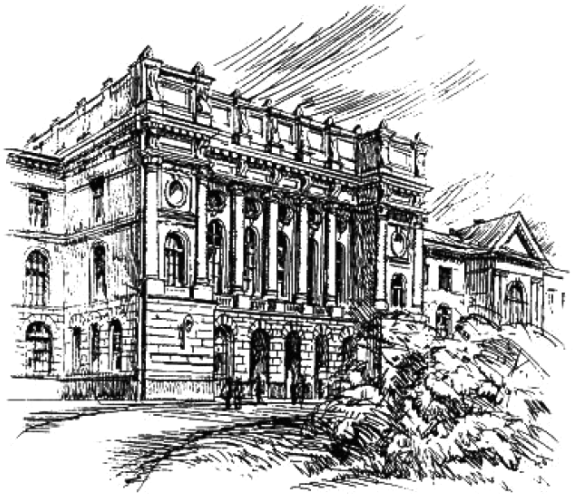
Министерство науки и высшего образования Российской Федерации

САНКТ-ПЕТЕРБУРГСКИЙ
ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО
DELL TECHNOLOGIES

СОВРЕМЕННЫЕ ТЕХНОЛОГИИ В ТЕОРИИ И ПРАКТИКЕ ПРОГРАММИРОВАНИЯ

Сборник материалов конференции

26 апреля 2022 года



ПОЛИТЕХ-ПРЕСС
Санкт-Петербургский
политехнический университет
Петра Великого

Санкт-Петербург
2022

УДК 004
ББК 32.973
С56

Современные технологии в теории и практике программирования : сборник материалов конференции, 26 апреля 2022 г. – СПб. : ПОЛИТЕХ-ПРЕСС, 2022. – 218 с.

В сборнике публикуются материалы докладов студентов, представленные на научно-практической конференции, проводимой Санкт-Петербургским политехническим университетом Петра Великого и организованной Институтом компьютерных наук и технологий при поддержке Санкт-Петербургского центра разработок Dell Technologies. Доклады отражают уровень подготовленности студентов – 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.

Печатается по решению
Совета по издательской деятельности Ученого совета
Санкт-Петербургского политехнического университета Петра Великого.

ISBN 978-5-7422-7680-7

Приветствие от Санкт-Петербургского Центра Разработок Dell Technologies

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

Санкт-Петербургский центр разработок Dell Technologies с 2013-го года поддерживает конференцию «Современные технологии в теории и практике программирования», которая позволяет молодым ученым продемонстрировать результаты их исследований в области информационных технологий и программной инженерии.

Сотрудничеству между Санкт-Петербургским центром разработок Dell Technologies и Политехническим университетом уже более тринадцати лет. За эти годы нами было организовано более двадцати совместных образовательных проектов. С 2016-го года в Политехническом университете реализуются студенческие проекты с открытым исходным кодом, в рамках которых участники вносят свой вклад в продукты, которыми пользуются ведущие ИТ-компании всего мира. Сотрудники компании проводят спецкурсы для студентов-политехников по тематикам современного программирования, управления программным продуктом, хранению данных, современным корпоративным сетям, тестированию программного обеспечения и DevOps подходам.

Целью как данной конференции в целом, так и в частности секции Dell Technologies является, в первую очередь, поддержка молодых ученых, их научных исследований в области программной инженерии, а также информирование молодых ученых о передовых технологических трендах в индустрии. Для компаний, занимающихся разработкой и сопровождением ПО, равно как и другой деятельностью в области ИТ, сотрудничество с университетами жизненно важно для дальнейшего устойчивого развития.

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

С уважением,

Павел Борисович Егоров

Генеральный директор

Санкт-Петербургского Центра Разработок Dell Technologies

Выпускник Санкт-Петербургского политехнического университета

Тезисы докладов конкурса-конференции

Секция «Программная инженерия: приложения, продукты и системы»

УДК 004.4`2

Балдус М.А, Кузьмин В.А. (2 курс магистратуры),
Амосов В.В., к.т.н., доцент

ПОЛЬЗОВАТЕЛЬСКИЙ СЕРВИС ПРОДВИЖЕНИЯ ТРАНСЛЯЦИЙ НА ВИДЕОХОСТИНГЕ

В настоящее время набирает популярность возможность продвижения своего творчества на онлайн площадках, таких как youtube, twitch и др. Онлайн площадки позволяют продемонстрировать своё творчество для пользователей со всего мира, обзавестись фанатами, а также получить материальную поддержку для его продвижения. Основной проблемой для начинающих контент-мейкеров является начало своего пути на этих площадках, так как алгоритмы работы на этих площадках нацелены на продвижение уже состоявшихся контент-мейкеров, которые уже обзавелись своей аудиторией. Алгоритмы этих площадок работают по системе наибольшего количества просмотров, из-за чего контент, который изначально не набрал просмотры, не получит нужного охвата от пользователей.

Целью нашей работы является создание пользовательского сервиса для начинающий авторов, который предоставит возможность им получить искусственно созданную активность на своем творчестве, которая позволит им набрать уже реальную аудиторию.

Техническая часть приложения приведена на Рисунке 1. Сборочный цикл приложения выглядит следующим образом. Кодовая база приложения хранится на приватном git репозитории [2]. Полученный после изменений образ загружается в Docker [5]. Для удобства и экономии времени разработчика докер разворачивается в облаке в данном случае на AWS. В случае необходимости подойдёт любой сервис предоставляющий облачные сервера с поддержкой виртуализации.

На докере загружен образ Jenkins, который необходим для автоматической сборки приложения на сервер и конечному пользователю на клиент. В ходе сборки обновлённый продукт проходит через тесты Backend части приложения при помощи внутренних инструментов языка программирования Go, Frontend часть проверяется при помощи бесплантого фреймворка для тестирования веб-интерфейсов Selenium. В случае положительного результата тестов приложение получает обновление.

Сервер написан на языке Go, данный язык был выбран по причине его особенностей работы с многопоточностью, что является критически важным для нашего проекта [4]. Для хранения данных используется бесплатная база данных MongoDB причина идентична, на данный момент среди бесплатных аналогов нереляционных баз данных, она является одной из лучших для задач связанных с многопоточностью.

Пользовательский интерфейс реализован через веб-клиент, при помощи frontend фреймворка SvelteJs.

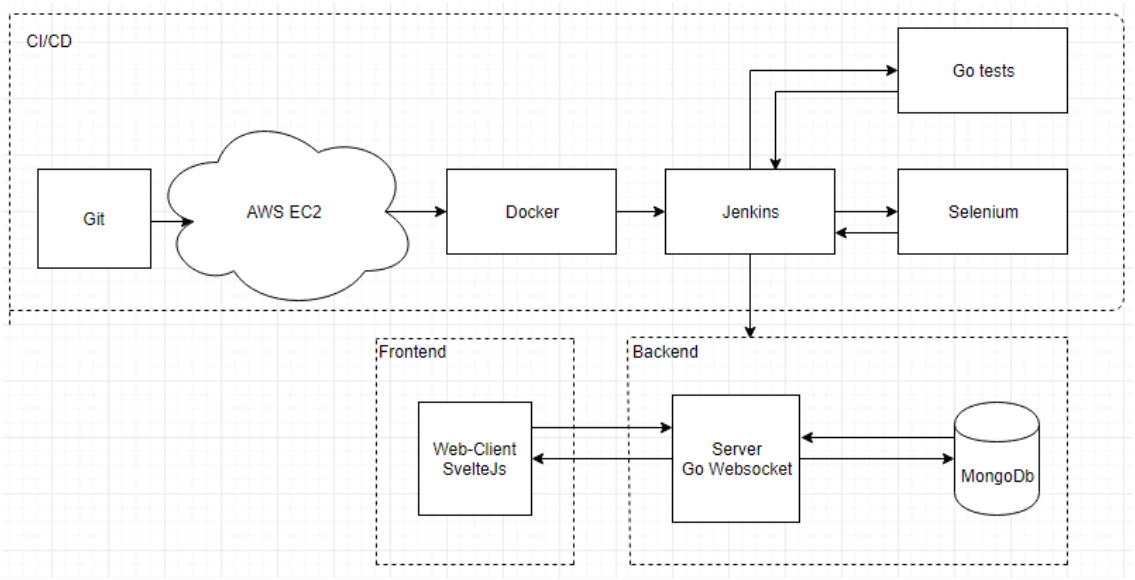


Рисунок 1 – Концептуальная схема архитектуры приложения

Функциональная спецификация сервера:

Площадка видеотрансляций twitch

- Накрутка зрителей через http-запросы [6]
- Накрутка подписчиков через twitch api [1]
- Эмуляция пользователей в чате

Видеохостинг youtube

- Накрутка просмотров, лайков и комментариев на видео
- Накрутка подписчиков на канал youtube api [2]
- Накрутка зрителей на онлайн трансляции
- Эмуляция пользователей в чате на онлайн трансляции

ЛИТЕРАТУРА

1. Twitch API documentation [Электронный ресурс]. Режим доступа:https://dev.twitch.tv/docs/api/
2. YouTube Data API v3 documentation [Электронный ресурс]. Режим доступа: https://developers.google.com/youtube/v3
3. Чакон С., Штрауб Б. Git для профессионального программиста. — Питер, 2017. — 496 с. — ISBN 978-5-496-01763-3.
4. Донован, Алан А. А., Керниган, Брайан, У. Язык программирования Go = The Go Programming Language. — М.: ООО «И.Д. Вильямс», 2016. — С. 432. — ISBN 978-5-8459-2051-5.
5. Э. Моуэт. Использование Docker. Разработка и внедрение программного обеспечения при помощи технологии контейнеров. Руководство = Using Docker: Developing and Deploying Software with Containers. — ДМК Пресс, 2017. — 354 с. — ISBN 978-5-97060-426-7.
6. W. M. Shbair, T. Cholez, A. Goichot, I. Chrisment. Efficiently bypassing SNI-based HTTPS filtering // 2015 IFIP/IEEE International Symposium on Integrated Network Management (IM). — 2015. — Май. — doi:10.1109/INM.2015.7140423.

УДК 004.031.2

Брусницына А.С., Шалгуева С.Л., Шаповалова И.А. (1 курс магистратуры),
Никифоров И.В., к.т.н., доцент

РАЗРАБОТКА РАСПРЕДЕЛЕННОЙ СИСТЕМЫ ПО АНАЛИЗУ ДАННЫХ ЛИЗААЛЕРТ

По данным МВД в России ежегодно пропадает около 180 тысяч человек. Одной из самых известных в России организаций по поиску пропавших без вести людей является ЛизаАлерт [1] – добровольческое некоммерческое общественное объединение. Данные ЛизаАлерт – лишь малая часть от всех пропавших людей, но анализ этих данных позволяет составить представление об общей статистике пропавших без вести.

Краткую ежемесячную статистику по пропавшим людям предоставляет ЛизаАлерт, а также ежегодные и ежеквартальные отчеты предоставляют МВД РФ, СК РФ и ЕМИСС. В нашем решении статистика предоставляется за 11 лет, начиная с 2010 года.

В представленном решении ставятся следующие аналитические группы задач:

– анализ ситуации по регионам;

– анализ поведения людей на форуме;

– анализ вероятности быть найденным;

– анализ условий, в которых пропадают люди;

– анализ возможности использования авиации в поисках.

У ЛизыАлерт есть 3 публичных источника данных: форум, страницы в социальных сетях ВКонтакте и Твиттер. Последние два дублируют первый, поэтому для сбора данных используется только форум. Общий объем полученных данных составил 175 Мб, или 34425 записей о пропавших людях.

Архитектура проекта состоит из четырех базовых модулей [2]: веб-парсинг, модуль обработки данных, модуль загрузки данных в БД и визуализация аналитики.

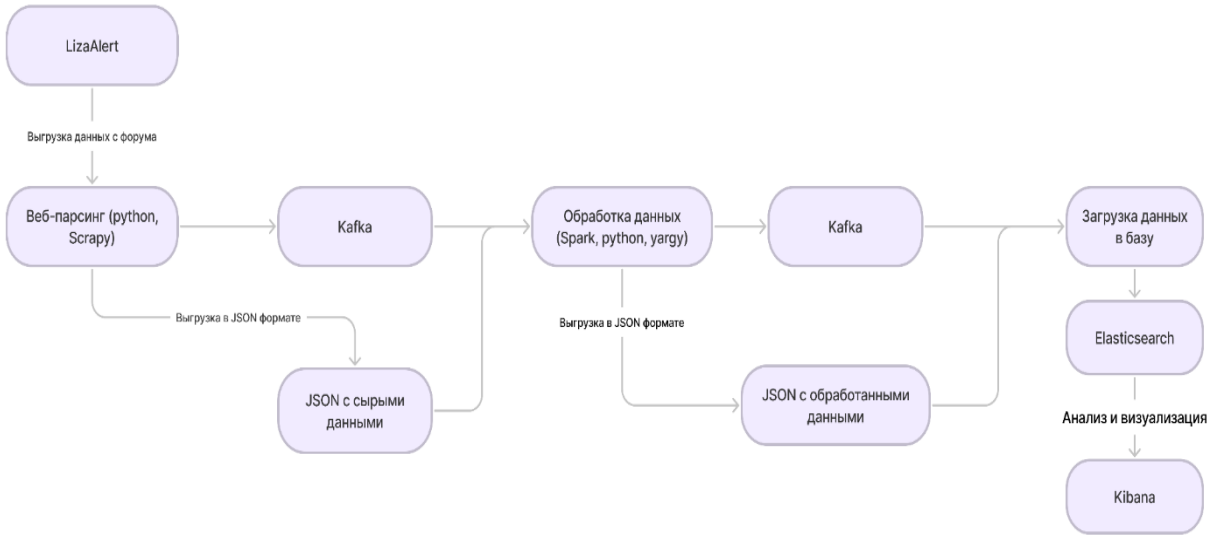


Рисунок 1 – Архитектура проекта

Разработка велась на языке Python версии 3.8.0.

Так как форум не предоставляет API, то был реализован модуль веб-пасинга [3], использующий фреймворк Scrapy [4]. Альтернативной библиотекой является BeautifulSoup.

Для структурирования данных на естественном русском языке с форума использовались инструменты из проекта Natasha [5], а именно парсер Yargy. На данный момент Yargy одним из наиболее популярных парсеров русского текста, распространяемых по открытой лицензии.

Хранение данных происходит в NoSQL базе данных Elasticsearch. Рассматривалось также использование NoSQL базы данных MongoDb, но плюсом Elasticsearch является поддержка полнотекстового поиска, а также встроенная интеграция с Kibana для визуализации данных и аналитики [6].

В качестве брокера сообщений используется Apache Kafka [7], который требует дополнительно установку ZooKeeper. Альтернативой является использование RabbitMQ.

Горизонтальное масштабирование системы обеспечивается за счет модульной архитектуры, использования Apache Kafka для связи между модулями, Apache Spark [7] в модуле обработки данных и Elasticsearch в связке с Kibana.

Система поддерживает три режима работы модуля обработки данных:

– последовательный;

– параллелизация вычислений в рамках одного хоста;

– масштабирование вычислений в кластере Apache Spark.

Для поставленных 5 аналитических групп было решено 18 задач, которые в свою очередь могут разбиваться на подзадачи (аналитические графики). Была собрана общая статистика по пропавшим в разных регионах Российской Федерации, представлен анализ зависимостей шансов быть найденным от различных характеристик, даны ответы на вопросы рациональности использования авиации при поисках.

Реализация проекта поставляется в виде ZIP-архива, в который входят руководство пользователя, скрипт для запуска, файл docker-compose.yml для развертывания необходимых сервисов. Дистрибутив может работать в двух режимах: открыть информационную панель в Kibana, где уже построены графики на заранее подготовленных данных, или запустить сбор данных снова, в режиме непрерывной работы.

ЛИТЕРАТУРА

1. Волонтёрское движение ЛизаАлерт: официальный сайт [Электронный ресурс]. Дата обращения: 21.12.2021. https://lizaalert.org/
2. И. В. Никифоров, 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.
3. D. K. Mahto, L. Singh, A dive into Web Scraper world // 3rd International Conference on Computing for Sustainable Global Development (INDIACom). – 2016. – P. 689-693.
4. D. S. Eyzenakh, A. S. Rameykov, I. V. Nikiforov, High performance distributed web-scraper // Proceedings of the Institute for System Programming of the RAS. – 2021. – Vol. 33. – No 3. – P. 87-100. – DOI 10.15514/ISPRAS-2021-33(3)-7
5. А. Г. Кромских, Подходы к использованию библиотеки natasha для извлечения структурированной информации из текстов на русском языке // Актуальные вопросы эксплуатации систем охраны и защищенных телекоммуникационных систем: Сборник материалов Всероссийской научно-практической конференции, Воронеж, 10 июня 2021 года. - Воронежский институт Министерства внутренних дел Российской Федерации, 2021. - 23-25.
6. К. В. Забелин, В. А. Максимчук, А. С. Шемякинская, И. В. Никифоров, Автоматизация сборки и развертывания системы анализа данных электронной библиотеки СПБПУ // Современные технологии в теории и практике программирования: Сборник материалов конференции, Санкт-Петербург, 22 апреля 2021 года. – СПб. : ПОЛИТЕХПРЕСС, 2021. – С. 207-209.
7. Y. Drohobytskiy, V. Brevus, Y. Skorenkyy, Spark Structured Streaming: Customizing Kafka Stream Processing // IEEE Third International Conference on Data Stream Mining & Processing (DSMP). – 2020. – P. 296-299. – DOI: 10.1109/DSMP47368.2020.9204304.

УДК 004.94

Булатов Д.Е. (2 курс бакалавриата),
Леонтьева Т.В., к.т.н., доцент

РАЗРАБОТКА СИСТЕМЫ БЕЗОПАСНОГО ПРОЕЗДА ПЕРЕКРЕСТКА СПЕЦТРАНСПОРТОМ

Ежегодно в России происходит большое количество ДТП, большая часть из них происходит в связи с несоблюдением правил дорожного движения и невнимательности водителя. Среди всех аварий отдельно можно выделить происшествия со спецтранспортом. Эти автомобили могут отступать от требований ПДД, поэтому при проезде ими перекрестка на красный сигнал светофора возникает риск столкновения с пересекаемым потоком [1].

Целью работы является разработка системы контроля перекрестка, позволяющая исключить ситуацию необходимости проезда спецтранспорта на красный сигнал светофора за счет его заблаговременного переключения для каждой единицы техники.

На данный момент в России реализуются системы интеллектуального управления перекрестком. Однако ни одно из представленных решений напрямую не решает существующую проблему, а лишь разгружает дороги. Кроме того, в основе некоторых из систем лежат физические датчики, встроенные в асфальт. Для их установки требуется качественное дорожное покрытие, время на установку и квалифицированные специалисты. Поэтому в своем проекте я применил другой подход, обнаружения через спутниковые системы автомобилей и обработки данных на серверах (Рисунок 1).

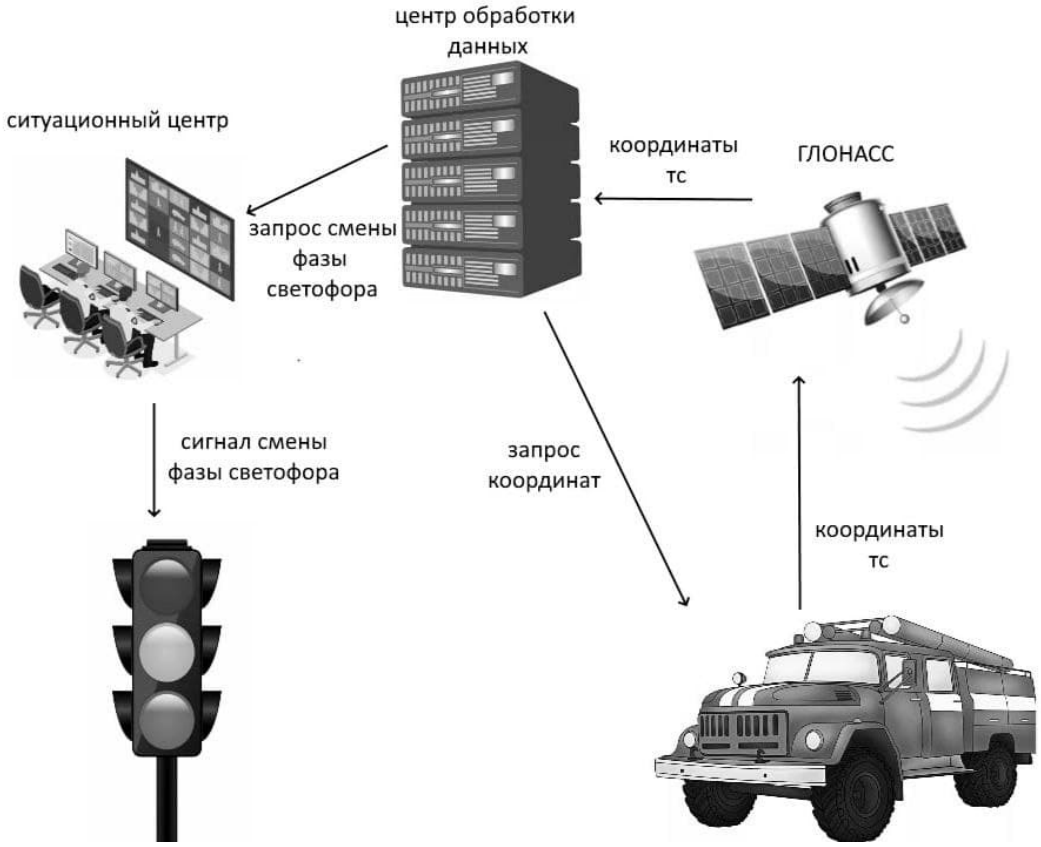


Рисунок 1 – схема передачи данных.

Для сбора информации о местоположении светофоров будут использованы геоинформационные сервисы. После получения данных о местоположении автомобиля через спутниковую систему ГЛОНАСС, и сопоставления его с картой светофорных объектов,

выполняется расчет расстояния до ближайшего перекрестка и определение направления движения. Запросы на получение геоданных производятся раз в 10 секунд, по ним вычисляется скорость транспортного средства. Имея необходимые параметры такие как скорость и расстояние, программное обеспечение всегда знает для каждой единицы транспорта через какой промежуток времени объект достигнет перекрестка. По проведенному исследованию было выяснено, что в среднем для освобождения проезжей части перед перекрестком необходимо 15 секунд с начала процесса смены фазы. Таким образом в момент, когда время достижения перекрестка становится равным 15 секундам, в ситуационный центр отправляется информация о направлении движения автомобиля спецслужб [2,3], после чего на перекрестке начинается смена фазы светофоров.

Разработана компьютерная модель движения транспорта по участку города с несколькими перекрестками. Программа написана на языке программирования C++ с применением библиотеки SFML, в среде разработки Microsoft Visual Studio 2019. В общем виде программу можно представить, как 3 составные части. Первая – обработка карты. На входе программа имеет карту с обозначенными дорогами и расставленными светофорами. По линиям дорог передвигаются автомобили, в том числе, автомобили специальных служб, которые передают информацию о своем положении во 2 модуль – модуль обработки данных. Этот модуль выполняет задачи центра обработки данных (определяет расстояние до перекрестка, направление движения и передает данные в нужный момент в 3 модуль). Третий модуль отвечает за управление светофорами. По полученной информации о направлении движения автомобиля и его координатам он определяет на каком светофоре необходимо переключить фазу. Таким образом, если сопоставить все модули программы и описанную ранее схему работы в реальных условиях, то первый модуль - это реальная дорога, второй модуль - серверная часть, а третий - ситуационный центр. Тестирование программы производилось путем запуска разного количества как обычных автомобилей, так и автомобилей спецтранспорта, заданием разного количества и типов перекрестков. При отключении разработанной системы наблюдается резкое увеличение столкновений объектов, а при ее работе столкновения возникают только в случаях пересечений траекторий при появлении новых объектов на карте. Конечно, такая модель не может точно отобразить поведение участников дорожного движения в реальных условиях, однако она показывает снижение риска столкновения транспортных средств вследствие невнимательности водителей, путем исключения возможности возникновения опасной ситуации на дороге.

Проведенные над компьютерной моделью тесты показывают, что система готова к внедрению в городскую среду, и она действительно способна снижать риск ДТП. Однако сама модель в дальнейшем продолжит свое существование для безопасной проверки новых возможностей и вариаций своего применения. Для этого будут добавлены новые возможности в программу. Среди них можно выделить введение возможности загрузки в программу карты конкретного города, для изучения целесообразности применения системы в этом населенном пункте со всеми его особенностями, которые не могут быть учтены в базовой модели.

ЛИТЕРАТУРА

1. Безопасность дорожного движения [электронный ресурс] URL: https://национальныепроекты.рф/projects/bezopasnye-kachestvennye-dorogi/bezopasnost_na_dorogakh.
2. Александров А.В., Сениченков Ю.Б., Модель оптимизации светофорных фаз, Современные технологии в теории и практике программирования: сборник материалов конференции, СПб, 2021.
3. Как управляются светофоры [электронный ресурс] URL: https://vasiliybalanyuk.livejournal.com/638110.html.

УДК 004.4

Гладун В.В. (4 курс бакалавриата),
Воинов Н.В., к.т.н., доцент,
Юсупова О.А., ассистент

РАЗРАБОТКА ПРИЛОЖЕНИЯ ДЛЯ АНАЛИЗА И ОБРАБОТКИ ЛОГ-ФАЙЛОВ С ПОЛЕТНОЙ ИНФОРМАЦИЕЙ

Работа посвящена разработке программы, позволяющей извлекать данные из лог-файлов с полетной информацией и анализировать их. Программа должна уметь работать с большим количеством таких файлов и структурировать полученные данные, строить графики и формировать отчеты [1]. Анализируются данные полета тестируемого самолета для дальнейших выводов и возможных корректировок конструкции самолета.

Данные из лог-файлов разделяются на отдельные параметры и по каждому параметру происходит анализ. Программа сверяет полученные значения с ожидаемыми. У каждого параметра разные допустимые значения. В конце работы программы пользователю будет представлен отчет обо всех параметрах. Будет выведена информация о попадании или промахе в допустимые значения каждого параметра [2].

Допустимые параметры задаются в отдельном конфигурационном файле. Пользователю будут представлены все заданные параметры, а также возможность задавать свои значения для каждого параметра на свое усмотрение.

В программе реализовано построение графиков на основе полученных данных из лог-файлов, в том числе график полного полета. Пользователь сможет построить график по тем или иным данным, выбранным им перед построением.

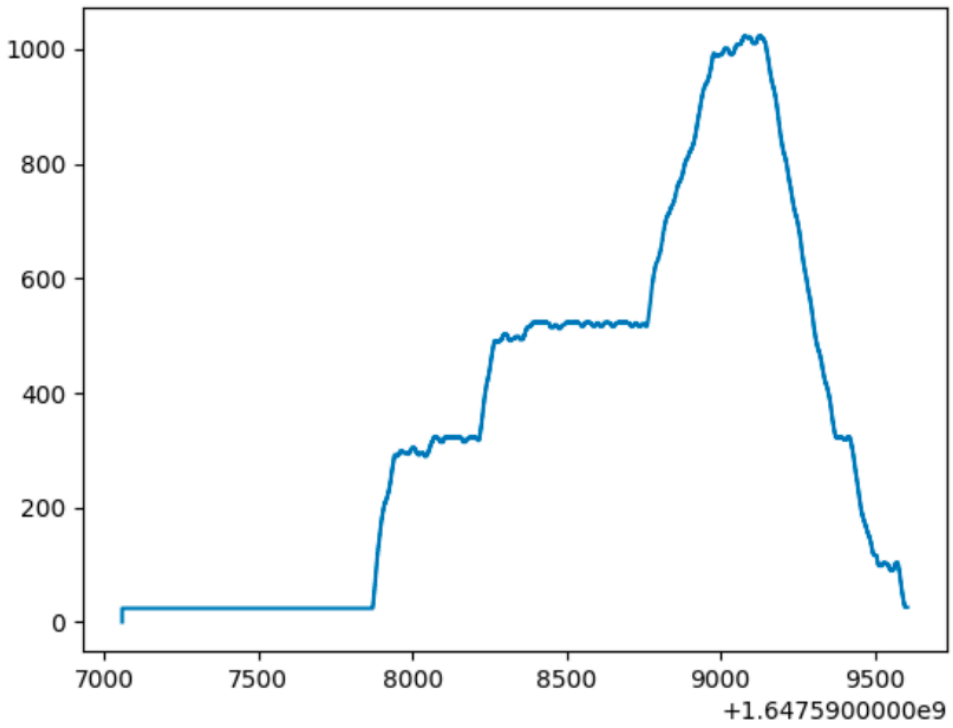


Рисунок 1 – Пример построения графика полета самолета в зависимости высоты от времени.

Пользователю будет предоставлена возможность загружать по несколько файлов за одну итерацию, программа определит правильный порядок файлов и начнет обработку.

Программа проверяет каждый считанный пакет подсчетом контрольной суммы CRC8. В значение CRC вносит вклад каждый бит сообщения. То есть, если хотя бы один бит исходного сообщения изменится при передаче, контрольная сумма тоже изменится, причём существенно. Это большой плюс такой проверки, так как он позволяет однозначно определить, исказилось исходное сообщение при передаче или нет [3].

Программный продукт разрабатывается на языке Python [4]. Разработка программного кода проходит в IDE PyCharm в операционной системе Mac OS. Программа разрабатывается под операционные системы Windows и Mac OS [5].

ЛИТЕРАТУРА

1. 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.
2. Derek Hawkins. [электронный ресурс]. Режим доступа: https://derek-m-hawk.medium.com/almost-end-to-end-log-file-analysis-with-python-59d4cb30f930
3. Целостность данных, CRC. [электронный ресурс]. Режим доступа: https://alexgyver.ru/lessons/crc/
4. Python Documentation. [электронный ресурс]. Режим доступа: https://docs.python.org/3/
5. Model Oriented Approach for Industrial Software Development / P. D. Drobintsev, V. P. Kotlyarov, N. V. Voinov, I. V. Nikiforov // Modeling and Analysis of Information Systems. – 2015. – Vol. 22. – No 6. – P. 750-762. – DOI 10.18255/1818-1015-2015-6-750-762.

УДК 004.42

Голдынская Е.К. (4 курс бакалавриата),
Самочадина Т.Н., старший преподаватель,
Самочадин А.В., к.т.н., доцент

РАЗРАБОТКА АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ ОФОРМЛЕНИЯ ЭЛЕКТРОННЫХ ПАСПОРТОВ ТРАНСПОРТНЫХ СРЕДСТВ

Целью работы является создание автоматизированной системы оформления электронных паспортов (ЭП) транспортных средств (АС ЭПТС) (Рисунок 1), а также разработка алгоритма для переноса ЭП из зарубежных шаблонов ЭП, в шаблон, поддерживаемый в РФ. Такая система должна предоставлять возможность оформлять, аннулировать, править ЭП, отправляя запросы на эти действия в ГИБДД, а также формировать различные отчеты.

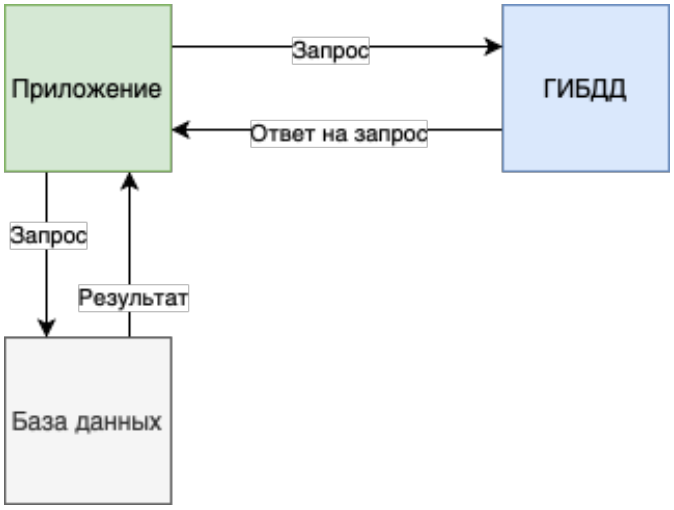


Рисунок 1 – Система ЭПТС

Особенности реализации системы ЭПТС:

Для удобного взаимодействия с системой ЭПТС реализовано web-приложение на java, через которое авторизованный пользователь может выполнять необходимые действия (Рисунок 2). Действия по формированию документов выполняются в автоматизированном режиме, а заполнение спецификаций, переноса данных между шаблонами и формирование отчетов выполняется автоматически.

Запросы в ГИБДД передаются в формате xml, ответы на эти запросы принимаются в том же формате. Действия с данными, не требующие обязательного подтверждения в ГИБДД, такие как формирование таблиц или сортировка данных, производятся посредством sql запросов в базу. При загрузке новых данных в базу и при формировании отчетов используются файлы формата xml.

Важной особенностью АС ЭПТС является алгоритм переноса данных из иностранного шаблона ЭП в российский. Его сложность заключается в разнообразии существующих шаблонов и отличиях между ними, поэтому алгоритм должен учитывать наличие лишних полей или наоборот отсутствие необходимых данных. Также, необходима проверка на корректность предоставляемого шаблона, который предстоит перенести.

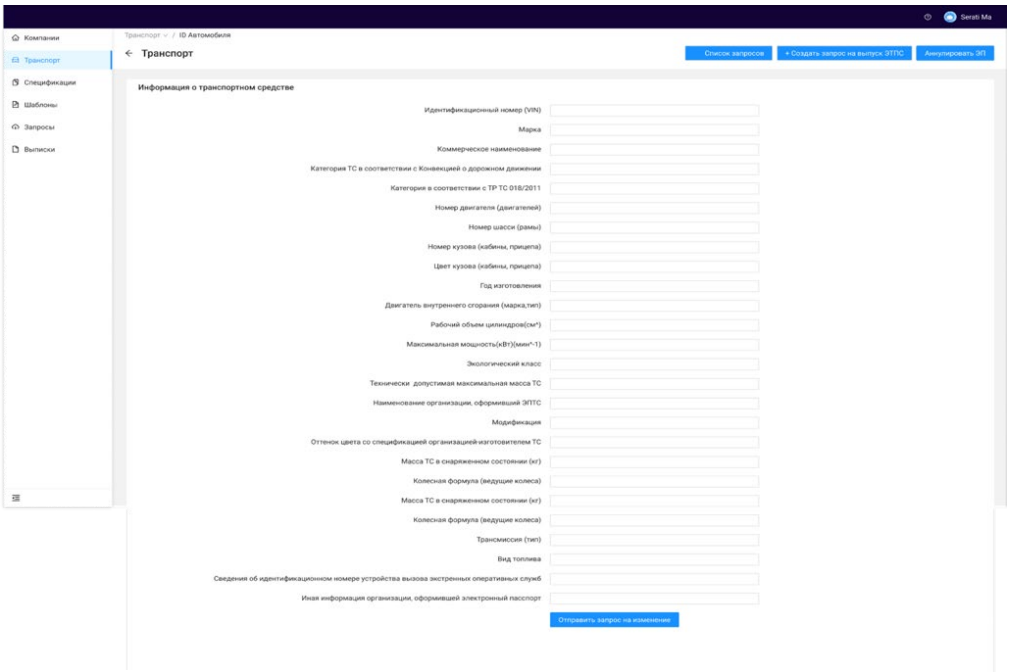


Рисунок 2 – Страница редактирования ЭПТС

ЛИТЕРАТУРА

1. Системы электронных паспортов. [Электронный ресурс]. Режим доступа: https://elpts.ru
2. Kundra S. Dureja A. Bhathagar R. The study of recent technologies used in E-passport system: IEEE Global Humanitarian Technology Conference - South Asia Satellite (GHTC-SAS) 26-27 Sept. 2014. [Электронный ресурс]. Режим доступа: https://ieeexplore.ieee.org/document/6967573
3. Jan L.Harrington Relational Database Design Clearly Explained, Elsevier : Morgan Kaufmann Publishers, 2002. [Электронный ресурс]. Режим доступа: https://books.google.ru/books?hl=ru&lr=&id=HHmBfmJRSL4C&oi=fnd&pg=PP2&dq=relational+database+design&ots=MgjMLZ80AR&sig=wetaqOE0ZlFyKANE50_GGGZ_Anc&redir_esc=y#v=onepage&q=relational%20database%20design&f=false
4. А.А.Соловьева Сравнение программного обеспечения для разработки пользовательского интерфейса и их прототипов: Наука без границ, 2020. [Электронный ресурс]. Режим доступа: https://cyberleninka.ru/article/n/sravnenie-programmnogo-obespecheniya-dlya-razrabotki-polzovatelskih-interfeysov-i-ih-prototipirovaniya
5. M. Rizwan Jameel Qureshi, F.Sabir A comparison of model view controller and model view presenter. [Электронный ресурс]. Режим доступа: https://arxiv.org/abs/1408.5786

УДК 004.051

Голубев А.В. (4 курс бакалавриата),
Григорьев Д.А., к.ф.-м.н., доцент

ИЗВЛЕЧЕНИЕ ТОРГОВЫХ РЕКОМЕНДАЦИЙ ИЗ АНАЛИТИЧЕСКИХ ОБЗОРОВ ВАЛЮТНОГО РЫНКА

Обработкой естественного языка (NLP) называется активно развивающаяся область науки, занимающаяся извлечением смысла и обучением на основании текстовых данных. Модели машинного обучения стремятся научиться правильному представлению входных данных для выполнения своей задачи. То, как модели учатся представлять слова с помощью NLP, в последние годы претерпело изменения, и в этой статье мы исследуем модели NLP для принятия финансовых решений. Мы фокусируемся на прямом применении NLP к финансовым рынкам: автоматизируем классификацию настроений текстового документа, чтобы делать быстрые и точные инвестиционные прогнозы, свободные от человеческих предубеждений. Процесс заключается в сборе аналитических обзоров на валютном рынке, содержащих, как прямую торговую рекомендацию эксперта, так и ее обоснование. После этого они разделяются и производится разметка текстовых обоснований с помощью извлеченных рекомендаций. Далее остается обучить модели машинного обучения для классификации аналитических обзоров на соответствие торговой рекомендации. Были исследованы след. модели: мешок слов (bag of words) [1], эмбеддинг (embedding) [2] и трансформер BERT [3].

Мешок слов - это упрощенное представление текста, часто используемое при обработке естественных языков, представляющее собой неупорядоченный набор слов (лексикон), входящих в обрабатываемый текст. Обработка текста заключается в поиске заранее готовых конструкций из мешка со словами в обрабатываемом тексте. Чем больше языковых конструкций наполняют мешок, тем в теории точнее работает метод. Наиболее популярным финансовым лексиконом является словарь Лофрана Макдональда [4]. На практике точность данная модели невелика, так как при классификации текста мы сильно полагаемся на слова, имеющие место в словаре.

Эмбеддинг-представления берут каждое слово в последовательности и проецируют их в многомерное пространство. Каждое измерение в пространстве является качественной характеристикой слова по критерию. Например, степень новизны слова, степень положительной окраски слова. Какие конкретно критерии будут входить в многомерное представление зачастую определяется на этапе обучения. Эмбеддинг сам по себе является просто представлением слова в многомерном пространстве и сам по себе не может классифицировать аналитические обзоры. Поэтому необходимо обучить модель, которая использует вложения в качестве входных данных и учится предсказывать настроения. В данной работе в качестве модели с эмбеддинг взята рекуррентная нейронная сеть долгой краткосрочной памяти LSTM архитектура которой кратко представлена на Рисунке 1.

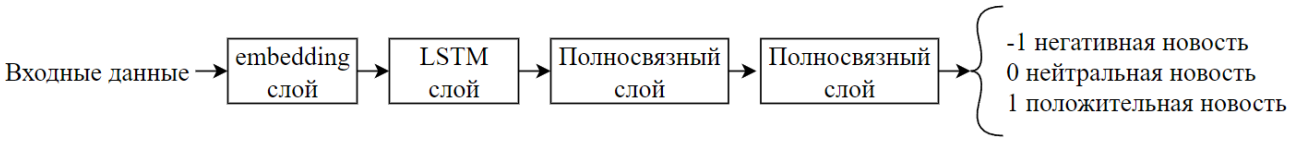


Рисунок 1 – Архитектура LSTM сети с использованием эмбеддинг

Другими более совершенным моделями NLP являются модели глубокого обучения. Одной из последних вех в этом развитии является BERT – это модель с открытым исходным кодом, которая побила несколько рекордов по тому, насколько хорошо она справляется с языковыми задачами. BERT строится на основе ряда идей, включая, помимо прочего, управляемое последовательное обучение, ELMo, ULMFiT, преобразователь OpenAI и трансферное обучение [5]. Данная модель работает по принципу обучения с учителем, что означает, что нам нужен размеченный набор данных.

В качестве примера были взяты данные с сайта Instaforex в количестве 819 аналитических обзоров по различным котировкам валют. Данные были размечены, и отправлены на вход описанным ранее NLP моделям. Результаты использования моделей представлены на Рисунке 2.

Рассмотрим пример положительного ожидания эксперта "… Until optimism about the US-China trade talks boosts AUD growth in the coming days, AUD is expected to dominate JPY for a while... As the price remains above 78.50 area with a daily close, the bullish pressure is expected to continue pushing higher in the coming days.". Ключевые слова 'AUD growth', 'AUD is expected to dominate' подтверждают рекомендацию покупать эту валюту.

Пример обзора с негативным ожиданием "… Therefore, even slight hints from the Federal Reserve about earlier-than-expected tapering stimulus programs will enable EUR/USD bears to reinforce selling pressure. The difference between the Fed's and the ECB rhetoric is going to be striking. The fact will exert strong pressure on EUR/USD. It makes sense to use such price jumps to open short positions." Здесь, по ключевым словам 'exert strong pressure', 'open short positions', видна рекомендация продавать валюту.

Пример нейтральной статьи в нашей выборке: "There is some lull in the market. EURUSD is consolidating below the resistance zone of 1.1200." – является нейтральной так как не содержит практических рекомендаций для принятия финансовых решений.

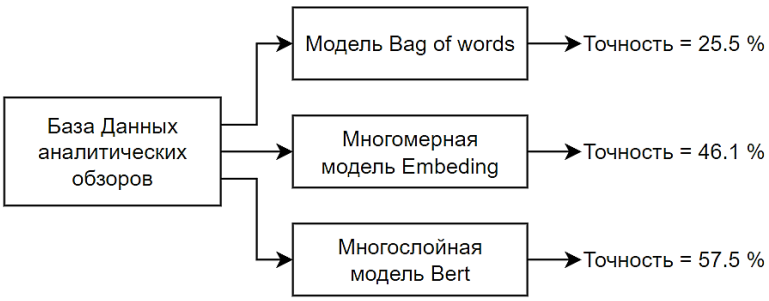


Рисунок 2 – Результаты предсказаний NLP модели

Как показывают результаты (Рисунок 2) модель Bert показывает более точные результаты классификации. Вероятно, это связано с тем, что использовалась предобученная модель BERT под названием finBERT [7], специализирующаяся на финансовых текстах. Собранные данные имеют специфику, например, одновременное наличие результатов фундаментального и технического анализа, что может приводить к ухудшению качества классификации. Дальнейшим направлением развития является применение полного NLP-конвейера для предобработки текста, расширение датасета и применение словарей с финансовой спецификой.

ЛИТЕРАТУРА

1. Мусаев, А. А., Григорьев Д.А. Формализованная постановка и краткий обзор технологий извлечения знаний из текстовых документов в задачах управления финансовыми // Техника и технология современных производств: Сборник статей II Всероссийской научно-практической конференции, Пензенский государственный университет, 2021. – С. 129-139.
2. Meng Y. et al. Spherical text embedding //Advances in Neural Information Processing Systems. – 2019. – Т. 32.
3. Choi H. et al. Evaluation of bert and albert sentence embedding performance on downstream nlp tasks // 2020 25th International Conference on Pattern Recognition (ICPR). – IEEE, 2021. – С. 5482-5487.
4. Loughran, T. and Mcdonald, B. When is a liability not a liability? Textual analysis, dictionaries, and 10-Ks. // The Journal of Finance 66. – 2011. – 35–65.
5. Tan C. et al. A survey on deep transfer learning // International conference on artificial neural networks. – Springer, Cham, 2018. – С. 270-279.
6. Мусаев А. А., Григорьев Д.А. Обзор современных технологий извлечения знаний из текстовых сообщений //Компьютерные исследования и моделирование. – 2021. – Т. 13. – №. 6. – С. 1291-1315.

УДК 004.42

Горячевская А.О. (4 курс бакалавриата),
Самочадина Т.Н., старший преподаватель,
Самочадин А.В., к.т.н., доцент

РАЗРАБОТКА МОБИЛЬНОГО ПРИЛОЖЕНИЯ ДЛЯ КОММУНИКАЦИИ И МОНИТОРИНГА В ТУРИСТИЧЕСКИХ ГРУППАХ

Целью разработки является создание мобильного приложения для контроля целостности некоторой туристической группы, а также взаимодействия между членами этой группы и отслеживания некоторых параметров группы, таких как количество устройств в группе, расстояние между ними, сила сигнала, время присоединения, и другие.

Такое приложение может быть применено в различных ситуациях – туристические походы в отдаленные местности, где может не быть доступа к внешней сотовой связи; экскурсионные маршруты, пешие или на транспорте, по городу и за городом; спасательные операции по поиску потерявшихся туристических групп на различных местностях, и во многих других.

Основными требованиями к разрабатываемому приложению являются следующие:

– Регистрации устройства с возможностью его идентификации;

– Передача информации между устройствами;

– Определение расстояния между устройствами;

– Уведомление устройств о некоторых событиях (например, потеря связи с некоторым устройством);

– Возможность использования приложения без доступа к внешней сети.

