnalyzer”. Разработанный ETL-процесс делает процесс загрузки данных проще и быстрее.

ЛИТЕРАТУРА

1. Reinsel D., Gantz J., Rydning J. The Digitization of the World. From Edge to Core, An IDC White Paper, #US44413318, November 2018.
2. Nielsen O.B. A comprehensive review of data governance literature”, Selected Papers of the IRIS, Issue Nr 8, 2017, pp. 120-133.
3. Иванов Г. В., Власов Д. В., Околович Л. Д., Никифоров И. В. Проектирование rest-интерфейса системы транскодирования видеопотоков в режиме реального времени // Современные технологии в теории и практике программирования: Сборник материалов научно-практической конференции, Санкт-Петербург, 22 апреля 2021 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2021. – С. 209-211.
4. Rate limits [Электронный ресурс] / Atlassian Developer. Режим доступа: https://developer.atlassian.com/server/hipchat/hipchat-rest-api-rate-limits/. (дата обращения: 12.03.2023).
5. Никифоров И. В. Курсовое проектирование по учебной дисциплине "Наука о данных и аналитика больших объемов информации": Учебное пособие / И. В. Никифоров. – Санкт-Петербург:

Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2017. – 62 с. – ISBN 978-5-7422-5638-0.

6. Voinov N., Rodriguez Garzon K., Nikiforov I., Drobintsev P. Big data processing system for analysis of GitHub events // Proceedings of 2019 22nd International Conference on Soft Computing and Measurements, SCM 2019: 22, St. Petersburg, 23–25 мая 2019 года. – St. Petersburg, 2019. – P. 187-190. – DOI 10.1109/SCM.2019.8903782.
7. Никифоров И. В., Петров А. В., Котляров В. П. Статический метод отладки тестовых сценариев, сгенерированных с использованием эвристик // Научно-технические ведомости Санкт-Петербургского государственного политехнического университета. Информатика. Телекоммуникации. Управление. – 2012. – № 4(152). – С. 114-119.

УДК 004.89:007.3

В. А. Ивлев, Г. В. Мироненков (2 курс аспирантуры),
И. В. Никифоров, к.т.н., доцент,
А. Д. Ковалев, ассистент

ГЕНЕРАЦИЯ ИНФОРМАЦИОННО-ТЕХНОЛОГИЧЕСКОЙ ИНФРАСТРУКТУРЫ ПРОЕКТА НА ОСНОВЕ НЕФОРМАЛИЗОВАННЫХ ТРЕБОВАНИЙ

Инфраструктура является одним из главных звеньев в экосистеме информационно-технологического (далее ИТ) проекта, благодаря которой повседневно решаются задачи любой сложности. От настройки и автоматизации работы ИТ-инфраструктуры напрямую зависит, насколько быстро и качественно будет поставляться программное обеспечение (далее ПО) конечному потребителю [1].

Но на сегодняшний день для обеспечения настройки инфраструктуры на проекте многие предприятия используют ручной труд отдельных категорий инженеров (например, эксплуатация [2, 3] или DevOps [4]), которые выполняют повседневные монотонные и однообразные задачи настройки инфраструктуры по заранее заданным требования заказчика (например, архитектора системы).

Поэтому задача автоматизации генерации ИТ-инфраструктуры отдельно взятого проекта внутри предприятия по заранее заданным настройкам пользователя является актуальной [1]. Ведь реализация подхода автоматизации потенциально повысит производительность создания ПО, а за счет этого будет снижаться затрачиваемое время на всех его этапах, таких как проектирование, разработка, тестирование ПО, предоставление его конечному пользователю и дальнейшая эксплуатация [2].

Таким образом, опираясь на представленную выше актуальность темы, а также актуальность автоматизированной настройки инфраструктуры ИТ-проекта [1] можно выдвинуть гипотезу, описанную ниже.

Если сократить ручную настройку инфраструктуры ИТ-проекта (исходя из того, что данный процесс является однотипным и постоянно повторяющимся), то можно уменьшить затрачиваемое время на создание проекта и дальнейшую разработку ПО на всех его этапах.

Рассмотреть настройку инфраструктуры для ИТ-проекта, как процесс, описанный выше, можно следующим образом (см. Рис. 1).

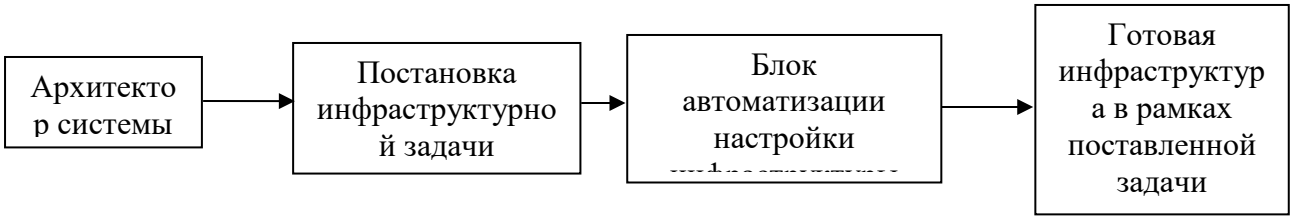


Рисунок 1 – Схема процесса настройки ИТ-инфраструктуры проекта

На представленной схеме, в первую очередь, стоит отметить блок «Постановка инфраструктурной задачи» со стороны архитектора. Данный блок являются довольно обширными и вводит множество факторов, которые в свою очередь повышают сложность и время выполнения работ по настройке инфраструктуры [5]. В их число может входить, например, количество выделяемых серверов и их настройка, типы соединения по сети, настройка сервисов экосистемы ИТ-проекта. Поэтому их важно учитывать при автоматизации настройки ИТ-инфраструктуры. Далее стоит обратить внимание на самый важный блок в схеме – «Блок автоматизации настройки инфраструктуры». Данный блок предусматривает выполнение описанных выше требований архитектора по настройке инфраструктуры стендов силами разработчика. Данный процесс может нести в себе много ручной и однообразной работы по настройке стендов, запуску вспомогательных систем как для всего проекта, так и для решения отдельно поставленных задач. На больших коммерческих проектах разработчикам приходится сталкиваться ежедневно с выполнением подобного рода задач, требующих настройку инфраструктуры ИТ-проекта. Поэтому для решения проблемы ручной настройки стендов при развертывании ИТ-инфраструктуры необходима автоматизация процесса, которая позволит сократить время выполнения повседневных и рутинных задач за счет выполнения развертывания и применения требуемых настроек без непосредственного участия инженера. Конечным результатом может быть увеличение производительности работы команды на проекте.

Для решения проблемы ручной настройки при развертывании ИТ-инфраструктуры предлагается рассмотреть подход автоматизации настройки ИТ-инфраструктуры для проекта (см. Рис. 2). Реализация данного подхода в качестве системы позволит снизить трудоемкость на этапе настройки инфраструктуры за счет распознавания текстовых требований по настройке при помощи технологии NLP [6,7], реализация которой предусматривает использование библиотеки DeepLearning4j [8]. Таким образом система составит требования на основе текста и выполнит их, тем самым автоматизирует процесс внедрения поставленных настроек архитектором в систему.

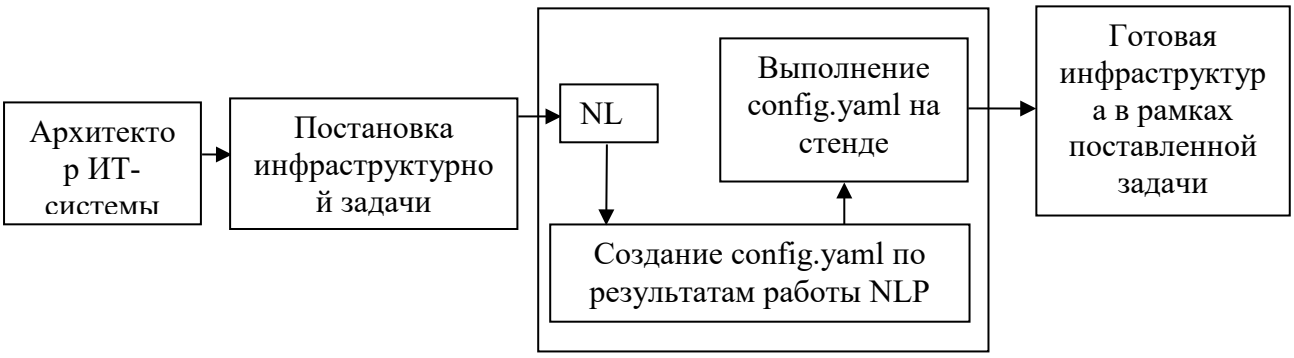


Рисунок 2 – Схема процесса автоматизации настройки ИТ-инфраструктуры проекта с использованием NLP

Отличительной особенностью данного метода является извлечение данных из поставленных требований архитектора системы и формирование на их основе конфигурационного файла [9] и его дальнейшее исполнение на стенде. Как следствие «Блок автоматизации настройки инфраструктуры» теперь выполняет настройку стенда на основе полученных параметров из требований архитектора.