Также можно определить некоторые ограничения:

– Скорость в использовании. Время определения устройств не должно зависеть от количества людей. Допустим, для определения 1000 устройств система не должна тратить больше 1 минуты;

– Универсальность. Система должна одинаково работать и на больших потоках людей, и на малых; и на больших расстояниях, и на малых при достаточном количестве устройств;

– Удобство сопровождения. Система должна быть построена по модульному принципу. При обнаружении ошибки есть возможность ее быстро идентифицировать;

– Начальная стоимость. В качестве оборудования должны быть использованы устройства пользователей без необходимости добавления дополнительных модулей.

Для реализации приложения с учетом всех вышеописанных требований был выбран подход Bluetooth mesh-сетей на базе BLE (Bluetooth Low Energy). Технология BLE поддерживается практически всеми современными (в том числе и бюджетными) смартфонами, так что не требуется никаких дополнительных модулей для реализации приложения. Также BLE имеет высокую энергоэффективность, так что работа приложения не сильно скажется на производительности смартфона.

Принцип работы mesh-сети:

Имеется главное устройство, которое и запускает mesh-сеть (вручную или по некоторому расписанию). Пользователем этого устройства, например, может быть экскурсионный гид, или организатор похода. При запуске mesh-сети устройство, при помощи технологии Bluetooth, связывается со всеми доступными устройствами в радиусе видимости. Далее все подключенные устройства производят также Bluetooth-сканирование, и, если найденного устройства еще нет в создаваемой mesh-сети, оно подключается. Эти шаги повторяются до тех пор, пока все устройства не будут найдены, или в радиусе видимости не останется искомых устройств.

Каждое устройство называется «узлом» сети. В терминологии mesh-сети эти части называются элементами.

На Рисунке 1 показана модель mesh-сети, которая управляется при помощи основного устройства (координатора), который инициирует создание mesh-сети, и устанавливает соединение с ближайшими устройствами (маршрутизаторами). В данном случае создается минимальное количество связей для возможности отправки сообщений между двумя любыми устройствами через координатора. При прекращении работы какого-либо узла создаются альтернативные пути, и сеть продолжает работать в обычном виде. При этом можно осуществить создание сети, где связи осуществляются между всеми доступными устройствами.

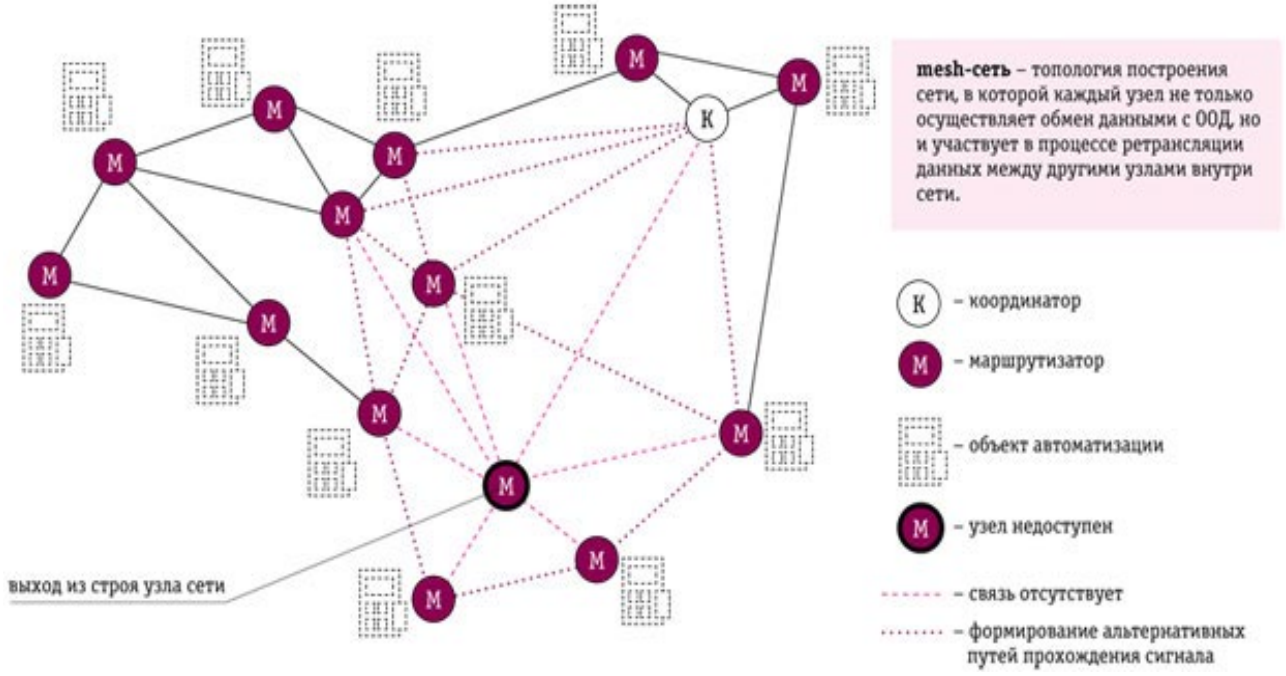


Рисунок 1 – Модель mesh-сети

Реализация приложения выполняется под платформу iOS на языке Swift, с использованием фреймворка CoreBluetooth, которое полностью поддерживает необходимое нам взаимодействие между устройствами.

ЛИТЕРАТУРА

1. Ad Hoc Mobile Wireless Networks, Prentice Hall Publishers / Chai Keong Toh, 2002.
2. Что такое сеть Bluetooth Mesh и как она работает? / Текст: электронный // Gadget-info.ru: Интернет-портал. – URL: https://ru.gadget-info.com/44019-what-is-bluetooth-mesh-networking-and-how-it-works – Текст: электронный.
3. Transferring Data Between Bluetooth Low Energy Devices, Apple: сайт. – URL: https://developer.apple.com/documentation/corebluetooth/transferring_data_between_bluetooth_low_energy_devices – Текст: электронный.
4. Мобильный mesh. Как создать сеть из смартфонов без единого роутера/ Евгений Зобнин - Текст: электронный // Xakep: Интернет-портал. – URL: https://xakep.ru/2017/08/15/mobile-mesh/ – Текст: электронный.

УДК 004

Донцов А.Д., Чикин Н.А. (2 курс магистратуры),
Дробинцев П.Д., к.т.н., доцент

РАЗРАБОТКА СЕРВИСА МЕНЕДЖМЕНТА МНОГОПОЛЬЗОВАТЕЛЬСКИХ МЕРОПРИЯТИЙ ДЛЯ КОЛЛЕКЦИОННЫХ КАРТОЧНЫХ ИГР

Несмотря на ограничения, возникшие в последнее время, для сбора людей, живые коллекционные карточные игры (ЖККИ, ККИ) остаются популярным и востребованным вариантом досуга. Одной из самых популярных коллекционных карточных игр сегодня является Magic: The Gathering (Магия). Данная игра имеет широкое сообщество и, до недавнего времени, активную поддержку соревнований и турниров со стороны разработчиков.

Несмотря на изменившиеся условия, а также развитие цифровых возможностей проведения мероприятий (развитие цифровой версии Magic: The Gathering), некоторые из многочисленных форматов этой игры, остаются недоступными в цифровом варианте. В частности, в цифровом формате не доступны многопользовательские форматы, а также большинство «вечных» форматов без ротации карт.

При проведении многопользовательских мероприятий, в отличие от классической игры «один на один», в партии принимает участие не менее трех, но не более четырех игроков. Похожие правила присутствуют в традиционной игре Маджонг [1].

На данный момент, крупнейшим проводимым мероприятием является турнир Marchesa [2], в России также проводятся крупные мероприятия с одновременным участием нескольких десятков человек.

В отличие от Маджонга, для многопользовательских мероприятий по Магии нет доступных сервисов, которые бы предоставляли генерацию игровых партий, а также хранили бы историю мероприятий и собирали аналитику. Все отчеты и аналитика собираются сообществом вручную, с помощью офисных программ, генерация партий также происходит в подобных программах, что приводит к очевидным проблемам как с удобством для пользователей, так и непосредственно с генерацией игровых партий, когда возникают ситуации неизменности оппонентов на протяжении нескольких раундов и т.д.

Таким образом, в результате анализа потребностей пользователей, было принято разработать веб-сервис, минимальная функциональность которого должна предоставлять возможности:

- Создание мероприятия,
- Добавление игроков на мероприятия,
- Формирование игровых партий, основываясь на системе очков и тайбрейков,
- Хранение истории мероприятий.

Также запланирована аналитика по колодам игроков, требующая интеграции с сервисом Moxfield для сбора данных о колодах игроков.

Для разработки сервиса был выбран Python с фреймворком FastAPI [3] для разработки сервера. Fast API - быстрый асинхронный фреймворк, позволяющий лаконично и быстро реализовать веб-сервер с автоматической генерацией документации в Swagger. Тесты, также, было решено реализовывать на языке Python с помощью фреймворка pytest [4].

Для реализации клиентской части было принято решение использовать сервис для быстрой генерации интерфейса Anvil [5], чтобы протестировать взаимодействие клиента и сервера, предоставить пользователям прототип для сбора данных. В качестве фреймворка для релизной версии клиента был выбран React [6]

В качестве базы данных была выбрана Mongo DB [7].

Для реализации необходимой функциональности были реализованы четыре независимых микросервиса. Архитектура приложения представлена на Рисунке 1.

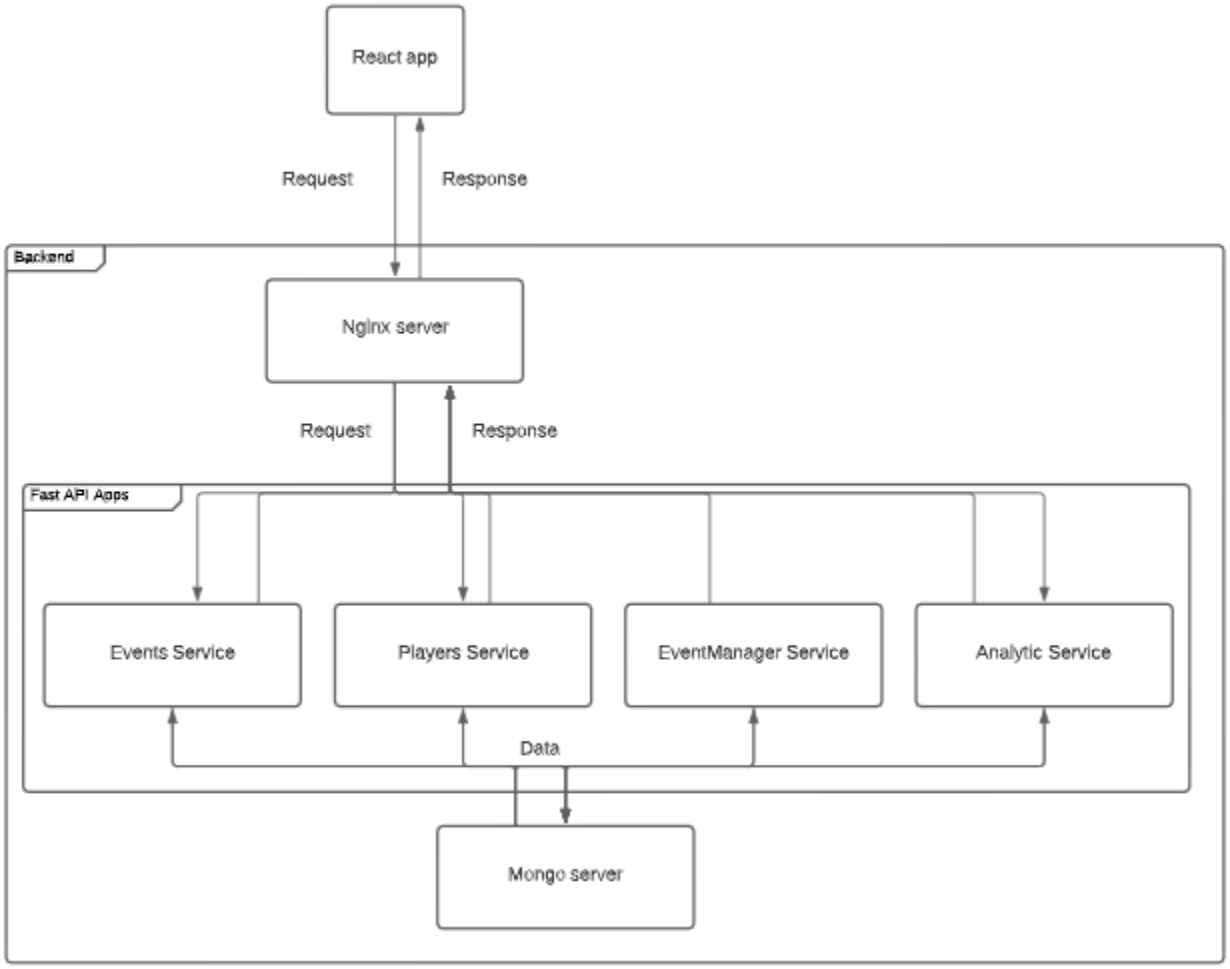


Рисунок 1 – Архитектура приложения.

Функциональность сервисов можно описать следующим образом:

- Event Service - базовая работа с мероприятиями: создание, изменение информации о мероприятии, удаление мероприятий,
- Player Service - получение информации об игроках на мероприятии,
- Event Manager Service - предоставляет основную функциональность работы с мероприятиями: генерация раундов, изменение очков, добавление игроков.

Для генерации игровых партий используется система очков и тайбрейков, которые, при изменении, с помощью формулы преобразуются в "скрытые очки". Данное решение было принято, чтобы жесткую систему "очки плюс тайбрейки" сгладить, что позволит учесть преимущества, которые предоставляет порядок хода и другие внеигровые факторы, а также разнообразить игроков, находящихся в середине турнирной сетки.

На данный момент реализована основная функциональность серверной части приложения и готовится прототип клиента. Наиболее неочевидной частью является формула, которая должна генерировать "скрытые очки", которые будут влиять на формирование турнирной сетки. Данная функциональность тестируется, с привлечением будущих пользователей, имеющих подтвержденный судейский статус игры Magic: The Gathering.

ЛИТЕРАТУРА

1. Cheng-Hung Lin, Yi-Chang Shan, and I-Chen Wu. Tournament framework for computer mahjong competitions. In 2011 International Conference on Technologies and Applications of Artificial Intelligence, pages 286–291, 2011.
2. Landon Liberator Nicholas Hammond. Marchesa about. https://www.monarch.cards/marchesa, 2022.

3. Fast api documentation. https://fastapi.tiangolo.com/
4. Pytest documentation https://docs.pytest.org/en/6.2.x/contents.html
5. Anvil overwiew. https://anvil.works/docs/overview.
6. React documentation. https://reactjs.org/docs/.
7. Mongodb documentation https://docs.mongodb.com/

УДК 004.023

Забелин К.В. (2 курс магистратуры),
Юсупова О.А., ассистент

МЕТОД ПОИСКА КОРРЕЛЯЦИИ МЕЖДУ НОВОСТНЫМИ СОБЫТИЯМИ И ОТКЛИКАМИ ПОЛЬЗОВАТЕЛЕЙ НА ОСНОВЕ ДАННЫХ ИЗ WIKIPEDIA

На данный момент в сети Интернет большое количество источников онлайн новостей [1] (порядка 18 тысяч источников в мире). Ежедневно миллионы людей читают новости онлайн. Однако, в большинстве случаев источники онлайн новостей не позволяют получать обратную связь от пользователей. Были опубликованы различные исследования, направленные на решение данной проблемы. В них использовались данные из различных источников для определения интереса людей к тому или иному событию.

В существующих исследованиях данные для подобного анализа брались из разных источников, например, из логов поисковых систем [2]. В нескольких исследованиях использовались логи количества посещений страниц онлайн энциклопедии – Wikipedia [3]. Wikipedia является подходящим источником, так как обладает большой аудиторией (374 млн. уникальных посетителей в год), а её содержимое не ограничено какой-то конкретной тематикой. Напротив, в Wikipedia обозреваются разные темы (55 млн. статей на разных языках).

В одном из исследований логи количества посещений страниц Wikipedia использовались для того, чтобы определить рост интереса пользователей Wikipedia к Олимпийским играм. Достигалось это за счет выявления резкого увеличения количества просмотров у статей смежной тематики. Однако, данное решение не позволяло определить, какое именно событие (или ряд событий) вызвало этот рост. Чтобы обнаружить данную связь, необходим ручной анализ статей, у которых был обнаружен резкий рост просмотров, и новостей, которые выходили в этот период.

Данная работа призвана закрыть этот пробел. В ней представлен метод, который позволяет автоматически определить резкий рост количества посещений страниц Wikipedia, а также определить, с чем был связан обнаруженный рост просмотров.

В качестве входных данных метода используется название статьи в Wikipedia и временной промежуток, в котором будет осуществляться поиск. В основе данного метода лежит следующий алгоритм:

Шаг 1: получение данных о количестве просмотров указанной статьи,
Шаг 2: выделение ключевых слов этой статьи,
Шаг 3: выделение промежутков возрастания с резким увеличением количества просмотров,
Шаг 4: определение списка дат, в которые могли произойти события, которые на это повлияли,
Шаг 5: получение списка новостных статей за данные даты,
Шаг 6: выделение набора ключевых слов из каждой новостной статьи,
Шаг 7: формирование списка итоговых новостей, которые могли потенциально повлиять на рост интереса к указанной тематике; выбираются статьи, в которых ключевые слова совпадают со списком ключевых слов статьи из Wikipedia (полученных на шаге 2).

На базе данного алгоритма было построено программное решение. Его архитектура схематично представлена на Рисунке 1.

Для получения данных о количестве просмотров страниц была написана специальная программа на языке Go (Connector), которая в несколько потоков скачивает логи просмотра всех страниц Wikipedia. Эти данные сохраняются в ElasticSearch в виде набора документов. Каждый из документов содержит в себе название статьи, количество просмотров данной статьи за час, дату и время. Полученные данные могут быть визуализированы с помощью инструмента Kibana, который легко интегрируется с ElasticSearch [4].

Далее подключается программа, реализованная на языке Python [5], которая собирает все необходимые данные о количестве просмотров указанной статьи из ElasticSearch, скачивает новостные статьи и ищет между этими данными корреляции. Для ускорения работы алгоритма, скачивание новостей и анализ ключевых слов распределяется между несколькими машинами в Spark кластере [6].

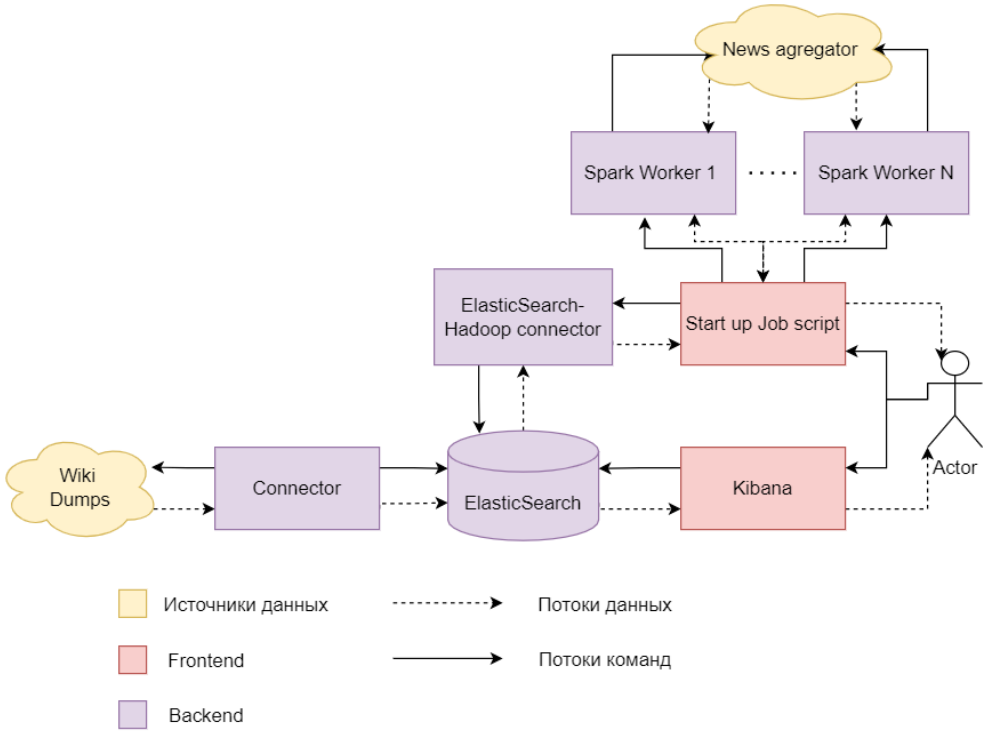


Рисунок 1 – Архитектура представляемого решения

Результаты тестирования данного решения представлены в Таблице 1.

Таблица 1 – Результаты тестирования

Название статьи в Wikipedia	**Количество обработанных логов, шт**	**Количество обработанных новостей, шт**	**Количество найденных подходящих новостей, шт**	**Время, с**
Facebook	8752	1646	8	31
Google	8587	2197	2	32
Netflix	8749	3256	2	46

Ручной анализ такого объема данных требует человеческих и временных ресурсов. Однако, из полученных результатов можно сделать вывод, что представленное решение позволяет автоматизировать ручную работу и получить результат за приемлемое время: около одной минуты при анализе временного промежутка в год.

ЛИТЕРАТУРА

1. В. К. Сычев, Ю. В. Ленькова, Е. А. Цветкова, И. В. Никифоров, 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.

2. B. Bardak and M. Tan, Prediction of influenza outbreaks by integrating Wikipedia article access logs and Google flu trend data // IEEE 15th International Conference on Bioinformatics and Bioengineering (BIBE), 2015, pp. 1-6, DOI 10.1109/BIBE.2015.7367640.
3. Y. Sun, Y. Tao, G. Yang and H. Lin, Visitpedia: Wiki Article Visit Log Visualization for Event Exploration // International Conference on Computer-Aided Design and Computer Graphics, 2013, pp. 282-289, DOI 10.1109/CADGraphics.2013.44.
4. Y. Gupta, “Chapter 1: An Introduction to Kibana” in Kibana Essentials. Birmingham, UK: Packt Publishing, 2015, p.1.
5. С. Э. Сараджишвили, В. В. Леонтьев, Н. В. Воинов, Введение в обработку изображений на языке Python // 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.
6. 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.

УДК 004.42

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

РАЗРАБОТКА ПРИЛОЖЕНИЯ ДЛЯ ИНФОРМИРОВАНИЯ О МЕСТАХ ПОВЫШЕННОЙ ОПАСНОСТИ ЗАРАЖЕНИЯ ТУБЕРКУЛЕЗОМ

Целью работы является исследование зависимости различных факторов окружающей среды инфицированных туберкулезом и создание на основе анализа входных данных мобильного приложения, информирующего пользователя о местах с высоким риском заражения и прокладывающего безопасный маршрут с минимальным риском заражения. Приложение должно включать пользовательские и административные функции для двух версий: мобильное и веб приложения.

Функциональность:

1. Формирование карты, содержащей информацию о местах с повышенным риском заражения туберкулезом (создание, редактирование, удаление);
2. Формирование наиболее безопасных (в смысле вероятности заражения) пешеходных маршрутов;
3. Оповещение пользователя о нахождении рядом с опасным местом;
4. Оповещение пользователя о необходимых мерах защиты (маска, обработка санитайзером и т.д.);
5. Добавление администратором в веб-версии входных данных, без предварительного подсчета коэффициента корреляции;
6. Возможность добавления опасных мест проверенными пользователями с последующей модерацией администратора;
7. Верификация администратором в мобильной версии предложений от проверенных пользователей.

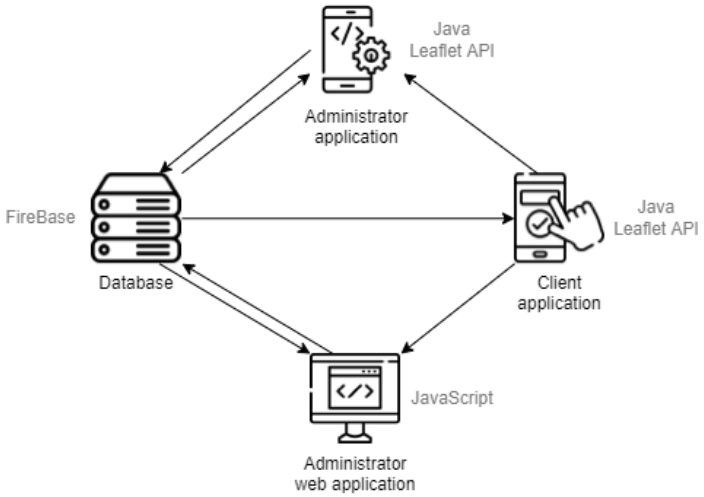


Рисунок 1 – Схема реализации приложения с указанием инструментов.

Особенности реализации приложения:
Мобильные элементы реализуются на Java в Android Studio с использованием open-source API Leaflet. Для реализации системы мониторинга не подходят популярные API, так как правила их использования запрещают применение к подобного рода приложениям. В связи с чем было принято решение обратиться к open-source решениям и был найден вариант в виде Leaflet API – базовая библиотека карт с большим количеством плагинов для расширения функциональности, что обеспечит наилучшую оптимизацию мобильного приложения.

Веб-приложение представляет собой one-page сайт, реализованный на JavaScript. Основная задача этой части приложения заключается в предоставлении возможности вводить входные данные в базу данных с ПК и без предварительного подсчета коэффициента корреляции. После добавления данных мобильное приложение пересчитывает коэффициенты и обновляет карту. Таким образом все вычисления «спрятаны», что минимизирует возможность ошибки подсчетов.

ЛИТЕРАТУРА

1. 3 open-source alternatives to Google Maps API [Электронный ресурс]. Режим доступа: https://opensource.com/life/15/11/getting-started-web-mapping
2. Теоретический материал: корреляционный анализ. [Электронный ресурс]. Режим доступа: https://e.vyatsu.ru/pluginfile.php/462616/mod_resource/content/3/Теоретический%20материал_корреляционный%20анализ.pdf
3. Leaflet API reference [Электронный ресурс]. Режим доступа: https://leafletjs.com/SlavaUkraini/reference.html
4. BSD licenses [Электронный ресурс]. Режим доступа: https://en.wikipedia.org/wiki/BSD_licenses

УДК 004.045

Калимуллина А. А., Кораблев Р. В. (2 курс бакалавриата),
Эйзенах Д. С. (1 курс аспирантуры),
Маслаков А. П., старший преподаватель

РАЗРАБОТКА ПЛАТФОРМЫ ДЛЯ ОНЛАЙН ТРЕНИРОВОК

Дистанционные фитнес-тренировки как деятельность существуют уже не первый год, однако актуальность обрели в период пандемии Covid-19. Тренировки на дому стали единственным вариантом для поддержания тела в форме, поэтому многие фитнес-клубы и персональные тренеры перешли на удаленную работу в период карантина, чтобы не потерять текущих клиентов.

Целью работы является создание веб-приложения, а также клиента под платформы iOS и Android, которые позволят клиентам заниматься фитнесом, не выходя из дома, а тренерам найти работу и обрести новых клиентов.

В приложении предусмотрена следующая функциональность:

– личный кабинет тренера с возможностью изменения информации на платформе
– покупка оптимального тарифа для пользователя
– добавление новых тренеров через панель администратора
– общение с персональным тренером через мессенджер
– отслеживание прогресса клиентов

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

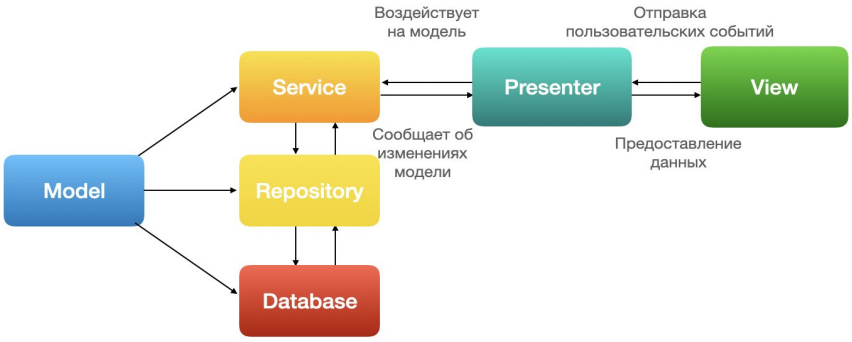


Рисунок 1 – Архитектура платформы

В качестве шаблона проектирования для разработки выбран MVP (Model-View-Presenter). Пользовательским интерфейсом является View, который отображает данные из модели. Presenter обрабатывает пользовательские команды и делегирует работу другим модулям системы. Model работает с данными, проводит вычисления и руководит всеми бизнес-процессами.

Для реализации фронтенда выбран фреймворк Vaadin [1] из-за кроссплатформенной поддержки. На Рисунке 2 приведена общая схема работы приложения на различных платформах. Vaadin специально ориентирован на доступность UX, позволяет программисту создавать настраиваемое веб-приложение, которое запускается как обычное Spring Boot приложение, без каких-либо дополнительных действий. Vaadin Flow — это фактическая часть инфраструктуры Java платформы Vaadin, которая заботится о взаимодействии клиент-сервер, и маршрутизации.

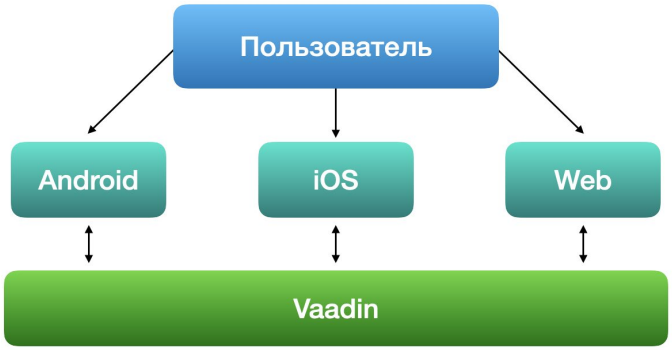


Рисунок 2 – Общая схема работы приложения

Для решения используем: Java 17 [2], сборщик Maven, Spring Boot, Vaadin, Spring Data. Авторизация и валидация выполнены силами Spring Security [3] и JSON Web Token (JWT). В качестве системы хранения была выбрана база данных PostgreSQL, для взаимодействия с которой используются JPA репозитории.

Данные о тренировках поступают из базы данных, которая заполняется тренером с помощью веб-интерфейса. У каждого видео есть определенный тег (категория), которая впоследствии используется для распределения видео по тарифам. В панели для администратора сайта имеется возможность добавлять новых тренеров, с целью увеличения количества производителей контента на сайте.

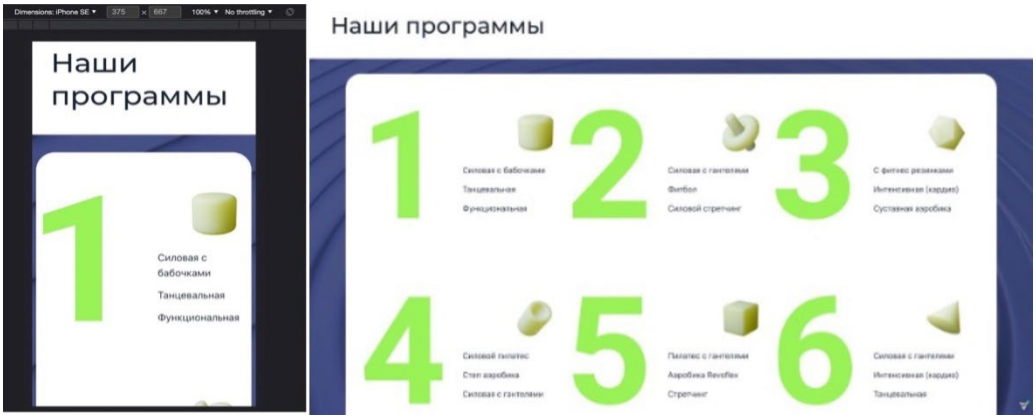


Рисунок 3 – Интерфейс страницы тренировок

Вся функциональность программного продукта была реализована и протестирована в работе, на Рисунке 3 можно увидеть пример разработанного интерфейса. Основная цель создания платформы была достигнута. У пользователей появилась возможность тренироваться с профессиональными тренерами в дистанционной форме. Дальнейшая работа предполагает, в частности, разработку и добавление рекомендательной системы для пользователей.

ЛИТЕРАТУРА

1. «Vaadin» [В Интернете]. Available: https://vaadin.com. [Дата обращения: 10 02 2022].
2. H. Schildt, Java - The Complete Reference, Eleventh Edition.
3. S. Security. [В Интернете]. Available: https://spring.io/projects/spring-security. [Дата обращения: 25 02 2022].

УДК 621.319

Каравашкин Л.А., Шкригунов А.Е. (2 курс магистратуры),
Молодяков С.А., д.т.н., доцент

ПРИМЕНЕНИЕ МЕТОДА АУДИО ОТПЕЧАТКА ДЛЯ ПОСТРОЕНИЯ МУЗЫКАЛЬНЫХ РЕКОМЕНДАТЕЛЬНЫХ СИСТЕМ

В современных аудио сервисах функции автоматического составления плейлистов, поиск похожей музыки и подбор музыки на основе вкуса пользователя становятся всё популярнее.

Существующие подходы включают в себя две категории: подходы на основе метаданных и подходы на основе анализа музыки. Цель данной статьи состоит в том, чтобы рассмотреть применение метода аудио отпечатков в системах на основе метаданных, тем самым объединяя эти два подхода. Предлагаемый подход заключается в использовании как минимум двух модулей для построения музыкальной рекомендательной системы.

Первый модуль основан на методе получения аудио отпечатков. На входе модуля – аудио файл всего произведения или его фрагмента. На выходе модуля – список аудио записей, схожих по звуковому содержанию. Для каждой аудио записи из списка указан коэффициент сходства. Алгоритм анализа сходства основан на алгоритме, описанном в статье “An Industrial Strength Audio Search Algorithm” [1]. Автор статьи применяет этот алгоритм для определения заранее проанализированной музыки через микрофон. Мы изменили алгоритм, чтобы можно

было распознавать не только заранее добавленную музыку, но и их обработанные версии. Для этого мы дополнительно пропустили входной файл через нейронную сеть Spleeter [2], основанную на базе данных musdb18 [3]. Эта нейронная сеть способна отделять мелодию от голоса. На Рисунке 1 представлен алгоритм получения похожих аудиозаписей.

Второй модуль отвечает за хранение метаданных музыкальных произведений, а также за составление и анализ связей между ними. Данный модуль используется для составления рекомендаций. Передавая ему на вход массив идентификаторов музыки, для которой нужно подобрать рекомендации, мы получаем на выходе список музыки, полученный на основе метаданных и информации о сходстве, предоставленной первым модулем. На Рисунке 2 представлен алгоритм, по которому происходит генерация рекомендаций, в нём участвует 2 параметра, отвечающих за пороговые значения коэффициента сходства, получаемого из первого модуля. K1 обозначает значение, начиная с которого две аудиозаписи считаются одинаковыми, а K2 - схожими. K2 должен быть меньше K1, при этом оба параметра должны быть в диапазоне от 0 до 1. Эти параметры применяются для того, чтобы использовать схожие аудиозаписи в подборе рекомендаций, но при этом будем удалять из результатов ту музыку, которая имеет высокий коэффициент сходства с входной. В ходе тестирования первого модуля были подобраны следующие значения: 0.2 для K1 и 0.05 для K2

На Рисунке 3 представлен граф, на котором присутствует две различных системы со связями, построенными на основе метаданных (similarity). Вершины 3 и 6 представляют собой разные версии одной и той же музыки. Первый модуль рассчитал коэффициент сходства, после чего он был внесён в граф и отображен в виде связи likeness. Теперь, при подаче на вход вышеприведенного алгоритма третьей вершины - на выходе будут все вершины, кроме 3 и 6. Шестая вершина не попадёт в выходной список потому, что у неё высокий коэффициент сходства с входной вершиной. Таким образом, имея на входе музыку из "синей" системы - получаем на выходе музыку из обеих систем, что и было целью данного алгоритма.

Рисунок 1 – Алгоритм получения списка похожей музыки

Рисунок 2 – Предлагаемый алгоритм составления рекомендаций

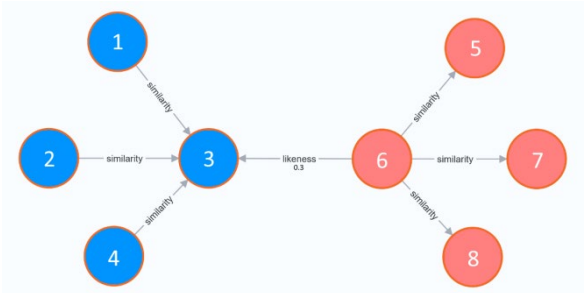


Рисунок 3 – Предлагаемый алгоритм составления рекомендаций

ЛИТЕРАТУРА

1. Wang A. L., Wang A. L., F T. F. B. An industrial-strength audio search algorithm // Proc. 4 TH Int. Conf. Music Inf. Retr. 2003.
2. Hennequin R. и др. Spleeter: a fast and efficient music source separation tool with pre-trained models // J. Open Source Softw. 2020. V. 5. № 50. P. 2154.
3. Rafii Z. и др. MUSDB18 - a corpus for music separation // 2017.

УДК 004.9

Корнилов Д. Ф. (4 курс бакалавриата),
Зайцев И. В., старший преподаватель

РАЗРАБОТКА ПРОТОТИПА СИСТЕМЫ КОНТРОЛЯ ЗНАНИЙ СТУДЕНТОВ С УЧЕТОМ ОСОБЕННОСТЕЙ ПРОЦЕССА ОБУЧЕНИЯ В РОССИЙСКИХ ТЕХНИЧЕСКИХ ВУЗАХ

В мире существует большое количество систем управления обучением (learning management systems), но их тестовые подсистемы обычно не учитывают специфики задач и реалий процесса обучения в российских технических вузах, и работа с такими системами далеко не всегда может быть достаточно удобной. В предлагаемом решении акцент сделан на функционале тестовой подсистемы, полезном для процесса оценки знаний при обучении на технических специальностях, и необходимом в российских университетах, но отсутствующем

в применяемых повсеместно зарубежных программных продуктах, таких как Moodle, Canvas и Open edX.

С целью облегчения внедрения и освоения предлагаемой системы на первом этапе предельно упрощено управление учетными записями студентов, на основе принятого в России фиксированного распределения обучающихся по группам в простом и понятном для преподавателей варианте (в отличие от вариативных групп в зарубежных системах управления курсами).

Вторая особенность — возможность пакетной загрузки тестовых заданий, посредством их импорта из файла в специальном формате. Данное решение позволит преподавателю хранить, быстро редактировать и добавлять задачи на портал, при этом не тратя много времени на работу с веб-интерфейсом, что позволит больше фокусироваться на процессе обучения студентов.

Третьей особенностью является реализация ввода множественного численного ответа, так как прототип системы разрабатывается для студентов технических ВУЗов. Это нововведение позволит преподавателю увеличить разнообразие задач с автоматической проверкой решения, а также создавать «многоэтапные» задания, востребованные в технических дисциплинах, что позволит в какой-то мере затруднить «списывание», что должно повысить точность результатов тестирования.

Следующая особенность предлагаемого решения заключается в гибко настраиваемом алгоритме распределения заданий, разделенных по уровню сложности. У обучающихся в ВУЗе разный уровень подготовки и владения предметом, поэтому в данном веб-сервисе предлагается более индивидуальный подход к развитию студента, что позволит каждому уверенно повышать свой уровень владения дисциплиной.

Заключительная особенность – возможность получить более подробные журналы попыток решения студентами задач для последующего анализа полученной информации вне разрабатываемой системы или для автоматического использования полученной информации внутри системы (в алгоритмах распределения заданий и вычисления итоговых баллов).

Разработка будет вестись по методологии Agile, так как необходимо гибко подходить к реализации системы с разными модулями, что облегчит подключение backend и frontend части приложений.

Backend часть приложения будет реализована с помощью фреймворка Spring Boot. Spring – фреймворк, для Java, делающий упор на скорость и простоту разработки, а также производительность ПО. Гибким библиотекам Spring доверяют разработчики по всему миру и является очень популярным и надежным решением для разработки серверной части приложений. В свою очередь, Spring Boot позволяет минимизировать время на конфигурацию приложения и получить многие библиотеки “из коробки”. Для импорта и экспорта тестов из документов Microsoft Office будет использована библиотека Apache POI, имеющая большую функциональность.

Frontend часть приложения будет реализована с помощью фреймворка React. React – фреймворк для JavaScript, позволяющий создавать простые представления для каждого состояния приложения, которые React будет эффективно обновлять и отображать только нужные компоненты при изменении данных. Данный фреймворк содержит большое количество библиотек, делающих разработку клиентской части приложения более продуктивной и надежной, а компоненты, написанные на React, позволяют эффективно их использовать повторно. Также будут использованы готовые компоненты MUI, которые предоставляет надежную, настраиваемую и доступную библиотеку базовых и продвинутых компонентов, позволяющую вам быстрее создавать свою систему проектирования и разрабатывать приложения React.

Для хранения данных будет использована база данных PostgreSQL. PostgreSQL — это мощная объектно-реляционная система баз данных с открытым исходным кодом с более чем 30-летним активным развитием, которая заслужила прочную репутацию за надежность, функциональность и производительность.

Для описания функциональности REST сервиса будет использована спецификация OpenAPI, а безопасность приложения (аутентификация и авторизация) будет обеспечена применением пакета Spring Security. Авторизация будет происходить с помощью JWT токенов.

Также планируется предоставить возможность развертывания программных слоев в Docker и настройка сообщения между ними с помощью сетевых протоколов.

ЛИТЕРАТУРА

1. Swagger specification [Электронный ресурс]. Режим доступа: https://swagger.io/docs/
2. Spring projects [Электронный ресурс]. Режим доступа: https://spring.io/projects
3. Блох, Джошуа Б70 Java: эффективное программирование, 3-е изд.: Пер. с англ. — СПб. ООО “Диалектика”, 2019. — 464 с.: ил. — Парал. тит. англ.
4. Moodle documentation [Электронный ресурс] Режим доступа: https://docs.moodle.org/
5. Docker documentation [Электронный ресурс] Режим доступа: https://docs.docker.com/

УДК 004.91

Крыжановская П. В. (4 курс бакалавриата),
Шошмина И. В., к.т.н., доцент

МОДУЛЬ ОЦЕНКИ ВОВЛЕЧЕННОСТИ ПРИ ДИСТАНЦИОННОМ ОБУЧЕНИИ

Целью работы является разработка средства статистической обработки входных данных, поступающих с онлайн курсов, для оценки вовлеченности студентов в дистанционное образование.

Работа выполнялась вместе с отделением открытого образования СПбПУ.

Отделением открытого образования подготовлен список оценок, на основе которых автор онлайн курса может делать выводы о вовлеченности и успеваемости студентов. В ходе моей работы оценки были разбиты по модулям (см. Рисунок 1).

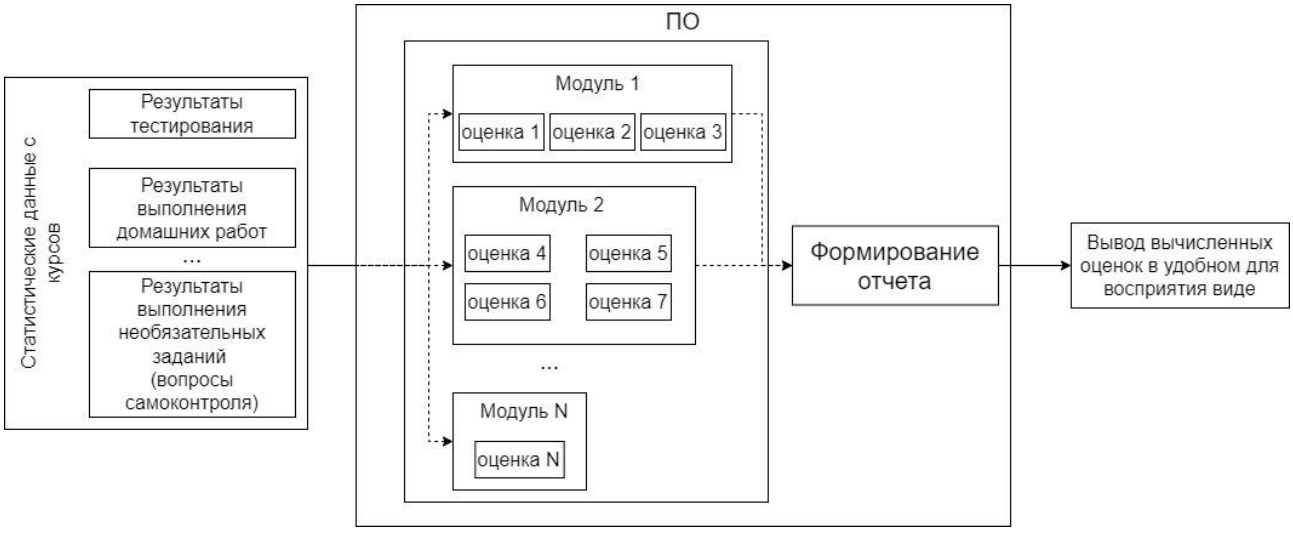


Рисунок 1 – Схема работы ПО.

Для расчета этих оценок выгружаются статистические данные с различных курсов и платформ (moodle, OpenEdu, Coursera) в формате csv. Перед вычислением оценок входной файл поддается предобработке для унификации структуры данных. На выходе будет представлен общий отчет с рассчитанными оценками в формате pdf.

Для удобства работы, будет разработано приложение с графическим интерфейсом, где пользователь сможет выбирать интересующие его оценки и разделы из курса.

ЛИТЕРАТУРА

1. Разработка инструментария для повышения вовлеченности обучающихся при дистанционном формате
2. Python 3.10.2 documentation [Электронный ресурс] https://docs.python.org/3/

УДК 004.031.43

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

ИНСТРУМЕНТАРИЙ ДЛЯ ИНТЕГРАЦИИ ГЕНЕРИРУЕМЫХ ТЕСТОВ НА COURSERA

Целью работы является создание инструментария, который позволит интегрировать генерируемые тесты в курсы платформы Coursera и таким образом создавать индивидуальные варианты под студента.

Эта задача появилась из-за того, что Coursera не предоставляет стандартных средств для создания тестов с динамически генерируемыми заданиями. В Coursera поддерживается динамический выбор вариантов тестов из заданного набора. Этот инструмент требует предварительной генерации вопросов и тестов, их содержащих, в больших объемах. Но даже в этом случае Coursera не позволяет создать больше 8 вариантов тестов, и обращение к этим вариантам происходит медленно, с потерями соединения.

Для решения этой задачи мною были изучены существующие программные средства Coursera:

– Плагины, небольшие JavaScript-приложения, которые встраиваются в платформу. Очень ограничены в объёме кода, производительности и функционале (установить безопасное соединение с внешним ресурсом невозможно).

– LTI (Learning Tools Interoperability) – спецификация которая стандартизирует взаимодействие между различными образовательными платформами и внешними инструментами. Coursera заявляет о поддержке стандартов LTI 1.1 и LTI 1.3.

Особенности реализации: реализация базируется на стандарте LTI 1.3, который определяет дополнительные стандарты безопасности и рекомендуется его разработчиками для новых систем, реализующих стандарт LTI. Разрабатывается инструментарий на языке JavaScript для программной платформы Node.js, используются также библиотеки и компоненты, поддерживаемые открытым сообществом, такие как fastify и handlebars.

Основной функционал инструментария – верификация входящих от образовательной платформы запросов, безопасное получение данных от платформы в условиях наличия посредника, участвующего в передаче этих данных (пользователь получающий тест), а также безопасная передача данных об оценках обратно платформе.

Результатом работы на данный момент является набор инструментов, решающий задачи валидации запросов от платформы Coursera и безопасного получения информации о пользователе и задании. Рисунок 1 демонстрирует процесс задействующий реализованный функционал. Первый и второй шаги - передача первичных данных запроса и инициализация сессии, если на этом этапе данные будут искажены посредником, то запрос будет отклонён одной из сторон; на третьем шаге данные передаются подписанными, если данные будут подменены, то при попытке верифицировать подпись подмена будет обнаружена и запрос будет отклонён, именно на этом этапе передаётся важная информация, такая как номер теста

и данные о студенте. Когда валидация запроса и верификация подписи пройдут успешно клиент получит запрашиваемый тест.

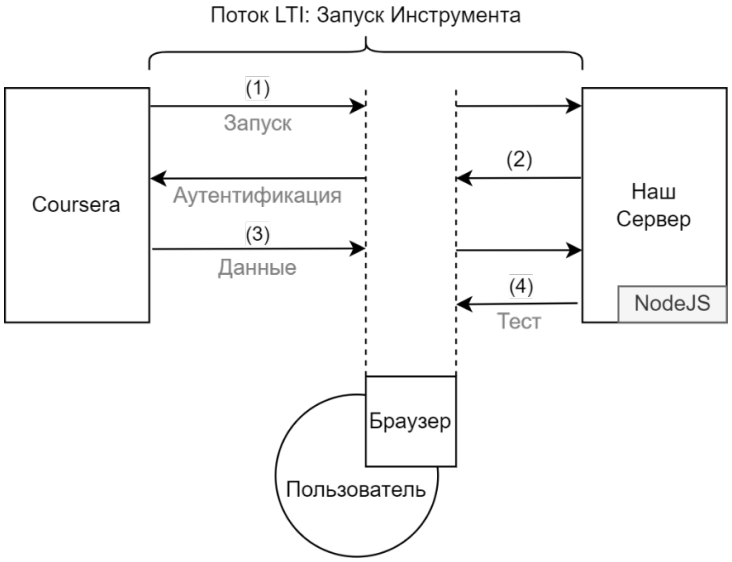


Рисунок 1 – Схема запуска инструмента (верификация запроса и передача данных о студенте и тесте)

ЛИТЕРАТУРА

1. Learning Tools Interoperability Core Specification [Электронный ресурс]. Режим доступа: https://www.imsglobal.org/spec/lti/v1p3/
2. IMS Security Framework [Электронный ресурс]. Режим доступа: https://www.imsglobal.org/spec/security/v1p0
3. LTI Developer Guide [Электронный ресурс]. Режим доступа: https://www.coursera.support/s/article/217416666-LTI-Developer-Guide?language=en_US
4. Node.js documentation [Электронный ресурс]. Режим доступа: https://nodejs.org/dist/latest-v16.x/docs/api/

УДК 004.415.2

Лоскутова М.А., Останина А.В. (4 курс бакалавриата),
Шошмина И.В., к.т.н., доцент

РАЗРАБОТКА СИСТЕМЫ ДЛЯ ПОДДЕРЖКИ ОБУЧАЮЩИХ ИГР

Целью работы является разработка библиотеки для поддержки обучающих игр и проектирование сервера. Приложение и разрабатываемые средства используют следующий стек технологий: Python используется для написания серверной части приложения, JavaScript для клиентской части приложения, Java, Kotlin используется для написания исходного кода сервисов и библиотеки, Spring Framework используется в качестве контейнера для внедрения зависимостей, Maven в качестве системы сборки проекта, технология Docker-контейнеров для развертывания приложений в изолированной системе, Git(система контроля версий), Github в качестве репозитория исходного кода.

Система будет использоваться при создании любых обучающих игр, должна обеспечивать администрирование игр и упрощать разработку игр в дальнейшем. Благодаря данной системы разработчикам игр не придется вручную настраивать администрирование и мониторинг активности пользователей, а также разработчики смогут добавлять в систему необходимые сервисы, которые впоследствии смогут использовать другие разработчики игр.

Игры, созданные с помощью данной платформы, могут использоваться как самостоятельные игры, так и встраиваться в курсы на платформах Moodle и OpenEdx.

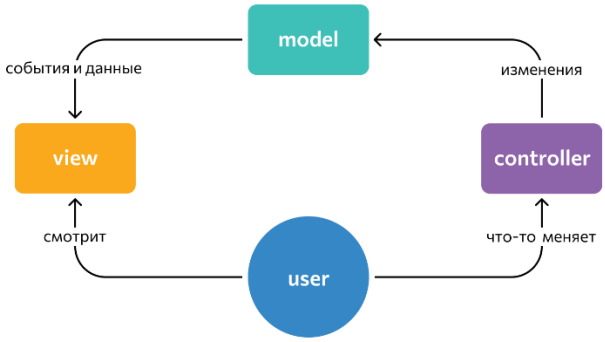


Рисунок 1 – Схема структуры MVC

Данная игровая система включает в себя интерфейс взаимодействия обучающей игры, приложений и сервисов. Взаимодействие удовлетворяет концепции Model-View-Controller (Рисунок 1).

Объединение приложений и сервисов выполняется с помощью единой игровой платформы. Единая игровая платформа содержит набор инструментов для работы с сервисами.

Функционал сервисов следующий: мониторинг активности пользователей, количество удачно завершенных игр, возможность для преподавателей оценивать сложность игры для студентов на основе статистики поражений и побед. Функционал сервисов слабо связан с другими сервисами и легко изменяем. В таком случае предпочтительно использование микросервисной архитектуры (Рисунок 2)

Сервисы и приложения работают независимо. Для независимой работы сервисов и приложений используется контейнерная архитектура. Контейнеры привязаны к конкретным портам хост-машины. Отправляя запрос на порт, сервис или приложение получает необходимую информацию о работе сервиса или приложения, если запрос был вызван сервисом, анализирует данные, обрабатывает и отправляет веб-клиенту или записывает во внутреннюю базу данных.

Рисунок 2 – Архитектура системы поддержки обучающих игр

В ходе работы было построенное решение системы для поддержки обучающих игр, проведен обзор и анализ средств для реализации системы. На данный момент написан прототип взаимодействия игры, использующей библиотеку, и сервиса, отправляющий ответ на запрос от библиотеки. Также написано приложение для решения курсовой по математической логике, которое будет использовано в качестве игры в системе для поддержки обучающих игр.

ЛИТЕРАТУРА

1. Ричардсон, К., 2019. Микросервисы. Паттерны разработки и рефакторинга. Питер.
2. Погружаемся в Docker: Dockerfile и коммуникация между контейнерами. Дата обращения: 14.03.2022 URL: https://blog.trukhin.com/погружаемся-в-docker-dockerfile-и-коммуникация - между-контейнерами-e3e79fb76bb7
3. Что такое MVC: рассказываем простыми словами Дата обращения: 25.04.2021 URL https://ru.hexlet.io/blog/posts/chto-takoe-mvc-rasskazyvaem-prostymi-slovami
4. Java Microservices: A Practical Guide Дата обращения: 12.01.2022 URL: https://www.marcobehler.com/guides/java-microservices-a-practical-guide#_deploying_and_testing_java_microservices
5. Craig, W., 2018. Spring in Action, Fifth Edition. Manning Publications.

УДК 004.415

Морева Е.И. (4 курс бакалавриата),
Петров А.В., старший преподаватель

MoBIE — ПЛАГИН ДЛЯ ПРОСМОТРА И АНАЛИЗА МУЛЬТИМОДАЛЬНЫХ ИЗОБРАЖЕНИЙ

Объем цифровых данных, генерируемых ежегодно в мире, растет экспоненциально, что связано с ростом количества записывающих датчиков, а также с увеличением емкости памяти. Основным следствием этой огромной числовой выборки реального мира является феномен мультимодальности. Для одной задачи сейчас используется множество датчиков с высокой чувствительностью и специфичностью, каждый из которых отображает определенный аспект источника. Эта мультимодальность записанных сигналов дает множество информации, позволяющей лучше представить и осмыслить воспринимаемый объект или явление. Вместе с тем она поднимает и ряд вопросов, касающихся оптимальной эксплуатации таких данных. В частности, существует реальная потребность в адаптированных инструментах просмотра и обработки биологических мультимодальных изображений.

Для решения этой проблемы был разработан активно развивающийся проект MoBIE.

MoBIE Fiji — это подключаемый модуль (плагин) Fiji (пакета для обработки изображений) с открытым исходным кодом, разрабатываемый сотрудниками Европейской молекулярно-биологической лабораторией (EMBL), позволяющий легко просматривать большие мультимодальные изображения и связанные с ними табличные данные.

Плагин предоставляет пользователю возможность удобного, плавного просмотра больших данных изображений; наложения многих типов изображений в одной системе координат; интерактивного исследования табличных данных вместе с изображениями, из которых они получены. Также плагин предоставляет возможности обмена данными или их фрагментами, выделения и разметки частей изображения, формирования сеточного представления изображений.

Быстрый темп инноваций в биологической визуализации и разнообразие ее приложений не позволили создать согласованный сообществом стандартизированный формат данных. MoBIE поддерживает чтение различных форматов данных, недоступных в аналогичных программах. В их числе файловый формат следующего поколения OME-Zarr, который

дополняет устоявшиеся открытые форматы (OME-TIFF и HDF5) и удовлетворяет большинству вариантов использования в биовизуализации.

Существует еще одна проблема мультимодальных данных, препятствующая их изучению - размер подобных данных (источник из нескольких изображений в сумме может достигать нескольких терабайт сжатых без потерь данных).

Благодаря “ленивой” загрузке в MoBIE обеспечивается интерактивный доступ к полному набору данных для всех, у кого есть стандартный компьютер и подключение к сети Интернет.

Приложение поддерживает различные источники данных: файловая система, удаленное хранилище Amazon S3, а также поддерживает чтение раздельно хранящихся данных — получение изображений с удаленного хранилища Amazon S3 и получение метаданных из GitHub репозитория, что позволяет пользователям MoBIE совместно изучать данные, использовать контроль версий для метаданных.

В плагине используется принцип проектов (сущностей, которые содержат всю информацию, необходимую для отображения данных). Каждый проект состоит из нескольких наборов данных, представляющих собой комбинацию изображений, которые могут отображаться вместе в одной и той же физической системе координат. Например, наборы данных могут быть изображениями из разных образцов, моментов времени или совершенно отдельных экспериментов. Плагин позволят не только просматривать, но и создавать собственные проекты.

Разработка плагина ведется по методологии FDD (Feature driven development), представляющей собой попытку объединения наиболее признанных в индустрии разработки программного обеспечения методик, принимающих за основу важную для заказчика функциональность (свойства) разрабатываемого программного обеспечения, и являющейся одной из гибких методологий разработки (Agile).

MoBIE реализуется на объектно-ориентированном языке программирования Java как плагин для пакета обработки изображений Fiji - дистрибутива ImageJ (проекта, упрощающего методы анализа изображений, включая обработку изображений, колокализацию, деконволюцию, регистрацию, сегментацию, отслеживание, визуализацию и многое другое).

Для реализации используется BigDataViewer — браузер для последовательностей изображений с несколькими представлениями размером в терабайт. BigDataViewer был разработан с учетом данных многоракурсной световой микроскопии, поставляется с настраиваемым форматом данных, оптимизированным для быстрого произвольного доступа к очень большим наборам данных, что позволяет просматривать любое место в многотерабайтной записи за доли секунды.

В данный момент реализуется поддержка применения моделей глубокого обучения BioImage Model Zoo (репозитория предобученных моделей искусственного интеллекта), что позволит проводить анализ изображений методами машинного обучения для широкого спектра научных задач.

MoBIE является передовым, удобным и универсальным плагином и содержит все необходимые инструменты для просмотра и анализа мультимодальных изображений, поскольку разрабатывается совместно с и для научного сообщества.

ЛИТЕРАТУРА

1. MoBIE: MultiModal Big Image Data Sharing and Exploration. [Электронный ресурс]. Режим доступа: https://mobie.github.io/tutorials/explore_a_project.html
2. Whole-body integration of gene expression and single-cell morphology // Cell: журнал. — Cell Press, 2021. — Т. 184, №18. — С. 4819-4837
3. Feature driven development. [Электронный ресурс]. Режим доступа: http://www.featuredrivendevelopment.com/
4. BioImage Model Zoo. [Электронный ресурс]. Режим доступа: https://bioimage.io/#/

5. Image processing & analysis in Java. [Электронный ресурс]. Режим доступа: https://imagej.nih.gov/ij/
6. Fiji is an image processing package. [Электронный ресурс]. Режим доступа: https://fiji.sc/
7. BigDataViewer. [Электронный ресурс]. Режим доступа: https://imagej.net/plugins/bdv/

УДК 004.9

Наумов А.А. (4 курс бакалавриата),
Андреев И.А., к.т.н.

РАЗРАБОТКА ВЕБ-СЕРВИСА УЧЕТА И АНАЛИЗА ФИНАНСОВ

Уже много тысячелетий люди пользуются деньгами как разменным и надежным ресурсом. Вместе с появлением первых денег человечеству понадобилось фиксировать долги и проведенные сделки. Были придуманы различные системы для записи такой информации. В наше время почти все хранится в цифровом виде в банках. Каждый день людям нужен хороший и достаточно простой инструмент для учета личных финансов, являющегося частью финансовой грамотности. Анализ расходов и доходов помогает двигаться в сторону финансовой независимости

Целью работы является разработка веб-сервиса учета и анализа финансов. Преимущества данного сервиса по сравнению с существующими решениями заключается в том, что акцент сделан на составление общей картины финансового поведения, на что и как уходят деньги, а не на детальный сбор статистики затрат. Постоянный ввод потраченной суммы до копеек и сортировка по множеству категорий приводят к тому, что пользователь теряет слишком много времени на детали, а по полученному множеству графиков и диаграмм не может провести анализ.

В предлагаемом решении по умолчанию будет выключен ввод копеек и будет предложен лаконичный список категорий. Таким образом внимание пользователя не будет сконцентрировано на том, что нужно ввести сумму с точностью до копейки, а потом еще выбрать наиболее подходящую категорию траты из длинного списка. Предложенный список будет стремиться снизить потенциальные пересечения. Для каждой траты будет предустановлен приоритет в виде численного показателя, что позволит на основе бюджета на месяц и трат по категориям составлять рекомендации, где стоит сократить расходы.

В качестве методологии разработки выступает Kanban доска на Trello. Для разработки архитектуры веб-сервиса были выбраны первые два уровня модели C4, System Context и Container Level. Данные диаграммы позволяют составить общее понимание архитектуры программного продукта. Сервис в основе имеет клиент-серверную архитектуру и будет состоять их трех частей: front-end, back-end и БД.

Клиентская часть будет React приложением, общающимся через API с back-end’ом. MUI предоставляет готовые React компоненты с гибкой настройкой как поведения, так и внешнего вида. Для написания модульных тестов будет использована библиотека Jest, поддерживаема разработчиками React и содержащая все необходимое для тестирования веб-приложений. Макет интерфейса разрабатывается при помощи веб-инструмента Figma.

Серверная часть представляет собой RESTful API поверх протокола HTTPS. Документирование спецификации API таким инструментом как OpenAPI. Языком программирования был выбран Go, так как он позволяет создавать быстрые и легковесные многопоточные решения. Gin веб-фреймворк заточен под высокую производительность, а GORM является полнофункциональной ORM. Аутентификация пользователей происходит при помощи JWT (JSON Web Token), зарекомендовавшего себя за счет простоты и надежности. Пароли хранятся в виде хэшей.

Решением для хранения пользовательских данных стала объектно-реляционная система управления базами данных PostgreSQL. Данная СУБД обеспечивает хорошую надежность и производительность, а также предоставляет широкий спектр функциональности.

Предполагается использование Docker и Docker Compose для обеспечения скорости разработки, тестирования и развертывания предлагаемого веб-сервиса. При помощи механизма создания сетей будет создано две сети для коммуникации клиента с сервером и сервера с БД.

ЛИТЕРАТУРА

1. gin package. [Электронный ресурс]. Режим доступа: https://pkg.go.dev/github.com/gin-gonic/gin
2. Justin Ellingwood. The Docker Ecosystem: Networking and Communication. [Электронный ресурс]. Режим доступа: https://www.digitalocean.com/community/tutorials/the-docker-ecosystem-networking-and-communication
3. JSON Web Tokens. [Электронный ресурс]. Режим доступа: https://jwt.io/
4. Алан А. А. Донован. Язык программирования Go/ Алан А. А. Донован, Брайан У. Керниган. – СПб: Диалектика, 2020. – 432 с.

УДК 004.415

Невзоров В.А. (2 курс магистратуры),
Шошмина И.В., к.т.н., доцент

РАЗРАБОТКА СЕРВИСА-ХРАНИЛИЩА РЕКЛАМНЫХ ПЛОЩАДОК

В работе рассматривается задача построения сервиса таргетированной рекламы. Сервис предоставляет инструменты для рекламодателя, которые могут создавать свои рекламные баннеры. А с другой стороны, к сервису подключаются различные сайты, мобильные приложения, на которых необходимо показывать эти рекламные баннеры. Подключаемые места размещения баннеров называются площадками.

До того, как я приступил к решению этой задачи, у моего работодателя существовало программное решение, которое использовалось для целей создания таргетированной рекламы. В существующем программном решении имелась проблема: для того, чтобы подключить новое место размещения баннеров, необходимо в программном коде изменить десятки констант и переменных, а также реализовать несколько дополнительных функций. Такой процесс отнимал у разработчика около 6 часов рабочего времени на одну новую площадку, а программный код превращался в сложно поддерживаемый, за счет огромных массивов констант. Кроме того, поскольку добавлять новые места размещения необходимо часто, а при существовавшем решении приходилось каждый раз привлекать высококвалифицированных специалистов, способных корректно редактировать программный код. Стоит заметить, что различные сервисы таргетированной рекламы — это узкоспециализированные разработки под требования конкретного бизнеса, поэтому готовых решений на рынке вовсе нет.

Исходя из вышеописанной проблемы, целью работы является разработка программного решения, которое будет хранить информацию о рекламных площадках и интегрировать его в существующую информационную систему, чтобы снизить затраты на добавление рекламных площадок.

Проанализировав текущие нагрузки на существующую систему таргетированной рекламы, к сервису-хранилищу мною были выдвинуты следующие требования:

- Возможность масштабируемости относительно количества рекламных площадок;
- Изменения информации о площадках должны применяться не дольше, чем 3 минуты;
- Выдерживать нагрузку 20 000 запросов в секунду на чтение и при этом отвечать не дольше, чем 50 мс.;
- Предусмотреть возможность масштабируемости решения относительно нагрузки;

- Предоставлять RPC (Remote Procedure Call) API для чтения, редактирования, создания и удаления рекламных площадок.

Особенности технической реализации:

Информация о площадке была разделена на 3 уровня:

- Publisher – хранит в себе общую информацию о площадке, такую как название, тип, качество рекламного трафика;
- Unit – это уникальное сочетание рекламного блока и платформы (Android, iOS, Web). Также в этом типе содержится уникальный идентификатор места показа. Каждый “Unit” имеет связь с одним родительским объектом “Publisher”;
- Slot – характеризует явное место показа рекламного баннера. Каждый “Slot” имеет связь с одним “Unit”.

На каждом из уровней Publisher, Unit, Slot имеется возможность определить сущность Config. Например, можно задать какие-то общие параметры для всего Publisher и переопределить другие параметры для конкретного Unit.

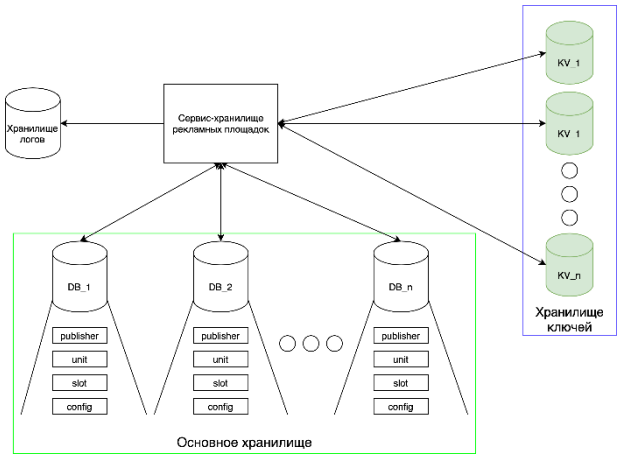


Рисунок 1 – Архитектура сервиса-хранилища

Архитектура сервиса-хранилища изображена на Рисунке 1. В качестве постоянного хранилища используется специальная база данных, которая была разработана внутри компании, ближайший публичный аналог – MySQL. Для отказоустойчивости в постоянном хранилище было развернуто 8 экземпляров баз данных. Сервис хранит данные в постоянном хранилище, “шардируя” [1] их по Publisher. То есть, все данные, которые привязаны к конкретному Publisher (Unit, Slot, Config) хранятся на одном физическом сервере. Таким образом, если один из 8 серверов отказывает, то теряется только часть площадок целиком, а все остальные – работают в штатном режиме.

В текущей схеме невозможно использование автоматического увеличения (auto increment) для идентификаторов, так как иначе в одних и тех же таблицах на разных серверах они будут дублироваться, что недопустимо. Для обеспечения уникальности идентификаторов используется хранилище на базе Memcached [2], в котором есть атомарная операция - increment.

Для того, чтобы удовлетворять требованию по времени ответа, каждые несколько минут сервис получает всю актуальную информацию из постоянного хранилища и строит хэш таблицу в оперативной памяти. Далее, все запросы в API за информацией о площадках будут возвращаться из полученной хэш таблицы.

Для отладки и сбора логов используется СУБД – ClickHouse [3]. В качестве основного языка разработки сервиса-хранилища используется – Golang. Сервис предоставляет API для системы таргетированой рекламы, в котором обмен сообщениями производится в бинарном формате, а для описания таких сообщение - используется Type Language [4].

Архитектура сервиса-хранилища позволяет запускать несколько экземпляров сервиса, так как вся информация хранится в базе данных – каждый запущенный экземпляр сервиса во время запуска – подгружает ее в оперативную память. Таким образом достигается горизонтальное масштабирование.

Сервис-хранилище был разработан и частично интегрирован в существующую систему, что позволило более активно расширять множество площадок. Теперь добавление новых площадок не требует участия квалифицированного специалиста, а время добавления одной площадки занимает не более 10 минут.

ЛИТЕРАТУРА

1. Sikha Bagui, Database Sharding: To Provide Fault Tolerance and Scalability of Big Data on the Cloud // International Journal of Cloud Applications and Computing: журнал. Апрель 2015, №5 С. 36-52
2. Memcached. [Электронный ресурс]. Режим доступа: https://memcached.org/
3. ClickHouse. [Электронный ресурс]. Режим доступа: https://clickhouse.com/docs/ru/
4. Telegram, TL Language. [Электронный ресурс]. Режим доступа: https://core.telegram.org/mtproto/TL

УДК 004.4

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

ПРИЛОЖЕНИЕ ДЛЯ АВТОМАТИЗАЦИИ ПЕРЕНОСА МУЗЫКИ МЕДЖУ СЕРВИСАМИ

Цель работы – разработать приложение, которое позволит пользователю переносить в автоматическом режиме музыкальную библиотеку большого объема между такими сервисами, как Spotify, Last.fm и др.

В качестве платформы, под которую будет разрабатываться приложение, была выбрана iOS, как одна из двух самых популярных мобильных операционных систем. Ввиду того, что стриминговые музыкальные сервисы в подавляющем большинстве случаев используются на мобильных устройствах.

В качестве основного фреймворка был выбран SwiftUI, поскольку, во-первых, он самый современный из существующих, и к тому же обеспечит возможность с минимальными затратами разработать приложение и под macOS.

На данный момент существует довольно много популярных сервисов стриминга музыки, таких как Spotify, Deezer, в России – Яндекс Музыка и Boom, и т.д. Наблюдается тенденция перехода от хранения музыки локально на мобильных устройствах к стриминговым сервисам, ввиду наличия доступного и быстрого интернета у большинства пользователей и удобства этих сервисов.

Ввиду этого существует потребность для многих пользователей быстро синхронизировать свою музыкальную библиотеку между сервисами, если они используют сразу несколько, или хотят перейти с одного на другой. Также такая функциональность востребована, поскольку главным преимуществом стриминговых сервисов являются алгоритмы анализа предпочтений пользователя для составления рекомендаций для него. Для этого пользователю требуется указать свои предпочтения.

Подобные приложения на iOS уже существуют, но их существенным недостатком является то, что для экспорта библиотеки пользователя они используют распознавание названий песен на скриншотах, а не API сервисов. Это создает значительные неудобства для пользователя, поскольку скриншот обычно вмещает 10–15 записей в списке, и процесс переноса библиотеки, количество песен в которой исчисляется сотнями, как это обычно бывает, будет очень долгим.

К тому же, при использовании такого способа могут возникнуть ошибки при распознавании текста на фотографиях, а также проблемы с поддержкой языков.

Приложение включает следующие сценарии использования:

- Пользователь может переносить свою музыкальную библиотеку между любыми двумя сервисами из поддерживаемых для того, чтобы использовать оба сервиса.
- Пользователь может экспортировать свою музыкальную библиотеку в файл. Это может потребоваться, в частности, для того, чтобы пользователь мог перенести свою музыку в сервис, который не поддерживается приложением, но поддерживает импорт из файла.
- Пользователь может перенести свою музыкальную библиотеку на сервер приложения, чтобы иметь возможность восстановления своей библиотеки независимо от музыкальных сервисов.

Система состоит из следующих компонентов:

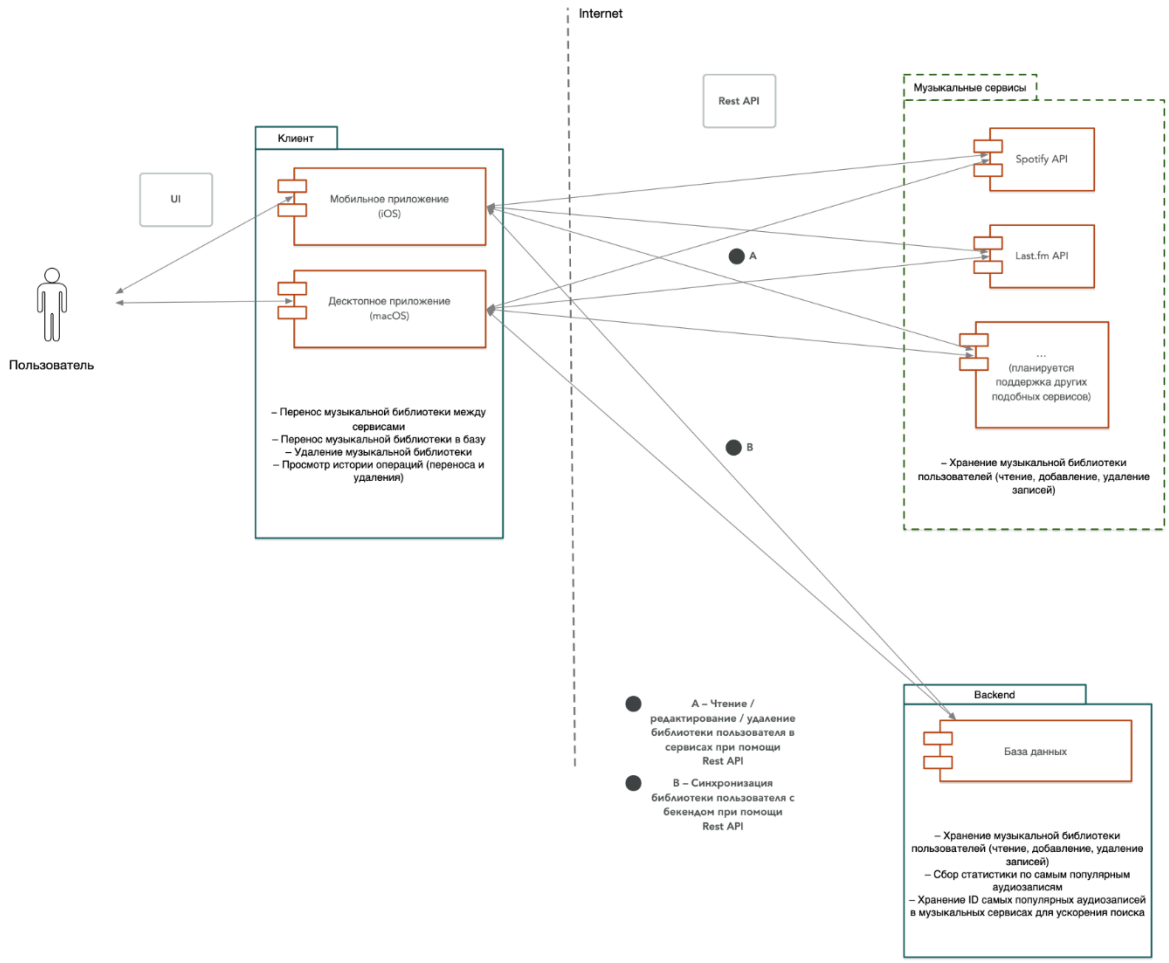


Рисунок 1 – компоненты системы

Основным преимуществом данного приложения перед конкурентами является использование API для переноса музыки. Это позволяет обеспечить большую скорость по сравнению с конкурентами. Также преимуществом приложения является возможность сихронизации музыкальной библиотеки пользователя с сервером, что облегчает перенос библиотеки между разными аккаунтами в одном сервисе, и предоставляет пользователю возможность использования резервного хранилища музыкальной библиотеки на случай возникновения проблем со стриминговыми сервисами.

ЛИТЕРАТУРА

1. Brian Mulloy. (2012) “Web API Design: Crafting Interfaces that Developers Love”
2. Apple Inc. (2021) “The Swift Programming Language (Swift 5.5)”

3. Antonio Bello, Audrey Tam, Bill Morefield, Sarah Reichelt (2018) “SwiftUI by Tutorials”
4. Eve Andersson, Philip Greenspun, and Andrew Grumet (2021) “Software Engineering for Internet Applications”

УДК 621.319

Погребной А. С, Алейников П. И. (2 курс магистратуры),
Молодяков С. А., д.т.н, доцент

РАЗРАБОТКА ПРИЛОЖЕНИЯ ПО ОБЪЕДИНЕНИЮ МЕССЕНДЖЕРОВ

Целью данной работы является разработка приложения по объединению некоторых существующих популярных мессенджеров, а также по возможности частей социальных сетей, таких как «Telegram», «ВКонтакте», «WhatsApp» и т.д. в единую систему с одинаковым пользовательским интерфейсом.

Существуют децентрализованные и централизованные решения по построению приложений. В случае децентрализованного решения осуществляет обмен информацией посредством мостов через различные сервисы. Для этой цели может использоваться, например, протокол Matrix [1]. В случае централизованного решения взаимодействие пользователей между собой обеспечивается через API различных сервисов, например, пользователь, авторизированный через мессенджер «Telegram», может обмениваться информацией через его сервер с другими участниками, какое клиентское приложение они бы не использовали. При этом пользователь может так же авторизироваться и в других сервисах получая возможность так же общаться через их API. Само API обычно распространяется свободно, как например «Telegram» [2] инструментарий которого активно используется в создании ботов.

В докладе рассматривается приложение по объединению мессенджеров, которое построено по централизованной схеме. Особенностью данного решения является централизация средств мгновенного обмена сообщениями, а также по возможности дополнительных решений сервисов по типу звонков, созданию групп, секретных чатов и т.д. с целью обхвата максимального количества аудитории. Преимуществом единой системы так же является поддержка кросс-сервисных функций, которые позволяют с легкость обеспечить поддержку взаимодействия между собой различных подключенных сервисов, таких как пересылка сообщений, встраивание видео файлов, массовая рассылка и других подобных возможностей.

Обобщенная схема функционирования приложения через сервера различных сервисов представлена на Рисунке 1. Для взаимодействия с сервером через API пишется по заготовленной схеме модуль для каждого мессенджера, в котором определяется список поддерживаемых им функциональных возможностей. Пользователь, пройдя авторизацию в мессенджере через его модуль, получает возможность обмена информацией через сервер с другими клиентскими приложениями. Также в программном продукте не планируется поддержка своего сервера с обработкой всей информации, что существенно снижает затраты на разработку и поддержку приложения, а сервисы мессенджеров при этом представляют собой отдельные совместимые между собой модули с перечнем доступного функционала. По этой же причине отпадает необходимость в дополнительной авторизации в самом приложении и хранению каких-либо сведений о пользователе. Таким образом не происходит передачи личной информации каким-то третьим лицам, а вся информация сохраняется исключительно только между пользователем и сервисом мессенджера.

Помимо основных возможностей мессенджеров, в данной системе можно так же обеспечить поддержку не поддерживаемых ими функциональности поверх существующего интерфейса. Например, встройку видеофайла через сервис «YouTube» и ему подобных на уровне отправки ссылки, которая в последствии обрабатывается самой системой с выдачей результатов, в данном случае видеофайла.

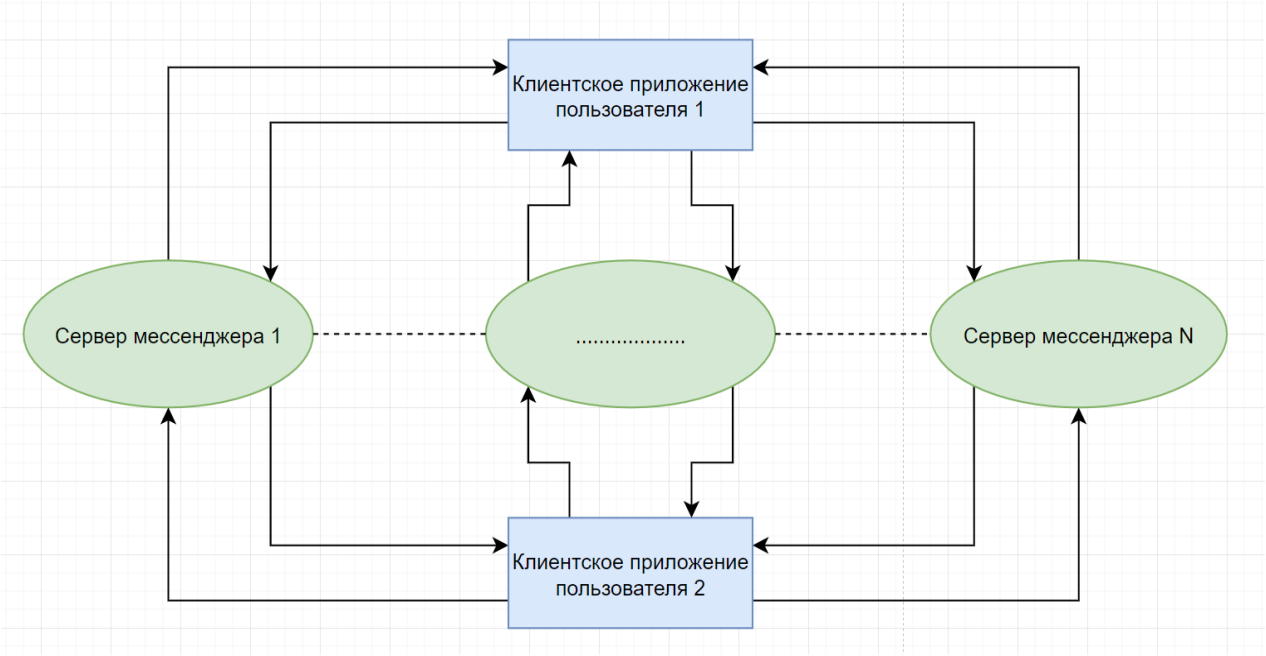


Рисунок 1 – Взаимодействие клинских приложений через сервера различных сервисов

В качестве основы для приложения был выбран современный фреймворк для создания нативных графических приложений использующий веб-технологии и библиотеку рендеринга Chromium и среду Node.js в качестве локально back-end'а [3] - Electron. А для написания самой графическо-интерфейсной Angular, который позволяет создавать сложные многокомпонентные веб-приложения.

ЛИТЕРАТУРА

1. Nathan Willis. "Matrix: a new specification for federated realtime chat". [Электронный ресурс]. Режим доступа: https://lwn.net/Articles/632572/
2. Telegram APIs. [Электронный ресурс]. Режим доступа https://core.telegram.org/#api-methods
3. "Electron Internals: Using Node as a Library". [Электронный ресурс]. Режим доступа: https://www.electronjs.org/blog/electron-internals-using-node-as-a-library

УДК 004.41

Пылаев Я.С., Голиков Г.Д. (4 курс бакалавриата),
Петров А.В., старший преподаватель

СЕРВИС ПО СОЗДАНИЮ ЭЛЕКТРОННЫХ КАРТ С ПОДДЕРЖКОЙ ТЕХНОЛОГИИ NFC ДЛЯ ЭЛЕКТРОННЫХ МОБИЛЬНЫХ КОШЕЛЬКОВ APPLE И GOOGLE WALLETS

В данной работе ставится задача разработки сервиса по созданию и конфигурации электронных карт с поддержкой технологии NFC, карт лояльности, билетов, проездных и сертификатов с QR-кодами для электронных кошельков Apple и Google Wallet, а также разработка Telegram-бота в качестве клиентского приложения.

Об актуальности темы работы можно судить по тому, как плотно цифровизация уже вошла в нашу повседневную жизнь. Использование электронных банковских карт для оплаты товаров, прикладывание штрих-кода электронного билета к турникету при входе в театр или проходе на платформу Аэроэкспресса является нормой нашего времени. При этом сервисы виртуальных картхолдеров (кошельков для карт) продолжают развивать свою функциональность, предоставляя бизнесу улучшенные каналы для коммуникации с пользователями за счет новых трендов в сфере электронных карт. В рамках научной работы мы исследуем технологию создания объектов хранения для электронных картхолдеров и

рассматриваем несколько сценариев взаимодействия с технологией как бизнеса, так и частного пользователя.

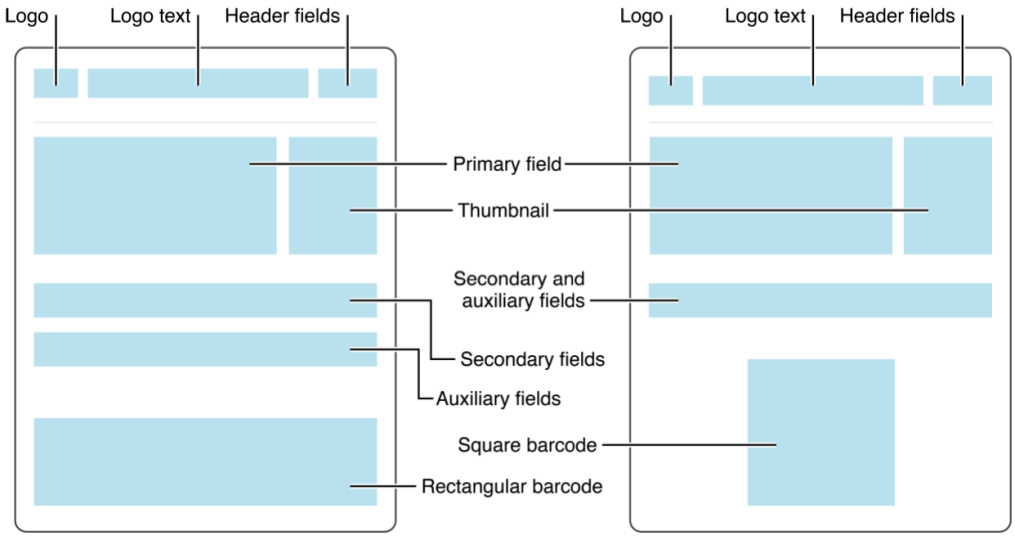


Рисунок 1 – Макет электронной карты

Далее представлен список функциональных требований, определяющих демоверсию разрабатываемого сервиса:

- Регистрация и авторизация в системе.
- Выбор типа создаваемой карты - посадочный талон, карта со штрих-кодом или с поддержкой NFC.
- Выбор электронного кошелька, для которого будет создана электронная карточка (Apple/Google Wallet).
- Конфигурирование информации и дизайна карты - ввод и загрузка параметров карты, например, логотип, цвет фона, данные о владельце или компании.
- Получение готовой электронной карты для добавления в сервис Apple/Google Wallet.
- Загрузка excel-таблицы с набором данных для создания сразу набора электронных карт и рассылка их на электронную почту пользователей, указанных в документе.
- Обновления данных электронной карты, созданной ранее в сервисе.

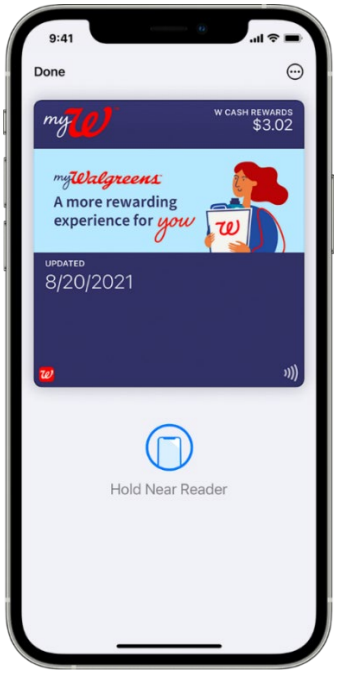


Рисунок 2 – Пример электронной NFC-карты в Apple Wallet

Для реализации описанной функциональности используется микросервисная архитектура, языки программирования Java с фреймворком Spring и JavaScript с программной платформой Node.js, база данных PostgreSQL для хранения информации об электронных картах пользователей.

Таким образом, основная задача работы – разработка сервиса по созданию электронных карт с поддержкой технологии NFC, включающий интеграции с Apple и Google API, а также следование современным процессам проектирования и разработки ПО.

ЛИТЕРАТУРА

1. Преимущества использование Apple Wallet, о которых вы не знали. [Электронный ресурс]. Режим доступа: https://appleinsider.ru/promo/preimushhestva-ispolzovaniya-apple-wallet-o-kotoryx-vy-ne-znali.html
2. Карты лояльности. Google Pay API for Passes в ASP.NET. [Электронный ресурс]. Режим доступа: https://habr.com/ru/post/434486/
3. Apple Wallet. Что это такое и как интегрировать в него свою карту. [Электронный ресурс]. Режим доступа: https://habr.com/ru/company/agima/blog/428437/
4. Pass Design and Creation. [Электронный ресурс]. Режим доступа: https://developer.apple.com/library/archive/documentation/UserExperience/Conceptual/PassKit_PG/Creating.html
5. Add, use, and share boarding passes, tickets, and other passes in Apple Wallet. [Электронный ресурс]. Режим доступа: https://support.apple.com/en-hk/HT204003
6. Добавление и использование карт в приложении Wallet на iPhone. [Электронный ресурс]. Режим доступа: https://support.apple.com/ru-ru/guide/iphone/iphe7aa3336/ios

УДК 004.42

Ригин Е.В. (2 курс магистратуры),
Молодяков С.А., д.т.н., доцент

ПРИМЕНЕНИЕ РАСПОЗНАВАНИЯ ЛИЦ В СИСТЕМЕ КОНТРОЛЯ ДОСТУПА И ПРОДАЖИ БИЛЕТОВ

Целью работы является разработка элементов программного обеспечения для системы контроля доступа и продажи билетов. Особенностью системы является использование распознавания лиц с помощью нейронной сети [1, 2].

Система контроля доступа и продажи билетов включает элементы, представленные на Рисунке 1: пользовательский интерфейс (UI), API сервер, контроллер турникетов и видеокамеру. Основной алгоритм работы системы определяется следующим образом. При покупке билетов пересылаются фотографии посетителей мероприятий в базу данных. При проходе на мероприятие камера считывает изображение. Разрабатываемая программа определяет возможность прохода через турникет.