Для корректного расчета результатов работы предложенного подхода по автоматизации настройки инфраструктуры ИТ-проекта, необходимо перед внедрением провести ручное выполнение поставленной инфраструктурной задачи и посчитать затраченное время на ее выполнение. Далее требуется выполнить аналогичные действия с использованием предложенного подхода. На основе поставленных критериев можно выдвинуть следующую формулу.

$$\frac{T_a - T_h}{T_h} * 100\%$$

где $T_a -$ среднее время выполнения с автоматизацией,

$T_h -$ среднее время выполнения задачи ручным способом.

При попытке расчета данных среднего времени выполнения задачи на проекте численностью 10-15 человек ожидается, что ручная настройка стендов с инфраструктурой сократится на ~60% за счет автоматизации процесса применения настроек в систему, а общая производительность команды будет увеличена на ~40%. Полученные результаты опираются на формулу, представленную выше. Данные для расчета были выдвинуты на основе физических суждений автора.

В дальнейшем авторами планируется на основе предложенного подхода автоматизации настройки ИТ-инфраструктуры для ИТ-проекта реализация прототипа программной системы, ее внедрение на ИТ-проекты с разной численностью штата сотрудников и проверка гипотез по уменьшению трудоемкости и увеличению производительности команд за счет внедрения системы, автоматизирующей процесс настройки ИТ-инфраструктуры для ИТ-проекта.

ЛИТЕРАТУРА

1. Ивлев В. А., Никифоров И. В. Актуальность автоматизированной настройки инфраструктуры ит-проекта // Современные технологии в теории и практике программирования : Сборник материалов конференции, Санкт-Петербург, 26 апреля 2022 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2022. – С. 79-80.
2. Jung D., Choi Y. Systematic review of machine learning applications in mining: Exploration, exploitation, and reclamation //Minerals. – 2021. – Т. 11. – №. 2. – С. 148.
3. Kovalev A., Voinov N., Nikiforov I. Using the Doc2Vec Algorithm to Detect Semantically Similar Jira Issues in the Process of Resolving Customer Requests // Studies in Computational Intelligence. – 2020. – Vol. 868. – P. 96-101. – DOI 10.1007/978-3-030-32258-8_11.
4. Sánchez-Gordón M., Colomo-Palacios R. Characterizing DevOps culture: a systematic literature review //Software Process Improvement and Capability Determination: 18th International Conference, SPICE 2018, Thessaloniki, Greece, October 9–10, 2018, Proceedings 18. – Springer International Publishing, 2018. – С. 3-15.
5. Ивлев, В. А. Обработка данных в геоинформационных системах для выбора местоположения рекламы / В. А. Ивлев, И. В. Никифоров, Т. В. Леонтьева // Современные технологии в теории и практике программирования : сборник материалов конференции, Санкт-Петербург, 19 апреля 2019 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2019. – С. 27-30. – EDN DNQPMC.
6. Kang Y. et al. Natural language processing (NLP) in management research: A literature review //Journal of Management Analytics. – 2020. – Т. 7. – №. 2. – С. 139-172.
7. Ковалев А. Д., Никифоров И. В., Дробинцев П. Д. Автоматизированный подход к семантическому поиску по программной документации на основе алгоритма Doc2Vec // Информационно-управляющие системы. – 2021. – № 1(110). – С. 17-27. – DOI 10.31799/1684-8853-2021-1-17-27.
8. Lang S. et al. Wekadeeplearning4j: A deep learning package for weka based on deeplearning4j //Knowledge-Based Systems. – 2019. – Т. 178. – С. 48-50.
9. Сысоев И. М., Никифоров И. В., Каплан Е. В. Создание подхода автоматизации обновления конфигурационных файлов программного обеспечения // Современные технологии в теории и практике программирования: сборник материалов конференции, Санкт-Петербург, 19 апреля 2019 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2019. – С. 126-127.

УДК 004.42

Н.А. Кубов (4 курс бакалавриата),
А. В. Самочадин, к.т.н., доцент,
И. В. Никифоров, к.т.н., доцент,
Л. П. Котлярова, ст. преподаватель

РАЗРАБОТКА РАСШИРЕНИЯ К СИСТЕМЕ НЕПРЕРЫВНОЙ ИНТЕГРАЦИИ JENKINS ДЛЯ АНАЛИЗА ЦЕПОЧЕК СБОРКИ

Современные подходы к разработке программных средств обязательно включают в себя практику непрерывной интеграции. Непрерывная интеграция [1, 2] (CI, англ. “Continuous Integration”) — это практика разработки программного обеспечения, которая заключается в постоянном слиянии рабочих копий в общую основную ветку разработки (до нескольких десятков раз в день) и выполнении частых автоматизированных сборок проекта для скорейшего выявления потенциальных дефектов и решения интеграционных проблем.

Интеграция включает в себя как процессы компиляции, сборки, тестирования и проверки, так и других рутинных специфичных для проектов задач. Выполняя эти задачи вручную, разработчику требуется большое количество времени. Сборка и тестирование часто увеличиваются по времени выполнения с ростом кодовой базы и часто могут заканчиваться ошибками. Поэтому все задачи и соответствующие им процессы стараются автоматизировать.

Система непрерывной интеграции Jenkins [3] является одной из систем поддержки непрерывной интеграции и позволяет автоматизировать рутинные операции. Несмотря на всю мощность технологии и большой набор инструментов доступных по умолчанию, у множества инженеров возникает потребность в использовании дополнительных расширений для анализа [4] выполнения цепочек сборки и упрощения собственной работы.

Целью данной работы является снижение трудоёмкости при анализе процесса непрерывной интеграции посредством визуализации цепочек сборки. Это достигается созданием программного средства, которое реализовано в виде расширения для среды непрерывной интеграции Jenkins.

Таким образом: необходимо провести обзор существующих решений для визуализации цепочек сборки и сравнительный анализ существующих решений по заранее сформулированным критериям для выявления их недостатков. Затем необходимо сформулировать предложения подхода к визуализации и основные функциональные возможности программного средства.

На рынке существует определённый набор плагинов, которые используются для визуализации CI-процессов. Для сравнения использовались плагины, находящиеся в открытом доступе, их список можно найти на сайте Jenkins с плагинами. Наиболее популярными у разработчиков являются следующие расширения: Blue Ocean [5], Pipeline Graph View [6], Yet Another Build Visualizer [7], Stage View [8].

	Название плагина	**Критерии**			
		Отображение пайплайнов в виде диаграмм Ганта	**Отображения пайплайнов в виде графа**	**Отображение метаданных**	**Отображение о сбоях и перезапусках**
1	BlueOcean	Отсутствует	Есть, но с ошибками	Есть	Есть
2	PipelineGraphView	Отсутствует	Есть, но с ошибками	Отсутствует	Есть, но в виде иконок и только о сбоях
3	Yet Another Build Visualizer	Отсутствует	Есть, но с ошибками	Отсутствует	Есть, но только о сбоях
4	StageView	Отсутствует	Отсутствует	Отсутствует	Есть, но только о сбоях

Рисунок 1 – сравнительная таблица существующих решений.

На Рисунке 1 представлена таблица со сравнением существующих решений. В ходе анализа было установлено, что все существующие решения соответствуют критериям сравнения частично, либо не соответствуют.

Из проведённого анализа следует необходимость в разработке нового программного средства в виде расширения для Jenkins, которое будет соответствовать следующим основным требованиям:

- отображать последовательные, ветвящиеся, параллельные, дочерние, родительские и другие структуры цепочек сборки в виде графа;
- отображать длительность и время запуска отдельных элементов цепочки сборки в виде диаграмм Ганта;
- отображать информацию о сбоях и перезапуске цепочек;
- отображать метаданные о структурах цепочек сборки.

Архитектура Jenkins работает по принципу сервлета. Запросы к браузеру осуществляются с помощью Jetty веб-сервер [9]. Внутри Jenkins расширение представлено отдельным модулем. На Рисунке 2 представлена структура расширения.

Рисунок 2 –структура предложенного расширения.

В модуле frontend находятся компоненты, содержащие методы визуализации. В качестве архитектурного подхода для реализации был выбран подход feature-sliced design (FSD) [10]. В качестве инструментов реализации были выбраны язык программирования typescript [11] и фреймворк React [12].

В модуле io/jenkins/plugins/pipelinevisualizer реализована бэкэнд часть. Извлечение данных о цепочке сборки происходит с использованием Jenkins workflow API [13]. Данный модуль осуществляет обработку информации о структуре цепочки сборки и состоянии отдельных её элементов. Также этот модуль использует API Jenkins для создания точек входа, необходимых для расширения функциональности. Обмен данными с модулем визуализации происходит с помощью Rest API. В качестве языка реализации этого модуля был выбран язык программирования Java [14].

В модуле resources находятся файлы ресурсов, используемых в модуле io/jenkins/plugins/pipelinevisualizer.

Таким образом, были рассмотрены существующие решения по визуализации цепочек сборки для Jenkins. В результате анализа были выявлены требования, которым должно соответствовать программное средство. Предложена архитектура и технологический стек для реализации. Далее планируется завершение реализации программного средства и демонстрация его возможностей.

Работа выполняется при постановке задачи от группы компаний YADRO.