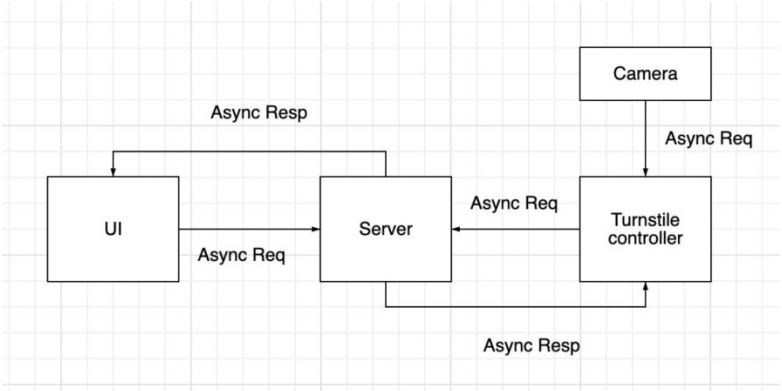


Рисунок 1 – Упрощенная схема системы контроля доступа и продажи билетов

В рамках данной работы нам необходимо провезти анализ существующих методов распознавания лиц, в которых используются нейронные сети. В качестве основных условий, усложняющих процесс распознавания, будут рассмотрены условия, специфичные для текущей эпидемиологической ситуации, требующей ношения масок.

В работе рассмотрены нейронные сети, в первую очередь те, для которых имеется достаточное количество документации и активного сообщества, используется язык разработки Pythoni открытый исходный код. Рассмотрены следующие нейронные сети:
FaceNet (https://github.com/davidsandberg/facenet),
Face Recognition (https://github.com/ageitgey/face_recognition),
InsightFace (https://github.com/deepinsight/insightface),
DeepFace (https://github.com/iperov/DeepFaceLab),
OpenCV [3] (https://github.com/opencv/opencv),
VGG-Face (https://github.com/rcmalli/keras-vggface).

Для анализа нейронных сетей был выбран датасет FaceID-550 (https://data.mendeley.com/datasets/zs2vnf9692/2), который состоит из 550 изображений лиц для обучения и одного изображения автора исследования для тестирования. Проверка качества распознавания проводилась при следующих условиях:

1. Полностью открытое лицо.
2. Надета медицинская маска, закрывающая область рта до носа.
3. Надета медицинская маска, закрывающая область рта и нос.
4. Надеты солнцезащитные очки.
5. Скрыта половина лица.

В работе предложен следующий показатель качества распознавания изображений:

$$Q = \frac{\sum_{i=N} \frac{N_{NET}}{5}}{N},$$

где N_{NET} – количество распознанных изображений нейросетью, 5 – количество тестируемых вариантов, N – количество тестируемых изображений.

Выделим 5 изображений лиц для тестирования, создав для каждого 5 вариантов из условий, описанных выше.

Таблица 1 – Результаты работы выбранных нейронных сетей

Нейросеть	Q
FaceNet	0.4
Face Recognition	0.6
InsightFace	**0.8**
DeepFace	0.2
OpenCV	0.6
VGG-Face	0.4

По результатам в таблице видно, что наиболее эффективным распознавание оказалось у нейронной сети InsightFace. Отдельно выделим, что в ходе эксперимента ни одна нейросеть не определила лицо человека, находящегося в солнцезащитных очках. Поэтому наиболее эффективной будет следующая методика распознавания человека по изображению лица:

1. Расположить камеру на уровне лица человека таким образом, чтобы в поле зрения попадала как минимум половина лица по вертикали.
2. Обеспечить у человека открытую область глаз.
3. Провести распознавание нейронной сетью InsightFace.

Выводы

В работе были сформулированы критерии выбора нейронных систем для распознавания лиц, критерии помех на изображениях для проверки нейронных сетей, соответствующие современным реалиям, проведен анализ качества распознавания при наличии помех,

предложена методика для наиболее эффективного распознавания человека по изображению лица. Рекомендована нейронная сеть InsightFace.

Дальнейшая работа будет посвящена разработке информационной системы контроля доступа и продажи билетов, в которой в качестве подсистемы будет входить нейронная сеть распознавания лиц.

ЛИТЕРАТУРА

1. Francoise Fogelman Soulie, Harry Wechsler, Jonathon P. Phillips, Thomas S. Huang, Vicki Bruce (2012-12-06). Face Recognition. Springer Berlin Heidelberg. ISBN 9783642722011.
2. S. Ramakrishnan (2016). Semisupervised Classification, Subspace Projection and Evaluation Methods. ISBN 9789535124214.
3. Молодяков С. А. Применение функций OpenCV в компьютерном зрении (60 примеров на Python): монография / С. А. Молодяков. – СПб. : ПОЛИТЕХ-ПРЕСС, 2022. – 296 с.

УДК 621.319

Смирнов П.А., Малиновская В.В. (4 курс бакалавриата),
Воинов Н.В., к.т.н., доцент

КЛИЕНТ-СЕРВЕРНОЕ ПРИЛОЖЕНИЕ ДЛЯ ДОНОРСКОЙ СЛУЖБЫ

Цель работы – разработать приложение для российского рынка, которое предоставит донорам возможность простого и удобного взаимодействия с пунктами сдачи крови. Создание такого приложения с учетом успешных зарубежных аналогов сможет решить существующие проблемы в данной области, сделать процесс добровольного донорства крови комфортным и привлечь новых волонтеров.

Планируется реализация Web-приложения с набором следующих основных функций.

У пользователя приложения должна быть возможность получить информацию о пунктах сдачи крови в своем городе: часы работы, возможность сдать плазму, актуальные новости и донорский светофор. Кроме того, должна быть реализована возможность фильтрации по удаленности от текущего местоположения и/или донорскому светофору.

Отзывы могут помочь донорам выбрать пункт сдачи крови, а администрации пункта получить обратную связь и предложения по улучшению процесса сдачи крови в том или ином пункте и сделать сдачу крови максимально удобной для доноров.

Раздел «Новости» поможет пунктам переливания донести важную информацию до доноров, а также привлечь больше доноров. Предполагается, что данный раздел будет содержать такую информацию, как введение дополнительных дат приема крови, оповещения о необходимости крови в случае чрезвычайных и критических ситуаций, а также о дополнительных поощрениях доноров, например, в день донора, что поможет способствовать поддержанию лояльности доноров и служить дополнительной мотивацией к совершению донаций.

На странице конкретного донорского пункта пользователь сможет посмотреть информацию о пункте, актуальные новости, оставить отзыв и ознакомиться с отзывами других доноров. Помимо этого, пользователь имеет возможность задать интересующие вопросы работникам донорского пункта в чате.

При записи на донацию предлагается заполнить анкету опроса донора крови с целью выявления временных противопоказаний, а также ускорения процесса сдачи крови.

В случае, если в результате анкетирования выяснится, что донор временно отстранен от сдачи крови, будет предложено установить напоминание, которое придет на почту по окончании срока действия противопоказаний. Также пользователь будет иметь возможность подключить уведомления с приглашением к повторной сдаче крови после минимального интервала между донациями.

В личном кабинете авторизованный пользователь сможет посмотреть историю своих донаций, а именно количество совершенных донаций с подробной информацией о каждой из них. Перед каждой донацией проводится исследование крови донора, с результатами анализов также можно будет ознакомиться в личном кабинете. Результаты представлены как в виде таблицы, так и в виде графиков, что позволит донорам отслеживать динамику своего здоровья. Также в личном кабинете будет отображено количество сдач крови и плазмы, которые необходимо совершить до получения звания “Почетный донор”.

Для реализации приложения была выбрана трехуровневая клиент-серверная архитектура [1-3], которая предполагает наличие в приложении трех типов компонентов: сервера приложения, сервера базы данных, и клиентского приложения.

Для создания сервера приложения были выбраны язык программирования Java и Spring Framework. Сам сервер разрабатываемого продукта представляет собой Spring Boot приложение, которое позволяет легко использовать все необходимые слои Spring Framework.

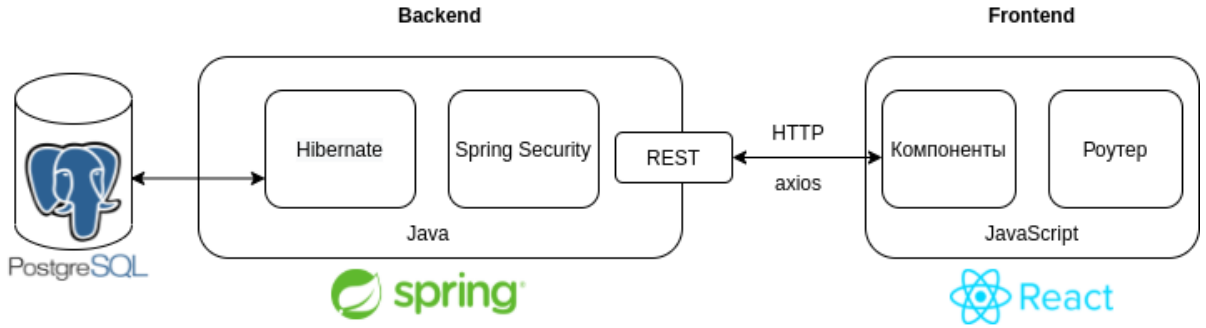


Рисунок 1 – Архитектура приложения

Взаимодействие сервера приложения и клиентского приложения предлагается организовать при помощи REST API, который предоставляет сервер. REST (от англ. Representational State Tranfer) – архитектурный стиль взаимодействия компонентов распределенного приложения. Обычно представляет собой вызов GET, POST, UPDATE и DELETE HTTP-запросов. Формат обмена данными – JSON.

Для хранения всей информации была выбрана система управления базами данных PostgreSQL. Сам сервер базы данных для удобства разработки планируется запускать при помощи соответствующего Docker-контейнера.

Для работы с СУБД используется модуль Spring – Spring Data JPA. Безопасный доступ к нашему REST API организуется при помощи еще одного модуля Spring – Spring Security. Для авторизации пользователей системы используются JWT (Json Web Token).

Обмен сообщениями в чате планируется организовать при помощи WebSocket – тонкого и легковесного слоя над TCP, а протокол обмена сообщениями, который планируется использовать – STOMP (Simple Text Oriented Message Protocol).

Для разработки клиентской части выбран язык программирования TypeScript и библиотека React в комбинации с библиотекой Redux для сохранения состояния сайта при переходах по истории. TypeScript расширяет возможности JavaScript путем добавления в него статической типизации.

В качестве Redux middleware для работы с REST API, а также WebSocket планируется использовать модуль Redux-Thunk. За организацию маршрутизации для сопоставления запросов к приложению с определенными компонентами интерфейса отвечает библиотека react-router.

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.

2. Тузов, Г. Д. Реализация асинхронного выполнения событий в высоконагруженном распределенном веб-приложении / Г. Д. Тузов, 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. 135-136.
3. Золотухин, Д. В. Разработка клиент-серверного приложения для автоматизации регрессионного тестирования / Д. В. Золотухин, Н. В. Воинов, В. В. Шаляпин // 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. 107-108.

УДК 004.41

Старовойтов А.Е. (2 курс магистратуры),
Самочадин А.В., к.т.н., доцент

РАЗРАБОТКА СИСТЕМЫ АНАЛИЗА ВАКАНСИЙ

Целью работы является разработка системы анализа описаний вакансий для извлечения требований вакансий. Данная работа относится к области обработки текстов на естественном языке, а именно к задаче извлечения информации – то есть автоматического построения структурированных данных из неструктурированных или слабоструктурированных.

Требования к разрабатываемой системе:

1) Извлечение требований для русского и английского языка.
2) Извлечение ранее неизвестных требований.
3) Возможность добавления новых источников данных.

В качестве языков разработки были выбраны C# и Python. В качестве базы данных используется ElasticSearch [1]. Система взаимодействует с внешними сервисами – веб-сайтами для поиска работы, такими как HeadHunter.

Подход к решению задачи основан на сопоставлении со словарем и векторном представлении слов, а именно модели word2vec [2].

Этапы работы системы следующие:

1. На вход подается набор текстов вакансий и словарь навыков.
2. Сопоставление со словарем навыков, получение предварительного списка требований.
3. Создание векторного представления текстов вакансий.
4. Для десяти наиболее встречаемых требований происходит поиск требований с похожими векторами в созданной модели word2vec и добавление их в словарь.
5. Сопоставление с новым словарем, вывод списка требований.

В качестве словаря навыков используется набор данных Emsi Skills [3]. Набор данных содержит 30500 технических навыков и личностных характеристик. В ходе работы к нему было добавлено 200 извлеченных навыков.

ЛИТЕРАТУРА

1. Elasticsearch. [Электронный ресурс]. Режим доступа: https://www.elastic.co/elasticsearch/
2. Mikolov T. et al. Efficient estimation of word representations in vector space //arXiv preprint arXiv:1301.3781. – 2013.
3. Emsi Skills. [Электронный ресурс]. Режим доступа: https://skills.emsidata.com/

УДК 004.42

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

РАЗРАБОТКА МОБИЛЬНОГО НАВИГАЦИОННОГО ПРИЛОЖЕНИЯ НА ПЛАТФОРМЕ IOS

Целью работы является предоставление пользователям удобной, быстрой, подробной и интерактивной навигационной системы, с помощью которой можно будет быстро найти, сориентироваться и построить маршрут от входа в комплекс до нужного кабинета. В качестве примера комплекса выбран кампус Политехнического Университета.

Для достижения цели необходимо решить следующие задачи:

- Провести обзор существующих программных решений.
- Определить методику картографии и построить высоко детализированную карту.
- Разработать и опубликовать мобильное приложение, отображающее интерактивную карту с возможностью построения маршрута.

В процессе работы был проведён обзор существующих решений и были выделены следующие основные недостатки: отсутствие стандартизации и/или документации форматов описания картографии, отсутствие привязки комнат к графу для построения маршрутов, отсутствие возможности описывать прилегающую территорию. В контексте кампуса Политехнического Университета ни одно из существующих решение не позволяет описать дорожки между корпусами.

Был проведён обзор способов хранение картографии, в результате которого для разработки проекта выбран открытый стандарт описания помещений IMDF[1], являющийся расширенной версией стандарта GeoJSON[2]. Стандарт подразумевает описание комплекса из нескольких зданий, однако в нём не предусмотрено описание окружения.

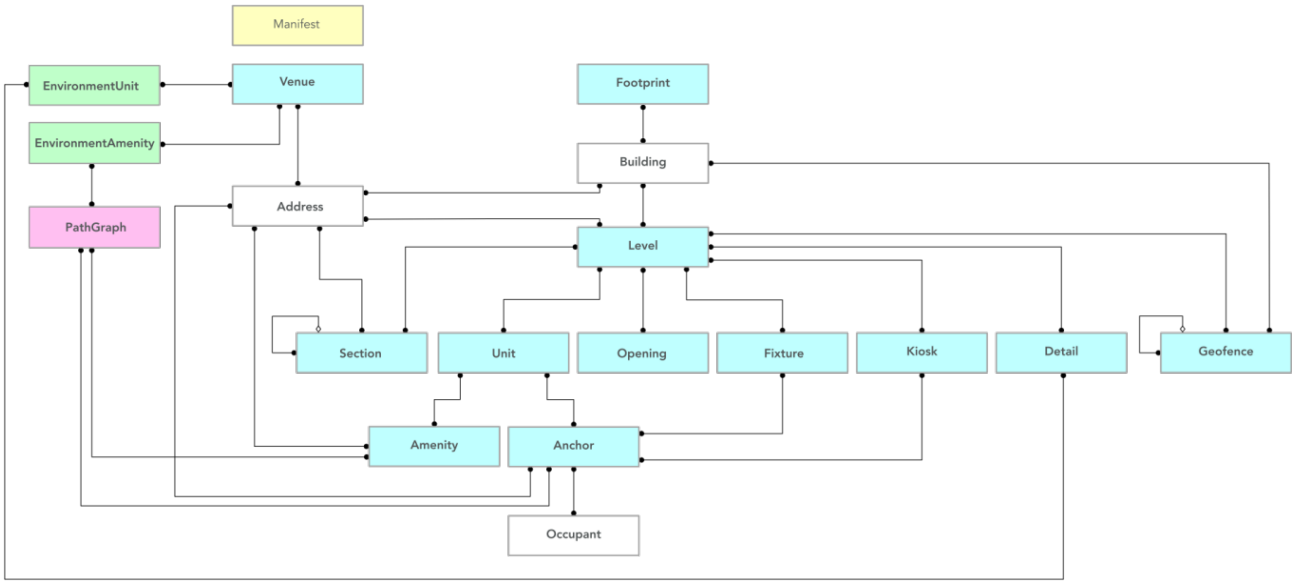


Рисунок 1 – Расширенный формат IMDF.

Стандарт был расширен для описания уличного окружения, в частности были добавлены EnvironmentUnit для описание геометрических форм (дорога, газон, лес и т. д.) и EnvironmentAmenity для описания аннотаций на карте (парковка, автобусная остановка, вход на территорию и т. д.). Для возможности построения маршрута, в стандарт был добавлен PathGraph, связывающий аннотации карты с вершинами графа маршрутов. Был разработан редактор карт, поддерживающий расширенный стандарт. Была создана карта северо-западной

части кампуса, включающая в себя план главного здания. В дальнейшем планируется составить план всего кампуса.

В данный момент ведётся разработка мобильного приложения на платформе iOS для отображения карты и построения маршрута по ней с использованием MapKit[3]. Приложение будет поддерживать технологию AppClips[4], которая позволяет открыть приложения с помощью QR кода, не скачивая его на устройство. С помощью этой технологии можно будет поделиться маршрутом до кабинета, например, для проведения мероприятий.

ЛИТЕРАТУРА

1. Indoor Mapping Data Format. [Электронный ресурс]. https://register.apple.com/resources/imdf/
2. GeoJSON Format. [Электронный ресурс]. https://datatracker.ietf.org/doc/html/rfc7946
3. Godbollon S., Kumar M. Adding Indoor Maps to your App and Website, WWDC, 2019
4. App Clips. [Электронный ресурс]. https://developer.apple.com/app-clips/ (дата обращения: 18.03.2022)

УДК 004.42

Субботин Д.И. (4 курс бакалавриата),
Самочадина Т.Н., старший преподаватель,
Самочадин А.В., к.т.н., доцент

РАЗРАБОТКА ПРИЛОЖЕНИЯ ДЛЯ ВИЗУАЛИЗАЦИИ ЧАСТИЧНО ИЛИ ПОЛНОСТЬЮ УТРАЧЕННЫХ ПАМЯТНИКОВ АРХИТЕКТУРЫ

Целью работы является создание мобильного приложения на базе операционной системы Android, которое позволяет визуализировать частично или полностью утраченные памятники архитектуры с помощью технологий дополненной реальности. Для получения внешнего вида частично или полностью утраченного памятника используются оцифрованные фотографии.

Для визуализации внешнего вида памятника до перестройки или разрушения и внешнего вида памятника после перестройки или разрушения используются следующие методы:

1. Сравнение с использованием слайдера «было/стало». С одной стороны, от слайдера находится фотография, сделанная до изменений, а с другой стороны находится фотография, сделанная после изменений. При перемещении слайдера меняется соотношение площадей этих фотографий.

Рисунок 1 – Сравнительная визуализация с помощью слайдера.

2. Сравнение с использованием наложения одного изображения на другое. В данном методе изображения контура и текстуры памятника до перестройки накладывается поверх изображения памятника после перестройки. Текстуры и контуры памятников извлекаются с помощью API Remove.bg, которое удаляет задний фон с изображения, и библиотеки алгоритмов компьютерного зрения OpenCV.

Рисунок 2 – Сравнительная визуализация с помощью наложения контура.

3. Сравнение с использованием дополненной реальности. В данном методе AR-модель в виде текстуры памятника до перестройки накладывается на изображение с камеры мобильного телефона, направленной на место постройки памятника. AR-моделирование осуществляется при помощи SDK ARCore.

Рисунок 3 – Сравнительная визуализация с помощью AR-моделирования.

ЛИТЕРАТУРА

1. Документация ARcore – URL: https://developers.google.com/ar/develop. [Электронный ресурс].
2. Palma, V. TOWARDS DEEP LEARNING FOR ARCHITECTURE: A MONUMENT RECOGNITION MOBILE APP [Электронный ресурс] / V. Palma // The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences. — 2019. — С. 551-556. — Режим доступа: https://doi.org/10.5194/isprs-archives-xlii-2-w9-551-2019

УДК 004.4

Трошев Д. М. (4 курс бакалавриата),
Медведев Б. М., к. т. н., доцент

ПРОГРАММНЫЕ СРЕДСТВА РАДИОКОНТРОЛЯ

Глобальное использование радио технологий привело к значительному росту парка радиосредств. В РФ работающая группировка стационарных радиосредств насчитывает около 2 миллионов радиоустройств, работающих на передачу и прием радиосигналов, а количество мобильных радиосредств персонального применения более 200 миллионов [1]. Рост загруженности радиочастотного спектра привел к усложнению электромагнитной и помеховой обстановки. Надлежащее использование радиочастотного спектра, организация рационального частотного планирования, учета и контроля определенных правил и норм использования радиочастот и радиоэлектронных средств обеспечивается системой радиоконтроля. Существующая система радиоконтроля ежегодно выявляет более 15 тысяч [2] нарушений норм использования радиочастот, однако позволяет обновлять информацию обо всех радиоэлектронных средствах не чаще, чем раз в три года [3]. Создаваемые новые системы радиоконтроля содержат элементы искусственного интеллекта и нейросетей для решения задач радиоконтроля таких, как определение местоположения источника и параметров радиосигнала.

Целью работы является разработка программных средств системы радиоконтроля, обеспечивающих контроль распределенных приемников, сбор и обработку результатов измерений. Для достижения поставленной цели нужно решить следующие задачи:

1. Разработка программных средств сбора измерений.
2. Разработка программных средств контроля распределенных приемников.
3. Разработка интерфейса пользователя для управления обучением нейросети, которая используется для определения местоположения источника сигнала.

На Рисунке 1 приведена архитектура разработанных программных средств:

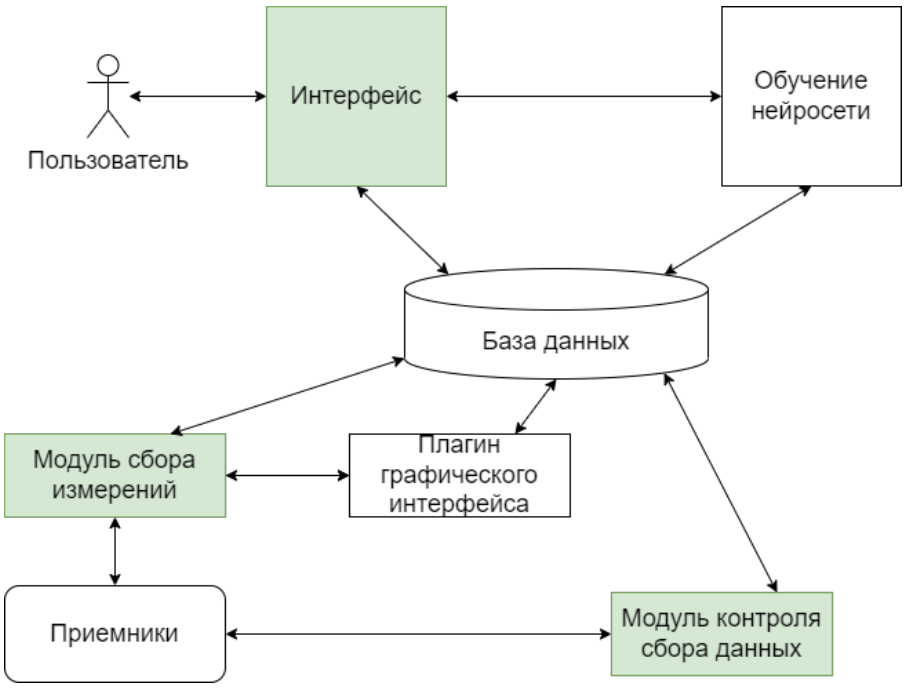


Рисунок 1 – Архитектура проекта.

Проект состоит из следующих компонентов:

Модуль сбора измерений предназначен для сбора информации о мощности, частоте, локации и времени измерения радиосигнала, которая формируется системой распределенных

на местности приемников. Результатом работы модуля являются файлы, которые образуют обучающую выборку для нейросети.

Модуль контроля сбора данных обеспечивает слежение за состоянием приемников и формирование сообщений об аварии. Так как все приемники подключены к коммутатору локальной сети и получают электропитание по кабелям с использованием power-over-Ethernet, то в качестве информации о работоспособности приемника используется мощность потребления и передача данных через порт, которые определяет коммутатор.

Интерфейс пользователя позволяет выбрать группу приемников и соответствующую обучающую выборку, которая будет использоваться в обучении нейросети. Кроме того, можно установить параметры обучения - коэффициент обучения, управляющий величиной весов на каждой итерации-эпохи, и количество итераций обучения [4].

Модуль формирования обучающих выборок собирает бинарные файлы в обучающие выборки. Нейросеть используется для определения параметров радиосигналов в автоматическом режиме в реальном масштабе времени и, в случае выявления нарушений правил использования радиочастотного спектра, сигнализировать об этом.

Для создания интерфейса пользователя использовался Qt фреймворк для разработки кроссплатформенного программного обеспечения на языке программирования C++ [5]. Для реализации модулей сбора измерений и контроля сбора данных – язык C++.

В результате проделанной работы были созданы программные средства системы радиоконтроля, которые обеспечивают сбор измерений и контроль состояния до 400 распределенных приемников, а также обеспечивают развитый интерфейс пользователя для выбора и настройки входных параметров и обучающих выборок нейросети.

ЛИТЕРАТУРА

1. Объекты и средства радиоконтроля. Совместное развитие технологий радиосвязи и радиоконтроля - С. В. Кизима. Режим доступа: https://niir.ru/wp-content/uploads/2020/03/%D0%9A%D0%B8%D0%B7%D0%B8%D0%BC%D0%B0-%D0%A1.%D0%92.-114.pdf
2. Результаты контрольно-надзорной и разрешительно-регистрационной деятельности в сфере связи в 2015г. – О. А. Иванов. Режим доступа: https://rkn.gov.ru/docs/Ivanov1.pdf
3. Концепция развития системы контроля за излучениями радиоэлектронных средств и (или) высокочастотных устройств гражданского назначения в Российской Федерации на период до 2025 года. Режим доступа: https://digital.gov.ru/uploaded/files/kontseptsiya-radiokontrolya.pdf
4. Нейронные сети и нейрокомпьютеры / П.Г. КРУГ // - Р. 93–94. Режим доступа: http://iit1.mpei.ac.ru/pubkrug1.pdf
5. Qt Documentation [электронный ресурс]. Режим доступа: https://doc.qt.io/

УДК 004.03

Фань Инган (2 курс магистратуры),
Никифоров И. В., к.т.н., доцент

ПЛАТФОРМА АНАЛИЗА БОЛЬШИХ ДАННЫХ ЧАЯ БАЗЕ PYTHON

В работе исследуется платформа для анализа больших данных о чае и предлагается концепция согласования ресурсов в цепочке поставок чая для содействия общей выгоде предприятий по производству чая [1]. Согласование ресурсов - это использование технологии больших данных цепочек поставок для реализации сбора информации о спросе и предложении чая с обеих сторон (поставщика и покупателя), а также для уточнения информации, необходимой обеим сторонам. При использовании методов согласования ресурсов, которые соответствуют собственному бизнесу предприятий снабжения и маркетинга, стороны спроса и предложения чая могут достичь более высокого уровня согласования ресурсов [2].

Анализ данных о производстве и продаже чая реализован в платформе анализа больших данных на основе языка программирования Python [3]. Общая архитектура платформы разделена на пять уровней, представленных на Рисунке 1: сбор данных, распределенное хранение, распределенные вычисления, прикладной уровень и пользовательский интерфейс.

В условиях растущей конкуренции на чайном рынке Китая, производители чая могут изменить свою традиционную модель управления, чтобы достичь "синергии" и "взаимовыгодного сотрудничества" со своими партнерами по переработке и сбыту в конкурентной борьбе, является актуальным вопросом, требующим решения.

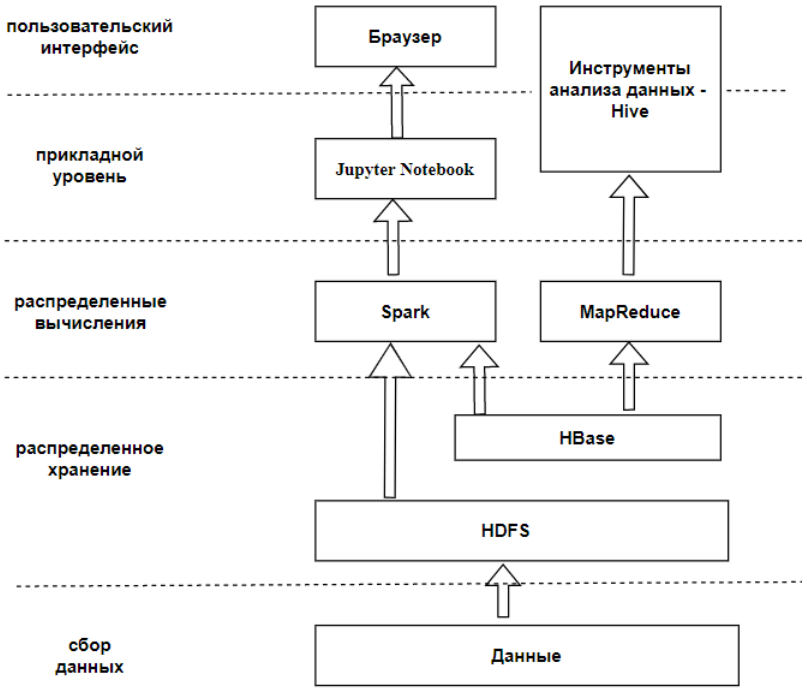


Рисунок 1 – Общая архитектура платформы больших данных чая.

Основные выделяемые уровни программного решения перечислены ниже.

1) Уровень сбора данных включает информацию о производстве чая, информацию о переработке чая, информацию о логистике и информацию о продажах чая.

2) Уровень распределенного хранилища включает HDFS и HBase. HDFS - это основа управления хранением данных в системе Hadoop [4]. Он обеспечивает механизм для одной записи и нескольких операций чтения, и данные распределяются блоками на разных физических машинах в кластере одновременно. HBase - это ориентированная на столбцы, масштабируемая, высоконадежная, высокопроизводительная, распределенная и ориентированная на столбцы динамическая база данных для структурированных данных, построенная поверх HDFS.

3) Уровень распределенных вычислений включает в себя Spark и MapReduce. Spark предоставляет более быструю и универсальную платформу для обработки данных. Данные, хранящиеся на уровне распределенного хранилища HBase, могут быть обработаны с помощью MapReduce, который идеально сочетает в себе хранение данных и параллельные вычисления.

4) Прикладной уровень предоставляет персоналу по оптимизации сети функции написания собственного кода, отладки программ и отображения результатов. Jupyter Hub используется для реализации управления записными книжками нескольких сотрудников по оптимизации сети. В тоже время он предоставляет традиционные инструменты статистического анализа больших данных, такие как HIVE и PIG, для персонала по оптимизации сети на выбор.

5) Уровень пользовательского интерфейса используется персоналом по оптимизации сети для взаимодействия с рабочей информацией и реализации преобразования между внутренней комбинированной формой сетевой информации и формой, приемлемой для персонала по оптимизации сети, в соответствии с установленной бизнес-логикой.

Программная система реализуется с помощью технологий, перечисленных ниже.

Jupyter Notebook это интерактивная онлайн-служба веб-приложений, которая может поддерживать более 40 языков программирования, включая Python, вызывая различные программы ядра. Он может осуществлять считывание данных, обработку данных, анализ данных, визуализацию данных и сохранение результатов данных. он стал очень практичным интерактивным вычислительным инструментом, а также идеальным средством для научных исследований и преподавания.

Jupyter имеет формат документа на основе JSON [5]. Ipynb может легко обмениваться контентом, таким как код, выходные результаты и изображения. В настоящее время на различных семинарах по Python популярным демонстрационным методом является использование Jupyter Notebook, который использует файлы ipynb.

ЛИТЕРАТУРА

1. Лю Цзяньи, Ся Чан, Сян И и др. Прогноз цен на чай в Гуйчжоу на основе временных рядов и логической регрессии. Информационные технологии и информатизация, 2021 (7): 70-75.
2. Чжао Сяньбин, Бай Донг. Система кластерного анализа данных о состоянии здоровья учащихся на основе Python. Электронные технологии и разработка программного обеспечения, 2021 (5): 183-185.
3. Пэн Гуаннянь, Лай Вэйкай. Разработка и внедрение инструмента табличного анализа Python. Фуцзянь Компьютер, 2021 (6): 77-78.
4. 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.
5. Хэ Цзунпин, Чжан Сяодун, Лю Юй. Микросервисная архитектура, основанная на платформе интерактивного анализа Jupyter. Приложения для компьютерных систем, 2019 (8): 63-70.

УДК 004.42

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

РАЗРАБОТКА СИСТЕМЫ ЭКСТРЕННОГО ОПОВЕЩЕНИЯ НА ОСНОВЕ ОПРЕДЕЛЕНИЯ ИНЦИДЕНТОВ В ПОВЕДЕНИИ ЛЮДЕЙ

Целью данной работы является создание системы наблюдения за подопечным и отправки уведомления в случае возникновения инцидентов, конкретно падений.

Для достижения поставленной цели были сформулированы следующие задачи:

- Провести обзор существующих реализаций систем наблюдения за людьми пожилого возраста и инвалидов
- Разработка архитектурной схемы данной системы.
- Разработка алгоритма определения падения при помощи существующий решений построение скелета человека по изображению.
- Разработка сервера обработки видеопотока.
- Разработка мобильного приложения под систему iOS

В процессе работы был проведён обзор существующих средств для наблюдения за лицами пожилого возраста и инвалидами. Преимущественно используется “тревожная кнопка” – мобильное приложение или специальное устройство, при взаимодействии с которым можно вызвать скорую медицинскую помощь. Однако это может оказаться сложным в критических ситуациях для подопечного, например, при возникновении инсульта.

В результате анализа была выявлена необходимость разработки собственного программного средства для решения поставленной задачи. Была разработана и впоследствии изображена в виде графической иллюстрации концептуальная схема работы системы и взаимодействия её компонентов.

Согласно концептуальной схеме система реализует клиент-серверную архитектуру, при которой клиентом является мобильное приложение на устройстве опекуна. Сервер взаимодействует с камерами и анализирует кадры видеопотока. При взаимодействии пользователя с приложением клиент запрашивает видеопоток с сервера. При возникновении инцидентов будет отправлено уведомление через Apple Notification Center Services на устройство опекуна.

Для определения падения человека было принято решение использовать открытую библиотеку MediaPipe, которая позволяет построить “скелет” человека по изображению [2]. С полученным набором координат за определенный период времени можно будет определять перемещение человека в пространстве, таким образом, можно определить падение.

Сервер будет разделен на две концептуальные части [1]:

- Первая часть - будет обрабатывать видеопоток с одной камерой и, в случае возникновения инцидентов, отправит уведомление [3].
- Вторая - оркестровая, будет создавать процесс первой части, передавая ссылку на видеопоток.

Мобильное приложение будет разработано под операционную систему iOS с возможность регистрации через AppleID. В данном приложении можно будет просматривать видеопоток от зарегистрированных на пользователя камерах.

Концептуальная схема работы системы в графическом виде представлена на Рисунке 1.

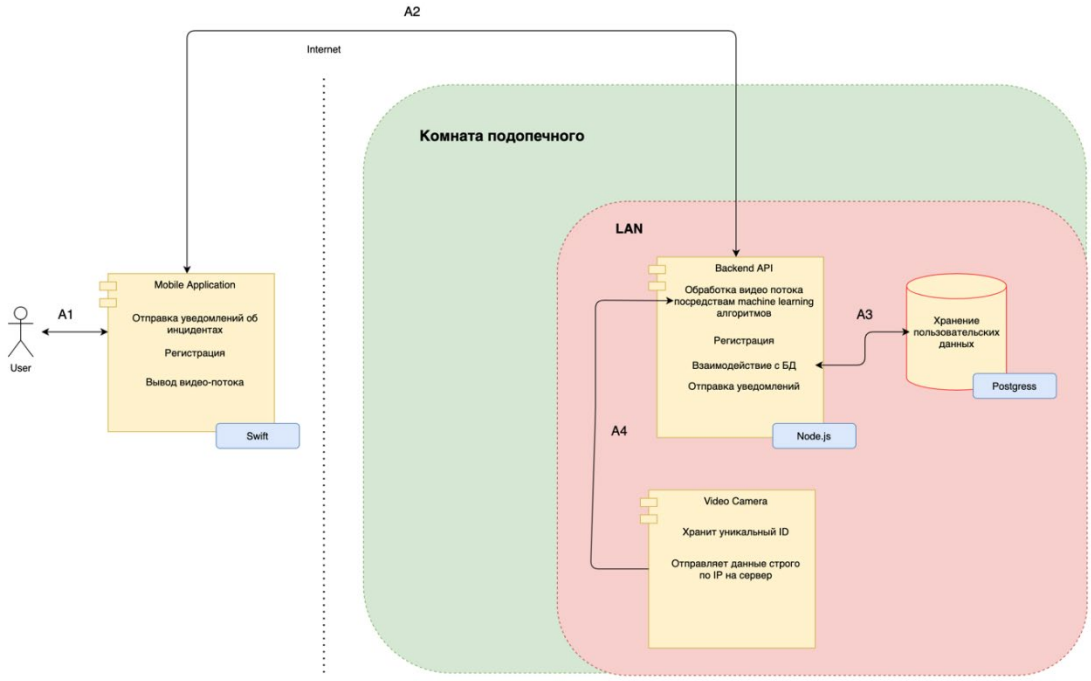


Рисунок 1 – Концептуальная схема системы.

В дальнейшем планируется осуществить программную реализацию описанной в данной статье системы.

ЛИТЕРАТУРА

1. Шаблоны интеграции корпоративных систем – Бобби Вульф, Грегор Хоп. 2016 г., 732 стр.
2. MediaPipe docs [Электронный ресурс] Режим доступа: https://google.github.io/mediapipe/.
3. Создания сервера потокового видео при помощи языка программирования JavaScript – [Электронный ресурс] Режим доступа: https://medium.com/swlh/building-a-video-streaming-service-in-javascript-7f751fe76564 .

УДК 004.4

Хисматуллин К.И. (4 курс бакалавриата),
Леонтьева Т.В., к.т.н., доцент

ПРИЛОЖЕНИЯ ДЛЯ ЧТЕНИЯ РАЗВЛЕКАТЕЛЬНОЙ ЛИТЕРАТУРЫ

Целью работы является разработка приложения для русскоязычного сегмента, которое предоставит пользователю возможность чтения развлекательной литературы с расширением книжной базы за счет плагинов. Создание такого приложения поможет пользователям организовать свою библиотеку с интересующей их развлекательной литературой. В качестве развлекательно литературы будет рассматриваться манга, манхва, маньхуа и западные комиксы.

С повышением популярности японской анимации (аниме) растет и популярность их первоисточников, а именно манги и ранобэ (японские романы, целевой аудиторией которых являются подростки). Популярность аниме затронула и Россию. На сайте MangaLib (один из самых популярных сайтов с переводом развлекательной литературы), можно заметить, что самое популярное произведение читают/прочитали примерно 350 тысяч человек. Следовательно, приложение, объединяющее любимые сайты читателей манги и прочей развлекательной литературы с возможностью чтения скачанных произведений без подключения к сети Интернет будет востребовано.

Подобные приложения уже существуют. Я взял 3 самых популярных приложений с русским переводом манги в Play Маркет:

- Mangas Viewer
- ЧитайМангу
- Manga Reader

У каждого приложения имеется некоторый набор источников (MangaReader – 4, Mangas Viewer – 2, ЧитайМангу – только 1 источник ReadManga), из которых берется манга, но они заранее определены. К тому же поиск может быть осуществлен только по одному источнику. Следующий недостаток связан с созданием библиотеки. У Manga Reader есть только один раздел избранного. У Mangas Viewer имеется 4 заранее определенных и неизменяемых раздела. У ЧитайМангу можно без проблем создавать коллекции с разными названиями.

Планируется реализация мобильного приложения под операционную систему Android с набором следующих основных функций.

У пользователя приложения должна быть возможность установить интересующие плагины для работы с сайтами, содержащими развлекательную литературу.

В разделе «Поиск» пользователь может осуществить поиск литературы по установленным плагинам. После нахождения интересующей литературы есть возможность добавить ее в библиотеку или начать чтение.

В разделе «Библиотека» хранится литература. Раздел изначально состоит из одной вкладки. Предполагается, что пользователь в дальнейшем создаст свои вкладки, даст им имя и распределит литературу по ним.

В разделе «Обновления» осуществляется поиск обновлений у литературы, находящейся в библиотеке. Обновления в библиотеке отображаются в виде списка с названием произведения и добавленными главами.

Из разделов «Поиск» и «Библиотека» есть возможность перейти к детальной информации о выбранном произведении, где отображается основная информация: название, описание, жанры и доступные для чтения главы.

У пользователя также должна быть возможность сохранить в памяти устройства выбранные главы для дальнейшего просмотра без доступа к сети Интернет.

Учитывая, что один пользователь может установить приложения на несколько устройств, у приложения должна быть возможность синхронизации библиотек на различных устройствах.

Приложение состоит из двух основных компонентов клиентская часть, в которой будет сосредоточена основная логика приложения, и серверная часть, отвечающая за синхронизацию данных.

Предполагается создать клиентское приложение и набор плагинов для устройств на операционной системе Android используя язык Kotlin. С помощью плагинов будет осуществляться преобразование данных с разметки сайта либо с открытого API сайта.

Для хранения данных в клиентской планируется использовать фреймворк Room, который использует SQLite, на серверной части PostgreSQL.

Предполагаемый фреймворк для разработки серверной части на данный момент не выбран. Есть выбор между Spring Boot для Java и KTor для Kotlin. Для работы с СУБД выбирается либо Spring Data JPA на Java, либо Exposed на Kotlin. Основная задача серверной части синхронизация. Также предполагается хранение и обновление расширений.

Аутентификация и авторизация пользователей осуществляется с помощью Firebase используя учетные записи Google.

Взаимодействие сервера и клиентского приложения предлагается организовать при помощи REST API, который предоставляет сервер.

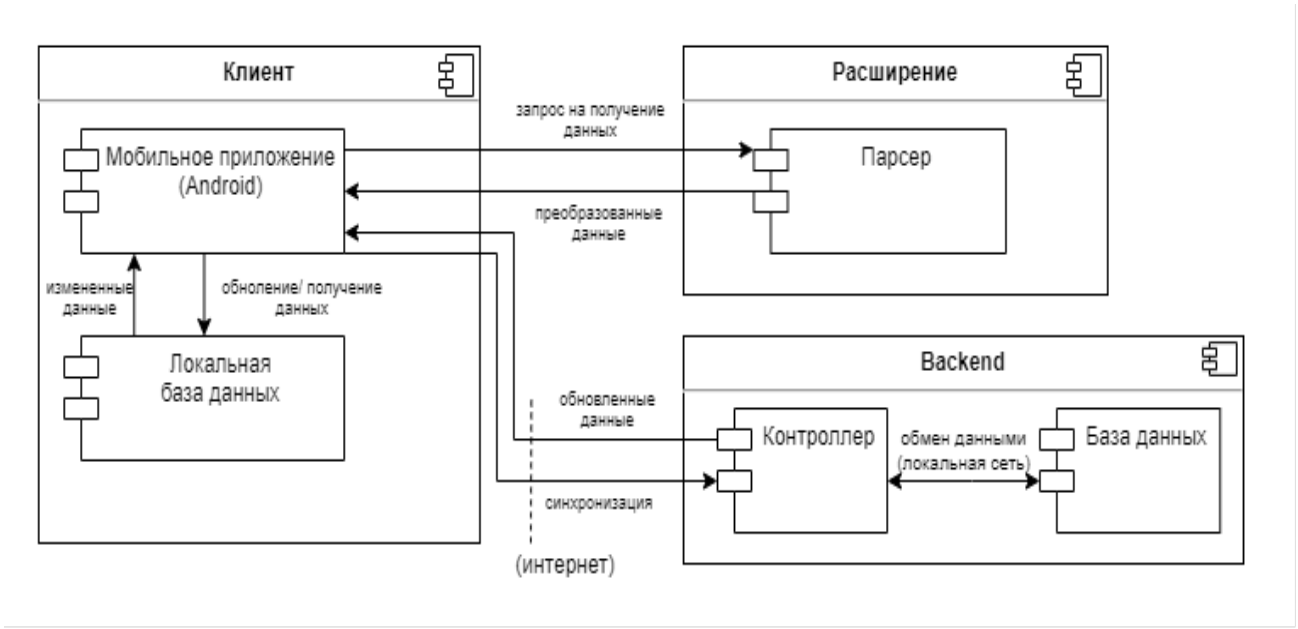


Рисунок 1 – компоненты системы.

Преимуществами разрабатываемого приложения являются расширяемость книжной базы с помощью легковесных плагинов, что также дает возможность легко добавлять/исправлять функционал, связанный с обработкой поиска без изменения основного приложения. Настраиваемая библиотека, включающая создание и изменение названий разделов для удобства сортировки литературы. Возможность синхронизации на разных устройствах под операционной системой Android.

ЛИТЕРАТУРА

1. Билл Филлипс, Крис Стюарт, Кристин Марсикано, Брайан Гарднер.Android. Программирование для профессионалов. 4-е издание. – СПб.: Питер, 2020.
2. Гриффитс Дон, Гриффитс Дэвид. Head First. Kotlin. – СПб.: Питер, 2020.
3. Арно Лоре. Проектирование веб-API / Пер. с англ. Д. А. Беликова. – М.: ДМК Пресс, 2020.

УДК 004.42

Шешукова О.С. (4 курс бакалавриата),
Самочадин А.В., к.т.н., доцент

КЛИЕНТСКАЯ ЧАСТЬ СИСТЕМЫ ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ СЦЕНАРИЕВ ПОВЕДЕНИЯ

Системы имитационного моделирования используются в разных областях, например, в механике, верификации программного обеспечения, моделировании физических процессов [1]. При этом системы имитационного моделирования достаточно сложны, так как имеют много разных настроек и разное поведение [2]. Стоит отметить, что систем моделирования сценариев поведения немного и нет сложившихся способов представления пользовательского интерфейса для ввода и исполнения сценариев. Таким образом, актуальной является задача разработки эффективной и удобной в использовании пользовательской части для системы имитационного моделирования сценариев поведения.

Целью работы является снижение трудоемкости работы оператора системы имитационного моделирования за счет проектирования и разработки клиентской части системы.

Перед непосредственной разработкой клиентской части необходимо рассмотреть и провести анализ существующих решений и методов. Это нужно как для составления требований к клиентской части, так и для ее реализации. Были рассмотрены четыре популярные системы имитационного моделирования, к которым относятся: AnyLogic [3], Arena [4], Plant Simulation [5, 6], GPSS Studio [7]. Для них были выявлены основные компоненты пользовательского интерфейса систем:

- Модуль симуляции, который включает в себя динамическое отображение объектов в процессе моделирования по мере изменения их состояния.
- Графический или текстовый редактор, который позволяет изменять параметры и настройки модели.
- Отображение результатов моделирования в различных видах, например, в виде текста, графика или картинки.

На основе результатов проведенного анализа были составлены требования для разрабатываемой клиентской части системы имитационного моделирования.

На Рисунке 1 представлена архитектура разрабатываемого программного решения.

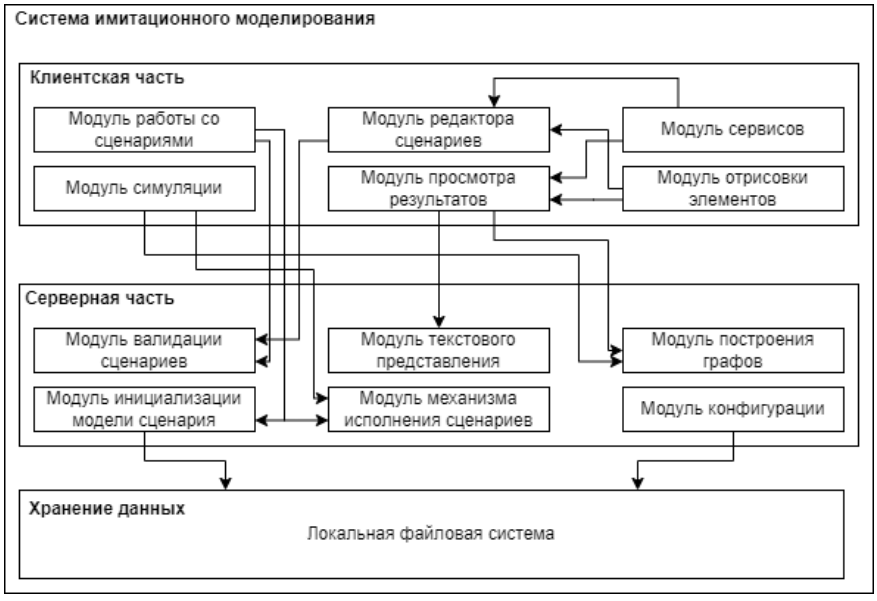


Рисунок 1 – Схема системы имитационного моделирования.

Разработанная система имитационного моделирования позволяет вводить, редактировать и исполнять пользовательские сценарии.

Функционал интерфейса разработан на языке программирования Python с помощью PySimpleGUI [8] – библиотеки Python GUI, которая инкапсулирует TKinter, Qt, Remi, WxPython в более простой и удобный интерфейс. Также в разработке использовались библиотеки NumPy, MatPlotLib, TKinter.На Рисунке 2 представлены результаты разработки пользовательского интерфейса.

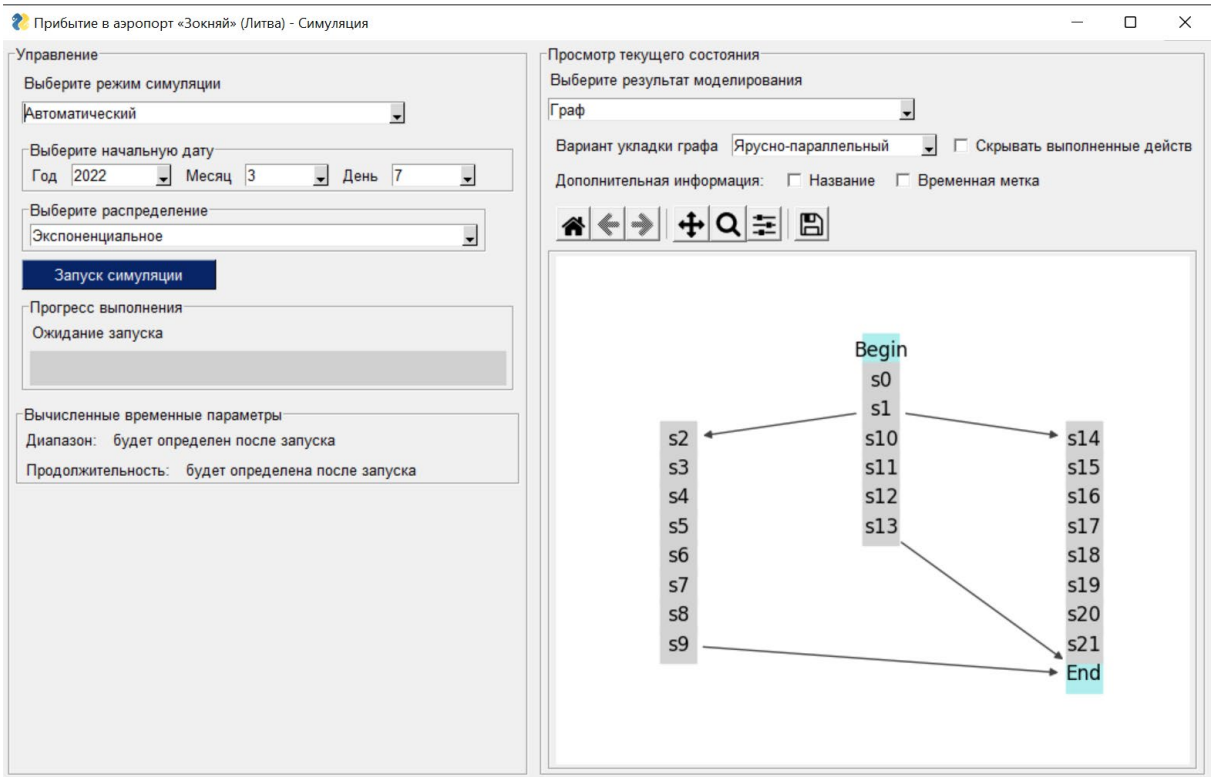


Рисунок 2 – Окно симуляции.

ЛИТЕРАТУРА

1. P. Drobintsev, V. Kotlyarov, I. Nikiforov, Conversion of abstract behavioral scenarios into scenarios applicable for testing // Proceedings of the Institute for System Programming of the RAS. – 2016. – Vol. 28. – No 3. – P. 145-160. – DOI 10.15514/ISPRAS-2016-28(3)-9.
2. И. М. Сысоев, И. В. Никифоров, Е. В. Каплан, 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.
3. The AnyLogic Company. // [Электронный ресурс] - режим доступа: https://www.anylogic.ru/ , свободный.
4. Rockwell Automation Inc., Wexford, PA, США. Сайт: www.arenasimulation.com. // [Электронный ресурс] - режим доступа: https://www.rockwellautomation.com/en-us/products/software/arena-simulation.html , свободный.
5. Bangsow, Steffen. Manufacturing Simulation with Plant Simulation and SimTalk. // Springer-Verlag Berlin Heidelberg, 2010 // ISBN 978-3-642-05073-2. // [Электронный ресурс] - режим доступа: http://simulation.su/static/plant-simulation-full-info.print, свободный.
6. О. Ю. Валеева, Имитационное моделирование с помощью системы // Трансформация современной науки в условиях цифровизации : Сборник статей Международной научно-практической конференции, Санкт-Петербург, 26 апреля 2019 года. – Санкт-Петербург:

Автономная некоммерческая организация дополнительного профессионального образования Межрегиональный образовательный центр, 2019. – С. 3-14.

7. Разработчик GPSS Studio компания ООО «Элина-компьютер» // [Электронный ресурс] - режим доступа: http://elina-computer.ru/static/about_gpss_studio.html свободный.
8. Документация PySimpleGUI // [Электронный ресурс] - режим доступа: https://pysimplegui.readthedocs.io/en/latest/ свободный

УДК 004.42

Шакола А.А. (4 курс бакалавриата),
Шмаков В.Э., к.т.н., доцент

СИСТЕМА УДАЛЁННОГО МОНИТОРИНГА ПАЦИЕНТОВ

Целью работы является создание системы удалённого отслеживания состояний пациентов путём численной оценки тяжести симптомов заболеваний с помощью валидизированных опросников.

По статистике на обсуждение жалоб пациента уходит примерно половина времени всего визита к врачу. Другая половина приходится на объективное обследование пациента, инструментальные методы диагностики и назначение терапии. В условиях огромного количества обращений пациентов к врачам, оценка жалоб становится очень затратным процессом (с индивидуальным подходом для каждого пациента). Для наблюдения хода заболеваний врачи используют стандартные валидизированные опросники и анкеты. Но все они разбросаны по интернету на разных ресурсах, что значительно затрудняет работу врачей. Единственное решение на рынке, которое было найдено, это приложение, состоящее из 10 опросов, без какого-либо функционала, не приспособленное для решения проблемы контроля состояний пациентов.

Система рассчитана на решение проблемы путём объединения всех опросников и анкет в единое приложение с развитым функционалом (выбор теста, сохранение результатов, оповещение врача и пациента и т.д) для удобного взаимодействия между врачами и пациентами.

Для разработки приложения выбран язык Kotlin, с подключенной Базой Данных PostgreSQL. Для уведомлений пользователей используется Telegram bot. Данные технологии обеспечивают удобство разработки мобильных приложений для смартфонов. Данный проект выполняется по запросу Первого Санкт-Петербургского государственного медицинского университета им. акад. И.П Павлова и планируется к внедрению в медицинскую область, связанную с Урологией.

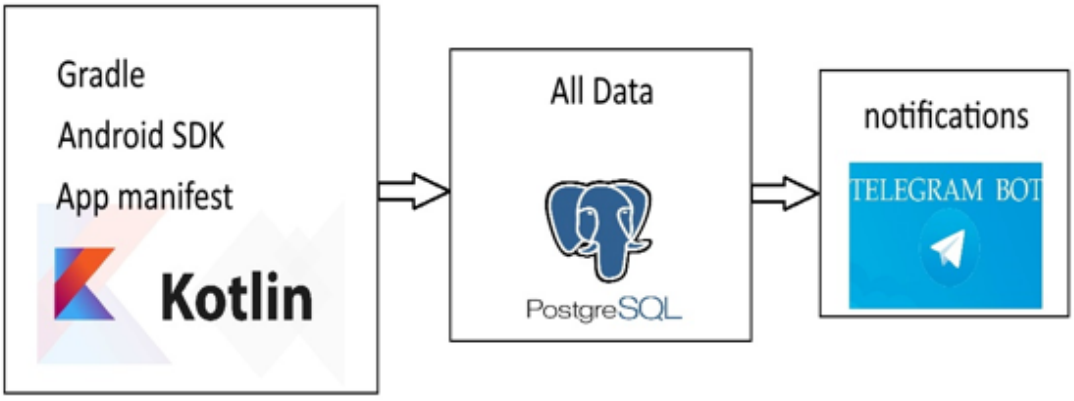


Рисунок 1 – Архитектура приложения

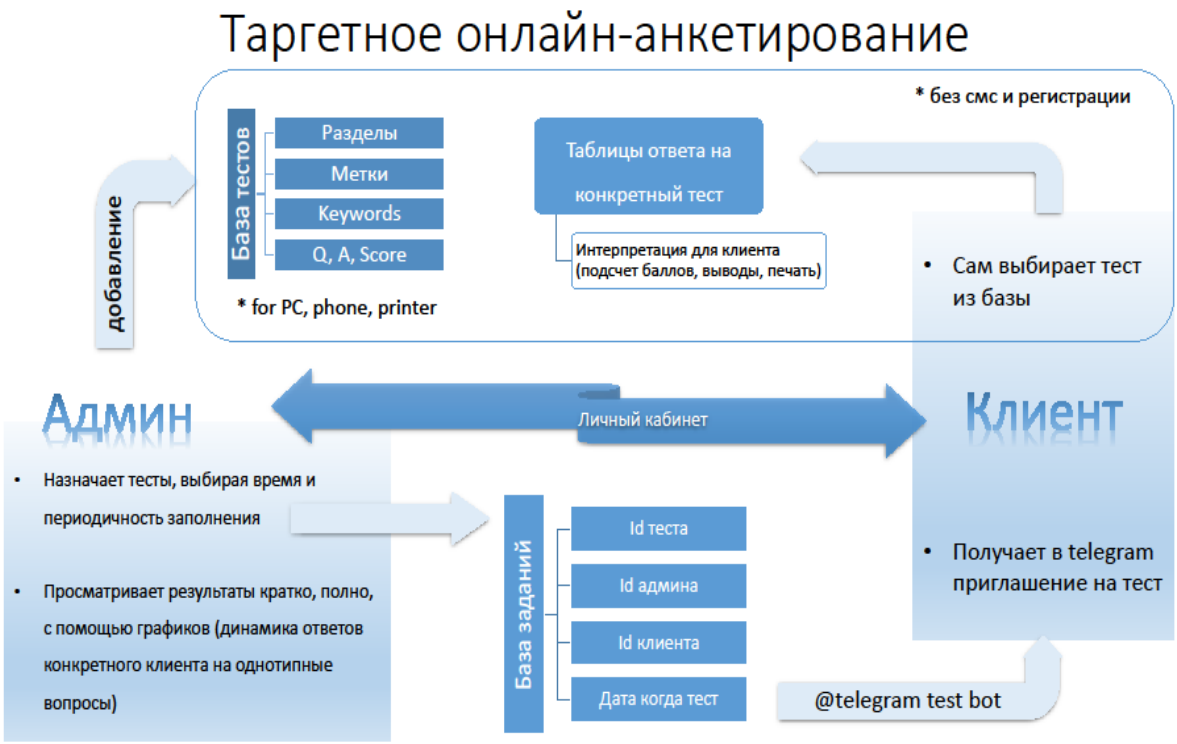


Рисунок 2 – Общее описание функционала.

На Главной странице сервиса представлено:

- Список медицинских опросников с возможностью поиска (по медицинским специальностям согласно Общероссийскому классификатору специальностей и по заболеваниям согласно классификации МКБ-10[1]);

- Блок с предложением возможности регистрации/авторизации врача в личном кабинете;

Страница Теста содержит:

- Заголовок теста, ключевые слова, описание, инструкцию по заполнению, ссылку на источник, текст вопросов, интерпретация ответов, кнопки назначить тестирование, онлайн-тест и версия для печати;

- Вопросы (текст) и Ответы, выбираемые из нескольких вариантов (визуально-аналоговая шкала);

- Подсчет баллов в виде общей суммы, а также возможности подсчета сумм по конкретным вопросам;

- Интерпретация в виде вывода различного текста в зависимости от значений;

В Телеграмм аккаунте клиента с помощью бота можно перейти по оповещению на конкретную анкету, выбранную врачом для пациента.

ЛИТЕРАТУРА

1. Международная классификация болезней 10-го пересмотра (МКБ-10) URL: https://mkb-10.com (дата обращения: 08.09.2021).
2. Э. Гамма, Р. Хелм, Р. Джонсон, Д. Влиссидес Приемы объектно-ориентированного проектирования. Паттерны проектирования. - СПб.: Питер: 2009. - 366 с.

УДК 004.912

Шандакова Г.С. (2 курс магистратуры),
Леонтьева Т.В., к.т.н., доцент

ОБНАРУЖЕНИЕ И СКРЫТИЕ НЕЖЕЛАТЕЛЬНОГО КОНТЕНТА

Как много времени мы тратим на серфинг интернета? Сложно представить хоть один день современного человека без проверки почты, общения в соцсетях, просмотра новостей и прослушивания музыки. Согласно глобальному исследованию проведенному компанией We are social за 2022 год[3] среднестатистический пользователь интернета в России находится онлайн 7 часов 50 минут. Из которых около 3 часов проводит в социальных сетях и еще час тратит на чтение новостей. За последние несколько лет наша жизнь все более и более привязывалась к цифровому миру. Это цифровизация имеет немало плюсов, однако есть и негативные последствия.

Психологи считают, что вместе с социальными сетями в нашу жизнь могут прийти стресс и тревожность[6]. Свобода выражения различных точек зрения, включая агрессивные и оскорбительные комментарии, может иметь долгосрочное негативное влияние на мнения людей и социальную сплоченность. Если вы когда-нибудь ловили себя на непрерывном чтении плохих новостей, то вы точно знаете, как тяжело оторваться от этого занятия. Данный феномен получил своё название — думскроллинг (от англ. doomscrolling). Специалисты приходят к мнению, что данное увлечение плохими новостями может привести к вредным психофизиологическим реакциям[1].

Контент, который мы потребляем действительно сильно на нас влияет — это признают и компании. Поэтому каждая компания, создающая площадки для размещения контента, всегда заботится о его модерации. Модерация контента — это процесс проверки размещаемой информации, на соответствие определенным правилам. Модерация предполагает первичную фильтрацию контента. Со временем модерация из полностью ручной области движется к автоматизации. На текущий момент в большинстве платформ, таких как youtube и facebook используется модерация как на основе алгоритмов, так и ручная[4][5]. Стоимость собственной разработки таких систем для модерации довольно высока, поэтому на рынке существуют решения, позволяющие развернуть систему для проверки и фильтрации “из коробки”. Из выдающихся примеров таких решений можно отметить Utopia AI Moderator[7], Community Sift[2], WebPurify[8]. Каждое решение способно модерировать текстовый, графический и видео-контент на более чем 15 языках, также к каждому решению можно запросить улучшение под собственные потребности.

Можно подумать, если на рынке уже существует множество решений для модерации, то проблем с контентом возникать не должно. Однако это не совсем так. Каждая платформа проводит модерацию и отбор контента согласно своим правилам. Учитывая, что правила обычно строятся на запрете определенных видов контента (например, бранных слов), но при этом стараются учитывать свободу слова, мы имеем довольно большую часть контента, которая пройдет правила, но будет по-прежнему негативно сказываться на пользователях. Также проблема думскроллинга на данный момент не решается компаниями совсем. Так как невыгодно останавливать пользователя от поглощения контента на площадке.

В качестве решения хочется предложить другой вариант схемы работы с контентом. В обычном нам виде каждая платформа занимается модерацией сама, как альтернативу этому можно предложить личный фильтр пользователя для всех площадок. Такой фильтр должен скрывать нежелательный контент, который задается персонально для каждого пользователя. Так мы не накладываем жестких правил к контенту, но при этом защитим пользователя от контента, который он бы не хотел видеть. К фильтру необходимо добавить возможность вводить свои стоп-слова, и таймер на чтение, который бы по истечению времени скрывал определенный тип контента. Для реализации такого фильтра предлагается использовать

плагины в браузере, которые пропускают через себя страницу браузера и выдают её же в уже отформатированном виде. Общая схема работы такой системы предоставлена на Рисунке 1.

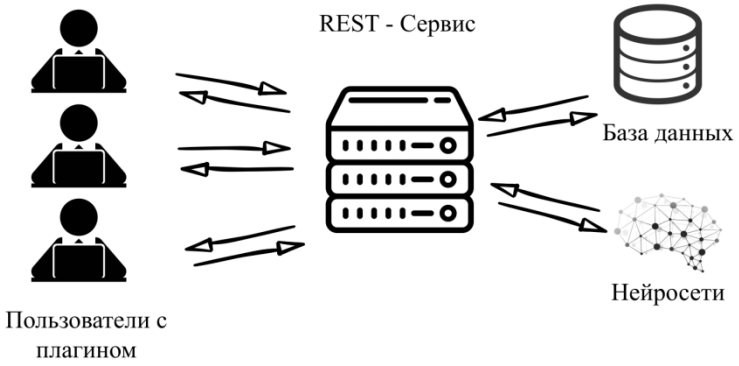


Рисунок 1 – Схема работы личных фильтров

Таким образом система состоит из frontend части — плагина в браузере у пользователя, который разбирает html страницы, находит элементы с текстом, отсылает их в backend часть и по получению типа контента, если пользователь предпочел его скрыть, генерирует графический элемент, закрывающий его. Backend часть содержит в себе логику авторизации пользователя, настройки (стоп-слова, типы контента), методы для доступа к нейросетям. Нейросети в этой схеме отвечают за задачу классификации контента, понадобятся несколько для распознавания разного контента: текста или изображений.

Для пользователя пользование решением не потребует больших усилий. Для работы потребуется установить плагин в браузер, выбрать из списка сайтов те, на которых будет работать плагин и отметить типы нежелательного контента, например, бранные слова и новости. При следующем заходе на выбранные сайты отмеченные виды контента будут скрыты.

Предполагается что такое решение проблемы поможет избежать избыточной цензуры и при этом обеспечит более здоровое времяпрепровождение в глобальной сети. Также стоит заметить, что это решение позволит использовать наработки по фильтрации контента, так как архитектура не привязана к конкретной платформе/правилу/нейросети и весьма гибкая для внесения изменений и новых наработок.

ЛИТЕРАТУРА

1. Anand N, Sharma MK, Thakur PC, et al. Doomsurfing and doomscrolling mediate psychological distress in COVID-19 lockdown: Implications for awareness of cognitive biases. Perspect Psychiatr Care. 2022;58(1):170-172. doi:10.1111/ppc.12803
2. Content Moderation Platform. [Электронный ресурс]. Режим доступа: https://www.twohat.com/solutions/content-moderation-platform/
3. Digital 2022: The Russian Federation. [Электронный ресурс]. Режим доступа: https://datareportal.com/reports/digital-2022-russian-federation
4. Facebook is now using AI to sort content for quicker moderation. [Электронный ресурс]. Режим доступа: https://www.theverge.com/2020/11/13/21562596/facebook-ai-moderation
5. Hunter Walk. Internet Content Moderation. 5 December 2017. [Электронный ресурс]. Режим доступа: https://hunterwalk.com/2017/12/05/internet-content-moderation-101/
6. Ing c.e. Hate speech in social media: An exploration of the problem and its proposed solutions. Doctoral Dissertation.2013
7. Utopia AI Moderator. [Электронный ресурс]. Режим доступа: https://utopiaanalytics.com/utopia-ai-moderator/
8. World-Class Image Moderation & More. [Электронный ресурс]. Режим доступа: https://www.webpurify.com

УДК 004.42

Щепетов В. В. (4 курс бакалавриата),
Никифоров И. В., к.т.н., доцент,
Самочадин А. В., к.т.н., доцент

АЛГОРИТМ ПОСТРОЕНИЯ ГРАФОВОГО ПРЕДСТАВЛЕНИЯ СЦЕНАРИЯ ПОВЕДЕНИЯ ОБЪЕКТА

В инструменте имитационного моделирования, часть которого будет описана в ходе работы, моделируются сценарии поведения объектов в реальном мире за счет создания поведенческих сценариев [3]. Так как сценарий – это набор связанных между собой действий, выполняемых синхронно или асинхронно, то наиболее подходящей структурой данных для их хранения и исполнения является граф, где вершины – этой действия, а направленные ребра между ними – это переходы от выполнения одного действия к выполнению следующего. Такие графы будем называть сценарными.

Так как при вводе пользователем информации о моделируемом поведении объекта, подаваемой на вход программной системы моделирования, могут возникнуть ошибки (как синтаксические, так и семантические), является целесообразным не допускать до процесса моделирования ошибочные наборы данных [2]. Отделять верно заполненные сценарии от ошибочных можно с помощью модуля проверки сценарных графов.

Для более удобной и эффективной работы со сценариями предлагается изменить традиционный сценарный граф таким образом, чтобы отражать на нем динамику выполнения сценария, например, с помощью добавления туда дополнительных вершин и рёбер.

Алгоритм расширения графа описан ниже.

Первый этап построения расширенного сценарного графа – построение традиционного сценарного графа на основе json-описания сценария. Затем вершины-действия расщепляются на два узла: узел типа «действие», который отражает смысл вершины-действия в традиционном сценарном графе, и узла типа «условие», который проверяет результат сполнения непосредственно предшествующего узла типа «действия». На третьем и четвертом этапе строятся новые рёбра-переходы так, как показано на Рисунке 1.

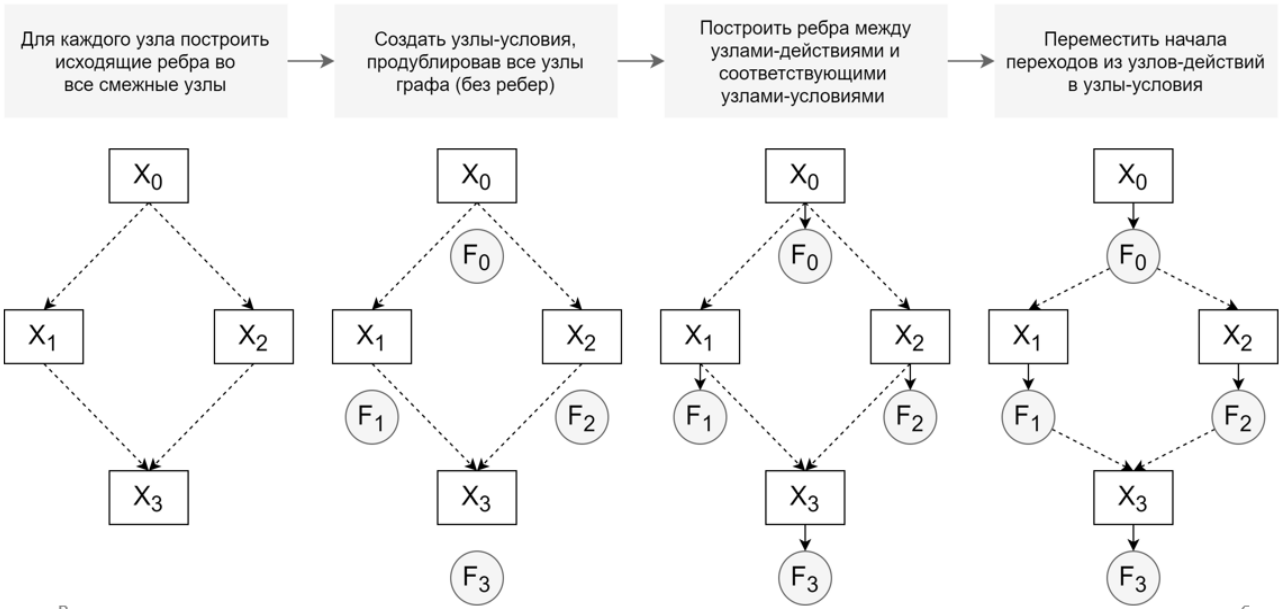


Рисунок 1 – Алгоритм расширения графа

Алгоритм проверки сценарного графа состоит из двух ступеней и описан ниже.

Валидация сценария происходит в процессе нескольких последовательных проверок, разбитых на две ступени. Первая – проверка структуры json-файла. Сначала проверяется

наличие всех обязательных полей. Если отсутствующих полей нет, проверяется заполненность всех полей. Вторая ступень – проверка графа, построенного по данным из json-файла. Она включает проверку на наличие повторяющихся идентификаторов состояний и проверку на наличие циклов в графе сценария.

Для реализации модуля отображения графа были использованы методы NetworkX – библиотеки для исследования сетевых структур, которая заимствует методы библиотеки Matplotlib для визуализации данных [1].

Результат реализации алгоритма построения расширенного сценарного графа можно продемонстрировать фрагментом пользовательского интерфейса (изображенного на Рисунке 2), на котором процесс результат моделирования отражены в виде расширенного сценарного графа.

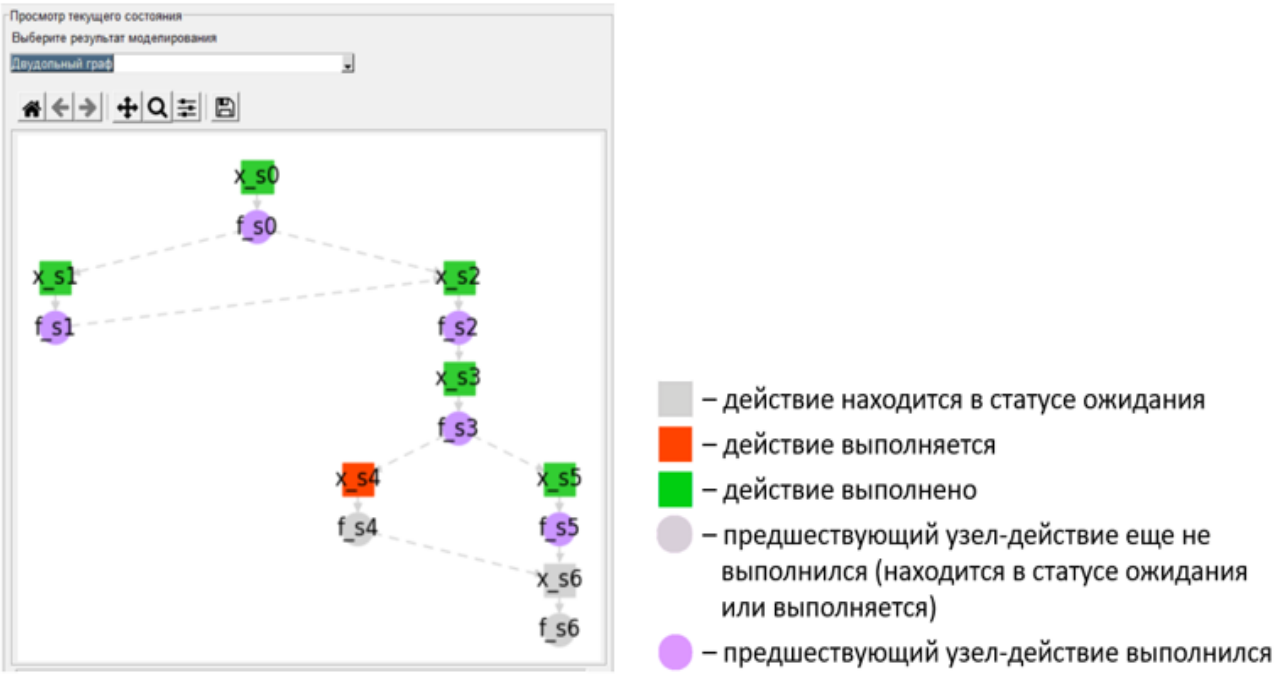


Рисунок 2 – Расширенный граф в пользовательском интерфейсе

ЛИТЕРАТУРА

1. NetworkX Documentation. [Электронный ресурс]. Режим доступа: https://networkx.org/documentation/stable/index.html
2. А. П. Маслаков, И. В. Никифоров, Л. П. Котлярова, Статический и динамический анализ UCM-модели на наличие синтаксических и семантических ошибок // Информатика и кибернетика (ComCon-2015): сборник докладов студенческой научной конференции Института информационных технологий и управления, Санкт-Петербург, 20–24 апреля 2015 года / Н. М. Вербова (отв. ред .). – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2015. – С. 143-146.
3. И. А. Селин, И. В. Никифоров, Н. В. Воинов, В. П. Котляров, Инструмент анализа покрытия ветвей UCM-модели тестовыми сценариями // Информатика и кибернетика (ComCon-2015): сборник докладов студенческой научной конференции Института информационных технологий и управления, Санкт-Петербург, 20–24 апреля 2015 года / Н. М. Вербова (отв. ред .). – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2015. – С. 140-143.

УДК 004.4

Полевич С.О. (2 курс магистратуры),
Леонтьева Т.В., к.т.н., доцент

ПРИЛОЖЕНИЕ ДЛЯ АНАЛИЗА ЗОН СЕРДЕЧНОГО РИТМА С ИНТЕРАКТИВНЫМ ПОСТРОЕНИЕМ ПРОГРАММ ТРЕНИРОВОК

Целью данной работы является исследование возможностей фреймворка React Native, вычисления и анализ данных, полученных с носимых устройств с последующим построением рекомендаций по физической активности. Носимые устройства приобретают массовую популярность среди российских пользователей. Драйвером рынка служат спортивные устройства, мотивирующие людей следить за физическими нагрузками и вести здоровый образ жизни. Так, в настоящее время для контроля физической активности многие люди используют различные фитнес-гаджеты: пульсометры, шагомеры, умные часы, а также фитнес-браслеты. Особое внимание, необходимо уделить именно фитнес-браслетам. Преимущество данного гаджета заключается в том, что он даёт людям возможность не только следить за своей физической активностью (подсчитывать количество шагов, пройденное расстояние, сожженные калории), но и контролировать физическое состояние (следить за пульсом, сном, давлением). В связи с совершенствованием технологий, данные, получаемые от использования браслета, предоставляются их владельцу достаточно точными и приближенными к реальным значениям. По итогам 1 квартала 2020 года рынок носимых устройств значительно увеличился. Так, за анализируемый период в России было продано более 1,4 млн. фитнес-трекеров и умных часов на общую сумму почти 8,7 млрд. рублей, что больше в сравнении с 2019 годом на 54% в натуральном выражении и 43% в денежном.

Как мы видим, большое количество людей использует фитнесс браслеты. И чаще всего пользование данных девайсов ограничивается простым мониторингом различных физических показателей для самоконтроля. Однако, эти данные возможно использовать для построения и контроля тренировочного процесса.

Для продуктивной и полезной тренировки необходим баланс. Если вы переусердствуете, то вы может увеличить риск травмы и выгорания. Для этого измеряется интенсивность тренировки с использованием сердечного ритма. Американская кардиологическая ассоциация рекомендует целевой пульс:

- Умеренная интенсивность физических упражнений: от 50% до 70% от максимального сердечного ритма
- Интенсивные физические упражнения: от 70% до 85% от максимального сердечного ритма.

Если вы стремитесь к целевой частоте сердечных сокращений в активном диапазоне от 70% до 85%, вы можете использовать метод резерва сердечного ритма (HRR). Для вычисления HRR используется ряд рекомендаций [6].

В результате вычислений мы получаем два числа. Они являются вашей средней целевой зоной сердечного ритма для интенсивности физических упражнений при использовании HRR для расчета частоты сердечных сокращений. Частота сердечных сокращений во время энергичных упражнений, как правило, должна быть между этими двумя цифрами.

Чтобы избавить пользователя от утомительной процедуры подсчета параметров, данное приложение должно автоматизировать вышеперечисленные рекомендации. Анализируя полученные данные, создается программа тренировок. Так как достаточно большое количество пользователей использует как операционную систему iOS, так и Android, было принято решение реализовать кроссплатформенное решение. В ходе сравнения инструментов решения задачи выбор пал на фреймворк ReactNative.

Данный фреймворк позволяет создавать приложение для различных платформ, используя одну и ту же кодовую базу. Например, мы можем разрабатывать полномасштабные мобильные приложения как для iOS, так и для Android, используя один язык, то есть JavaScript,

что экономит много времени. Архитектура React Native выступает неким переводчиком, где на одной стороне ваше приложение, набранное на языке JavaScript, а на другой, уже преобразованное самим фреймворком, Objective-C/Swift для iOS или Java/Kotlin для Android. Кроме этого, React Native приложение имеет web версию, однако, для данной работы это не востребовано, так как носимые устройства чаще всего используются именно со смартфонами.

Цель разрабатываемого приложения, предоставить «тренера», который получает результаты вашей активности в реальном времени и соответственно им дает рекомендации по ведению тренировочного процесса. Функционал приложения на начальном этапе предоставляет набор базовых интерактивных тренировок для различных целей (силовые, аэробные и т.п.), сводка осуществляемого прогресса в виде графиков. Также приложение имеет ряд перспективных возможностей для развития, например, более детальные тренировки для людей с некоторыми ограничениями. Кроме этого, рынок фитнесс браслетов постоянно растет и развивается, с новыми моделями трекеров добавляются новые датчики, которые позволяют более детально и точно анализировать реакцию организма и соответственно этому давать более полезные индивидуальные рекомендации.

ЛИТЕРАТУРА

1. Информационно-аналитическое агентство NBPrice.RU: новости, обзоры, аналитика о лаптопах, смартфонах и носимых устройствах [Электронный ресурс] // – URL: http://www.nbprice.ru/info/details/23370
2. Рынок носимых IoT- гаджетов в 2020 г. [Электронный ресурс] // Технологии и средства связи. – URL: https://www.tssonline.ru/news/rinok-nosimih-iot-gadjetov-v-2020g-prodolzhit-rost
3. Фитнес браслеты – мониторинг будущего [Электронный ресурс] // Manor Medical Center. – URL: https://manormedicalgroup.com/novosti/fitnes-braslet-monitoring-budushhego/
4. Цветкова, А.Б. Оценка восприятия цифровой медицины молодежным сегментом потребителей [Электронный ресурс] / А.Б. Цветкова, А.В. Шишкин // Статистика и экономика. – 2018. – No6. – С. 46-56. – URL: https://cyberleninka.ru/article/n/otsenka-vospriyatiya-tsifrovoy-meditsiny-molodezhnym-segmentom-potrebiteley
5. Elevated resting heart rate, physical fitness and all-cause mortality, [Электронный ресурс]. Epidemiology, 2013. – URL: http://heart.bmj.com/content/99/12/882.full?sid=90e3623c-1250-4b94-928c-0a8f95c5b36b
6. Target Heart Rate and Estimated Maximum Heart Rate, [Электронный ресурс] // Centers for Disease Control website – URL: https://www.cdc.gov/physicalactivity/basics/measuring/heartrate.htm

Секция «Программная инженерия: инструментальные средства и технологии проектирования и разработки»

УДК 681.3.06

Белореченский Д.А., Сараев М.И. (3 курс бакалавриата),
Семенова-Тян-Шанская В.А., к.т.н., доцент

РАЗРАБОТКА ВЕБ-ПРИЛОЖЕНИЯ ДЛЯ АНАЛИЗА ХАРАКТЕРИСТИК СУДОВ С ИСПОЛЬЗОВАНИЕМ МЕТОДОВ ETL

Известно, что средства ETL процесса выполняют задачу извлечения данных из источников, подвергают данные трансформации для приведения формата данных к формату, принятому в хранилище, и загружают данные в хранилище данных, попутно выполняя согласование с уже имеющимися значениями в базе данных [1]. Целью представленной работы являются следующие задачи:

1. Разработать базу данных, которая хранит данные о судах Российского речного регистра.
2. Произвести анализ исходных (“грязных”) данных, предоставленных в виде документов Excel, содержащих большое количество ошибок и несоответствий
3. Произвести очистку данных, их структурирование, приведение к единому виду. Далее загрузить очищенные данные в созданную базу данных.
4. Разработать веб-приложение, позволяющее производить CRUD операции в базе данных.
5. Построение фильтров к базе данных.

ETL процесс, реализованный в приложении, схематично показан на Рисунке 1.

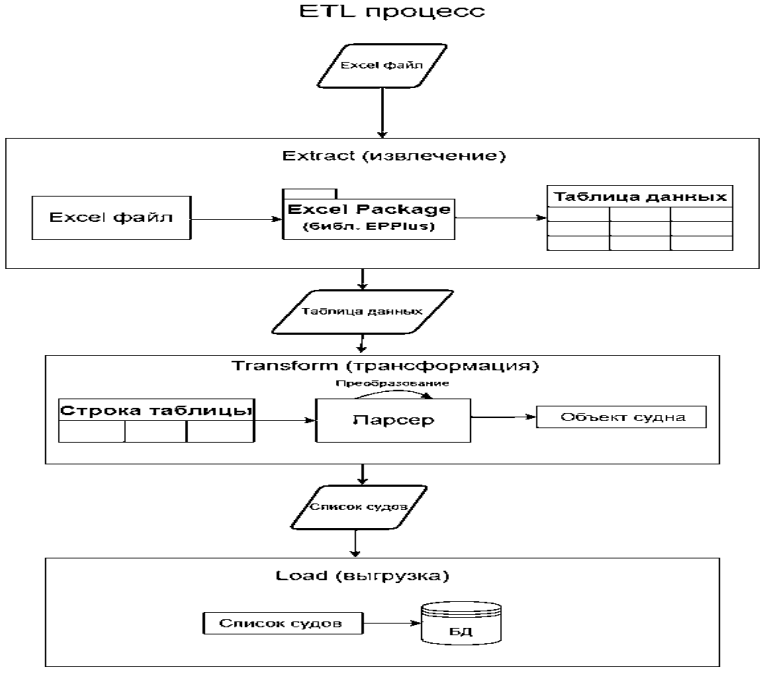


Рисунок 1 – Схема ETL процесса

Записи в таблицах Excel имеют множество ошибок и несоответствий (Таблица 1), поэтому требуется разработать парсер, который будет структурировать и приводить к единому виду записи из документов Excel и вставлять обработанные записи в базу данных.

Таблица 1

Типы ошибок в Excel таблице	Описание
Теплоход пассажирский Пассажирский теплоход	Одинаковый тип судна, но слова в разном порядке
Несамоходный плавкран Н/с плавкран	Одинаковый тип судна, но в названии одного из типов присутствует сокращение
Танкер для перевозки нефтепродуктов с t всп.паров >60° С Танкер, перевозка нефтепродуктов с Твсп.>60Гр.С	Требуется отделить тип судна (Танкер) от его назначения (перевозка нефтепродуктов с $T_{всп.} > 60°C$)

Структура базы данных основана на полях соответствующих документов Excel (Рисунок 2).

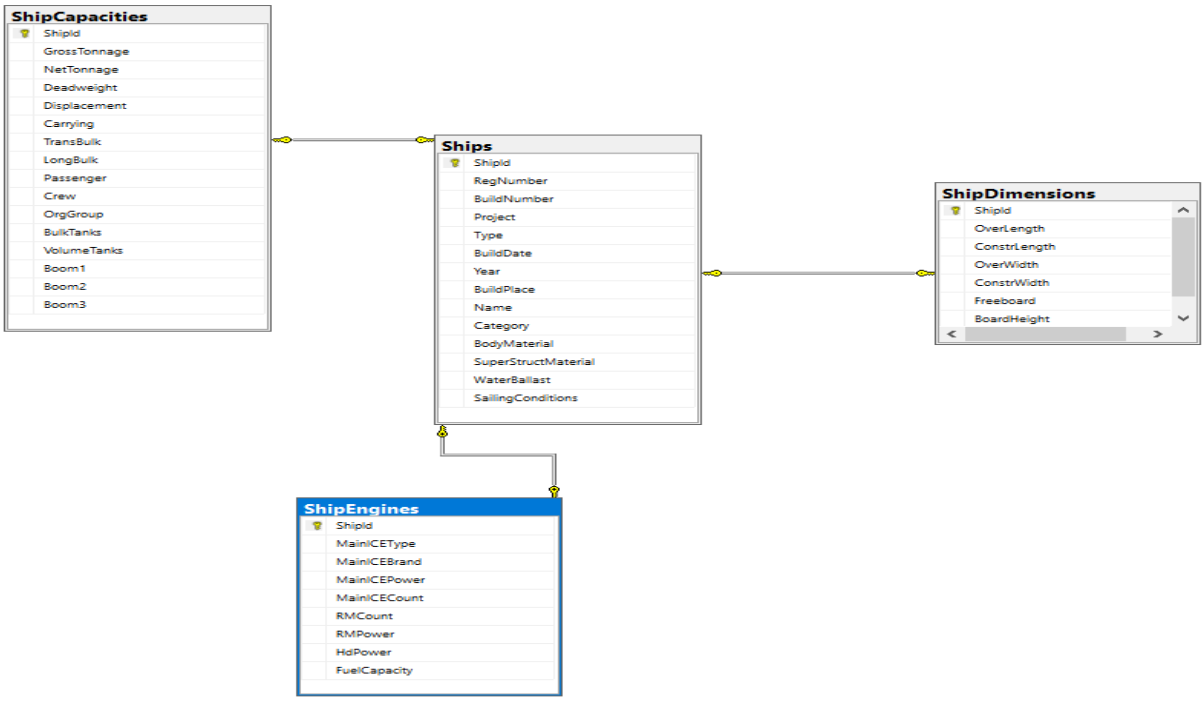


Рисунок 2 – Схема базы данных

В работе разработано веб-приложение, которое направлено на извлечение необходимых данных из гетерогенных источников (в Excel файлах и на веб-сайтах) и преобразование их в базу данных. Для его реализации используется следующие технологии: серверная часть (платформа - ASP.NET Core 5 [2], СУБД – MS SQL Server, ORM – EF Core, библиотеки – EPPlus (для работы с таблицами Excel) [3], AngleSharp - Html парсинг; клиентская часть: *UI-Фреймворк - React.*

Фильтрация судов происходит на стороне сервера. Поиск с фильтром осуществляется с помощью GET запроса /api/Ship[?param=…&param2=…]. Пример фильтра: найти все судна, которые построены до 1960 года, материал корпуса – сталь, город постройки – Астрахань. Ответ с сервера представлен на Рисунке 3 и включает: 1) необходимую информацию для реализации пагинации на стороне клиента, 2) список судов.

Рисунок 3 – Ответ с сервера

ЛИТЕРАТУРА

1. Барсегян А. А., Куприянов М. С., Степаненко В. В., Холод И. И. Методы и модели анализа данных: OLAP и Data Mining. СПб.: БХВПетербург, 2004. С. 27-44, 49-66
2. ASP.NET MVC 5 [Электронный ресурс]: Документация по продукту; Руководства. – Режим доступа: https://docs.microsoft.com/en-us/aspnet/mvc/mvc5
3. Библиотека EPPlus [Электронный ресурс]: Документация по продукту; Руководства. – Режим доступа: https://epplussoftware.com/

УКД 004.4

Белошицкий Д.Р. (2 курс бакалавриата),
Петров А.В., старший преподаватель

ИСПОЛЬЗОВАНИЕ ФРЕЙМВОРКА ARKIT ДЛЯ СОЗДАНИЯ МОБИЛЬНЫХ AR-ПРИЛОЖЕНИЙ

Актуальность работы обусловлена активному распространению дополненной реальности во все сферы жизни, начиная от медицины, заканчивая играми (яркий пример — игра Pokemon GO).