ЛИТЕРАТУРА:

1. Никифоров И. В., Кольцов А. А. Автоматизация конфигурации системы непрерывной поставки и интеграции по - Jenkins // НЕДЕЛЯ НАУКИ СПбПУ: материалы научной конференции с международным участием, Санкт-Петербург, 19–23 ноября 2019 года / Санкт-Петербургский политехнический университет Петра Великого. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2019. – С. 56-59.
2. M. Shahin, M. Ali Babar and L. Zhu, "Continuous Integration, Delivery and Deployment: A Systematic Review on Approaches, Tools, Challenges and Practices," in *IEEE Access*, vol. 5, pp. 3909-3943, 2017, doi: 10.1109/ACCESS.2017.2685629.

3. Jenkins [Электронный ресурс] / Jenkins. Режим доступа: https://www.jenkins.io/ (дата обращения: 14.03.2023)
4. Сычев В. К., Ленькова Ю. В., Цветкова Е. А., Никифоров И. В. Анализ данных трендов YouTube // Современные технологии в теории и практике программирования : сборник материалов конференции, Санкт-Петербург, 23 апреля 2020 года / Санкт-Петербургский политехнический университет Петра Великого; Dell Technologies; EPAM Systems. – Санкт-Петербург: ПОЛИТЕХ-ПРЕСС, 2020. – С. 131-132.
5. Web for Blue Ocean [Электронный ресурс] / Web for Blue Ocean. Режим доступа: https://plugins.jenkins.io/blueocean-web/. (дата обращения: 14.03.2023)
6. Pipeline Graph View Jenkins [Электронный ресурс] / Pipeline Graph View. Режим доступа: https://plugins.jenkins.io/pipeline-graph-view/https://www.jenkins.io/. (дата обращения: 14.03.2023)
7. Yet Another Build Visualizer Jenkins [Электронный ресурс] / Yet Another Build Visualizer Режим доступа: https://plugins.jenkins.io/yet-another-build-visualizer/. (дата обращения: 14.03.2023)
8. Stage View [Электронный ресурс] / Pipeline: Stage View. Режим доступа: https://plugins.jenkins.io/pipeline-stage-view/. (дата обращения: 14.03.2023)
9. Jetty [Электронный ресурс] / Eclipse foundation. Режим доступа: https://www.eclipse.org/jetty/. (дата обращения: 14.03.2023)
10. Feature-sliced design documentation [Электронный ресурс] / Feature-Sliced Design. Режим доступа: https://feature -sliced.design/. (дата обращения: 14.03.2023)
11. Typescript programming language [Электронный ресурс] / Typescript. Режим доступа: https://www.typescriptlang.org/. (дата обращения: 14.03.2023)
12. React framework [Электронный ресурс] / React Developer. Режим доступа: https://react.dev/. (дата обращения: 14.03.2023)
13. Workflow API [Электронный ресурс] / Pipeline: API. Режим доступа: https://plugins.jenkins.io/workflow-api/. (дата обращения: 14.03.2023)
14. Java programming language. [Электронный ресурс] Режим доступа: https://www.java.com/ru/https://plugins.jenkins.io/workflow-api/. (дата обращения: 14.03.2023)

УДК 004.428.4

Е. В. Порсев (4 курс бакалавриата),
С. И. Салищев, к.ф.-м.н.

РАЗРАБОТКА БИБЛИОТЕКИ ДЛЯ КОНКУРЕНТНЫХ ВЫЧИСЛЕНИЙ

В области конкурентного программирования развитие языков и библиотек идет в направлении уменьшения сложности разработки. Современные языки предоставляют высокоуровневые интерфейсы для конкурентного программирования, гарантируя отсутствие в программах определенных трудно воспроизводимых ошибок при использовании этих интерфейсов [1].

Интерфейсы для разработки конкурентных программ предоставляют гарантии за счет ограничений возможностей пользователя: не ограничивающий интерфейс не может давать никаких гарантий.

Языки программирования общего назначения можно разделить на две группы. Некоторые предоставляют единый (неограниченный) интерфейс для разработки конкурентных программ [2, 3]. Это позволяет пользователям избежать необходимости изучать несколько различных интерфейсов, но такие языки *не могут защитить пользователя* от многих трудно воспроизводимых ошибок, таких как взаимные блокировки [4]. Другие языки предоставляют одновременно интерфейсы с более сильными гарантиями и интерфейсы с широкими возможностями, но со слабыми гарантиями [1, 5, 6, 7]. В таких языках одновременное использование нескольких интерфейсов затруднено из-за их отличий, и для перехода между интерфейсами требуется переписывать значительные части кода.

Таким образом, для современных языков общего применения справедливо следующее противоречие:

— Если язык предоставляет единый интерфейс конкурентной разработки, то невозможно проверять код на ошибки.
— Если язык предоставляет несколько интерфейсов, то для перехода от одного интерфейса к другому требуется изменения значительной части существующего кода.

Данная работа стремится разрешить это противоречие, возникающее при проектировании языков программирования. Результаты данной работы актуальны при разработке современных конкурентных библиотек и языков программирования.

Цель работы – разработать библиотеку для конкурентного программирования, предоставляющую несколько способов написания конкурентных программ, имеющих единый интерфейс и предоставляющих возможность верификации кода.

Для достижения данной цели необходимо решение следующих задач:

1. Обзор математических моделей конкурентности и конкурентных библиотек в существующих языках программирования.
2. Доработка и уточнение предложения по реализации конкурентной библиотеки, сформулированным научным руководителем.
3. Реализация прототипа библиотеки на языке Java.
4. Внедрение библиотеки в новый язык программирования.

Интерфейс нашей библиотеки опирается на модель сопрограмм, общающихся через каналы. Пользователь определяет вычислительный граф, создавая экземпляры сопрограмм и соединяя их каналами для обмена сообщений. При поддержке со стороны компилятора программист может вызывать сопрограммы как обычные функции, а построение графа может происходить автоматически.

Библиотека позволяет пользователю выбирать режим работы. В неограниченном режиме можно определять любые вычислительные графы, что позволяет решать произвольные задачи конкурентного программирования. В других режимах на графы накладываются ограничения – возможна верификация кода: отсутствие зависания и блокировок для ациклических и синхронных графов [8].

Способы написания конкурентных программ, предоставляемые нашей библиотекой, отличаются предположениями, которые делаются относительно графов сопрограмм. Таким образом интерфейс неограниченного режима является расширением интерфейса ограниченного режима. Следовательно, при переключении между режимами не требуется изменение существующего кода.

В результате был произведен обзор математических моделей и библиотек в существующих языках программирования, на основе чего было сделано предложение по реализации конкурентной библиотеки для нового языка, был разработан прототип этой библиотеки на языке Java, идет внедрение в новый язык программирования. Предложенный интерфейс и архитектура конкурентной библиотеки актуальны при разработке конкурентных библиотек и языков программирования.

ЛИТЕРАТУРА

1. Developers, The Rust Project, «Fearless Concurrency,» [В Интернете]. Available: https://doc.rust-lang.org/book/ch16-00-concurrency.html. [Дата обращения: 05 03 2023].
2. Ericsson AB, «Erlang -- Concurrent Programming,» [В Интернете]. Available: https://www.erlang.org/doc/getting_started/conc_prog.html. [Дата обращения: 05 03 2023].
3. The Go Authors, «The Go Programming Language Specification,» 15 12 2022. [В Интернете]. Available: https://go.dev/ref/spec. [Дата обращения: 05 03 2023].
4. R. Cieslak, «Undecidability results for deterministic communicating sequential processes,» *Proceedings of the 28th IEEE Conference on Decision and Control,* т. 3, pp. 2701-2707, 1989.

5. Oracle, «Package java.util.concurrent,» 18 03 2014. [В Интернете]. Available: https://docs.oracle.com/javase/8/docs/api/index.html?java/util/concurrent/package-summary.html. [Дата обращения: 05 03 2023].
6. Kotlin Foundation, «Coroutines | Kotlin Documentation,» [В Интернете]. Available: https://kotlinlang.org/docs/coroutines-overview.html. [Дата обращения: 05 03 2023].
7. Python Software Foundation, «Concurrent Execution -- Python 3.11.2 documentation,» [В Интернете]. Available: https://docs.python.org/3/library/concurrency.html. [Дата обращения: 05 03 2023].
8. D. Messerschmitt, «Synchronous Data Flow,» *Proceedings of the IEEE, vol. 75, no. 9,* pp. 1235 - 1245, 1987.

УДК 004.053

М. В. Рощин, С. Л. Шалгуева (2 курс магистратуры),
Д. Г. Штенников, к.т.н., доцент

АВТОМАТИЗАЦИЯ УПРАВЛЕНИЯ СКАЛИРОВАНИЕМ МИКРОСЕРВИСОВ С ПОМОЩЬЮ KUBERNETES И KNATIVE SERVING

В последнее время технология бессерверных вычислений получила широкое развитие.[1]. С помощью неё появляется возможность выполнить отдельный взятый кусок кода n-ое количество раз и при этом платить только за время выполнения вычислений, а не за время использования виртуальной машиной. Такие услуги уже предоставляют многие компании такие как Yandex, Amazon, Microsoft, Google [2]. Но что, если возникла потребность чтобы микросервис начинал работал только в тот момент, когда его вызывают, а в остальное время не потреблял ресурсы. Для решения данной задачи отлично подходит Knative.