Целью работы является исследование особенностей и возможностей нового фреймворка Apple ARKit, создание SWOT-анализа технологии, а также её сравнение с существующими аналогами.

В 2017 году на конференции WWDC компания Apple анонсировала SDK ARKit, как универсальный инструмент для разработки мобильных приложений для iOS,

взаимодействующих с дополненной реальностью. ARKit является высокоуровневым API, который предлагает разработчику простой интерфейс, но в то же время широкую функциональность.

Ключевые особенности технологии:

1. Распознавание движений с помощью ИВО (инерциальной визуальной одометрии). Визуальная одометрия — метод отслеживания положения объекта, с помощью анализа изображений, полученных с камеры устройства. Более того, устройство сопоставляет полученную информацию с камеры с информацией с датчиков (CoreMotion data). Комбинация камеры и датчиков позволяет устройству понимать, где оно находится в данный момент с высоким уровнем точности. Стоит отметить, что ИВО не требуется предварительная калибровка или изначальная информация о локации, где планируется использование устройства.

2. Продвинутое распознавание окружающей среды: с использованием ARKit устройство может анализировать поверхности с помощью камеры и точно определять горизонтальные поверхности вокруг. ARKit так же использует сенсор камеры для расчёта света вокруг и его дальнейшего наложения поверх размещенных виртуальных объектов. A hit-testing позволяет понимать топологию реального мира для корректного размещения объектов поверх существующих (например, поставить виртуальную лампу на реальный стол).

3. Поддержка популярных графических движков: ARKit предоставляет непрерывный поток изображений и информации о них, сведений об окружающей среде, которые могут служить входом для разных движков, например, Apple Metal, а также решений от сторонних разработчиков, например, Unity или Unreal Engine.

Процесс создания приложения:

Первый этап — это создание ARSession. ARSession — это объект, который контролирует все процессы, связанные с дополненной реальностью. Для конфигурации настроек сессии нужно использовать класс ARSessionConfiguration, который дает возможность тонкой настройки.

Второй этап — это запуск ARSession, который создает AVCaptureSession (объект, ответственный за обработку изображений) и CMMMotionManger (объект, ответственный за обработку движений). ARSession объединяет полученную информацию и выдает объект ARFrame. ARFrame представляет собой контейнер, содержащий текущее состояние сессии и информацию для последующей обработки сцены.

Третий этап — передать полученный ранее ARFrame в графический движок и получить требуемый результат.

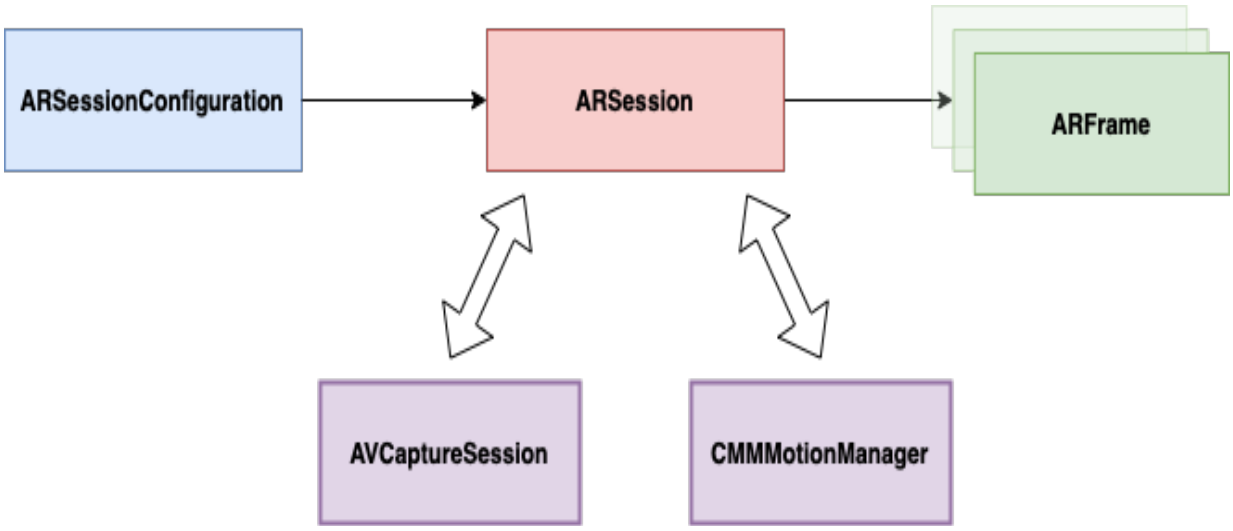


Рисунок 1 – Схема работы ARKit

Таблица 1 – SWOT-анализ технологии

Сильные стороны: – Глубокая интеграция в систему iOS; – Бесплатное распространение; – Исчерпывающая документация и наличие большого количества примеров; – Низкий порог входа для конечного пользователя (поддерживаются устройства, начиная с iPhone 6s)	Возможности: – Спрос на AR-приложения растет с каждым годом; – AR-приложения могут быть использованы в большой количестве сфер; – Растущее число пользователей с устройствами, имеющих поддержку ARKit;
Слабые стороны: – Отсутствие поддержки устройств на ОС Android; – В некоторых сильно загруженных локациях ARKit может работать не так точно, как этого ожидает разработчик;	Угрозы: – Конкуренция с Google ARCore, PTC Vuforia и аналогами;

Сравнительный анализ существующих технологий для создания приложений, работающих с дополненной реальностью: к прямым конкурентам ARKit можно отнести Google ARCore и PTC Vuforia. В сравнении с ARKit они обладают большим количеством поддерживаемых устройств (за счет поддержки Android устройств) и списком поддерживаемых графических движков. Но если рассматривать только разработку под iOS, то нативная поддержка устройств и глубокая интеграцию в систему, а также оптимизации связанные с распознаванием пространства у ARKit перевешивают все сильные стороны конкурентов.

Дополненная реальность открыла пользователям новый способ взаимодействия с привычным миром, а разработчикам поставила новые задачи по программной реализации этого взаимодействия. Благодаря современным фреймворкам, таким как ARKit, интеграция AR в приложения стала проще, а итоговый результат качественнее. Более того, компании активно дорабатывают свои решения, расширяя их функциональность.

ЛИТЕРАТУРА

1. ARCore Documentation. [Электронный ресурс]. Режим доступа: https://developers.google.com/ar/develop
2. ARKit Documentation. [Электронный ресурс]. Режим доступа: https://developer.apple.com/documentation/arkit/
3. Vuforia Developer Library. [Электронный ресурс]. Режим доступа: https://library.vuforia.com
4. Apple ARKit press release. [Электронный ресурс]. Режим доступа: https://www.apple.com/newsroom/2018/06/apple-unveils-arkit-2/
5. AR – Habr. [Электронный ресурс]. Режим доступа: https://habr.com/ru/post/419437/

УДК 004.4

Амирасланов Э.Г. (2 курс магистратуры)
Леонтьева Т.В., к.т.н., доцент.

СРАВНИТЕЛЬНЫЙ АНАЛИЗ СПОСОБОВ РАСПОЗНАВАНИЯ РЕЧИ

В статье рассматриваются существующие методы распознавания речи. Объектом исследования являются современные технологии распознавания речи и методы построения систем распознавания речи. Целью создания систем распознавания речи является способность машины распознавать слова с эффективностью, равной человеческой. Важным направлением автоматизированных систем является создание систем голосового управления, например, для

пунктов пропуска, в системе «умный дом», в голосовом управлении телефоном, для облегчения жизни людей с ограниченными возможностями.

В Таблице 1 приведены данные сравнительного анализа эффективности распознавания отдельных слов:

Таблица 1 – Сравнительный анализ систем распознавания речи

Система	WER, %	WRR, %	SF
HTK	19,8	80,2	1.4
CMU Sphinx (pocketsphinx/sphinx4)	21.4/22.7	78.6/77.3	0.5/1
Kaldi	6.5	93.5	0.6
Julius	23.1	76.9	1.3
iAtros	16.1	83.9	2.1
RWTH ASR	15.5	84.5	3.8

WER (Word Error Rate) – процент ошибок в распознанных словах

WRR (Word Recognition Rate) – процент правильно распознанных слов

SP (Speed Factor) – скорость распознавания

Исходя из данных таблицы, можно сделать вывод, что система распознавания Kaldi имеет наименьший процент ошибок при распознавание отдельных слов, наибольший процент распознанных слов и занимает лидирующие позиции по скорости распознавания.

Именно эту систему распознавания речи, я буду использовать в своей работе по разработке модуля голосового управления для системы “умный дом”. Kaldi способна предоставить пользователю наиболее богатый выбор алгоритмов для разных задач и очень удобна в использовании.

Система позволяет использовать множество алгоритмов для уменьшения размера акустических признаков сигнала, и, соответственно, увеличивать производительность системы. Kaldi написана на языке программирования C++, что положительно сказывается на скорости работы системы, и имеет модульную структуру. С точки зрения удобства использования Kaldi, также является одной из первых систем. Она предоставляет подробную документацию. Она кроссплатформенна, то есть запускается на большинстве современных операционных систем.

Можно выделить пять основных способов распознавания речи:

1. Распознавание отдельных команд – требуется раздельно произносить слово\словосочетание, а затем проводится его распознавание. Качество распознавания данного способа ограничено размером звукового сигнала.

2. Распознавание по грамматике –проводится распознавание уже по фразам, которое соответствует определенному набору правил.

3. Поиск ключевых слов в потоке слитной речи – здесь речь вполне может быть, как соответствующая набору неких правил, так и соответствующей определённым правилам.

4. Распознавание речи с помощью нейронных систем – является довольно сложным методом, однако на основе нейронных сетей появляется возможность создания обучаемых и самообучаемых систем.

В своей работе использую первый способ, а именно распознавание отдельных команд. Опишу поэтапную реализацию данного способа в моем случае. После того, как пользователем произносится речь, происходит фильтрация речи от шума. Далее в обработку вступает модуль Alphacep, являющийся неотъемлемой частью работы системы Kaldi. Модуль в основе своей работы использует акустическую и языковую модель. Акустическая модель определяет какой набор фонем соответствует звуковому сигналу. Этому она учится на большом корпусе начитанных дикторами текстов и затем сопоставляет каждому звуку определенный фонему. Акустическая модель, обработав частотные признаки фрейма, выдаёт не одну конкретную фонему, а несколько — и у каждой из них свой коэффициент вероятности. Языковая модель работает уже не с признаками звука, а с цепочкой вероятных фонем. Как и акустическая модель, языковая тоже обучается на большом корпусе текстов. Далее полученный текст, с распознанными командами, сравнивается с командами, прописанными в базе данных. При выявлении совпадений – происходит взаимодействие с устройством. Если же совпадений не выявлено, или же после прохождения акустической и языковой модели вовсе никаких команд программа не получила, пользователю будет передано соответствующее сообщение и план дальнейших действий.

ЛИТЕРАТУРА

1. Фланаган Дж.Л. Анализ, синтез и восприятие речи / пер. с англ. А. А. Пирогова. М.: Связь, 1968. 397 с
2. Вишнякова О. А., Лавров Д. Н. Применение преобразования Гильберта-хуанга к задаче сегментации речи // Математические структуры и моделирование. 2011. вып. 24. С. 12–18
3. Kaldi [Электронный ресурс]. – URL: http://kaldi-asr.org/doc (дата обращения: 19.02.2022)

УДК 004.51

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

ОСНОВНЫЕ ПРОБЛЕМЫ ПРИ РАЗРАБОТКЕ ГРАФИЧЕСКОЙ СОСТАВЛЯЮЩЕЙ МОБИЛЬНОГО ПРИЛОЖЕНИЯ

Целью работы является исследование и анализ различных техник, приемов и технологий, используемых при создании графической составляющей мобильного приложения.

В ходе работы будут исследованы основные проблемы при разработке графической составляющей мобильного приложения, а также будут предложены способы их решения.

Разработка пользовательского интерфейса проходит два этапа: разработка UX и разработка UI составляющих. Основная цель человека, разрабатывающего UX продукта – сделать опыт работы пользователя с приложением наиболее приятным. При этом желательно затратить разумное количество времени и ресурсов на разработку интерфейса приложения.

Специалист, работающий с UI отвечает, как за функциональность, так и за внешний вид интерфейса.

Чем интерфейс на компьютере отличается от разработки интерфейса на мобильном устройстве? Разработка UX/UI для мобильных приложений осложнена многими факторами, относящимися исключительно к мобильным устройствам. Ограничения включают скромную вычислительную мощность, относительно небольшой объем памяти, меньший размер дисплея, меньший размер, а также ограниченный заряд батареи. Также, работая с компьютером, пользователь ограничен функционалом клавиатуры и мыши, но на мобильном устройстве он может выполнить гораздо больше различных действий, так как на большинстве мобильных устройствах сейчас имеется сенсорный экран.

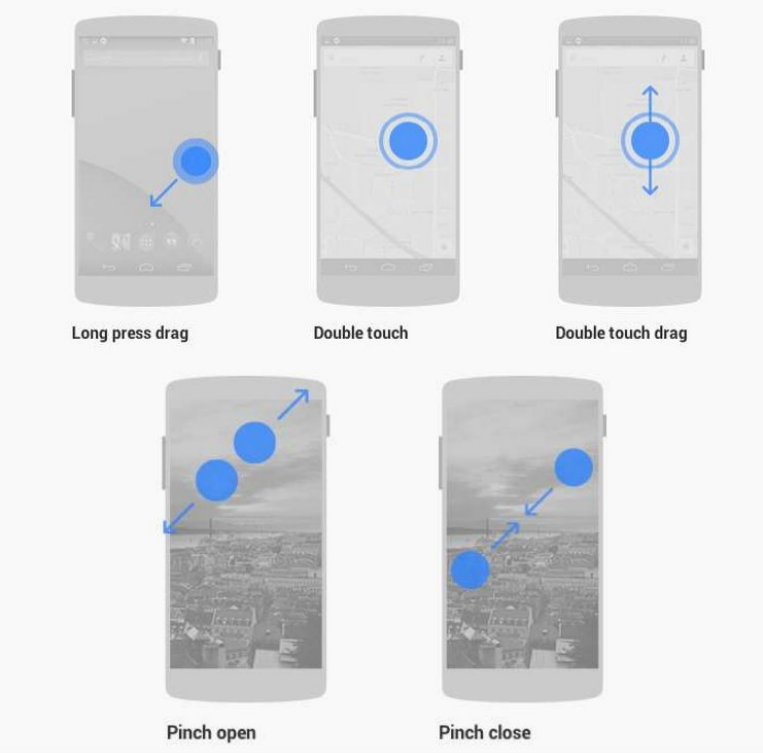


Рисунок 1 – Типы касаний

В связи с этим были разработаны дизайн паттерны для мобильных приложений.

Дизайн паттерны представляют собой решение часто встречающихся проблем. Они также представляют собой решение, которое обеспечит пользователю наиболее легкий опыт обращения с приложением, также они призваны сократить время разработки. Существует множество примеров таких паттернов.

Для разработки UI необходимо понимать приоритетность разных компонентов. Участвует ли объект в истории или занимает ли место в пространстве игры? В зависимости от этих характеристик, объект может быть диегетическим, недиегетическим, пространственным или мета-объектом. Понимая разницу между ними, разработчик может избежать ошибок в их расположении. Ограничения, связанные с пользователями мобильных приложений, включают ограниченную продолжительность концентрации внимания, так что стоит фокусировать внимание пользователя на наиболее важных аспектах приложения. Поэтому разработчик UI должен понимать правильное размещение объектов, задавать им необходимый размер и цвет.

Тестирование интерфейса может вызывать некоторые проблемы. Самым простым методом тестирования UX является сбор отзывов пользователей, однако трудно узнать мнение пользователей об интерфейсе если он еще не до конца разработан. В результате рекомендации по его улучшению попадают на финальный этап разработки, когда вносить существенные изменения уже поздно.

Существуют другие методы тестирования интерфейса: эвристическая оценка, перспективная проверка вмешательства пользователя, когнитивное прохождение, плюралистическое прохождение и формальные проверки использования. Каждый из этих методов имеет разную эффективность, затрачивает разное количество времени и ресурсов и требует разных навыков тестирования.

Что касается дизайна внутриигровых элементов, дизайнер, или, во многих случаях, разработчик, берущий на себя роль дизайнера, должен понимать ключевые элементы, на которых он должен сосредоточиться. Постоянство дизайна объектов, персонажей и локаций является одной из важнейших составляющих дизайна игры. Все спрайты должны быть выполнены в одном стиле и сочетаться по цвету, дизайнер все время должен придерживаться общего дизайна игры.

За последние десять лет большинство компаний перешло на простые и минималистичные дизайны, упростило логотипы и элементы интерфейса. Это связано с тем, что пользователь предпочитает такие дизайны более сложным и содержащим большое количество деталей. Если говорить конкретно о графике в мобильных играх, то, несмотря на то что некоторые из них пытаются придерживаться реализма, наиболее популярной остается казуальная графика, также содержащая небольшое количество деталей и цветов, упрощающая формы.

ЛИТЕРАТУРА

1. Mobile UI design patterns. [электронный ресурс] // статья. URL: https://www.academia.edu/35546063/Mobile_UI_Design_Patterns
2. Mobile content UI [электронный ресурс] // статья. URL: https://www.academia.edu/4717201/Mobile_Content_UI
3. User Interface Design & Evaluation of Mobile Applications. [Электронный ресурс] // статья. URL: https://www.researchgate.net/publication/349087972_User_Interface_Design_Evaluation_of_Mobile_Applications
4. Mobal UX design. [Электронный ресурс] // статья. URL: https://www.academia.edu/41126886/Mobile_UX_design
5. Level Up: A Guide to Game UI. [Электронный ресурс] // статья. URL: https://www.toptal.com/designers/gui/game-ui

УДК 621.319

Забабурина Е.А. (2 курс магистратуры),
Малеев О.Г., к.т.н., доцент

РАЗРАБОТКА СЕРВИСА УПРАВЛЕНИЯ МЕТАДАННЫМИ В ГЕТЕРОГЕННОМ ХРАНИЛИЩЕ

Целью проекта является разработка веб-сервиса для управления хранилищем, содержащим технические сведения и бизнес-описания различных источников данных. Созданный сервис позволяет получать конечные данные из нужного источника, не имея прямого доступа к базе данных (БД) или серверу, на котором он фактически находится.

Хранилище метаданными представляет собой СУБД PostgreSQL, которая предусматривает следующие возможности для пользователя-сервиса:

– получение общих сведений об источниках, к которым имеется доступ;

– получение сведений о полях источника;

– получение данных из указанного источника.

Взаимодействие веб-сервиса с хранилищем осуществляется посредством вызовов определенных функций на языке запросов SQL. Доступ к данным удаленных источников предоставляется компонентом dblink. Каждый источник содержит атрибут is_dblink, указывающий возможность получить данные через функцию хранилища. При отстутствии

такого атрибута, веб-сервису необходимо подключиться к источнику напрямую с сохранением всех возможностей SQL-запроса (фильтрация, группировка, постраничная выборка и т.д.).

Сервис для управления метаданными является API-сервисом и реализован с использованием веб-фреймворка Flask на языке программирования Python. Данный фреймворк обладает возможностью гибкой настройки и реализован так же на Python, что предоставляет обширные возможности работы с несколькими БД одновременно и json-данными.

SQLAlchemy является инструментом для работы с реляционными СУБД с применением технологии ORM. С помощью специализированной конфигурации (SQLAlchemy Binds) реализована привязка к нескольким БД одновременно, что позволяет осуществлять прямое подключение к источнику данных.

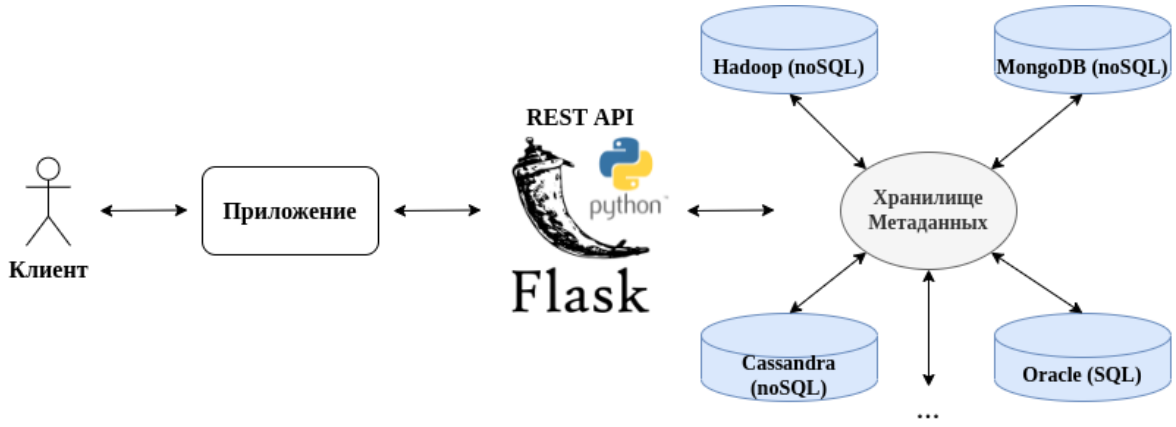


Рисунок 1 – Взаимодействие с сервисом управления метаданными

На Рисунке 1 схематично представлен вариант использования сервиса. Во взаимодействии участвуют следующие акторы:

– пользователь, взаимодействующих с приложением посредством GUI;
– приложение, которое использует сервис управления метаданными;
– реализованный API-сервис для взаимодействия с хранилищем;
– гетерогенное хранилище.

Преимуществами гетерогенных хранилищ являются высокая скорость доступа к данным и их обработки, гибкие средства масштабирования, полное сокрытие реализации и возможность получения данных из любой точки системы, имея доступ только к API-сервису.

Таким образом, предоставляется универсальный инструмент, позволяющий в одном месте аккумулировать сведения о различных источниках данных.

ЛИТЕРАТУРА

1. Cattell R. Scalable SQL and NoSQL data stores. ACM SIGMOD Rec. 2011;39:12. [Электронный ресурс] Режим доступа: https://doi.org/10.1145/1978915.1978919
2. Flask-RESTful. Библиотека для построения API. [Электронный ресурс] Режим доступа: https://flaskrestful.readthedocs.io/
3. Flask-SQLAlchemy [Электронный ресурс] Режим доступа: https://flask-sqlalchemy.palletsprojects.com/en/2.x/
4. Gibson G.A. Network attached storage architecture. Communications of the ACM, 43(11):37–45, 2000.
5. PostgreSQL [Электронный ресурс] Режим доступа: https://www.postgresql.org/
6. Sveen, A.F. Efficient storage of heterogeneous geospatial data in spatial databases. J Big Data 6, 102 (2019). [Электронный ресурс] Режим доступа: https://doi.org/10.1186/s40537-019-0262-8

7. Архипенков, С. Я. Хранилища данных: от концепции до внедрения : практическое пособие : [16+] / С. Я. Архипенков, Д. Голубев, О. Максименко ; ред. С. Я. Архипенков. – Москва : Диалог-МИФИ, 2002. – 528 с.
8. Гринберг, М. Разработка веб-приложений с использованием Flask на языке Python / М. Гринберг. - М.: ДМК, 2014. - 272 с.
9. Коннолли, Т. Базы данных. Проектирование, реализация и сопровождение. Теория и практика / Т. Коннолли. - М.: Вильямс И.Д., 2017. - 1440 с.

УДК 004.514

Дац П., (4 курс бакалавриата),
Шошмина И. В., к. т. н., доцент

РАЗРАБОТКА ГРАФИЧЕСКОГО ИНТЕРФЕЙСА ДЛЯ ГЕНЕРАТОРА ТЕСТОВ

Целью работы является разработка графического интерфейса для генератора тестов. Интерфейс должен быть написан на языке программирования Python.

Ранее уже был создан графический интерфейс:

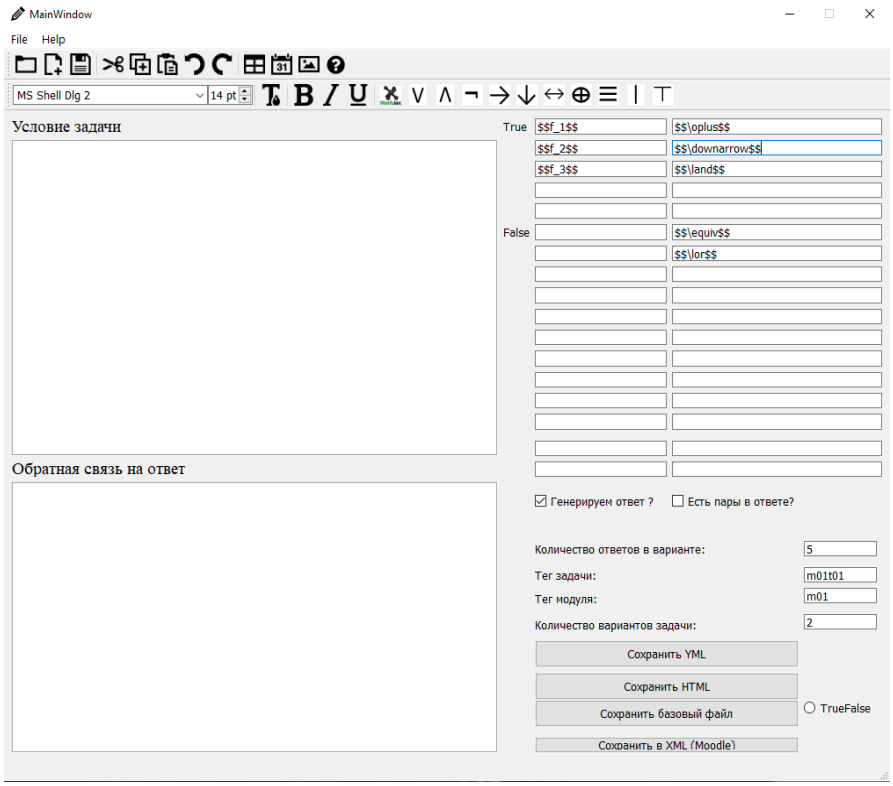


Рисунок 1 – Старый графический интерфейс

Данный проект был написан на Python с использованием Qt Designer. Qt Designer прост в использовании, но имеет маленький набор возможностей. Потребность в новом графическом интерфейсе возникла в связи с расширением функций и настроек приложения и добавления сохранений в новых форматах.

В рамках данной работы были проанализированы различные графические библиотеки, поддерживаемые на Python, например, Qt Designer, Kivy, Tkinter. Самой подходящей оказалась Tkinter, на которой и был написан новый графический интерфейс.

Одним из основных требований к моему интерфейсу было то, чтобы создать динамическое окно программы. В новом интерфейсе должна быть возможность раздвигать окно приложения, открывая дополнительные секции (секцию ответов или секцию готового результата) или сворачивать их, получая окно, содержащее только поле для ввода условия

задачи, добавлять любые изображения, необходимые для задачи, без потери качества, создавать динамическую таблицу, которую в любой момент можно редактировать, добавлять неограниченное количество ответов и добавлять математические формулы в формате Latex.

Для реализации динамических окон в приложении были созданы классы, каждый из которых создает окно и эти окна взаимодействуют друг с другом. Далее были добавлены кнопки [1], чтобы открыть соответствующее окно или скрыть его и были придуманы функции открытия и сворачивания окон.

Были созданы новые кнопки для сохранения ответов, а именно добавлена кнопка для сохранения в QTI формате и кнопка для сохранения в XML формате.

Для динамических ответов была создана кнопка для добавления нового ответа, разработана функция, которая выдает каждому новому полю уникальное имя и записывает его в словарь, после чего было добавлено поле для общего комментария, было создано поле комментария для каждого ответа, был разработан класс, который создает уникальный объект, который добавляет подсказку в поле.

Для качественного добавления изображений была внедрена новая иконка в панель инструментов [1], которая открывает проводник для выбора изображения [2] и написана функция, которая обрабатывает изображение в нужный для вывода формат.

В программе можно создавать и таблицу [3]. Для нее было создано новое окно, созданы кнопки для добавления и удаления строк и столбцов, были разработаны функции для добавления и удаления строк и столбцов и была разработана кнопка сохранения таблицы, которая скрывает окно создания и добавляет таблицу в поле условия задачи. Повторное нажатие на кнопку создания таблицы дает возможность редактировать таблицу, а при сохранении старая таблица меняется на новую.

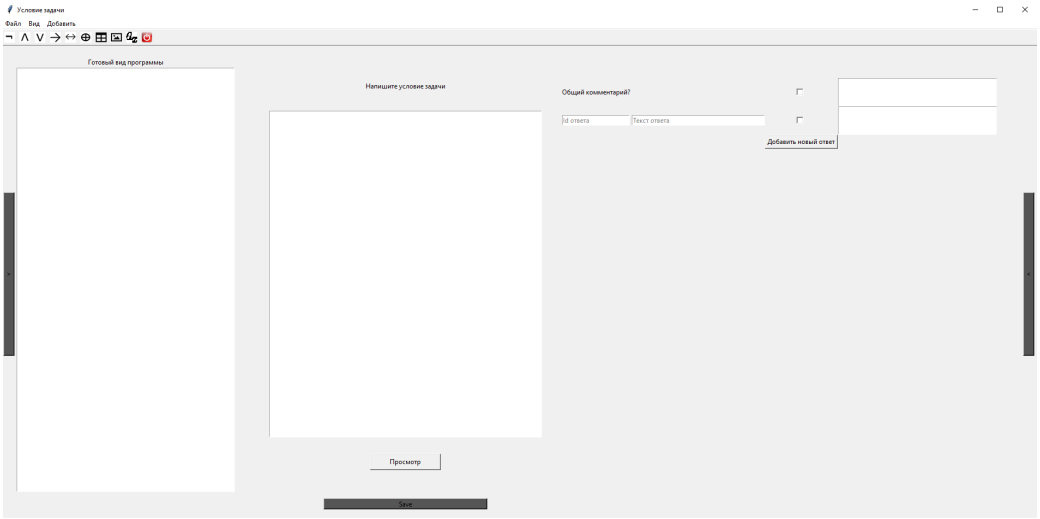


Рисунок 2 – Динамический графический интерфейс

Теперь интерфейс генератора тестов имеет большой набор возможностей: может открывать или сворачивать секции ответов и готового результата, появилось окно готового результата задачи, имеет динамическое количество ответов и возможность написания единого комментария, умеет работать с изображениями и может создавать динамические таблицы. Вдобавок, получилось сделать user-friendly интерфейс для программы благодаря тому, что все функции вынесены на панель инструментов, каждая из которых имеет свою иконку, интуитивно понятную для любого пользователя

ЛИТЕРАТУРА

1. Изучаем Python 3 на примерах. [Электронный ресурс]. URL: https://python-scripts.com/tkinter
2. John E. Grayson "Python and Tkinter Programming", Manning Publications Co., 2000, ISBN 1-884777-81-3
3. The Python Package Index. [Электронный ресурс]. URL: https://pypi.org/

УДК 004.9

Ивлев В. А. (1 курс аспирантуры),
Никифоров И. В., к.т.н., доцент

АКТУАЛЬНОСТЬ АВТОМАТИЗИРОВАННОЙ НАСТРОЙКИ ИНФРАСТРУКТУРЫ ИТ-ПРОЕКТА

Информационные технологии (ИТ) стремительно развиваются и с каждым днем охватывают новые сферы деятельности человека, решая возложенные на них задачи. Именно поэтому ИТ-структуры выстраивают свои процессы таким образом, чтобы быстро и качественно выпускать требуемые рынку программные продукты [1].

Актуальность темы исследования охватывает одну из основных концепций разработки программного обеспечения сегодняшнего времени – разработку с использованием полноценной ИТ-инфраструктуры [2]. Решение задачи автоматизации развертывания ИТ-инфраструктуры нового проекта повышает производительность разработки и снижает временные трудозатраты на создание программного продукта, что в свою очередь актуально как для малых, так и для огромных ИТ-компаний [3]. Ведь именно им важно исключить ручную настройку локального окружения, т.к. это трудоемкий и постоянно повторяющийся процесс, отнимающий у разработчиков время на создание нового продукта.

Результатом работы предполагается появление следующих пунктов, выражающих научную новизну:

1. Метод интеграции системных сред разработки в едином ИТ-окружении, который позволяет снизить трудозатраты на всех этапах создания программного обеспечения;
2. Автоматизация настройки ИТ-окружения, позволяющая повысить пропускную способность команды разработки за счет исключения из процесса ручной настройки локального окружения;
3. Разработка и внедрение системы, за счет которой повышается скорость введения нового сотрудника в проект и тем самым увеличивается скорость разработки нового функционала программного продукта;
4. Внедрение алгоритма подготовки и настройки окружения для начала разработки нового проекта, который повышает эффективность команды при разработке программного обеспечения;
5. Представление результатов разработанного подхода и его сравнение с существующими подходами автоматизации развертывания ИТ-инфраструктуры.

Исходя из этого целью исследования является снижение трудоемкости настройки инфраструктуры ИТ-проекта за счет новых методов автоматизированной настройки, реализованных в программной системе. Таким образом вытекают задачи, необходимые для достижения цели:

1. Проведение исследования в рамках выбранной области методов автоматизированной настройки ИТ-инфраструктуры;
2. Представление сравнительного анализа существующих подходов по автоматизации настройки ИТ-инфраструктуры;
3. Предложение метода интеграции системных сред разработки [4] в едином ИТ окружении;
4. Разработка программной системы, реализующих предложенные методы автоматизация настройки ИТ-окружения, которые позволят повысить пропускную способность команды разработки;
5. Демонстрация результатов разработки, показывающих снижение трудоемкости работы с проектом в ИТ-инфраструктуре.

Опираясь на вышеизложенные пункты актуальности, научной новизны и поставленных целей можно сделать вывод о необходимости разработки прототипа системы для автоматизации настройки инфраструктуры ИТ-проекта. Сравнительный анализ

существующих систем, архитектурную проектировку, процесс разработки прототипа и конечные результаты работы будут представлены в дальнейших исследованиях.

ЛИТЕРАТУРА

1. Дробинцев, П. Д. Интегрированная технология обеспечения качества программных продуктов с помощью верификации и тестирования: специальность 05.13.11 "Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей": диссертация на соискание ученой степени кандидата технических наук / Дробинцев Павел Дмитриевич. – Санкт-Петербург, 2006. – 238 с.
2. С. Унагаев, Организация ИТ-инфраструктуры компании // Системный администратор. – 2013. – № 4(125). – С. 40-41.
3. Т. Г. Буриченко, Анализ модели оценки эффективности использования облачных технологий // Молодой ученый: вызовы и перспективы: Сборник статей по материалам III международной научно-практической конференции, Москва, 25 января – 04 2016 года / Ответственный редактор: Бутакова Е.Ю.. – Москва: Общество с ограниченной ответственностью "Интернаука", 2016. – С. 252-260.
4. N. Voinov, K. Rodriguez Garzon, I. Nikiforov, P. Drobintsev, 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.

УДК 004.42

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

РАЗРАБОТКА МОБИЛЬНЫХ ПРИЛОЖЕНИЙ ДЛЯ УДАЛЕННОЙ РАБОТЫ СО СРЕДАМИ РАЗРАБОТКИ НА ПЛАТФОРМЕ INTELLIJ

При создании программного обеспечения используются интегрированные среды разработки (Integrated development environment – IDE). Такие среды, помимо непосредственно кодирования, также предоставляют возможности сборки, отладки и запуска программ. Более того, наиболее прогрессивные из них в дополнение ко всему интегрируют в себе множество различных компонентов, таких как набор средств для разработки под определенную платформу (Software development kit – SDK), библиотеки и фреймворки.

IDE предъявляют высокие требования к системному программному и аппаратному обеспечениям, что может ограничивать возможность их использования.

Помимо производительности, также существует проблема, когда разработчик физически не может получить доступ к рабочему месту. В таком случае ему необходимо решение, предоставляющее удаленное подключение и работу со средствами разработки.

На данный момент существует ряд систем, предоставляющих возможность удаленной работы с IDE. Среди них можно выделить как отдельные приложения, так и встроенные непосредственно в среду разработки. Однако такие решения либо являются исключительно десктопными приложениями, либо используют веб-браузер для отображения интерфейса. В первом случае возможность подключения с мобильных устройств полностью отсутствует, в то время как во втором случае такая работа возможна с некоторыми ограничениями. Главными из них являются низкая производительность и невозможность использования стандартных для IDE сочетаний клавиш.

Целью работы является разработка нативных мобильных приложений для предоставления возможности удаленной разработки в средах, построенных на платформе IntelliJ [1], с устройств, работающих на операционных системах iOS и Android.

Такое решение, помимо устранения зависимости от браузеров, также увеличивает производительность отрисовки, уменьшает нагрузку на устройство, и позволяет задействовать специфические возможности, такие как использование жестов для определенных сценариев использования.

Приложения реализуются на базе существующей системы Projector [2]. Общая архитектура Projector'а - клиент-серверная (Рисунок 1).

Сервер (находится в свободном доступе на платформе Github) [3] отвечает за прием поступающих к нему сигналов отрисовки, агрегируя их в параллельную связанную очередь, а затем отправляя сформированный буфер клиентам с частотой около 100 раз в секунду. Сервер реализован на языке Kotlin.

Для подключения к серверу необходим клиент. Главная задача клиента – принимать события от пользователя и отправлять их на сервер, а также «слушать» сервер, принимать и отрисовывать посылаемые им события. В данной работе разрабатываются мобильные клиенты с использованием мультиплатформенной технологии Kotlin Multiplatform Mobile [4], которая позволяет объединить большую часть разработки.

Для соединения клиента с сервером существует специальный протокол. Для передачи событий устанавливается и используется WebSocket-соединение, что делает протокол событийно-управляемым.

Для кодирования событий используется JSON или Protobuf формат. Для уменьшения времени передачи событий поддержан формат сжатия данных Gzip.

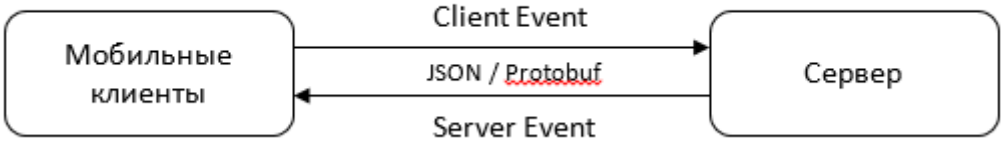


Рисунок 1 – Архитектура системы Projector

Архитектура мобильных приложений состоит из модулей нативного и общего кода (Рисунок 2).

Модуль нативного кода включает в себя реализацию представлений (View) приложения, которые являются уникальными для каждой платформы.

Модуль общего кода состоит из компонент, составляющих бизнес-логику приложений. Для работы с платформозависимыми API, соответствующая компонента подразделяется на две части (actual и expect), тем самым позволяя работать с интерфейсом каждой из платформ.

Модуль общего кода включает в себя реализацию протокола, работу с сетью и логику отрисовки примитивов. Для работы с сетью и отрисовки задействуются платформенные API, поэтому компоненты Сеть и Canvas помимо общей части имеют отдельную реализацию для каждой платформы.

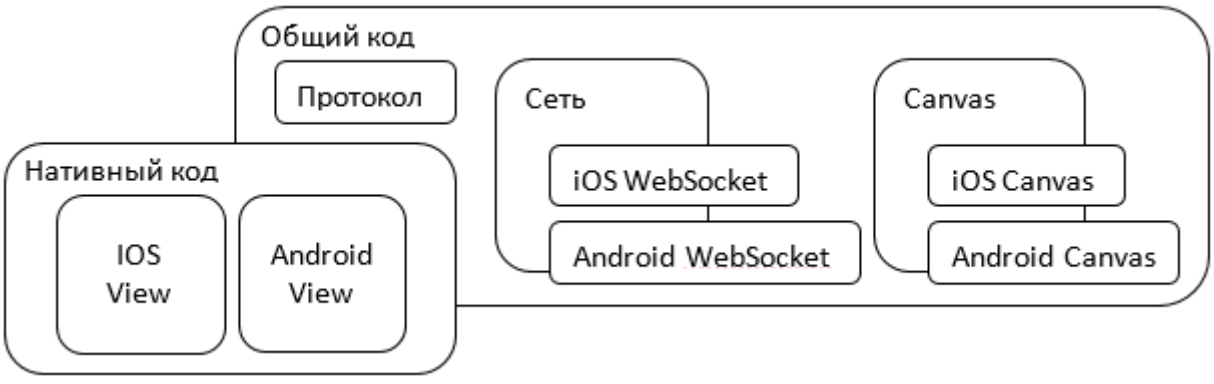


Рисунок 2 – Архитектура мобильных приложений

В перспективе такая система позволит не только избежать проблем, связанных с ресурсоемкостью IDE, но также предоставит возможность разработки ПО с мобильных устройств.

ЛИТЕРАТУРА

1. JetBrains. IntelliJ Platform: Open-Source Platform for Building Developer Tools [Электронный ресурс]. 2022. URL: https://www.jetbrains.com/opensource/idea/ (дата обращения: 05.03.2022).
2. JetBrains. Projector [Электронный ресурс]. 2022. URL: https://lp.jetbrains.com/projector/ (дата обращения: 05.03.2022).
3. projector-server [Электронный ресурс]. 2022. URL: https://github.com/JetBrains/projector-server (дата обращения: 05.03.2022).
4. Kotlin. Write the business logic for your iOS and Android apps just once, in pure Kotlin [Электронный ресурс]. 2022. URL: https://kotlinlang.org/lp/mobile/ (дата обращения: 05.03.2022).

УДК 004.415.2

Квашнин А. А. (4 курс бакалавриата),
Петров А. В., старший преподаватель,
Леонтьева Т. В., к. т. н., доцент

РАЗРАБОТКА АРХИТЕКТУРЫ ДЛЯ ВЫДЕЛЕННОЙ МОДЕЛИ АРЕНДЫ SAAS ПРИЛОЖЕНИЙ В KUBERNETES

Программное обеспечение как услуга (далее SaaS), представляет собой модель распространения программного обеспечения (далее ПО), согласно которой, вместо загрузки ПО для локального запуска на вашем ПК, программа размещается у стороннего поставщика, а затем доступна пользователям через Интернет [2]. Чаще всего архитектура приложений имеет мультитенантную архитектуру, что позволяет сократить расходы на обслуживание, но ухудшает отказоустойчивость системы в целом. Переход на выделенную (однотенантную) архитектуру повысит надёжность системы, а также позволит рассчитывать стоимость аренды для каждого подписчика индивидуально, размещать сервисы в региональных центрах рядом с каждым клиентом, обновлять сервисы индивидуально и многое другое [7]. Выделенная архитектура подразумевает использование отдельной инфраструктуры для размещения ПО для каждого арендатора [1]. Основная проблема выделенной архитектуры заключается в сложной организации модулей непрерывной интеграции (далее CI) и доставки (далее CD), а также хранении секретов сервисов [7].

Цель работы заключается в проектировании CI/CD процессов, а также организации хранения секретов приложений в выделенной архитектуре на базе оркестратора Kubernetes.

Реализовать CD на различные окружения поможет инструмент Argo CD. Основным функционалом которого является отслеживание git или helm репозитория, содержащего манифесты сервиса, при изменении которых автоматически происходит обновление подов в кластере Kubernetes [4]. Все манифесты сервисов располагаются в отдельном git репозитории, сопровождением которого занимаются DevOps специалисты. Другой немаловажной функцией Argo CD является обновление контейнера приложения в Kubernetes на данный момент отслеживание изменений из реестра контейнеров возможно через плагин Argo CD image updater.

Со стороны разработчика после нового коммита исходного кода микросервиса в репозиторий запускается процесс CI при помощи одного из инструментов GitLab CI, Jenkins, TeamCity и т. д. По завершению сборки на выходе получается артефакт, который загружается в реестр контейнеров. Для окружений stage и production настроено автоматическое слежение за соответствующим ветками git репозитория, а для development окружения процесс CD выполняется вручную, запуском конвейера сервиса CD, например, Azure DevOps.

Администратор сервиса может подключать и отключать модули сервиса через UI интерфейс. Модули включают в себя некоторое количество микросервисов, поэтому должна существовать база данных, содержащая всю надлежащую информацию по каждому сервису (git url, ключи доступа, аннотации и т. д.). Сервис по управлению инфраструктурой должен автоматически добавлять или удалять приложения из Argo CD через REST API, этим требованиям отвечает CMDB, которая позволит быстро создать модуль управления архитектурой без затрат на разработку.

Для каждого сервиса необходимо подключить свой набор секретов (например, строк подключения к базе данных). Kubernetes не предоставляет такой возможности [3], поэтому архитектура включает в себя плагин External Secrets Operator, который отслеживает подключенный Vault и монтирует секреты в нужный под. Таким образом, каждый экземпляр сервиса может получить свой набор секретов и использовать его. Такой подход позволяет реализовать автоматическое обновление секретов внутри работающего сервиса, для этого достаточно монтировать секреты как файл (например, JSON), а не как переменные окружения.