Knative состоит из двух частей Knative serving и Knative Eventing. В данной работе внимание сосредоточенно на Knative serving. Knative Serving занимается развертыванием и запуском контейнерных приложений в Kubernetes. Разработчики используют Knative Serving для развертывания и запуска своих приложений без необходимости заниматься управлением кластером или сервером. Что является бессерверными вычислениями. Также Knative Serving выполняет работы по маршрутизации трафика, и автоскалирование pod. [3]

Если использовать исключительно Kubernetes, то у пользователя не может вручную определять, когда сервис начинает использоваться и поднимать кол-во pod до 1 или более. При использовании Knative serving вместе с Kubernetes появляется возможность автоматического скалирования pod из 0 в 1. В этот момент сервис не имеет запущенных экземпляров и ждет пока произойдет обращение к данному сервису, как только запрос поступает на сервис Knative serving запускает процесс скалирования сервиса в 1. Если в течение определенного времени, которое пользователь может настроить самостоятельно для каждого сервиса, сервис не получает никаких новых запросов, то контейнер останавливается, pod скалируется в 0 и сервис вновь переходит в режим ожидания. Так же Knative serving может увеличивать кол-во pod до двух и более если на сервис приходит много запросов, тем самым помогая избежать перегрузки сервиса и успешность ответа на запрос.

Процесс автоскалирования в Knative Serving представлена на рисунке 1. На нем отображены основные элементы системы такие как ServerlessService, Activator и Autoscaler.[4]

ServerlessService являет абстракцией над Kubernetes service которая управляет переключением трафика между активатором и прямым запросом к пользовательской поде. Это происходит за счет создание двух Kubernetes service для каждого сервиса: public service и private service. Private Service – это обычный Kubernetes service его селектор указывает, на экземпляр приложения. Public Service – это Kubernetes service который не подчиняется Kubernetes напрямую, у него нет селектора и он не управляется endpoint автоматически. Endpoint в данном случае управляет SKS. SKS в свою очередь имеет 2 режима работы Serv и

Proxy. В режиме Serve весь трафик будет идти на пользовательскую поду как на Рисунке 1. В режиме Proxy трафик будет проходит по пути на Рисунке 2.

Activator участвует в масштабировании до/от нуля и в балансировке. Когда pod масштабируется до нуля, трафик будет идти через активатор как это показано на рисунке 2. Если запросы попадают к активатору, то они буферизуются, передаются в Autoscaler metric и удерживает запросы до тех пор, пока не появятся экземпляры приложения.

Autoscaler отвечает за сбор метрик из queue proxy различных экземпляров приложения и определяет желаемое количество pod, необходимых приложению.

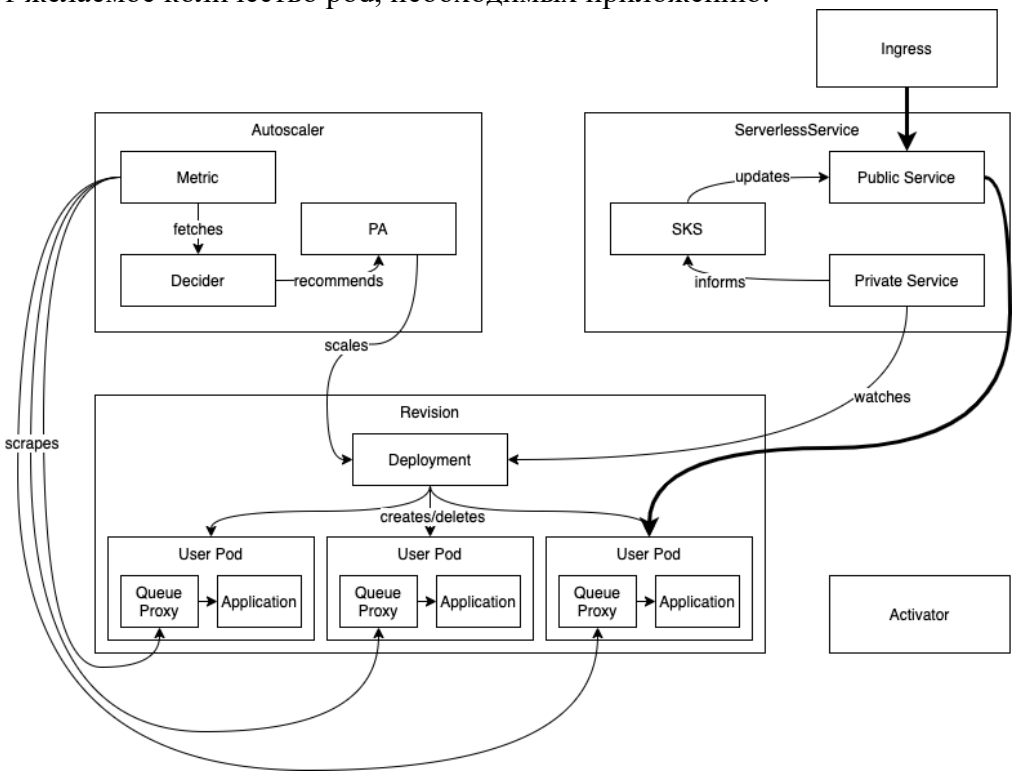


Рисунок 1 – Архитектура Knative Serving

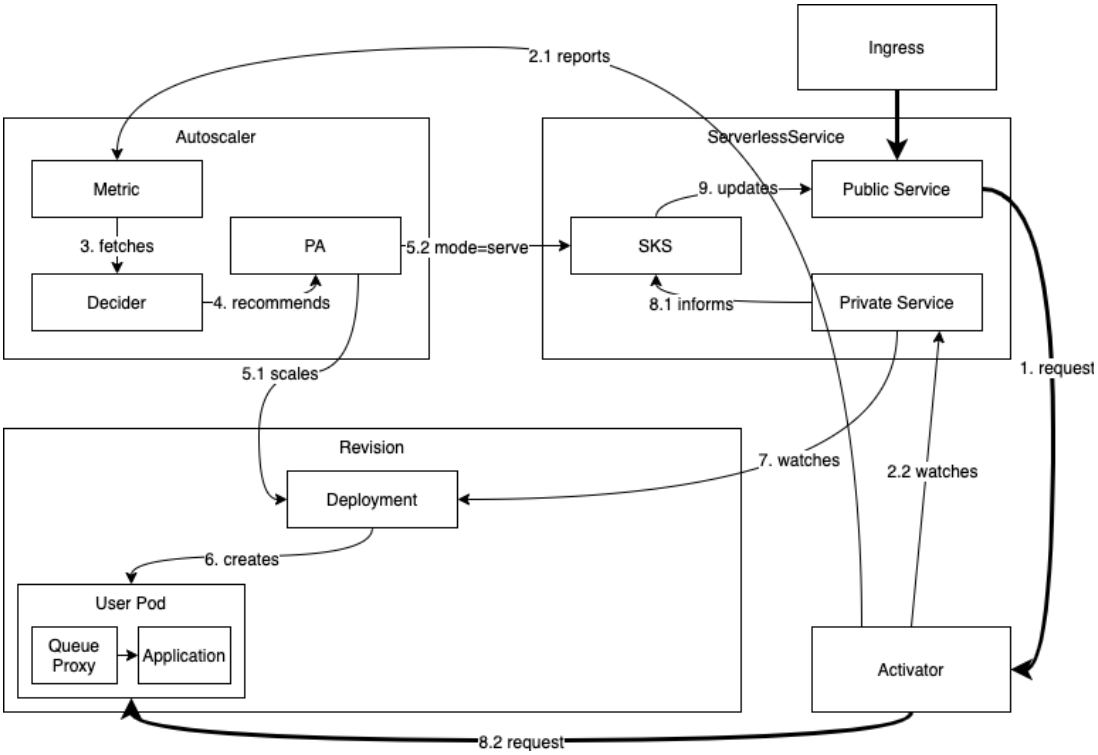


Рисунок 2 – Скалирование из 0 в 1

Использование данной технологии позволит сократить потребление ресурсов сервисов, которые не используются в системе на постоянной основе. За счет чего можно сократить расходы на сервера. Однако максимальную эффективность достигается при использовании с сервисами с низкой полезной работой. Применение с высоконагруженными сервисами не принесет практической пользы.

ЛИТЕРАТУРА

1. Eismann S. et al. A review of serverless use cases and their characteristics //arXiv preprint arXiv:2008.11110. – 2020. (дата обращения: 03.03.2023).
2. Jake Grogan, Connor Mulready. A Multivocal Literature Review of Function-as-a-Service (FaaS) Infrastructures and Implications for Software Developers // In book: Systems, Software and Services Process Improvement – C. 5.
3. Towards Serverless as Commodity: a case of Knative // ResearchGate URL: https://www.researchgate.net/publication/336567672_Towards_Serverless_as_Commodity_a_case_of_Knative (дата обращения: 10.02.2023).
4. Особенности технологий бессерверных вычислений // КиберЛенинка URL: https://cyberleninka.ru/article/n/osobennosti-tehnologiy-besservernyh-vychisleniy (дата обращения: 25.02.2023).