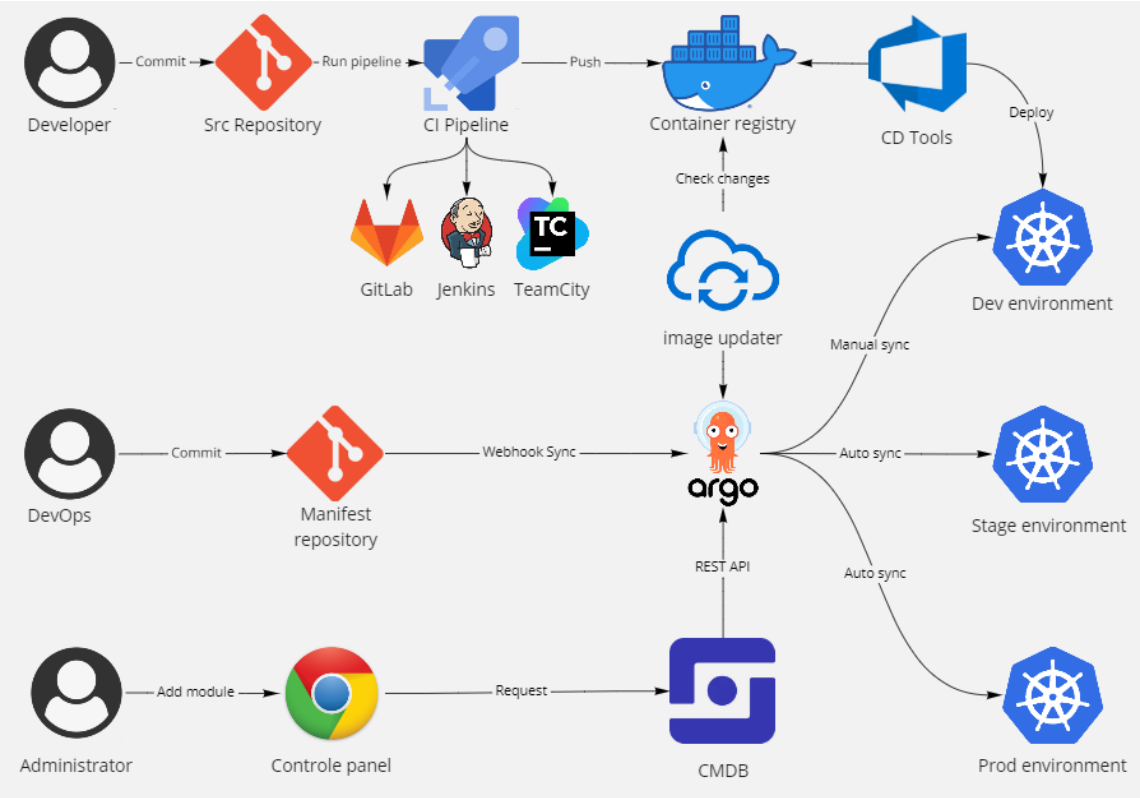


Рисунок 1 – Архитектура CI/CD процессов.

Разработанная архитектура позволяет повысить отказоустойчивость системы, быстро и интуитивно настроить систему, автоматизировать процесс тестирования и релиза ПО, выставлять счёт аренды для каждого клиента индивидуально, размещать систему территориально близко к клиентам, сократить расходы на поддержание инфраструктуры, стандартизировать и гибко управлять жизненным циклом ПО, обеспечить горячую замену секретов.

ЛИТЕРАТУРА

1. Мультитенантные шаблоны клиента в базе данных SaaS. [Электронный ресурс]. Режим доступа: https://docs.microsoft.com/ru-ru/azure/azure-sql/database/saas-tenancy-app-design-patterns.
2. Создание предложения SaaS. [Электронный ресурс]. Режим доступа: https://docs.microsoft.com/ru-ru/azure/marketplace/create-new-saas-offer.
3. Документация по Kubernetes. [Электронный ресурс]. Режим доступа: https://kubernetes.io/ru/docs/home.
4. Документация Argo CD. [Электронный ресурс]. Режим доступа: https://argo-cd.readthedocs.io.

5. Ричардсон Крис. Микросервисы. Паттерны разработки и рефакторинга – Питер, 2019, 544 с.
6. Дэниелс Кэтрин, Дэвис Дженнифер. Философия DevOps. Искусство управления IT – Питер, 2017, 416 с.
7. Мультитенантная архитектура для SaaS приложений. [Электронный ресурс]. Режим доступа: https://habr.com/ru/company/microsoft/blog/145027/.

УДК 32.973

Кириллов Н.И. (4 курс бакалавриата),
Муравьев Е.А., к.т.н., доцент

ДОМЕННАЯ СЕМАНТИКА В ЗАДАЧАХ ТЕХНОЛОГИИ ПРОГРАММИРОВАНИЯ

В докладе обсуждается техника формализации доменной семантики (семантики предметной области) в задачах технологии разработки программного обеспечения. Развивается метод извлечения семантики из документации библиотек прикладных программ, пространств имен, рассматриваемых как источник эмпирических данных о доменной семантике. Семантика формализована на основе аппарата онтологического моделирования – RDF[1].

Применение доменной семантики в форме онтологий состоит в следующем. Код на языке программирования (ЯП), содержащий вызовы членов прикладного пространства имен, компилируется в онтологию, RDF-граф формата N3. Онтология, результат компиляции, представляет семантику кода. Семантический анализ кода выполняется с применением инструментов стека семантических программ, основным из которых является язык запросов SPARQL[2].

Компилятор ЯП → RDF реализован с применением утилиты Yacc[3]. RDF темплеты пространств имен .NET, примененные в компиляторе, расширяют аппарат работы с семантикой, реализованный в Yacc. В качестве тестовых пространств имен для извлечения семантики были взяты: в части общей семантики ЯП – пространство имен System.CodeDom; в части примера доменной семантики – пространство имен System.Drawing, оба члены .NET. Онтологическое моделирование выполнено на базе программной реализации стека семантических программ STARDOG[4] и его API.

Рассмотрим пример из спецификации класса CodeMethodInvokeExpression из пространства имен System.CodeDom, приведенный на сайте Microsoft:

C# Копировать

```
CodeMethodInvokeExpression methodInvoke = new CodeMethodInvokeExpression(
    // targetObject that contains the method to invoke.
    new CodeThisReferenceExpression(),
    // methodName indicates the method to invoke.
    "Dispose",
    // parameters array contains the parameters for the method.
    new CodeExpression[] { new CodePrimitiveExpression(true) } );

// A C# code generator produces the following source code for the preceeding
example code:

// this.Dispose(true);
```

Рисунок 1 – пример из спецификации CodeMethodInvokeExpression.

Определения данной конструкции на естественном языке:
CodeMethodInvokeExpression используется для ***определения выражения*** вызова ***метода***.
CodeMethodInvokeExpression имеет параметр ***CodeThisReferenceExpression***.

Запишем приведенные определения в виде RDF триплетов:

```
:CodeExpression rdf:type rdfs:Class.
:InvokeExpression rdfs:subClassOf :CodeExpression.
:CodeMethodInvokeExpression rdf:type rdfs:Class.
:CodeMethodInvokeExpression :CreatesExpression :InvokeExpression.
:CreatesExpression rdf:type rdf:Property.
:CreatesExpression rdf:domain :CodeMethodInvokeExpression.
:CreatesExpression rdf:range :CodeExpression.
:CodeMethodInvokeExpression :hasParameter :CodeThisReferenceExpression.
:hasParameter rdf:type rdf:Property.
:hasParameter rdf:domain rdf:CodeMethodInvokeExpression.
:hasParameter rdf:range rdfs:Class.
```

Некоторые термины для данного примера были доопределены (путем обобщения). Если разделить полученные термины на группы, то имеем:

1. Термины пространства имен CodeDom, например: CodeMethodInvokeExpression
2. Члены RDF/RDFS: rdf:type, rdf:Property, rdf:domain, rdf:range, rdfs:subClassOf.
3. Доопределенные сущности: CodeExpression, InvokeExpression, CreatesExpression.

Преобразование естественно языковых определений в RDF-триплеты выполнено пока вручную. Ведется работа по формализации и автоматизации этого процесса.

Онтология загружается в STARDOG Server и представляет пример фрагмента доменной семантики. Исходная программа в процессе компиляции интегрируется с онтологиями, извлеченными из других доменных пространств имен, а также других членов CodeDom. Онтология - результат представляет полную формальную семантику программы, включая доменную.

Ниже приведен пример семантического анализа, приведенного на Рисунке1 кода, а именно, запросы к онтологии-семантике программы.

Запрос: основная функция программы?
Ответ: CodeMethodInvokeExpression.

Запрос: содержит ли вызов метода фактические параметры?
Ответ: содержит.

Запрос: тип фактического параметра? Приведем формулировку запроса на языке SPARQL:

```
SELECT ?param WHERE{ ?constructor :hasParameter ?param}
```

Ответ:

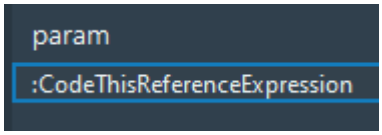


Рисунок 2 – результат SPARQL запроса в STARDOG.

Развиваемая техника может применяться при решении семантических задач анализа кода, таких, как: понимание программы, обучение на основе программы как примера, семантический поиск реализации по абстрактной спецификации кода, верификация доменных

требований и др. Ведется работа по формализации пространства имен домена геометрического моделирования System.Drawing.

ЛИТЕРАТУРА

1. Концепции RDF 1.1 и абстрактный синтаксис http://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/
2. Язык запросов SPARQL11 https://www.w3.org/TR/2013/REC-sparql11-query-20130321/
3. Ахо Альфред В. Компиляторы: принципы, технологии и инструментарий, 2-е изд.: Пер. с англ. – М.: ООО “И.Д. Вильямс”, 2017. – 1184 с. : ил. – Парал. тит. англ.
4. STARDOG Server - https://www.stardog.com/

УДК 004.42

Ковальчук Е.В. (2 курс магистратуры),
Самочадин А.В., к.т.н., доцент

РАЗРАБОТКА ИНСТРУМЕНТАЛЬНЫХ СРЕДСТВ ДЛЯ ИЗВЛЕЧЕНИЯ МОДЕЛЕЙ СЦЕНАРИЕВ ИЗ СЛАБОСТРУКТУРИРОВАННЫХ ТЕКСТОВ

Целью работы является разработка инструментальных средств для извлечения моделей сценариев из слабоструктурированных текстов на естественном языке. Построенная модель сценария должна использоваться в системе ИМ.

Рассмотрен такой термин как моделирование процессов и выделены способы извлечения моделей процессов. Проведено исследование существующих методов и подходов извлечения моделей процессов, выделены их недостатки.

Были проанализированы сценарии в виде слабоструктурированного текста. На основе результатов анализа были выделены рекомендации и правила написания сценариев на естественном языке.

Разработан собственный метод извлечения моделей сценариев из слабоструктурированных текстов на естественном языке. Для упрощения создания сценариев разработан шаблон, который является связующим звеном межу сценарием на неформальном языке и сценарием в виде JSON-файла, пригодном для исполнения модели. Таким образом, особенностью метода является шаблон, с помощью которого создаются сценарии.

Разработанный метод состоит из двух шагов (Рисунок 1):

- Перевод сценария на естественном языке в некоторый шаблон;
- Создание модели сценария на основе шаблона.

Рисунок 1 – Схема метода.

Шаблон был реализован на языке программирования python. Для создания сценариев разработано приложение с интерфейсом, упрощающее создание новых сценариев и редактирование уже имеющихся. Так как при создании сценария очень легко допустить ошибки, были реализованы алгоритмы, проверяющие корректность заполненных полей шаблона.

ЛИТЕРАТУРА

1. Процессная аналитика [Электронный ресурс]. Режим доступа: https://ru.wikipedia.org/wiki/Процессная_аналитика.
2. Глоссарий. Регламент процесса [Электронный ресурс]. Режим доступа: https://www.businessstudio.ru/articles/article/glossariy_reglament_protsessa/.

3. Ланцев Е.А., Доррер М.Г. Создание агентных имитационных моделей с применением технологии интеллектуального анализа процессов.
4. Ланцев Е.А., Доррер М.Г. Агентное имитационное моделирование бизнес-процессов в нотации eEPC // Научно-технический вестник информационных технологий, механики и оптики. – 2013. – No 3. – С. 86–92.
5. W.M.P. van der Aalst. Process Mining: Discovery, Conformance and Enhancement of Business Processes. Springer-Verlag, Berlin, 2011.
6. Д.Ю. Чалый, Р.Р. Яиков. Извлечение моделей процессов из неаннотированных логов событий с использованием методов кластеризации.
7. К.В. Соколов, Д.А. Тимофеев, А.В. Самочадин. Извлечение описаний бизнес-процессов из текстов на естественном языке.
8. Тимофеев Д.А., Самочадин А.В. Извлечение моделей бизнес-процессов из обучающих материалов // Научно-технические ведомости СПбГПУ. Информатика. Телекоммуникации. Управление. 2018. Т. 11. No 4. С. 162—170. DOI: 10.18721/JCSTCS.11412.
9. Сергея Буракова «Бизнес-процессы. Извлечение BPMN-модели из документа. Часть 1». Режим доступа: https://habr.com/ru/post/439934/.
10. Мишланов, В. А., Чуганская, А. А., Смирнов, И. В., Суворова, М. И., Курузов, И. А. Разработка методов анализа сценариев поведения (на материале инструктивных интернет-текстов).
11. И. В. Смирнов, А. И. Панов, А. А. Скрынник, Е. В. Чистова. Персональный когнитивный ассистент.

УДК 004.738.5: 004.451.55

Красиков Р.В., Ситникова К.А. (1 курс магистратуры),
Горавнева Т.С., к.т.н., доцент

РАЗРАБОТКА ПРИЛОЖЕНИЯ И СРАВНИТЕЛЬНЫЙ АНАЛИЗ ЗАРАЖЕННЫХ АУДИТОРИЙ КОРОНАВИРУСНОЙ ИНФЕКЦИЕЙ

В настоящее время остро стоит вопрос обеспечения безопасности студентов и работников университета в период пандемии коронавируса. Несмотря на принятые меры, случаи заражения все равно происходят и приводят к отправке зараженной группы на карантин. Значимость данного исследования заключается в том, что в условиях текущего комбинированного проведения занятий важно вовремя отследить полное дерево распространения инфекции, а не брать под строгое наблюдение исключительно нулевого пациента. Поэтому разработка приложения для учета распространения инфекции в аудиториях университета, анализ и выработка рекомендаций по составлению расписания являются в настоящее время важными и актуальными задачами.

Общая постановка данной сложной задачи сводится к следующим моментам:

- разработке базы данных, хранящей данные расписания университета,
- созданию алгоритма заполнения базы данных,
- разработке логического модуля, осуществляющего анализ распространения инфекции,
- проведению ряда вычислений количества зараженных аудиторий,
- разработке интерактивной карты аудиторий учебного корпуса, содержащей зараженные аудитории.

Список основных программных средств разработки включает в себя: язык программирования C#, ASP.Net MVC Framework, сервер баз данных Microsoft SQL Server, библиотеку AngleSharp, технологию ADO.Net EntityFramework, среду Microsoft Visual Studio [1]. Некоторые моменты данной разработки были изложены на конференции Всероссийского фестиваля «НАУКА 0+» в СПБГМТУ и представлены в сборнике докладов [2].

Целью дальнейшей работы является анализ распространения короновирусной инфекции в аудиториях университета по текущему расписанию занятий для различных групп и дней недели. Разработка приложения и сравнительный анализ зараженных аудиторий является актуальной на сегодняшний день темой, так как в связи с коронавирусной инфекцией вопрос оптимизации расписания как никогда актуален. Благодаря анализу, можно увидеть, для каких групп и по каким дням недели было большое количество заражений, вследствие чего можно сделать вывод об оптимальном/неоптимальном расписании.

Таким образом, конечная цель научно-практического исследования – переработка расписания занятий, чтобы уменьшить распространение короновируса в аудиториях университета. Для решения задачи также требуется исследовать, подходят ли какие-либо методы оптимизации, применить их или разработать свой алгоритм и приложение для успешного решения задачи.

В ходе работы на данный момент времени были решены следующие задачи:

1. Разработан модуль сравнительного анализа зараженных аудиторий.
2. Выполнен вывод результатов анализа в графическом виде.

Работу приложения можно разделить на 3 части:

– Ввод исходных данных (курс и период времени, в течение которого проводится анализ);

– Получение данных по зараженности:
для каждой группы выбранного курса по каждому дню недели из указанного периода подсчитывались аудитории, каждый день недели из указанного периода считался началом заражения, в заражении учитывались 2 последующих дня (всего 3 дня, включая день начала заражения);

– Вывод результатов в графическом виде:
2 графика: по группам и по дням недели, данные по максимальной и минимальной зараженности (Рисунок 1).

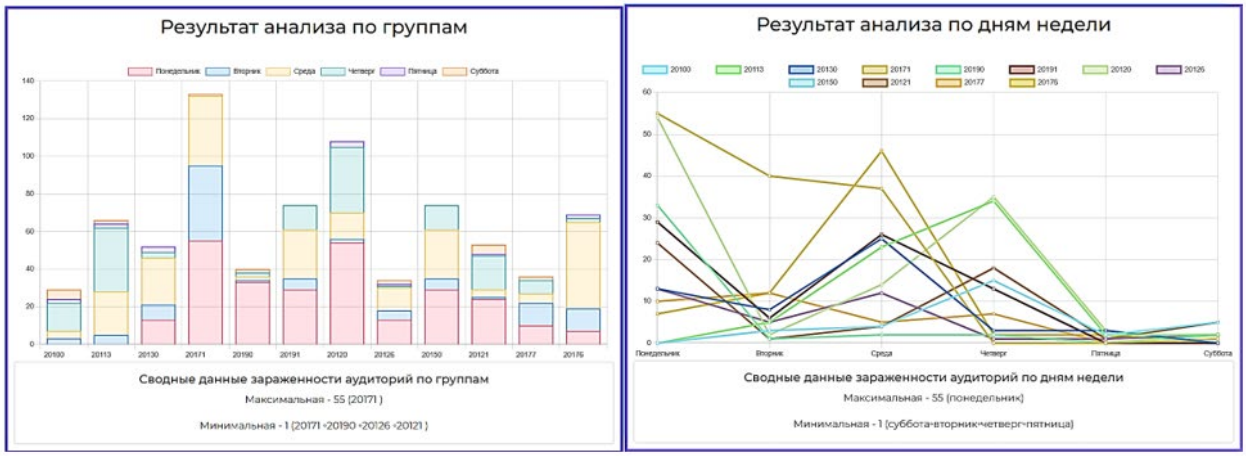


Рисунок 1 – Результат анализа для 1-го курса

На графике (Рисунок 1) видно, что на промежутке дней недели с понедельника по среду имеется большое количество заражений, а на других промежутках (вторник – четверг и др.) меньше. Это означает, что когда зараженный студент заболевает в понедельник, то это приводит к наибольшему распространению инфекции по аудиториям. На основе полученных данных можно также определить, в какие дни расписание занятий слишком загруженное и неоптимизированное, что поможет при дальнейшей его переработке.

Разработка приложения осуществлялась в среде Microsoft Visual Studio 2022 с применением ASP.NET Core MVC, Entity Framework Core.

ЛИТЕРАТУРА

1. Фримен А. ASP .NET MVC 5 с примерами на C# 5.0 для профессионалов // Вильямс: 2013. – 688 с.
2. Ситникова К.А., Горавнева Т.С. Разработка приложения моделирования распространения коронавирусной инфекции по аудиториям университета // Неделя науки СПбГМТУ-2021: сборник докладов Всероссийского фестиваля науки «Nauka 0+». – СПб.: Изд-во СПбГМТУ, 2021.

УДК 004.415.25

Ладвищенко И.В. (4 курс бакалавриата),
Самочадина Т.Н., старший преподаватель,
Самочадин А.В., к.т.н., доцент

РАЗРАБОТКА МИКРОСЕРВИСОВ ИМПОРТА И ЭКСПОРТА ДАННЫХ ДЛЯ PAYROLL-СИСТЕМЫ

Целью работы является разработка средств импорта и экспорта данных для веб-приложения, осуществляющего payroll (автоматический расчет различных выплат работникам предприятия и их учет). Приложение и разрабатываемые средства используют следующий стек технологий: Angular JS с библиотекой Kendo UI, C# .NET (вместе с Entity Framework и SQL).

Эти средства должны обеспечивать загрузку (и обработку согласно пользовательским настройкам) данных о работниках и соответствующих выплатах в веб-приложение, а также выгрузку составленных финансовых отчетов из приложения в указанные пользователем хранилища.

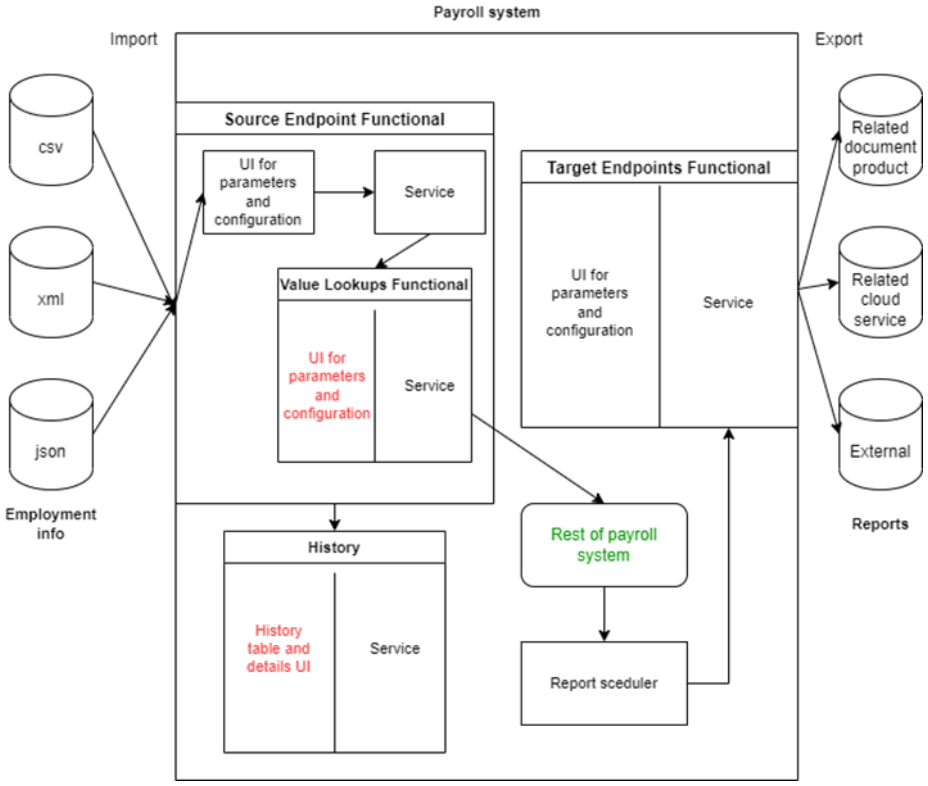


Рисунок 1 – Схема микросервисов и их компонентов, разрабатываемых в рамках проекта.

В рамках данной работы разрабатываются 2 системы – импорт и экспорт, состоящие из различных компонентов. На рисунке красным выделены компоненты, непосредственно разработанные автором работы. Зеленым выделен функционал существующего приложения.

Импорт – микросервис, который получает информацию о работниках из сторонних систем (например, workday.com) в некоем формате (например, csv, xml, json), анализирует эти данные, обрабатывает и проверяет, а затем отправляет в основное приложение для дальнейшего составления отчетов (например, для расчета заработных плат или выплат отпускных). Импорт состоит из таких компонентов, как Source Endpoint, Value Lookups и History, в которых производятся описанные выше действия в соответствии с указанными пользователем настройками и параметрами. В Value Lookups дополнительно производится

замена указанных вхождений другими, а в History для облегчения мониторинга заносится информация о полученных файлах и статусах их обработки.

Экспорт представляет собой микросервис, регулирующий посылку отчетов в указанные пользователем хранилища. Экспорт состоит из компонентов Scheduler и Target Endpoint. Приложение может формировать и отправлять отчеты как в режиме реального времени, так и в отложенном режиме. Последний реализован при помощи компонента Scheduler, который позволяет выбрать дату и время посылки отчета, а также установить ее ежедневно, ежемесячно и т.д. Компонент Target Endpoint в соответствии с указанной конфигурацией отправляет отчет в одно или несколько хранилищ. Здесь осуществлена интеграция с различными дружественными продуктами владельца приложения (продуктом для работы с документами, с облачным хранилищем), а также имеется возможность посылки во внешние хранилища.

ЛИТЕРАТУРА

1. Роботизация бухгалтерского учета и процессов / ROBIN, 2018. // [Электронный ресурс] - режим доступа: https://www.rpa-robin.ru/blog/avtomatizaciya-rabochego-mesta-buhgaltera, свободный. Дата обращения: 17.01.22
2. Katie McBeth, Ken Boyd. What is payroll? A 2021 guide to processing payroll / QuickBooks, 2020. // [Электронный ресурс] - режим доступа: https://quickbooks.intuit.com/r/payroll/what-is-payroll/, свободный. Дата обращения: 17.01.22
3. Compare All Payroll Software / Software Advice, 2022. // [Электронный ресурс] - режим доступа: https://www.softwareadvice.com/hr/payroll-software-comparison/, свободный. Дата обращения: 17.01.22
4. Daniel Epstein. 15 Best Payroll Software Systems / FinancesOnline, 2021 // [Электронный ресурс] - режим доступа: https://financesonline.com/top-15-payroll-management-software-systems/, свободный. Дата обращения: 17.01.22

УДК 004.415.2

Лебедев И.В. (2 курс магистратуры),
Медведев Б.М., к.т.н., доцент

ПРОГРАММНЫЕ СРЕДСТВА ПРОТОТИПА УСТРОЙСТВА ИНТЕРНЕТА ВЕЩЕЙ НА БАЗЕ ARDUINO

Быстрый рост интернета вещей (Internet of Things, IoT) от 26,66 миллиарда подключенных устройств IoT в 2019 году до ожидаемых 75 миллиардов к 2025 году [1] требует подготовки большого числа разработчиков программных средств, встраиваемых в устройства IoT, реализующих обработку данных в облачных сервисах и приложений пользователя для мониторинга состояния и управления в таких, например, системах как умный дом, умный город, промышленный интернет вещей, умное сельское хозяйство, электронное здравоохранение и т.д.

В процессе подготовки специалистов для проведения практических занятий необходимо иметь программируемое устройство IoT, а также предоставить доступ к платформе IoT, которая обеспечивает подключение устройств IoT, реализует функции брокера сообщений, а также дает возможность развернуть инструменты обработки передаваемых данных.

В качестве устройства IoT можно использовать смартфон, содержащий набор датчиков и средства обработки и передачи данных. Однако, применение одной из популярных систем на модуле обладает преимуществом, заключающемся в том, что знание алгоритмов, навыки программирования и разработанные в рамках курса обучения программные средства можно использовать в будущих проектах интернета вещей. В Таблице 1 приведены базовые характеристики популярных систем на модуле. В качестве прототипа устройства IoT можно рекомендовать модуль Arduino Uno, который обладает развитой периферией, пригодной для задач IoT производительностью при малых энергопотреблении и стоимости.

Таблица 1 – Системы на модуле

	Arduino Uno [2]	Intel Edison [3]	Raspeberry Pi 4 Model B [4]	Espressif ESP-32 [5]
Процессор	ATmega328, 16 МГц	Intel Atom, 500 МГц	ARM Cortex A72 1,5 ГГц	Xtensa LX6, 240 МГц
RAM	2 Кбайт	1 ГБ LPDDR3	1 ГБ LPDDR4	520 Кбайт
память	32 Кбайт	4 GB eMMC, micro SD	SD-карта	448 Кбайт
Сеть	подключаемые модули	802.11 a/b/g/n Bluetooth 4.0	802.11 b/g/n/ac Bluetooth: v5.0	802.11 b/g/n, Bluetooth: v4.2
Периферия	USB, UART, I2C, SPI, АЦП 6 каналов, 20 GPIO	USB, 2x UART, 2x I2C, SPI, I2S; 12 GPIO	2x USB 2.0, 2x USB 3.0; MIPI CSI & DSI, HDMI, Gigabit Ethernet, 40 GPIO	4x SPI, 2x I2S, 2x I2C, 3x UART АЦП 18 каналов; ЦАП 2 канала;
Энерго-потребление	Спящий режим: 12 мА; Активный: 48 мА	Спящий режим: 50 мА; Активный: 120 мА	Спящий режим: 240 мА; Активный: 700 мА	Спящий режим: 10 мА; Активный: 260 мА
Размеры	69×53 мм	35.5×25×3.3 мм	85×56×17 мм	18×25.5×3.1 мм
Стоимость	3540 рублей	9735 рублей	15990 рублей	2110 рублей

В качестве платформы интернета вещей можно использовать Microsoft Azure IoT, Amazon Web Services IoT, Google Cloud, IBM Watson, Yandex IoT Core. Применение таких платформ обеспечивает быстрый старт проекта без значительных затрат, масштабируемость от единиц до миллионов подключенных устройств, возможность использовать развитые средства обработки данных с применением искусственного интеллекта [6]. Сравнение характеристик платформ показало, что для учебных курсов и проектов IoT целесообразно использовать Yandex Cloud [7], имеющий различные программы поддержки образования, 50 сервисов для работы с инфраструктурой и сетью, включая сервис интернета вещей Yandex IoT Core, бесплатный доступ к ресурсам при разумных ограничениях (количество устройств в одном облаке до 1000, первые 100000 сообщений в месяц не тарифицируются).

Целью работы является создание ряда демонстрационных проектов и сценариев выполнения курсовой работы с использованием Arduino в рамках практических занятий по курсу «Интернет вещей».

Выполнение курсовой работы разделено на два этапа. В рамках первого этапа каждый студент использует демонстрационный проект, включая создание аккаунта в Yandex Cloud, создание устройства на базе Arduino Uno, программирование и подключение устройства к платформе Yandex IoT Core по инструкции. На втором этапе на базе демонстрационного проекта выполняется разработка программных средств по индивидуальному заданию [8].

Разработанные демонстрационные проекты содержат:

- встраиваемое программное обеспечение Arduino для обработки сигналов датчиков движения, освещения, температуры.
- средства обеспечивания энергетической эффективности работы устройства;
- систему мониторинга состояния устройства с помощью платформы Node Red;
- обработку сообщений в Yandex IoT Core, в связке со средой Node Red;
- приложение пользователя, обеспечивающее контроль состояния устройства.

Использованы следующие средства разработки: язык программирования C++, среда разработки Arduino IDE, Node Red, а также пакет библиотек для работы с платами расширения и датчиками.

Разработанные сценарии выполнения курсовой работы с использованием Arduino содержат индивидуальные траектории обучения.

ЛИТЕРАТУРА

1. Проницательная статистика «Интернета вещей»/ Safeatlast [Электронный ресурс]. – URL: https://safeatlast.co/blog/iot-statistics/#gref/ (дата обращения: 17.03.2022).
2. What is Arduino/ Arduino [Электронный ресурс]. – URL: https://www.arduino.cc/en/Guide/Introduction (дата обращения: 17.03.2022).
3. Intel Edison. Первый запуск/ Хабр [Электронный ресурс]. – URL: https://habr.com/ru/post/256089/ (дата обращения: 17.03.2022).
4. Raspberry Pi/ Wikipedia [Электронный ресурс]. – URL: https://ru.wikipedia.org/wiki/Raspberry_Pi (дата обращения: 17.03.2022).
5. Espressif Modules/ Espressif [Электронный ресурс]. – URL: https://www.espressif.com/en/products/modules (дата обращения: 17.03.2022).
6. Топ-10 инструментов IoT-разработки в 2020/ Хабр [Электронный ресурс]. – URL: https://habr.com/ru/company/ruvds/blog/525798/ (дата обращения: 17.03.2022).
7. Yandex IoT Core/ Yandex Cloud [Электронный ресурс]. – URL: https://cloud.yandex.ru/services/iot-core (дата обращения: 17.03.2022).
8. Разработка демонстрационных проектов для устройства интернета вещей на базе Arduino// Современные технологии в теории и практике программирования: сборник материалов конференции, 22 апреля 2021 г. – СПб.: ПОЛИТЕХ-ПРЕСС, 2021. – 236 с. / [Электронный ресурс]. – URL: https://icst.spbstu.ru/userfiles/files/sbornik.pdf (дата обращения: 17.03.2022).

УДК 004.415.2

Линде Д.В. (4 курс бакалавриата),
Прокофьев О. В., старший преподаватель

ИСПОЛЬЗОВАНИЕ ПРОТОКОЛА GOPROXY ДЛЯ УПРАВЛЕНИЯ ЗАВИСИМОСТЯМИ В ЯЗЫКЕ ПРОГРАММИРОВАНИЯ GO

В языке программирования Go управление зависимостями децентрализовано и осуществляется с помощью модулей, которые хранятся в открытых репозиториях с помощью различных систем контроля версий[6]. По умолчанию роль системы контроля версий выполняет git. Модуль представляет собой набор директорий, называемых пакетами, в которых хранится программный код, предлагаемый к использованию сторонним разработчиком[2]. Таким образом модули выполняют роль библиотек в языке программирования Go. Для включения модуля в проект разработчику достаточно в начале файла с исходным кодом указать имя модуля. Компилятор при сборке проекта сам загрузит модуль и добавит его в проект.

У такого подхода есть недостатки. Сервера системы контроля версий могут быть недоступны. В таком случае процесс разработки будет невозможно продолжить без доступа к программным средствам, используемым в проекте. Для предотвращения подобных ситуаций компания Google, являющаяся основным разработчиком компилятора языка программирования Go, разработала протокол GOPROXY.

GOPROXY – это протокол http сервера, который должен отвечать на заданные http запросы[4]. URL адрес сервера, реализующего данный протокол должен быть указан в переменной среды GOPROXY[4]. При использовании модулей, хранящихся в приватных репозиториях, следует указать их через переменную среды GOPRIVATE[5].

Когда система контроля версий и искомые модули доступны, компилятор находит нужные модули и устанавливает их в проект, и, если модуля нет на GOPROXY сервере, то он его там сохраняет. При отсутствии модуля в системе контроля версий или при отсутствии доступа к системе контроля версий компилятор будет искать модуль на GOPROXY сервере и установит его оттуда.

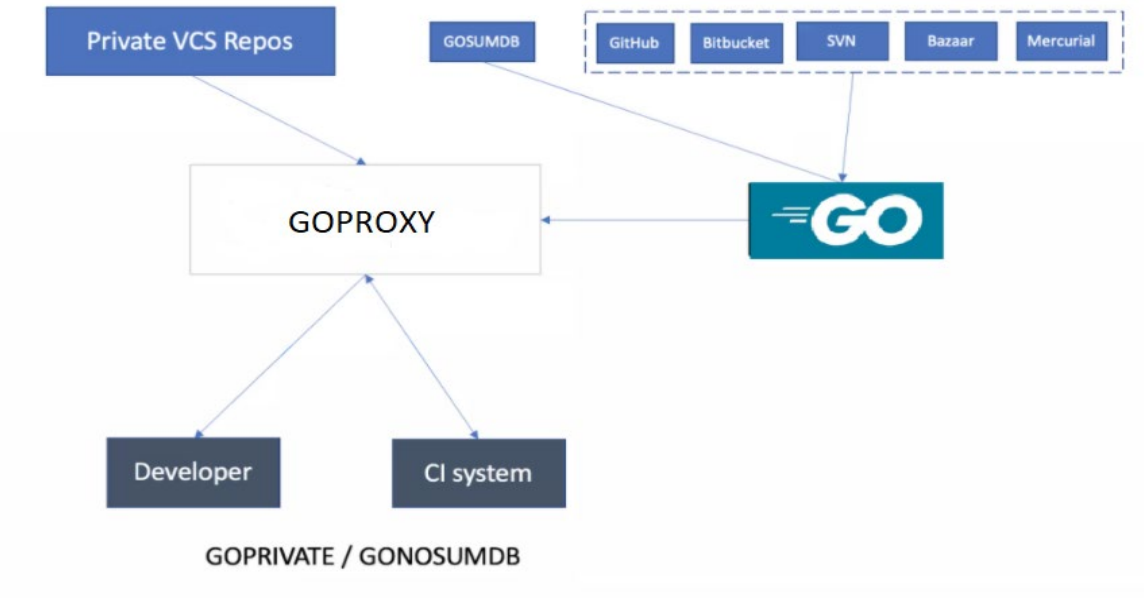


Рисунок 1 – Схема подключения GOPROXY

Актуальность данного подхода.

Большинство модулей хранятся в открытых репозиториях GitHub. GitHub – это крупнейший веб-хостинг IT-решений, принадлежащий компании Microsoft. Несмотря на то, что руководство GitHub обещало не прекращать работу сервиса на территории Российской Федерации, сервис принадлежит американской компании, которая уже приостанавливает работу территории страны[7]. По этой причине не следует исключать вероятность прекращения работы GitHub на территории Российской Федерации. В данной ситуации разработка российских решений на основе протокола GOPROXY может быть решением проблемы, так как подобные сервера, могут стать заменой GitHub, если сохранить на них заранее наиболее распространённые в разработке модули. Готовых российских решений на данный момент нет.

Самые распространённые решения GOPROXY.IO и JFROG принадлежат компаниям Google и JFROG, которые так же являются иностранными компаниями. По этой причине данные аналоги не являются надёжным решением проблем, которые возникнут при отключении GitHub. Следует разработать свою реализацию GOPROXY, которое будет использовать российские сервера, например, облачное решение Yandex Cloud.

ЛИТЕРАТУРА

1. Go Modules Reference. Документация по модулям в Go [Электронный ресурс]. Режим доступа: https://go.dev/ref/mod
2. Managing Dependencies. Руководство по управлению зависимостями в Go. [Электронный ресурс]. Режим доступа: https://go.dev/doc/modules/managing-dependencies
3. Документация по GOPROXY. [Электронный ресурс]. Режим доступа: https://go.dev/ref/mod#goproxy-protocol
4. Руководство по выбору GOPROXY от JFROG. [Электронный ресурс]. Режим доступа: https://argo-cd.readthedocs.io.
5. Дэниелс Кэтрин, Дэвис Дженнифер. Философия DevOps. Искусство управления IT – Питер, 2017, 416 с.
6. Донован, Алан А. А., Керниган, Брайан, У. Д67 Язык программирования Go : Пер. с англ. — М. : ООО “И.Д. Вильямс”, 2016. — 432 с.
7. Microsoft приостановила работу в России. [Электронный ресурс]. Режим доступа: https://www.rbc.ru/technology_and_media/04/03/2022/62223c1a9a7947383d7c45ee

УДК 004.4

Марашов А.С. (4 курс бакалавриата),
Галкин А.С., Смирнов Н.Г., ассистент

СЕРВИС СОЗДАНИЯ АГРЕГИРУЮЩИХ ОЧЕРЕДЕЙ ДЛЯ ОТЛОЖЕННОГО ВЫПОЛНЕНИЯ ЗАПРОСОВ ПО УСЛОВИЮ

Большинство современных сервисов так или иначе предоставляют пользователям возможность публикации своего контента. Недостаточная или некачественная модерация таких публикаций в итоге может уменьшить привлекательность сервиса для пользователей, вплоть до прекращения его использования.

В сервисе модерации социальной сети «Одноклассники» решения о приемлемости модерируемого контента принимаются на основе обработки входящих данных различными инструментами: от поиска плохих паттернов до исследования контента нейронными сетями. Временная сложность получения этой информации различна, поэтому долгие операции, такие как применения нейронных сетей, должны проводиться в другом сервисе – отдельно от общего конвейера модерации.

Проблемы взаимодействия и синхронизации работы таких сервисов решаются использованием брокера сообщений (например, Kafka). Этот инструмент позволяет системам обмениваться информацией и сохранять в очереди данные до тех пор, пока целевой сервис их не обработает.

Реализация такого подхода требует, чтобы в каждый сервис был интегрирован клиент брокера сообщений, однако на практике это не всегда осуществимо. Часть сервисов предоставляют только REST интерфейс взаимодействия, прямое использование которого в данном случае неприемлемо.

Для решения этой проблемы предлагается реализовать инструмент-посредник между сервисами, который сможет организовывать очередь из сущностей и при накоплении нужного количества (или по таймеру) осуществлять запросы в сторонние системы с целью дальнейшего возвращения информации.

Такой инструмент позволит управлять нагрузкой на конкретный сервис путём задания ограничения на количество запросов в секунду и передачи данных партией, а не поштучно. Однако главным преимуществом этого решения перед брокерами сообщений будет подключение дополнительных сервисов, которые поддерживают только HTTP взаимодействие, без внесения изменений в их исходный код, что позволит существенно сэкономить на времени интеграции с ними.

В данной работе используется объектно-ориентированный язык программирования Java. Для создания REST-сервиса, к API-метода которого будут обращаться сторонние системы, используется Spring Framework.

Поскольку сервис и его окружение могут временами быть недоступны, то необходимо поддерживать согласованность данных даже в случае перезапуска системы или проблем с сетью. Поэтому в качестве хранилища данных выступает распределённая система управления базами данных Apache Cassandra. Использование этого инструмента гарантирует не только согласованность данных, но и оптимальное время чтения при высокой скорости записи, свойственной NoSQL-системам.

Приложение будет запущено в нескольких Docker-контейнерах на компьютерах, расположенных в разных дата центрах, что позволит повысить отказоустойчивость сервиса, увеличит его суммарную пропускную способность и обеспечит среду для репликации хранимых данных.

Наряду с такими требованиями как отказоустойчивость сервиса и согласованность данных, важно добавить и минимизацию нагрузки на процессоры. Анализ сервиса модерации,

который имеет похожую архитектуру, показал, что высокая требовательность к ресурсам процессора – одно из его слабых мест.

Упрощённо, структура данных в Кассандре имеет следующий вид: информация разбита на таблицы, которые состоят из строк. Каждая строка – это ключ, по которому к ней можно обращаться, и набор колонок с произвольными данными. И проблема в том, что прикладное ПО, используемое в сервисе модерации для работы с Кассандрой, позволяет указать время жизни информации только на уровне колонки, и для общего случая решает задачу выявления кандидатов на удаление путём регулярного обхода хранимых данных.

В данной задаче рассматривается более частный случай, когда все хранимые сущности должны быть выстроены в очередь по времени их поступления в систему. Организация структуры данных таким образом, чтобы ключом строки была временная метка, а в колонках хранились сущности, которые поступили в систему в этот момент, позволяет не только упрощать поиск и обработку информации, но и помечать сущности для удаления без выполнения регулярного обхода данных. Таким образом, предложенная структура хранения информации позволяет существенно снизить требовательность системы к ресурсам процессора в сравнении с подходом, который используется в сервисе модерации.

ЛИТЕРАТУРА

1. Карпентер Д., Хьюитт Э. Cassandra. Полное руководство. 2-е изд. / пер. с англ. А. А. Слинкина. - М.: ДМК Пресс, 2017. - 400 с.: ил.
2. Как мы модерируем объявления. [Электронный ресурс]. Режим доступа: https://habr.com/ru/company/youla/blog/455128/
3. One-cloud – ОС уровня дата-центра в Одноклассниках. [Электронный ресурс]. Режим доступа: https://habr.com/ru/company/odnoklassniki/blog/346868/

УДК 004.007

Опарко Я. В. (2 курс магистратуры),
Воинов Н. В., к.т.н., доцент,
Дробинцев Д. Ф., старший преподаватель

РАЗРАБОТКА ПРИЛОЖЕНИЯ ДЛЯ КРОССПЛАТФОРМЕННОЙ ВЫСОКОСКОРОСТНОЙ ПЕРЕДАЧИ ФАЙЛОВ ПО WiFi

Целью работы является создание приложения для высокоскоростной передачи файлов по WiFi между устройствами, работающими под управлением следующих операционных систем: Android (начиная с Android Q и новее), Windows 10 (в том числе и Windows 11) и Ubuntu. Для передачи файлов используется только встроенный модуль WiFi, при этом одно из устройств работает в режиме “точка доступа”, тем самым организовывая временную локальную сеть для передачи файлов, защищенную по стандарту WPA2-Personal, а в будущем, возможно, по WPA3. Для аутентификации в сети пользователь может как ввести пароль сети, так и войти с помощью QR-кода (только Android-версия, так как компьютеры обычно не имеют хорошей камеры). Поддерживается режим множественной передачи файлов (когда несколько файлов передаются друг за другом), а также, возможно, для Android версии будет поддерживаться режим передачи установленных приложений. Если оба устройства поддерживают стандарт WiFi 5, то передача должна осуществляться по этому высокоскоростному стандарту на скорости порядка 150 Мбит/с. Также должна иметься возможность поставить передачу на паузу и возобновить ее в дальнейшем.

Для реализации поставленной задачи использовалась среда разработки Android Studio v4.1.3 для мобильной версии приложения и IntelliJ IDEA v2020.3.3 для компьютерной версии. Обе версии приложения написаны на языке Kotlin v1.6.10. Графический интерфейс Android-приложения реализован по принципу: “Одна Activity – несколько фрагментов” [4][5][6].

Управление фрагментами осуществляется с помощью стандартного класса Fragment–Manager [1]. Для генерации QR-кода, который представляет из себя 10-значное число, используется общедоступная библиотека QRGenerator, а для сканирования QR-кода – BarCode Scanner [2][9]. При запуске приложение сначала создает точку доступа, работающую на частоте 2.4 ГГц, а в случае поддержки WiFi 5 обоими устройствами происходит переключение на этот высокоскоростной стандарт. Это сделано в целях совместимости. Все Android-смартфоны, имеющие операционную систему Android Q и новее поддерживают стандарт WiFi 5, однако до сих пор множество компьютеров поддерживают только устаревший стандарт WiFi 4, который использует частоту 2.4 ГГц (частота 5 ГГц для стандарта WiFi 4 используется очень редко). Если приложение будет создавать сеть стандарта WiFi 5, то устройства, поддерживающие только WiFi 4, просто не смогут подключиться к такой сети.

Отдельного внимания заслуживает описание принципа сетевого взаимодействия. Так как приложение должно работать и на Android – смартфонах, и на компьютерах, то библиотека сетевого взаимодействия также должна быть доступна для всех требуемых устройств. Поэтому выбор пал на сетевые TCP – сокеты из стандартной библиотеки Java.net. Протокол TCP гарантирует доставку пакетов до клиента именно в той последовательности, в которой они были отправлены. Это избавляет от необходимости проверки доставленных данных [3]. Передать файл одним большим пакетом невозможно, так как размер такого пакета может превышать несколько Гбайт, что негативно скажется на объеме доступной оперативной памяти и сделает невозможным реализацию паузы передачи. Поэтому файл передается пакетами по 500 Кбайт каждый (размер последнего пакета обычно меньше 500 Кбайт). Перед отправкой самого файла получателю отправляется имя файла и его размер. На основе этой информации получатель будет считывать соответствующие данные из своего сокета. Так как файлов для отправки может быть несколько, то сначала отправляется пакет, содержащий количество файлов, их общий размер и информацию поддержки стандарта WiFi 5 отправителем. Таким образом, все виды пакетов можно поделить на такие типы: пакет, содержащий часть файла, пакет с именем файла и его размером, пакет с количеством отправляемых файлов и их размером, служебные 1 или 2-байтовые пакеты.