УДК 004.42

Н. В. Сорин (4 курс бакалавриата),
Л. П. Котлярова, ст. преподаватель,
А. Д. Ковалев, ассистент

ВИЗУАЛИЗАЦИЯ ГРАФА ЭТАПОВ СБОРКИ ПРОГРАММНЫХ ПРОДУКТОВ С ИСПОЛЬЗОВАНИЕМ БИБЛИОТЕКИ D3.JS

Количество и сложность программных продуктов со временем все возрастает. Код приходится постоянно изменять, и соответственно заново собирать код, проводить тестирование с учетом изменений и развертывание. Для эффективной работы со сложными проектами трудно обойтись без постоянных сборок программного продукта.

Одним из популярных инструментов для этой цели является Jenkins [1]. Инструмент Jenkins - это программная система с открытым исходным кодом, предназначенная для обеспечения процесса сборки, тестирования и развертывания программного обеспечения. При необходимости непрерывной интеграции [2] часто используют именно Jenkins.

В процессе сборки на разных этапах могут происходить различные ошибки, например, тесты проекта могут быть провалены, или пройдены не с первого раза. Графовое представление - один из способов быстро обнаружить факт падения тестов и этап, на котором они упали. Базовый интерфейс Jenkins не предоставляет возможности увидеть сборку в виде графа. Однако это возможно с подключением в Jenkins плагинов [3].

На данный момент существующие решения имеют различные недостатки, в частности в графовом представлении. А именно, неправильное отображение этапов сборки, недостаточная детализация, или наоборот недостаток общей информации. Таким образом, есть смысл в устранении перечисленных недостатков в визуализации графа шагов сборки программного обеспечения.

Целью данной работы является корректное отображение последовательных, ветвящихся, параллельных, дочерних, родительских и других структур Pipeline [4] в виде графового представления с использованием библиотеки d3.js для последующего отображения данного графа посредством плагина Jenkins.

Для достижения этой цели необходимо решение следующих задач:

1. Провести исследование существующих программных средств, отображающих граф шагов сборки программного продукта.

2. Провести исследование средств визуализации графа, которые могут быть использованы с последующим выводом результата в Jenkins посредством плагина.

3. Предложить архитектуру рисующего граф средства и дизайн самого графа шагов сборки программного продукта.

4. Реализовать модуль, отображающий граф шагов сборки программного продукта, который далее может быть использован в плагине для Jenkins.

Для отображения графа шагов сборки программных продуктов в первую очередь необходима библиотека для визуализации, позволяющая рисовать граф и настраивать его. Также этот граф должен быть использован в плагине Jenkins далее, поэтому это должна быть Java Script библиотека. В ходе проведения сравнительного анализа была выбрана библиотека d3.js [5].

На Рис. 1 представлена концептуальная схема решения. Данные о сборке получаются из Jenkins и преобразовываются в более удобную модель данных. Необходима обработка сложных структур Pipeline, чтобы в дальнейшем без ошибок представлять ветвящиеся, параллельные, вложенные и другие структуры. Модули, отвечающие за графовое представление и за представление в виде диаграмм Ганта, используют упомянутую выше модель данных. Графовое представление реализуется при помощи библиотеки d3.js и отображает всю информацию в виде графа в ярусно-параллельной форме. При нажатии на узел отображается информация дочерних Job. Результат работы этих модулей отображается в пользовательском интерфейсе Jenkins.

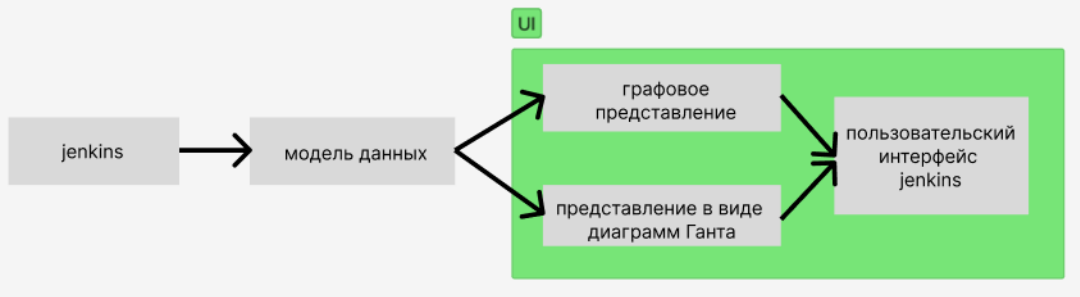


Рисунок 1 – концептуальная схема решения

Пример отображения Job представлен на Рис. 2. По такому отображению можно видеть как общий статус Job-ы, так и общую внутреннюю информацию о состоянии дочерних Job.

Рисунок 2 –UI элементы для отображения Job

Подход реализован в программном средстве на языке Java [6] – родном языке разработки Jenkins и на языке Type Script [7] (более удобен чем Java Script и преобразуется в последний, также имеет возможность использования библиотеки d3.js). Пользователь устанавливает плагин на Jenkins заходит в сборку Pipeline и нажав на встроенную в пользовательский интерфейс Jenkins кнопку видит представление сборки в виде графа. Далее он может оперативно собрать информацию о том, как проходила сборка и о месте и причинах проблем, если они есть.

ЛИТЕРАТУРА

1. Никифоров И. В., Кольцов А. А. Автоматизация конфигурации системы непрерывной поставки и интеграции по - Jenkins // НЕДЕЛЯ НАУКИ СПбПУ: материалы научной конференции с международным участием, Санкт-Петербург, 19–23 ноября 2019 года / Санкт-Петербургский политехнический университет Петра Великого. – Санкт-Петербург: Федеральное

государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2019. – С. 56-59.

2. Mojtaba Shanin, Muhammad Ali Babar, Liming Zhu. Continuous Integration, Delivery and Deployment: A Systematic Review on Approaches, Tools, Challenges and Practices // This work was supported by Data61, a business unit of CSIRO, Australia. The work of M. Shahin was supported by the Australian Government Research Training Program Scholarship. DOI 10.1109/ACCESS.2017.2685629
3. Jenkins Plugin Tutorial. [Электронный ресурс] Режим доступа: https://www.jenkins.io/doc/developer/tutorial/
4. Jenkins Pipeline. [Электронный ресурс] Режим доступа: https://www.jenkins.io/doc/book/pipeline/
5. Data-Driven Documents. [Электронный ресурс] Режим доступа: https://d3js.org/
6. Манаков Н. А., Никифоров И. В. Методика оценки производительности работы виртуальных Java- машин при использовании JIT компиляции // Современные технологии в теории и практике программирования: сборник материалов конференции, Санкт-Петербург, 23 апреля 2020 года / Санкт-Петербургский политехнический университет Петра Великого; Dell Technologies; EPAM Systems. – Санкт-Петербург: ПОЛИТЕХ-ПРЕСС, 2020. – С. 67-69. – EDN KEKOTY.
7. Type Script – npm. [Электронный ресурс] Режим доступа: https://www.npmjs.com/package/typescript

УДК 004.457

С. В. Тагильцев (3 курс бакалавриата),
К. К. Смирнов, ст. преподаватель

PACEMAKER: УПРАВЛЕНИЕ РЕСУРСАМИ ЧЕРЕЗ ПЛАГИНЫ

В современном мире многие большие IT-компании имеют сервера с критически важными базами данных или бизнес-приложениями, которые должны быть доступны практически всегда. Для выполнения этой задачи существуют отказоустойчивые кластеры [1]. IT-компания YADRO, ведущий российский разработчик и производитель высокоэффективных серверов и систем хранения данных корпоративного класса, не является исключением. Отсутствие High-Availability кластера приводит к тому, что из-за выхода из строя одного из серверов становятся недоступными и приложения, которые были на нём запущены. Такая проблема может привести к потерям данных, клиентов, а вследствие чего и прибыли. Отказоустойчивая кластеризация позволяет мониторить состояние своих серверов и своевременно обнаруживать неисправности, а также запускать процессы борьбы с ними без вмешательства системных администраторов, тем самым минимизируя риск сбоев на стороне пользователя. Зачастую это достигается за счет разворачивания новых серверов, которые в свою очередь являются копиями неисправных, или соответствующей конфигурации имеющихся [2].

Одним из основных компонентов отказоустойчивых систем компании ClusterLabs является менеджер ресурсов с открытым исходным кодом pacemaker [3]. Ресурсы кластера управляются при помощи агентов, которые представляют собой внешние скрипты, запускаемые при помощи стандартного для Unix систем метода – fork-exec. Что касается компании YADRO, то в своих высоко-доступных кластерах они используют большое количество ресурсов, которые зачастую лишь пишут заранее подготовленные данные в файлы. Иначе говоря, подготовка и создание нового процесса для таких простых действий выглядит излишней.

Возможным усовершенствованием pacemaker, а также и решением данной проблемы, является добавление поддержки разделяемых библиотек как альтернативы внешним скриптам. Такие библиотеки можно будет динамически подгружать в код, не порождая при этом дополнительных процессов, что должно положительно сказаться на производительности.

При этом важно обеспечить обратную совместимость: в случае если не получилось подгрузить ресурс динамически, инициировать стандартный механизм запуска.

Таким образом, *целью* работы является разработка системы плагинов.

Для достижения данной цели были поставлены следующие *задачи*:

1. Погрузиться в предметную область: развернуть простой HA кластер на двух виртуальных машинах.
2. Разработать и реализовать пример плагина с шаблоном.
3. Разработать и реализовать системы загрузки динамических библиотек.
4. Провести замеры производительности.

Для написания агентов для pacemaker существует стандарт, который называется Open Cluster Framework (OCF) [4]. Агенты представляют собой исполняемые файлы, которые соответствуют основным положениям, изложенным в документации стандарта OCF.

На Рис. 1 представлены архитектура классического pacemaker (слева) и архитектура системы динамической загрузки плагинов (справа)

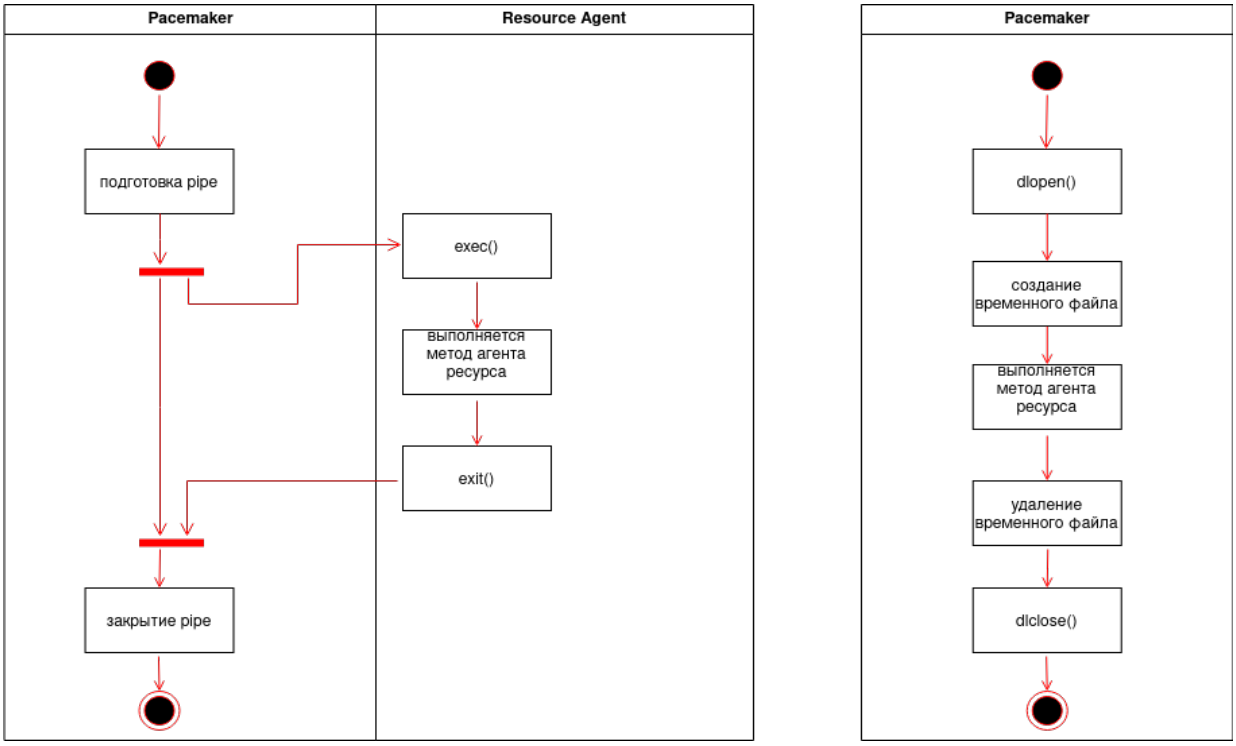


Рисунок 1 – Старая/новая архитектура вызова агентов pacemaker

В рамках знакомства с предметной областью проекта был развернут простенький кластер по гайду от ClusterLabs. Сама же работа велась в рамках специально подготовленного тестового стенда, который собирался из исходников при помощи Vagrant [5].

Сама система загрузки плагинов, включая пример плагина с шаблоном и загрузку динамических библиотек, была реализована на языке программирования C [6] – основном языке разработки pacemaker.

Работа выполнена при поддержке Лаборатории технологий программирования инфраструктурных решений, СПбГУ.

ЛИТЕРАТУРА

1. Introduction To Clustering and High Availability. [Электронный ресурс] Режим доступа: https://docs.geoserver.geo-solutions.it/edu/en/clustering/clustering/introduction.html
2. Active-Active vs Active-Passive High-Availability Clustering. [Электронный ресурс] Режим доступа: https://www.jscape.com/blog/active-active-vs-active-passive-high-availability-cluster
3. ClusterLabs > Pacemaker. [Электронный ресурс] Режим доступа: https://clusterlabs.org/pacemaker

4. OCF Resource Agents. [Электронный ресурс] Режим доступа: http://www.linux-ha.org/wiki/OCF_Resource_Agent
5. Vagrant. [Электронный ресурс] Режим доступа: https://www.vagrantup.com
6. C (programming language). [Электронный ресурс] Режим доступа: https://en.wikipedia.org/wiki/C_(programming_language)

УДК 004

И. С. Томилин (2 курс магистратуры),
С. А.Фёдоров, ст. преподаватель,
В. Э. Шмаков, к.т.н., доцент

РАЗВЕРТЫВАНИЕ МИКРОСЕРВИСНОЙ АРХИТЕКТУРЫ С ИСПОЛЬЗОВАНИЕМ CONSUL, VAULT, NOMAD

Целью работы является демонстрация развертывания кластера состоящего из набора микросервисов используя современные технологии.

Для выполнения данной работы основной упор был сделан в пользу трех основных инструментов: оркестрации(**nomad**[1]), хранения секретов(**vault**[2]) и системы обнаружения сервисов(**consul**[3]) все эти технологии созданы разработчиком инфраструктурных решений — HashiCorp[4] и используя их вместе не требуется использовать сторонние плагины. Продукты HashiCorp имеют хорошую документацию, открытый исходный код, большое сообщество, а также широко используются в крупных компаниях[5].

На Рисунке 1 показана укрупненная схема архитектуры кластера, которая включает в себя не только внутренние микросервисные компоненты системы, но и сторонние фреймворки например БД и систему централизированного журналирования. Итоговая система получается очень крупная, поэтому будут показаны самые основные моменты.

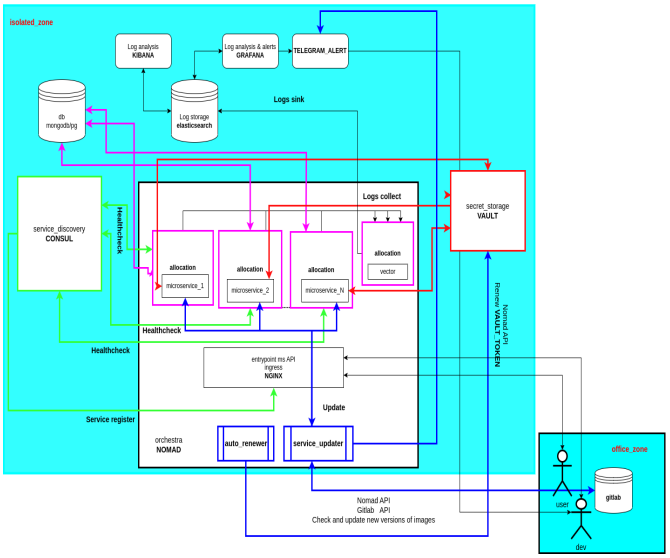


Рисунок 1 – Схема полной архитектуры кластера

```
{
  "SERVICE_NAME": "ms_name",

  "MONGODB_URL": "conn_str",
  "MONGODB_DBNAME": "dbname",

  "APP_PORT": "3000",

  "BODY_LIMIT_BYTES": "2001000",

  "ZEEBE_URL": "http://...",

  "DISCOVERY_URL": "http://...",
  "DISCOVERY_TOKEN": "token"
}
```

Рисунок 3 – Конфигурация микросервиса

```
task "${SERVICE_NAME_CUT}" {

  logs {
    max_files = 1
    max_file_size = 100
  }

  resources {
    cpu    = 300
    memory = 256
  }

  driver = "docker"

  config {
    image = "${CI_REGISTRY_IMAGE}:${SERVICE_VERSION}"
    ports = [ "http_${SERVICE_NAME_CUT}" ]

    extra_hosts = split( "\n", chomp( file("./hosts.txt") ) )

    auth {
      username = "${REGISTRY_LOGIN}"
      password = "${REGISTRY_PASSWORD}"
    }

    volumes = [
      "secrets/.env:/usr/src/app/.env"
    ]
  }

  template {
    change_mode   = "restart"
    change_signal = "SIGHUP"

    data = <<EOF
    {{- with secret "${NS}/${SERVICE_NAME_CUT}/config" -}}
    {{- range $key, $value := .Data.data }}
    {{- printf "%s=\"%s\"\n" $key $value -}}{{ end }}
    {{- end }}
    EOF

    env = true
    destination = "secrets/.env"
  }

  template {
    data = <<EOF
    {{ with secret "gitlabci/config" }}
    REGISTRY_LOGIN="{{ .Data.data.REGISTRY_LOGIN }}"
    REGISTRY_PASSWORD="{{ .Data.data.REGISTRY_PASSWORD }}"{{ end }}
    EOF

    env = true
    destination = "secrets/service-name.env"
  }

  service {
    name = "${NS}-${SERVICE_NAME_CUT}"
    port = "http_${SERVICE_NAME_CUT}"

    check {
      type     = "http"
      port     = "http_${SERVICE_NAME_CUT}"
      interval = "15s"
      timeout  = "3s"
      path     = "/healthcheck"
    }
  }
}
```

Рисунок 2 – Часть nomad job файла

На Рисунке 2 представлена часть конфигурационного файла nomad, которая демонстрирует взаимодействие nomad без сторонних плагинов с consul и vault. Блок **service** регистрирует задачу в consul и затем система обнаружения сервисов переодически выполняет проверку работоспособности микросервиса. Блок **template** позволяет читать данные из системы хранения секретов vault. Запись секрета в виде json файла пример которого показан на Рисунке 3 выполняется по конкретному пути по аналогии файловой системы <**path/to/my/secret**>, чтение секретов из vault реализовано с помощью go templates.

Consul имеет удобный UI интерфейс для отслеживания зарегистрированных сервисов Рисунок 4.

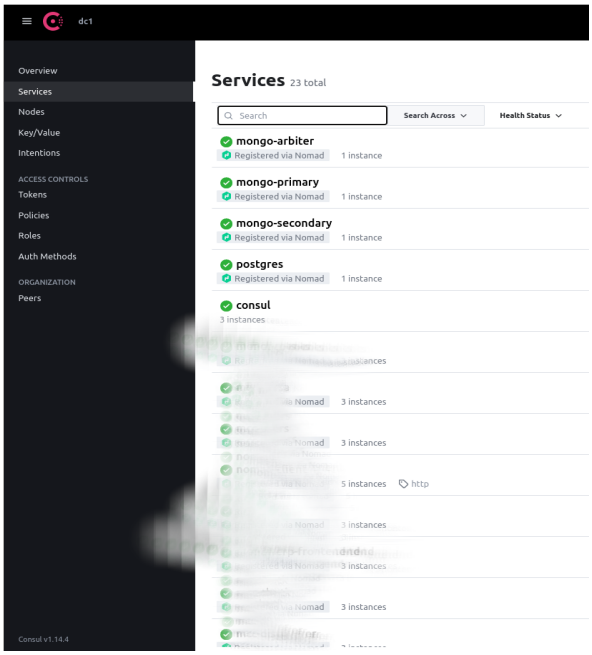


Рисунок 4 – Графический интерфейс consul со списком зарегистрированных сервисов

Nomad также, как и consul имеет удобный UI, показанный на рисунке 5 для отслеживания состояния запущенных микросервисов, просмотр отладочных журналов, запуска/остановки сервисов, выбора пространства имен для показа группы сервисов и многое другое.

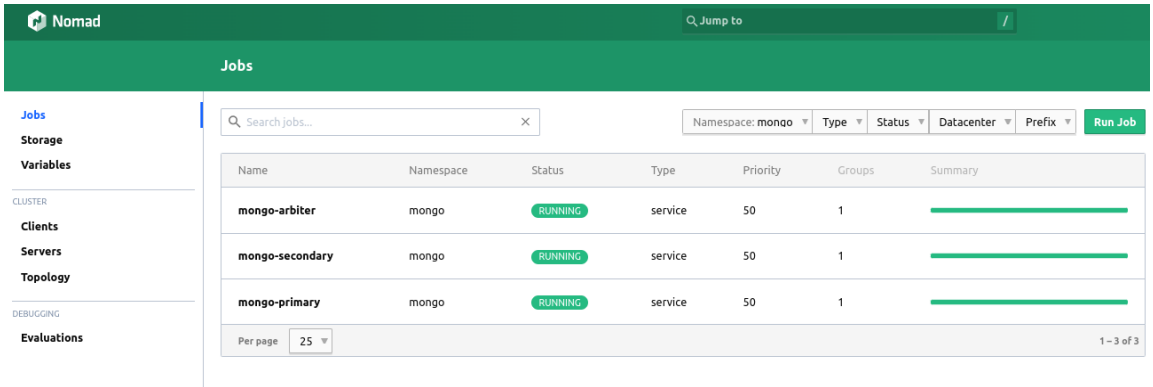


Рисунок 5 – Графический интерфейс nomad

ЛИТЕРАТУРА

1. [Электронный ресурс] https://www.nomadproject.io/
2. [Электронный ресурс] https://www.vaultproject.io/
3. [Электронный ресурс] https://www.consul.io/
4. [Электронный ресурс] https://www.hashicorp.com/
5. [Электронный ресурс] https://developer.hashicorp.com/nomad/docs/who-uses-nomad

УДК 004.05

М. В. Тянутов (4 курс бакалавриата),
И. В. Никифоров, к.т.н., доцент,
Л. П. Котлярова, ст. преподаватель

ПРИМЕНЕНИЕ АЛГОРИТМА РАСЧЕТА МЕТРИКИ TCO ДЛЯ ЦЕНТРОВ ОБРАБОТКИ ДАННЫХ

В современном мире центры обработки данных являются неотъемлемой частью сектора услуг в различных областях человеческой деятельности [1, 2]. Рост спроса на услуги информационных технологий привел к увеличению числа, масштабов и затрат центров обработки данных (ЦОД) по всему миру [3]. В связи с этим возникает необходимость эффективного использования ресурсов дата-центра. Решение этой проблемы требует систематического мониторинга результатов принятых решений. Для этого используются показатели, которые позволяют легко сравнивать результаты [4].

Особенно интересна в этом плане сугубо финансовая метрика Total Cost of Ownership (TCO) [5] – общая стоимость владения. Данная метрика позволяет бизнесу оценивать стоимость обслуживания центра обработки данных. Уменьшение TCO без уменьшения вычислительной мощности ЦОД является одним из наиболее очевидных способов увеличения эффективности ЦОД.

Хоть TCO довольно популярная метрика, она требует индивидуального подхода к каждому продукту.

Целью данной работы является разработка подхода к расчету TCO, который позволит реализовывать гибкий и легко масштабируемый алгоритм для расчета данной метрики. В ходе работы был проведен анализ возможных категорий расходов для TCO. Все категории можно разделить на разовые (капитальные) и операционные затраты:

Капитальные затраты (Capex):

- цена оборудования (ИТ-оборудование - серверы и т.д.);
- стоимость подготовки помещения (ремонт помещения, подготовка электрической системы, системы водоснабжения и т.д.);
- стоимость установки дата-центра (установка систем охлаждения, серверов, стоек и другие);
- цена программного обеспечения (лицензии на проприетарное программное обеспечение, такое как антивирусы, системы мониторинга, ОС и т.д.).

Операционные затраты (Opex):

- цена электроэнергии, потребляемой сервером (серверами);
- расходы на техническое обслуживание (стоимость ремонта оборудования, зарплата обслуживающего персонала и т.д.);
- цена электроэнергии, потребляемой системами охлаждения;
- цена электроэнергии, потребляемой системами освещения;
- аренда помещений.

Данные категории не являются всеобъемлющими или финальными, их можно легко объединять или ещё более разделить в зависимости от желаемой глубины анализа метрики. Так, расходы на энергопотребление можно объединить в одну категорию, или же, наоборот, разделить на несколько, например, отдельно подсчитывать расходы на энергопотребление внутренними и внешними системами охлаждения.

Сам алгоритм расчета может быть представлен следующей формулой (1):

$$TCO = C + \sum_{t=1}^{T} O_t \quad (1)$$

где C – общие капитальные затраты, O_t – общие операционные затраты за период t, T – длина жизненного цикла (в ед. t).

Хоть формула и достаточно проста, сразу применить ее не получится из-за неоднородности операционных расходов. Ниже представлен более подробный алгоритм расчёта TCO:

1. Суммируйте все значения энергопотребления (серверы, освещение, охлаждение).
2. Умножьте значение потребляемой мощности на стоимость электроэнергии, чтобы получить цену электроэнергии за месяц.
3. Суммируйте затраты на электроэнергию за месяц с другими ежемесячными расходами
4. Умножьте ежемесячные расходы на срок службы дата-центра в месяцах, чтобы получить общие операционные расходы.
5. Суммируйте все капитальные затраты, чтобы получить общие капитальные затраты.
6. Суммируйте общие операционные расходы и общие капитальные затраты, чтобы получить общую стоимость владения.

Алгоритм был реализован в виде полноценного программного обеспечения, которое позволяет рассчитывать TCO автоматически. ПО представляет собой консольное приложение, которое использует данные из мок-таблиц и представляет результат в удобном виде круговой диаграммы. В дальнейшем возможно использование Kubernetes [6] для автоматизации разворачивания утилиты мониторинга аппаратурного окружения на множестве узлов.

Так как количество потребляемой сервером электроэнергии зависит достаточно сильно от его нагрузки, то для получения более достоверных данных применяется метрика Server Compute Efficiency (SCE) [7] за некоторое время работы сервера, которая позволяет определять расход электроэнергии сервера, основываясь на минимальном и максимальном электропотреблении.

Таким образом, был представлен гибкий и легко модифицируемый алгоритм расчета метрики TCO для центров обработки данных. Алгоритм был реализован в виде полноценного программного обеспечения и продемонстрированы результаты его работы. Результатом вычисления показателя TCO является круговая диаграмма, показывающая отношение категорий затрат друг к другу, а также таблица, содержащая рассчитанное значение TCO и затраты, для которых это значение было суммировано. Алгоритм показал себя достаточно универсальными и масштабируемыми, а программное обеспечение позволяет получить общую картину затрат машины или всего центра обработки данных.

Работа выполнена при постановке задачи от компаний Dell Technologies.

ЛИТЕРАТУРА

1. Денисенко Б. А., Тянутов М. В., Кубов Н. А., Никифоров И.В Анализ открытых источников данных по нагрузкам на ЦОД // Современные технологии в теории и практике программирования: Сборник материалов конференции, Санкт-Петербург, 26 апреля 2022 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2022. – С. 191-193.
2. Denisenko B., Tyanutov M., Nikiforov I., Ustinov S. "Algorithm for calculating TCO and SCE metrics to assess the efficiency of using a data center," Proc. SPIE 12564, 2nd International Conference on Computer Applications for Management and Sustainable Development of Production and Industry (CMSD-II-2022), 1256403 (5 January 2023); doi:10.1117/12.2669285.
3. B. Schödwell, K. Erek, R. Zarnekow. Data Center Green Performance Measurement: State of the Art and Open Research Challenges // Nineteenth Americas Conference on Information Systems, Chicago, Illinois, August 15-17, 2013.
4. A. M. Ferreira and B. Pernici. Managing the complex data center environment: an integrated energy-aware framework // Computing, vol. 98, no. 7, pp. 709–749, 2014.
5. N. Rasmussen. Determining total cost of ownership for data center and network room infrastructure // Schneider Electric, White Paper 6, 2011.

6. Shemyakinskaya A. S., Nikiforov I. V. Hard drives monitoring automation approach for Kubernetes container orchestration system // Proceedings of the Institute for System Programming of the RAS. – 2020. – Vol. 32, No. 2. – P. 99-106. – DOI 10.15514/ISPRAS-2020-32(2)-8.
7. M. Blackburn. The green grid data center compute efficiency metric: DCcE // The Green Grid, White Paper-34, 2010.

УДК 004.453

А. Ю. Шестакова, А. А. Афанасьев (1 курс магистратуры),
Е. А. Павлов, ассистент

ПОДХОД АВТОМАТИЧЕСКОГО ДЕПЛОЯ DOCKER-ОБРАЗА ИЗ ПРИВАТНОГО DOCKER-РЕПОЗИТОРИЯ В KUBERNETES-КЛАСТЕР С ИСПОЛЬЗОВАНИЕМ ОПЕРАТОРА KEEL

DevOps – это технологическая структура, обеспечивающая взаимодействие между командами разработчиков и операционными командами для более быстрого развертывания кода в производственных средах с возможностью повторения действий и автоматизации. Благодаря использованию практик и подходов DevOps уменьшается количество возникающих при развертывании ошибок, сокращается время выхода сервиса на рынок, создаваемые системы становятся более гибкими, надежными, безопасными и отказоустойчивыми [1]. Основными этапами проекта, следующего принципам DevOps, являются непрерывные развитие, интеграция, тестирование, развертывание, мониторинг и постоянная обратная связь. В ходе работы будет рассмотрен этап непрерывного развертывания.

Для разработки современных облачных приложений необходимо автоматически управлять контейнерами, поскольку приложение может содержать тысячи микросервисов в соответствующих контейнерах. Для решения этой задачи используется оркестрация контейнеров [2].

Kubernetes [3] – популярный оркестратор контейнеров с открытым исходным кодом, который разработчики программного обеспечения используют для развертывания, масштабирования огромного количества микросервисов и управления им.

Целью работы являются настройка и реализация автоматизации деплоя Docker-образа из Docker-репозитория с последующим его развертыванием в Kubernetes-кластере.

Для достижения цели были поставлены следующие задачи:

– Развертывание кластера Kubernetes с применением minikube/docker desktop;

– Развертывание Kubernetes Operator (Keel.sh) [4];

– Настройка Keel.sh для автоматизации деплоя Docker-образа из Docker-репозитория.

Для обеспечения изолированности кластера требуется создать и установить в качестве текущего контекста новое пространство имён.

Так как репозиторий приватный, также требуется создать секретный ключ, который далее будет использован в конфигурационном файле деплоймента в поле imagePullSecrets.

Для передачи образа из DockerHub во внутреннюю структуру Kubernetes применяется WebhookRelay – инструмент ретрансляции, предоставляющий возможность получения веб-хуков во внутренний кластер Kubernetes без настройки общедоступного IP или балансировщика нагрузки [5]. После его установки требуется создать новый веб-хук на официальном сайте [6] и, сгенерировав токен, получить два параметра: RELAY_KEY и RELAY_SECRET. Далее требуется развернуть оператор WebhookRelay, указав полученные параметры.

Следующий шаг после развёртывания оператора – создание пользовательского ресурса, который позволит получать и пересылать веб-хуки.

При создании deployment в WebhookRelay автоматически было создано хранилище для образа Docker, в котором указаны публичный входной эндпоинт, используемый в DockerHub для передачи образа, и выходной эндпоинт, используемый для получения образа в Kubernetes.

Для передачи образа из DockerHub при его обновлении требуется указать имя и публичный эндпоинт созданного WebhookRelay хранилища в DockerHub.

Следующий шаг – установка оператора Keel.sh – фонового инструмента, который автоматически обновляет приложения, запущенные в Kubernetes [7]. Оператор был выбран для обеспечения автоматического извлечения образа и обновления соответствующего развертывания при обнаружении изменения искомого образа.

Для автоматизации деплоя требуется внести следующие изменения в ранее созданный конфигурационный файл deployment.yaml:

– Для автоматического обновления деплоя необходима принудительная политика оператора Keel.sh: keel.sh/policy: force. Она гарантирует, что деплоймент будет обновлён, даже если тег билда не изменился;

– Также для контейнера была установлена политика постоянного пула образа: imagePullPolicy: Always. Она необходима для принудительного извлечения образа.

Далее после обновления конфигурации деплоймента можно наблюдать, что при обновлении сборки изменяется идентификатор деплоймента, а также появляется новая запись в его истории.

ЛИТЕРАТУРА

1. Shemyakinskaya A. S., Nikiforov I. V. Hard drives monitoring automation approach for Kubernetes container orchestration system // Proceedings of the Institute for System Programming of the RAS. – 2020. – Vol. 32, No. 2. – P. 99-106. – DOI 10.15514/ISPRAS-2020-32(2)-8. – EDN ABZBLX.
2. Д. Сафронов, И. В. Никифоров Автоматизация подготовки окружения в среде Kubernetes с помощью расширения инструмента Kind для тестирования функциональности горизонтального масштабирования приложений // Современные технологии в теории и практике программирования: сборник материалов конференции, Санкт-Петербург, 26 апреля 2022 года / Санкт-Петербургский политехнический университет Петра Великого; Dell Technologies. – Санкт-Петербург: ПОЛИТЕХ-ПРЕСС, 2022. – С. 202-204. – EDN WYLKSN.
3. Kubernetes : официальный сайт [Электронный ресурс]. Дата обращения: 12.03.2023. https://kubernetes.io/.
4. Stonozhenko K.M., Nikiforov I.V., Ustinov S.M. Automated object storage management approach with operator SDK and custom resource definition // Proceeding of the Institute for System Programming of the RAS 34 (2), 2022. – P. 123-134. – DOI: 10.15514/ISPRAS-2022-34(2)-10.
5. Webhook-Relay : документация [Электронный ресурс]. Дата обращения: 12.03.2023. https://github.com/webhookrelay?ysclid=lfe3owraa6129217083/.
6. WebhookRelay : официальный сайт [Электронный ресурс]. Дата обращения: 12.03.2023. https://my.webhookrelay.com/.
7. Keel : официальный сайт [Электронный ресурс]. Дата обращения: 12.03.2023. https://keel.sh/docs/.

СОДЕРЖАНИЕ

СОВРЕМЕННЫЕ ТЕХНОЛОГИИ В ТЕОРИИ И ПРАКТИКЕ ПРОГРАММИРОВАНИЯ

Сборник материалов
научно-практической конференции студентов,
аспирантов и молодых ученых

26–27 апреля 2023 года

Налоговая льгота – Общероссийский классификатор продукции
ОК 005-93, т. 2; 95 3004 – научная и производственная литература

Подписано в печать 20.04.2023. Формат 60×84/16. Печать цифровая.
Усл. печ. л. 16,75. Тираж 200. Заказ 1988.

Отпечатано с готового оригинал-макета,
предоставленного редакционной коллегией,
в Издательско-полиграфическом центре Политехнического университета.
195251, Санкт-Петербург, Политехническая ул., 29.
Тел.: (812) 552-77-17; 550-40-14.