Для хранения данных о приостановленных передачах в мобильной версии используется база данных SQL–Lite [7][8]. Для работы с этой базой данных используется библиотека RoomDatabase [10]. В компьютерной версии приложения вероятней всего будет использоваться обычный текстовый файл.

ЛИТЕРАТУРА

1. Брайан Харди, Билл Филлипс, Крис Стюарт, Кристин Марсикано. Программирование под Android. 2-е издание/ Изд-во - Питер, 2019. - 582с.
2. Котеров Д., Симдянов И. PHP 7/ Изд-во БХВ-Петербург, 2017. – 1073с.
3. Кей Хорстманн, Гари Корнелл. Java библиотека профессионала. Том 1. Основы/ Изд-во - Вильямс, 2018. – 864с.
4. Мэтт Зандстра. PHP объекты, шаблоны и методики программирования/ Изд-во – Вильямс, 2017. – 576с.
5. П.Дейтел, Х.Дейтел, Э.Дейтел. Android для разработчиков/ Изд-во – Питер, 2017. – 390с.
6. Сатия Коматинени, Дэйв Маклин. Android 9 для профессионалов - создание приложений для планшетных компьютеров и смартфонов/ Изд-во – Вильямс, 2019. – 873с.
7. Ян Ф. Дарвин. Android. Сборник рецептов. Задачи и решения для разработчиков приложений/ Изд-во – Диалектика, 2018. – 768с.
8. «7 Android библиотек, о которых должен знать каждый разработчик» [Электронный ресурс] – Режим доступа: URL: http://www.mobilab.ru/androiddev/7androidlibraries.html/ (дата обращения: 10.03.2022).
9. Программирование для Android. Самоучитель / Колисниченко Д. - СПб.: Санкт-Петербург, 2019. - 736 с
10. «Документация по REST» [Электронный ресурс] – Режим доступа: URL: https://dev.1c-bitrix.ru/rest_help/ (дата обращения 12.03.2022).

УДК 004.056

Павлова А.А. (4 курс бакалавриата),
Орлов Е.С., старший преподаватель

РАЗРАБОТКА ДЕЦЕНТРАЛИЗОВАННОГО ПРИЛОЖЕНИЯ НА ОСНОВЕ ТЕХНОЛОГИИ БЛОКЧЕЙН

Основной целью данной работы является разработка децентрализованного приложения на основе технологии Блокчейн для работы с данными о пассажирах и авиарейсах. В ходе выполнения работы необходимо разработать приложение, предоставляющее информацию о покупке билетов у авиакомпаний. Разрабатываемая система должна получать и хранить данные о пассажирах, а именно паспортные и контактные данные каждого из пассажиров.

В настоящее время вопрос безопасного хранения личных данных особо актуален. Когда дело касается покупки билетов на самолет, нам необходимо предоставить свои паспортные данные, а это одни из тех данных, разглашение которых может принести серьезный ущерб. Недопустимо, что эти данные могли быть получены в случае несанкционированного доступа. Также нельзя допускать, чтобы эта информация была каким-либо образом изменена. Гарантировать защиту данных от порчи или изменений третьими лицами может технология Блокчейн, а также шифрование данных, заложенное в основе этой технологии.

Блокчейн — это выстроенный определенный алгоритм, непрерывная последовательная цепочка блоков, хранящая информацию. Главным принципом работы данной технологии является прозрачность совершаемых операций, с недоступностью их изменить лицам, не имеющим санкционированного к ним доступа.

В этой работе рассматривается создание децентрализованного приложения для отслеживания и работы с данными о пассажирах, покупающих билеты на авиарейсы. Это приложение удостоверяет каждую запись о покупке билетов специальной записью с хеш-суммой. Из строк данных выстраиваются цепочки, которые невозможно исправить или удалить. В случае если произошла подмена данных или их удаление, то система сможет определить исправленную строку и в автоматическом режиме запросить испорченные данные с других серверов. В итоге полностью исключается потеря или порча данных.

В качестве языка программирования данной работы был выбран язык C#. C# - это объектно-ориентированный язык программирования, который используется для создания масштабируемых приложений, совместимых с .NET Framework. Его происхождение восходит к 2000 году и с тех пор используется для разработки надежных кроссплатформенных кодов, которые работают с разными типами ОС, такими как Android, Mac, Windows и Linux.

ЛИТЕРАТУРА

1. Using blockchain technology for data protection.
International Journal of Open Information Technologies ISSN: 2307-8162 vol. 9, no. 9, 2021
(дата обращения: 20.03.2022)
2. Blockchain Programming in C#. https://finbuzzactu.files.wordpress.com/2017/06/blockchain-programming-in-csharp.pdf
(дата обращения: 20.03.2022)
3. Blockchain as a new technology for development.
International Journal of Open Information Technologies ISSN: 2307-8162 vol. 7, no.1, 2019
(дата обращения: 20.03.2022)

УДК 004.942

Сабуткевич А.М., Вихляев Д.А. (1 курс магистратуры),
Никифоров И.В., к.т.н., доцент,
Самочадин А.В., к.т.н., доцент

ИМИТАЦИОННОЕ МОДЕЛИРОВАНИЕ ПОВЕДЕНИЯ СЛОЖНЫХ МНОГОАГЕНТНЫХ СИСТЕМ С ИСПОЛЬЗОВАНИЕМ ВЕРОЯТНОСТНОЙ МОДЕЛИ

Имитационное моделирование является наиболее эффективным и перспективным средством проектирования и дальнейшего исследования сложных систем [1]. Использование имитационного моделирования в многоагентных системах позволяет в процессе моделирования исследовать поведение каждого агента в частности, а также влияние их поведения друг на друга в рамках модели. Динамика исследования таких систем определяется правилами, которые формируются на основе результатов индивидуальной активности агентов [2]. Данный подход наиболее актуален в социально-экономических сферах [3].

Целью данной работы является рассмотрение подхода к принятию решений в процессе имитационного моделирования в сложных многоагентных система на основе оценок эффективности, описывающихся вероятностной моделью.

В рамках исследования данной темы были рассмотрены решения и подходы, описанные в работах [4], [5] и [6]. В качестве общего недостатка было выделено отсутствие влияния изменения оценки эффективности текущего решения на зависимые последующие, что сводит схему принятия решения к применению жадных алгоритмов.

В поведенческой модели сложных систем на прикладном уровне можно выделить следующие компоненты [4]:

– создание и актуализация модели мира [7];
– формирование и принятие решений;
– исполнение решений.

Внутренняя модель мира представляется как совокупность следующих компонентов:

– множества действий, характеризующихся набором параметров и отношениями порядка, заданными на данном множестве. Значения указанных параметров определяют текущее состояние соответствующего действия.
– множества агентов, в текущий момент времени находящихся в состоянии выполнения конкретного действия.

В совокупности текущие состояния действий и агентов определяют общее состояние модели мира.

Актуализация модели мира во времени происходит на основе информации, поступающей из внешнего мира. Данная информация нацелена на изменение состояния модели мира посредством изменения параметров и характеристик ее компонентов.

Процесс, связанный с формированием и принятием решений, основывается на анализе текущего состояния внутренней модели мира. На прикладном уровне указанный процесс можно декомпозировать на следующие подпроцессы:

– распознавание состояния агента;
– формирование текущих целей;
– принятие решений.

В ходе актуализации модели наравне с обновлением значений параметров ее компонентов может поступать информация об изменении положения агентов относительно выполняемых ими действий во внешнем мире. В связи с этим на этапе распознавания состояния агента необходимо определить текущее состояние всех агентов внутренней модели, чтобы в дальнейшем спровоцировать их переход к выполнению тех действий, которые в текущий момент выполняются ими во внешнем мире на основе полученной информации.

В сложных системах возможно выделение нескольких уровней целей [4]:

– главная цель – результирующее целевое действие, завершение выполнения агентом которого характеризует завершение текущего сеанса моделирования;

– промежуточная цель – промежуточное действие, выполнение агентом которого позволяет наиболее эффективно достичь главной цели в условиях текущего состояния модели мира.

У агента в текущий момент времени может быть выработано несколько промежуточных целей, каждая из которых может иметь равную оценку эффективности достижения главной цели. В связи с этим на данном этапе необходимо определить набор правил, позволяющих агенту самостоятельно принимать решение о выборе промежуточной цели.

В качестве оценки эффективности достижения главной цели для промежуточных используется вероятностная модель. В начальный момент времени каждому действию сопоставляется положительное числовое значение, которое определяет вероятность перехода агента в данное действие. Задание начальных значений может осуществляться как на основе экспертных оценок, так и при помощи равновероятностной схемы.

Вероятность перехода в действие определяется как отношение вероятности перехода в предшествующее действие к количеству его смежных действий, умноженное на произведение всех раннее примененных к этому действий корректирующих коэффициентов:

$$P_{following} = \frac{P_{previous}}{n_{following}} * k,$$

где $P_{following}$– вероятность перехода в действие, связанное с текущим отношением следования (смежное действие);

$P_{previous}$– вероятность перехода в предшествующее действие;

$n_{following}$– количество действий, связанное с текущим отношением следования;

k–произведение всех раннее примененных к этому действий корректирующих коэффициентов.

Информация, поступающая из внешнего мира, может оказывать влияние на значения данных оценок, так как они относятся к параметрам действий и вследствие этого также определяют состояние внутренней модели мира. На прикладном уровне это реализуется посредством умножения вероятности перехода в действие на специальный коэффициент, инкапсулированный во внешнюю информацию, и осуществлением повторного вычисления оценок для всех действий, находящихся в отношении следования с действием, обновление вероятности которого осуществляется.

На этапе исполнения решения система инициирует переход агента к выбранной промежуточной цели. Дополнительными ограничениями на изменение состояния агента относительно выполняемого им действия могут выступать необходимость соблюдения логических условий, сформулированных относительно компонентов модели мира, а также условий, накладываемых на временные характеристики модели.

Описанный инструмент моделирования разработан в формате настольного приложения. В качестве основного языка для программной реализации инструмента выбран Python. Внутренняя модель мира в рамках модели данных на верхнем уровне представлена посредством двух коллекций: действий и агентов. На абстрактном уровне модель мира описывается ациклическим ориентированным графом, узлами которого являются действия, выполняемые агентами, а дуги определяют отношения порядка. Использование данной абстракции позволило свести метод вычисления вероятностей перехода к модифицированному алгоритму обхода в глубину графа DFS [8]. Для графического представления модели мира использована библиотека NetworkX.

Разработанный инструмент позволяет осуществлять процесс моделирования поведения сложных систем с возможностью оказания влияния на внутреннюю модель мира посредством поэтапной ее актуализации на основе информации из внешнего мира. Использование предложенной вероятностной модели для оценки эффективности выбора текущей цели агентом позволяет, во-первых, учитывать влияние изменения оценки текущего решения на

зависимые, а, во-вторых, оказывать влияние на выбор текущей цели для каждого агента в частности за счет информации из внешнего мира.

ЛИТЕРАТУРА

1. Городецкий В.И., Самоорганизация и многоагентные системы. I. модели многоагентной самоорганизации // Известия Российской академии наук. Теория и системы управления. – 2012. – № 2. – С. 92.
2. Шпаркин А.М. Исследование существующих подходов к имитационному моделированию воздушной тактической обстановки в автоматизированных информационных системах / Воздушно-космические силы. Теория и практика. – 2019. – №12. – С. 168–175.
3. Макова А.С. Перспективы развития имитационного моделирования / Современные наукоемкие технологии. – 2014. – №7. – С. 59–60.
4. Балута В.И., Яковенко О.Ю. Формализация описания сложного поведения объектов в задачах имитационного моделирования систем физической защиты // Препринты ИПМ им. М.В. Келдыша. 2016. № 3.
5. Бабкин Е. А. Разработка событийного имитационного графа // Auditorium. – 2016.
6. Антонова В.М. Моделирование графов с различными видами достижимости спомощью языка Python / Антонова В.М , Захир Б.М., Кузнецов Н.А. // Информационные процессы. – 2019. – № 2. – С.159–169.
7. Никифоров, И. В. Методы автоматизации построения поведенческой модели программного продукта на основе UCM-спецификаций: специальность 05.13.11 "Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей" : диссертация на соискание ученой степени кандидата технических наук / Никифоров Игорь Валерьевич. – Санкт-Петербург, 2013. – 205 с.
8. Кормен Т. Алгоритмы. Построение и анализ / Т. Кормен, Ч. Лейзерсон, Р.Ривест. – М: Диалектика, 2019. – 471 с.

УДК 004.415.2

Селезнев В.А. (4 курс бакалавриата),
Прокофьев О.В., старший преподаватель

РАЗРАБОТКА ПРИЛОЖЕНИЯ ПО РАСПОЗНАВАНИЮ ПЛАГИАТА С ИСПОЛЬЗОВАНИЕМ POSTGRESQL

Программное обеспечение, которое будет реализовываться в рамках научно-исследовательской работы, будет представлять собой web-сервис. Используя машинное обучение, будет создано приложение для проверки на плагиат. Суть его в поиске украденного контента в огромной базе данных.

В приложении будет доступно две роли: администратор и клиент и будет реализовано два интерфейса. Это позволит каждому из них взаимодействовать с базой данных PostgreSQL из своего аккаунта.

Администратор будет загружать новые данные в PostgreSQL с целью переобучения модели. Использоваться будет модель среднего количества сопоставлений слов. Это алгоритм обучения без учителя для получения векторных представлений слов. Обучение проводится на основе агрегированной глобальной статистики встречаемости слов.

У клиентов будет возможность загружать файлы, из которых впоследствии будет получена информация, которая находится внутри них. Это необходимо, чтобы воспользоваться SDK Pinecone — это алгоритм поиска заимствований. Данный инструмент работает путем сравнения векторов. На выходе алгоритма получаем те фрагменты текста, которые были скопированы.

Логика приложения будет реализована на языке программировании Python в среде разработки PyCharm. Фреймворк Flask будет использован для создания веб-приложения. Пользовательский интерфейс будет выполнен на JavaScript.

Актуальность данного подхода.

Актуальность заключается в том, что в разрабатываемом приложение в качестве хранилища информации будет выступать база данных PostgreSQL. Более того, будет доступна роль администратора. Все это позволит регулярно переобучать модель, чтобы она была способна распознавать плагиат в различных областях: отчеты студентов, научные статьи и тд.

Реализуемое решение позволит, например, распознавать скопированный контент в рамках академической группы с условием, что человек, с которого был заимствован фрагмент, не использовал информацию из сети интернет.

В России самым популярным средством по распознаванию плагиата считается “Антиплагиат.ру”. Этот ресурс не может справиться с данной задачей, так как суть его заключается в поиске информации из интернета.

ЛИТЕРАТУРА

1. Фреймворк для создания веб-приложений на языке программирования Python. [Электронный ресурс] https://flask.palletsprojects.com/en/2.0.x/
2. PostgreSQL and Machine Learning. [Электронный ресурс] https://mljar.com/blog/postgresql-machine-learning/
3. Библиотека векторного поиска. [Электронный ресурс] https://www.pinecone.io/
4. Средство проверки на плагиат. [Электронный ресурс] https://habr.com/ru/post/582094/
5. Поиск заимствований. [Электронный ресурс] https://habr.com/ru/company/antiplagiat/blog/429634/
6. Описание Антиплагиат.ру. [Электронный ресурс] https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D1%82%D0%B8%D0%BF%D0%BB%D0%B0%D0%B3%D0%B8%D0%B0%D1%82

УДК 004.4

Скрипчук А.В. (4 курс бакалавриата),
Воинов Н.В., к.т.н., доцент,
Каплан Е.В., старший преподаватель

КЛИЕНТ-СЕРВЕРНОЕ ПРИЛОЖЕНИЕ ДЛЯ УПРАВЛЕНИЯ ПАРОЛЯМИ

Цель работы – разработать приложение для управления паролями. Создаваемое приложение призвано упростить организацию паролей, обеспечить безопасное хранение паролей и других данных учетных записей пользователя, предоставить возможность генерации сложных паролей.

Безопасность хранимых данных обеспечивается применением алгоритмов шифрования к хранимым данным и установкой мастер-пароля – пароля, который требуется вводить для доступа к базе паролей [1].

Планируется реализация десктопного приложения с набором следующих основных функций:

Возможность создать базу паролей, задав для доступа к ней мастер-пароль.

Возможность сгенерировать случайный пароль, задав такие параметры, как длина пароля и типы входящих символов (цифры, латинские буквы верхнего и нижнего регистров, некоторые специальные символы) [2].

Возможность сохранить и изменить запись, состоящую из логина, пароля и опциональных атрибутов (например, URL соответствующего ресурса, текстовые комментарии) [3].

Возможность импортировать и экспортировать хранимые записи в формате XML.

Возможность хранить не только записи с паролями, но и файлы, так же в зашифрованном виде.

Возможность проанализировать пароль на слабость на основе его длины, типов входящих символов, повторяемости отдельных символов и подстрок. Анализ применим как к одному указанному паролю, так и ко всем хранимым паролям.

Возможность проверить, фигурирует ли указанный пароль в базе утечек паролей. Проверка применима как к одному указанному паролю, так и ко всем хранимым паролям [3].

Для реализации приложения были выбраны язык программирования Java и фреймворк Swing. Пользовательский интерфейс будет реализован средствами Swing [4]. Для реализации серверной части планируется использование средств Java-платформы, в частности, компонента Java cryptography API, включающего в себя реализацию алгоритма AES, который будет использован для шифрования данных [5].

ЛИТЕРАТУРА

1. Best Password Managers For Windows In 2022. [Электронный ресурс]. Режим доступа: https://www.forbes.com/advisor/business/software/best-password-managers-for-windows/
2. Password Generator powered by NordPass. Режим доступа: https://nordpass.com/password-generator/
3. The Bitwarden Password Manager. [Электронный ресурс]. Режим доступа: https://bitwarden.com/products/
4. Как создать графический интерфейс с примерами Swing на Java. [Электронный ресурс]. Режим доступа: https://hr-vector.com/java/swing-graficheskij-interfejs
5. Java Cryptography. [Электронный ресурс]. Режим доступа: http://tutorials.jenkov.com/java-cryptography/index.html

УДК 004.4`2

Соколова О.А. (2 курс магистратуры),
Амосов В.В., к.т.н., доцент

ВНЕДРЕНИЕ ВЫБОРОЧНОГО ТЕСТИРОВАНИЯ В ПРОЦЕСС НЕПРЕРЫВНОЙ ИНТЕГРАЦИИ ДЛЯ ОПТИМИЗАЦИИ ПРОЦЕССА РАЗРАБОТКИ

Тестирование является важнейшим этапом разработки качественного программного продукта. При добавлении новой функциональности в продукт необходимо регулярно проверять, что существующая функциональность работает корректно. Данный процесс называют регрессионным тестированием. По мере роста проекта увеличивается количество тестов, необходимых для подтверждения работоспособности существующей функциональности, и их выполнение может занимать довольно продолжительное время.

Существует подход, позволяющий ускорить процесс выявления появившихся в проекте дефектов. Выборочное тестирование подразумевает выполнение не всего тестового набора, а лишь его части, покрывающей внесенные при последнем слиянии изменения. Многие методы выборочного регрессионного тестирования объединяют два этапа: поиск зависимостей между исходным кодом программы и тестами и идентификация изменений, сделанных в коде. Среди методов выборочного тестирования можно выделить группу методов, использующих статический анализ кода для поиска зависимостей [1], [2] или динамический [3], [4], [5]. Также существуют гибридные методы, включающие в себя оба подхода. Кроме того, методы могут отличаться уровнем гранулярности: строка, метод, класс, модуль.

На Рисунке 1 приведена архитектура предлагаемого подхода. После добавления изменений в Git репозиторий модуль ChangesProcessor обрабатывает полученные изменения и вызывает модуль TestsListCreator. На начальном этапе разработки в качестве выходных данных модуля ChangesProcessor может выступать список изменённых файлов. При развитии

системы можно провести эксперименты с изменёнными методами/строками кода. Далее модуль TestsListCreator, используя информацию из хранилища, создаёт список тестов, необходимых для покрытия сделанных изменений и передаёт их в Jenkins [6], инструмент непрерывной интеграции.

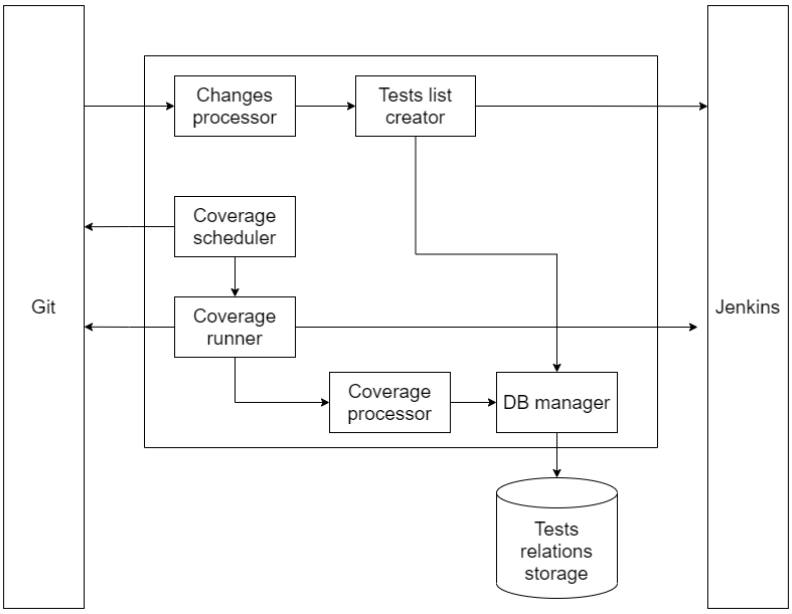


Рисунок 1 – Архитектура предлагаемого подхода

Предлагаемый алгоритм можно разделить на 2 части: получение и обработка информации о покрытии кода (1), использование информации о покрытии кода для формирования тестового набора (2). Ниже описываются обе части алгоритма.

Часть (1):

1. Выполнение регулярного регрессионного тестирования с подключённым JaCoCo агентом [7].
2. Обработка выходного файла JaCoCo агента и формирование отчёта о покрытии кода при помощи JaCoCo CLI [8].
3. Добавление в базу данных информации о связи каждого файла проекта с тестами, которые его покрывают.

Часть (2):

1. Срабатывание триггера на добавление изменений в Git репозиторий.
2. Получение с помощью GitHub REST API [9] списка изменённых файлов (классов/методов).
3. Обращение к базе данных и формирование списка тестов, полностью покрывающих изменения.

Таким образом, предлагаемый подход поможет сократить время между внесением дефекта разработчиком и моментом его выявления при регрессионном тестировании. Сценарий использования данного подхода может быть следующим: регулярное выполнение регрессионного тестирования (например, один раз в 24 или 48 часов, зависит от размера регрессионного набора) и выполнение выборочного тестирования после добавления изменений в исходный код проекта.

Важно отметить, что наряду с аналогичными системами, предлагаемый подход имеет преимущество, заключающееся в том, что благодаря использованию JaCoCo агента, не требуется внесение изменений в исходный код проекта, а также система будет собирать информацию о покрытии кода независимо от используемого инструмента тестирования.

ЛИТЕРАТУРА

1. Owolabi Legunsen, August Shi D.M. STARTS: STAtic Regression Test Selection // 2017 32nd IEEE/ACM International Conference on Automated Software Engineering (ASE). 2017. № 6(3). С. 1–61.

2. Shi A., Hadzi-Tanovic M., Zhang L., Marinov D., Legunsen O. Reflection-aware static regression test selection // Proceedings of the ACM on Programming Languages. 2019. № OOPSLA(3). DOI:10.1145/3360613.
3. Gligoric M., Eloussi L., Marinov D. Practical regression test selection with dynamic file dependencies // 2015 International Symposium on Software Testing and Analysis, ISSTA 2015 - Proceedings. 2015. (520). С. 211–222. DOI:10.1145/2771783.2771784.
4. Zhang L. Hybrid regression test selection2018. С. 199–209. DOI:10.1145/3180155.3180198.
5. OpenClover [Электронный ресурс]. URL: https://openclover.org/ (дата обращения: 11.02.2022).
6. Документация Jenkins [Электронный ресурс]. Режим доступа: https://www.jenkins.io/doc/book/.
7. Документация JaCoCo [Электронный ресурс]. Режим доступа: https://www.jacoco.org/.
8. Документация JaCoCo CLI [Электронный ресурс]. Режим доступа: https://www.jacoco.org/jacoco/trunk/doc/cli.html.
9. Документация GitHub REST API [Электронный ресурс]. Режим доступа: https://docs.github.com/en/rest.

УДК 004.925.3

Стоцкая Е.Ю. (4 курс бакалавриата),
Леонтьева Т.В., к.т.н., доцент

РЕАЛИЗАЦИЯ ДЕКАЛЕЙ С ПОМОЩЬЮ МЕТОДА ТРАССИРОВКИ ЛУЧЕЙ

Трассировка лучей – один из самых распространенных на сегодняшний день методов построения изображений в компьютерной графике. Его основное преимущество состоит в реалистичности получаемых изображений – так как алгоритм вычисления цвета в каждой точке строится на формулах оптики, метод позволяет довольно точно отображать, например, тени, преломление света, зеркальные и прозрачные материалы и др. На данный момент этот метод используется для широкого круга задач, а также продолжают разрабатываться его новые разновидности, обеспечивающие еще большую точность изображений.

В некоторых разделах компьютерной графики, в основном в разработке видеоигр, используются декали. Это объекты, накладывающиеся поверх других элементов сцены и изменяющие их текстуру и свойства материала в той области, на которую они спроецированы. По большей части эта техника применяется для отображения динамически меняющихся деталей сцены или небольших предметов, отличающихся по свойствам от окружающих поверхностей. Однако не для любых динамических элементов сцены нужны декали – обычные текстуры также могут быть динамическими и отображать движущиеся и меняющиеся объекты, например, такие, как огонь или морские волны. Основная особенность декалей состоит в том, что, в отличие от текстуры, являющейся одним из свойств объекта, декаль рассматривается как отдельный объект, независимый от поверхности, на которую его проецируют. Из этого следует, во-первых, что декаль обладает, кроме текстуры, другими свойствами, такими, как прозрачность и отражательная способность, и будет выделяться на фоне поверхности не только отличающимся рисунком, но и оптическими характеристками. Во-вторых, текстуры обычно представляют собой некоторый повторяющийся рисунок, автоматически дублирующийся при изменении размеров объекта. Следовательно, с их помощью невозможно изобразить элемент, не повторяющийся с остальной текстурой. В таких случаях и могут использоваться декали – например, если в сцене присутствует стена, покрытая стандартной повторяющейся текстурой (кирпич, обои), с помощью декали можно изобразить, к примеру, картину, висящую на стене, или динамически появляющуюся на ней трещину. В основном данная технология сейчас реализована для более простых методов компьютерной

графики, более часто используемых для видеоигр в силу большей скорости вычислений. Пример использования декали в сцене представлен на Рисунке 1 [1].

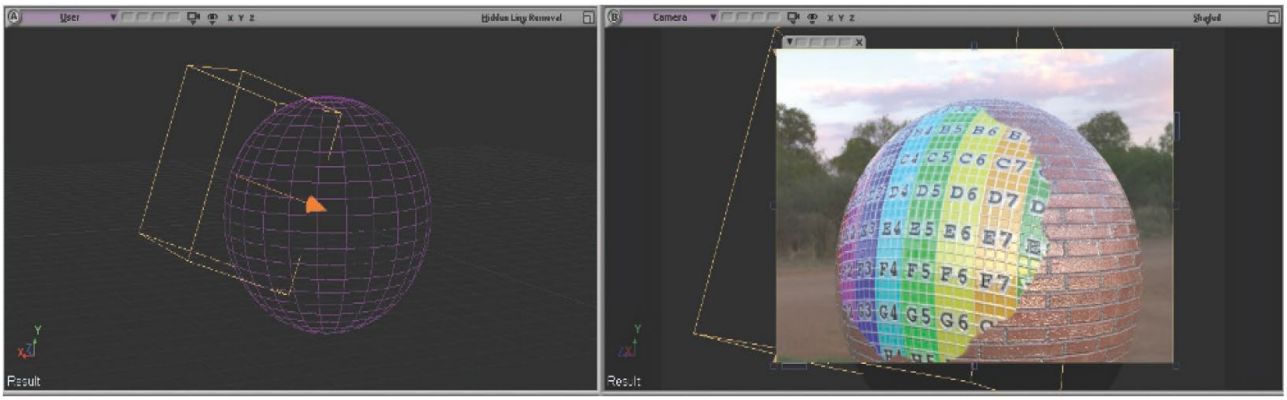


Рисунок 1 – Пример использования декали для изменения свойств материала для части объекта (слева – отображение 3D-модели в редакторе, справа – итоговое изображение)

Целью работы является разработка программного обеспечения, реализующего декали в рамках метода трассировки лучей, и сравнение реалистичности изображений, построенных с помощью данной программы, с результатами использования других алгоритмов.

Так как данную тему начали исследовать недавно, была найдена только одна публикация [1] решения подобной задачи, где для реализации использовалась среда DirectX [2]. Одним из инструментов этой среды является библиотека функций для трассировки лучей. В работе описано несколько возможных подходов к релизации декалей с ее использованием и проведено сравнение производительности разных вариантов. Полученные результаты имеют высокую ценность как ориентиры для дальнейшей разработки. Наше решение предполагает использование другой графической библиотеки, OpenGL [3], отличающейся от DirectX универсальностью для различных операционных систем, а также тем, что она основана на открытом стандарте. При этом, в этой библиотеке нет стандартных встроенных функций для трассировки лучей, однако существуют доступные реализации, написанные с использованием OpenGL [4, 5], которые не стандартизованы и их характеритики изначально неизвестны. Поэтому алгоритм должен быть реализован на более низком уровне, что позволит оптимизировать отдельные детали решения и выбирать наиболее подходящие варианты в процессе разработки.

Для решения задачи предполагается взять за основу проект [5], в котором уже реализован метод трассировки лучей с помощью инструментов библиотеки OpenGL. Декали планируется добавить в виде отдельного класса геометрических объектов и реализовать алгоритм их обработки, обеспечивающий корректную замену текстур и других свойств на нужных этапах конструирования сцены. Данный пример [5] был выбран в качестве основы для работы как достаточно полный и качественный, поскольку он, в частности, позволет работать с широким кругом геометрических объектов. Другое его преимущество состоит в использовании известных и общедоступных вспомогательных инструментов, таких, как GLFW, что делает его более универсальным по сравнению с реализациями, использующими для тех же целей самодельные или малоизвестные библиотеки.

После создания программы результаты ее работы могут быть оценены путем сравнения полученных с ее помощью изображений с образцами, построенными с использованием более старых алгоритмов, с точки зрения реалистичности. Несмотря на то, что для нее не существует количественной меры, реалистичность (или в частном случае фотореалистичность) часто используется как показатель качества методов компьютерной графики. Этот показатель мы считаем наиболее важным. С другой стороны, для алгоритмов, используемых в таких задачах, как разработка видеоигр, не менее важным параметром является производительность, которая закономерно ухудшается при повышении сложности метода. Имеет смысл сравнить производительность разработанной программы с результатами, полученными в описанной

реализации на базе DirectX, в частности, для оценки целесообразности использования каждой из библиотек для различных вариантов алгоритма.

ЛИТЕРАТУРА

1. Bahnassi W. (2021) Ray Tracing Decals. In: Marrs A., Shirley P., Wald I. (eds) Ray Tracing Gems II. Apress, Berkeley, CA.
2. DirectX graphics and gaming / Интернет-портал Microsoft. URL: https://docs.microsoft.com/en-us/windows/win32/directx (дата обращения 03.03.2022).
3. OpenGL. The Industry's Foundation for High Performance Graphics / Интернет-портал OpenGL. URL: https://www.opengl.org/ (дата обращения 03.03.2022).
4. PathTracer / Интернет-портал Github. URL: https://github.com/MomoDeve/PathTracer (дата обращения 03.03.2022).
5. Realtime raytracing / Интернет-портал Github. URL: https://github.com/engilas/raytracing-opengl (дата обращения 03.03.2022).

УДК 004.42

Соколова А.Е. (2 курс магистратуры),
Ковалев А.Д., ассистент,
Никифоров И.В., к. т. н., доцент

ПРОГРАММНАЯ СИСТЕМА АВТОМАТИЗАЦИИ ПРОВЕДЕНИЯ РЕЧЕВОЙ АУДИОМЕТРИИ

Существует множество причин нарушений слуха, к которым можно отнести: генетическую предрасположенность, перенесенные травмы, высокий уровень постоянного шума в городе, и другие [1]. При возникновении проблем со слухом повысить качество жизни позволяют слуховые аппараты, однако для обеспечения корректной и эффективной работы они требуют гибкой индивидуальной настройки под каждого пациента [2].

Настроить слуховой аппарат позволяет проведение речевой аудиометрии, для которой используются программно-аппаратные комплексы. Существующие решения требуют от врача-сурдолога дополнительной ручной настройки, поэтому актуальной является задача минимизации времени и усилий, затрачиваемых на проведение речевой аудиометрии [3-6]. Это достигается за счет создания программного средства для автоматизации ручных этапов исследования остроты слуха с применением инструментов распознавания речи.

На сегодняшний день для работы с аудиометрами существуют специализированные программные продукты, например, Audibase [7], ampliSuite [8], SIBELMED W50 [9]. Они позволяют вести базу данных пациентов, управлять настройками аудиометра и считывать с него информацию. Сравнительный анализ таких систем представлен в Таблице 1.

Таблица 1 – Обзор программных средств проведения аудиометрии

Название программы	Поддерживаемая модель аудиометра	Стоимость программного средства	Поддержка речевой аудиометрии	Распознавание речи	Построение аудиограммы	Формирование отчета
Audiometry C#	Не зависит от модели аудиометра	Открытое программное обеспечение	+	-	+	+
SIBELMED W50	SIBELSOUND DUO, SIBELSOUND 400	Активация от аудиометра	+	-	+	+
Audibase	Model 116, Model 170, PC850, Otosure	Поставляется с аудиометром	+	-	+	+
OTOSuite	Madsen Astera	Поставляется с аудиометром	+	-	+	+
ampliSuite	Model 116, Model 170, PC850, Otosure, Model 240, Model 260, Model 270, Model 270+	Поставляется с аудиометром	+	-	+	+

На основе сравнительного анализа можно сделать следующие выводы: в большинстве своем программные средства зависимы от определенных моделей аудиометров, не обладают возможностью распознавания речи и не полностью поддерживают весь бизнес-процесс аудиометрии.

Традиционно речевая аудиометрия требует от врача-сурдолога ручной работы на всех этапах тестирования. Предлагаемый подход к проведению исследования объединяет большинство этапов в последовательную цепочку, выполняемую в рамках одного программного средства. Информация о пациенте и враче, а также настройки заполняются в последовательных панелях приложения. Записи для прослушивания в формате wav можно выбрать случайным образом из всего доступного списка или же по определенным частям речи. Проигрывание записи может производиться как поэтапно традиционным способом, так и автоматически. Ответ автоматически распознается и результат для записи сохраняется. После проигрывания всех записей врачу предоставляется возможность проверить и скорректировать при надобности результаты, сгенерировать отчет в формате docx и построить аудиограмму в виде картинки формата png. Исходные данные аудиограммы сохраняются в табличном формате csv. Также есть возможность провести следующую итерацию тестирования, вернувшись на панель настроек. Данная схема позволяет сократить усилия, затрачиваемые сурдологом на проведение тестирования. Разработанное программное средство, реализующее предлагаемый подход, было написано с использованием библиотек языка Python.

В качестве инструмента распознавания в данной работе был использован готовый сервис, предварительно обученный на датасете, собранном на видеохостинге YouTube. Сервис содержит инструменты Vosk [10] и Kaldi [11], доступ осуществляется по веб-сокетам. Инструмент поддерживался ресурсами Санкт-Петербургского политехнического университета, на данный момент его внешний порт отключен.

Для оценки преимуществ использования системы было проведено несколько экспериментов с разным количеством прослушиваемых записей: 5, 50, 150. Эксперименты для каждого количества проводились традиционным методом и с помощью автоматизированной системы.

Автоматизация наиболее заметна на нескольких этапах работы программы, а именно выборе случайных записей, этапе распознавания речи и генерации отчета. В ходе каждого эксперимента записывалось время, затраченное на каждый из этапов. Также подсчитывалось общее время проведения речевой аудиометрии для эксперимента с каждым числом звуковых записей. Результаты представлены на Рисунке 1.

Как можно заметить, трудоемкость проведения речевой аудиометрии автоматизированным методом снижается линейно по сравнению с ручным методом в зависимости от количества записей, от 47% до 67% для проведенных тестов.

На данный момент в результате работы были выполнены следующие задачи:

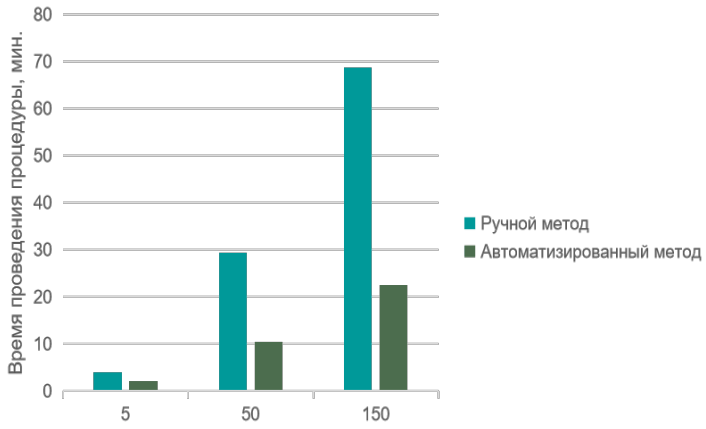


Рисунок 1 – Среднее время, затраченное на проведение процедуры

1. Проведен обзор существующих программных решений для проведения речевой аудиометрии.
2. Предложена концептуальная схема подхода с применением инструментов распознавания речи.
3. Реализовано программное средство, генерирующее отчет с результатами процедуры аудиометрии.
4. Проведено тестирование инструмента в процедуре речевой аудиометрии, время проведения

снижается линейно в зависимости от количества записей.

5. Представлены преимущества использования системы.

Дальнейшие исследования будут направлены на улучшение качества распознавания и оптимизацию работы системы.

ЛИТЕРАТУРА

1. Deafness and hearing loss. [Электронный ресурс]. Режим доступа: https://www.who.int/news-room/fact-sheets/detail/deafness-and-hearing-loss, свободный
2. J. B. Nielsen, J. Nielsen. Efficient individualization of hearing aid processed sound // 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, 2013, pp. 398-402.
3. Drobintsev P., Voinov N., Kotlyarova L., Selin I., Aleksandrova O. Optimization of Technological Processes at Production Sites Based on Digital Modeling // Lecture Notes in Electrical Engineering. 2020. Т. 634 LNEE. С. 600-607.

УДК 004.4

Чавес Кирос Г.Г. (2 курс магистратуры),
Воинов Н.В., к.т.н., доцент,
Каплан Е.В., старший преподаватель

РАЗРАБОТКА JAVASCRIPT-ФРЕЙМВОРКА ДЛЯ ГЕНЕРАЦИИ СЕРВЕРНЫХ ВЕБ-ПРИЛОЖЕНИЙ

Разработка веб-приложений является одной из наиболее востребованных областей разработки программного обеспечения. В области веб-разработки есть две четко определенные среды: первая — это фронтенд, представляющая собой уровень представления веб-приложения[1], то есть часть, с которой пользователь взаимодействует через графическую среду, интерфейс приложения, а второй — бэкенд, отвечающий за обработку данных[2], который является частью, где устанавливается связь между действиями пользователя и базой данных. Эта часть не видна, пользователь делает запросы, подключаясь к бэкенду через интерфейс приложения на фронтенде, и как только запрос получен, бэкенд возвращает ответ на запрос.

В настоящее время существует множество языков программирования и инструментов, доступных для веб-разработки как для фронтенда, так и для бэкенда, но при их разработке вы должны учитывать множество факторов: время, необходимое для процесса разработки, количество разработчиков рабочей группы, требования к веб-приложению и др. Для решения этой проблемы сегодня существуют пакеты библиотек и фреймворков, которые представляют собой инструменты, позволяющие оптимизировать работу по созданию программы или приложения за меньшее время и с меньшими усилиями. Рассмотрим JavaScript как язык разработки. Он был создан как язык для разработки фронтенд веб-приложений, хотя несколько лет назад стало возможным использовать его для бэкенда благодаря Node.js, платформе, позволяющей использовать код JavaScript для создания серверов для взаимодействия с фронтендом и выполнять некоторые функции, такие как подключение к другим серверам и обмен информацией, подключение и управление базами данных, управление сессиями пользователей. На основе Node.js был создан Express.js, фреймворк для разработки веб-серверов с использованием JavaScript, он предоставляет упрощенный API для основных функций Node.js[3], и, хотя существует множество библиотек для выполнения задач, которые может дополнять Express.js, многие другие фреймворки были построены поверх Express.js на основе Express.js, чтобы предоставить больше возможностей и преимуществ при его использовании по сравнению с использованием только Express.js.

Хотя в настоящее время существует множество вариантов выбора фреймворка для создания веб-серверов и API-интерфейсов, однако, проблема быстрого старта разработки с

готовой средой библиотек и фреймворков, выполняющих реализацию некоторых функций в рамках внутреннего проекта, ещё полноценно и успешно не решена. Это снижает вероятность более быстрого запуска приложения без особых сложностей, поскольку, как правило, весь этот процесс выполняется с нуля, и необходимо просмотреть документацию каждой библиотеки и фреймворка, чтобы начать процесс установки и импорта инструмента в папку проекта приложения с последующей его реализацией.

Цель этой работы сосредоточена на бэкенд разработке и состоит в разработке инструмента, позволяющего создавать бэкенд проекты разработки, специализирующиеся на разработке веб-сервисов[4] под архитектурой REST[5], с использованием библиотек и фреймворков на языке программирования JavaScript, что облегчит начало создания проектов данного типа и уменьшит их сложность, положительно влияя на ускорение разработки серверного приложения.

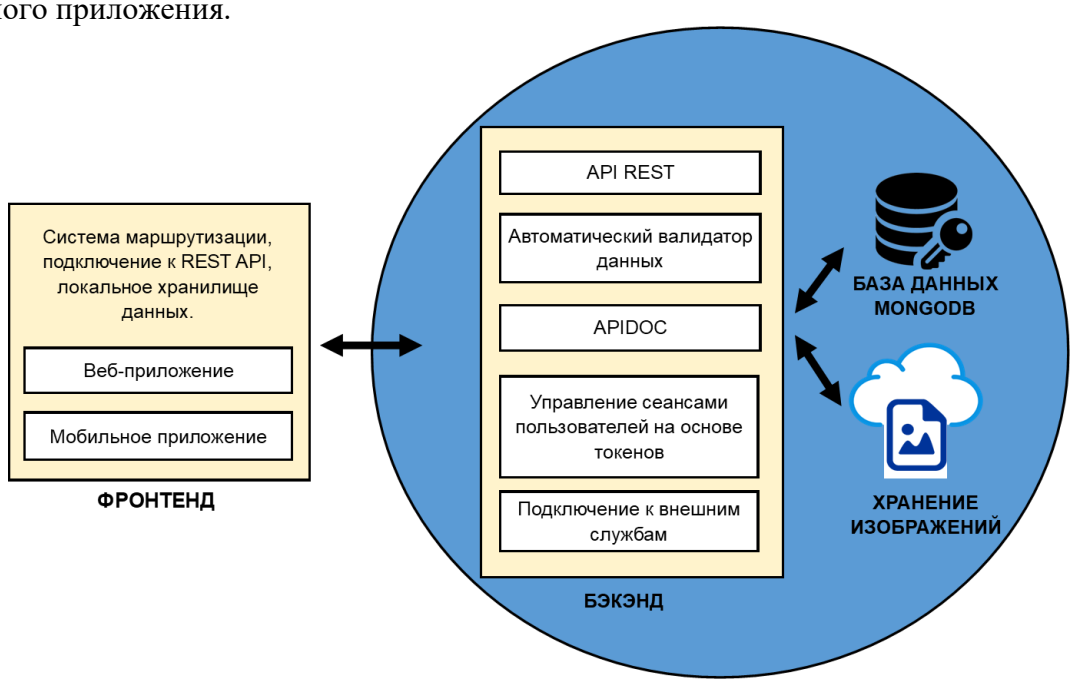


Рисунок 1 – Архитектура предлагаемого решения

Возможности демоверсии:

– Структура проекта: через командную консоль будет загружен файл JSON, который определит структуру проекта и инструменты, которые будут использоваться.

– Автоматическое создание APIDOC: создает все необходимое для автоматического создания документации API при создании API.

– Автоматический валидатор: он имеет ряд предопределенных функций, которые позволяют API проверять и преобразовывать данные, когда они поступают в промежуточное ПО или контроллер.

– CRUD: по умолчанию модели данных имеют функциональные возможности для выполнения операций создания, чтения или поиска, обновления и удаления.

– Аутентификация: можно управлять разрешениями для выполнения операций CRUD в API через пользователей.

– Хранение изображений в выделенной службе: для хранения изображений для этой функции используется выделенная внешняя служба. В API будут сохранены только URL-адреса, на которых изображения размещены во внешней службе (Cloudinary).

ЛИТЕРАТУРА

1. B. Paul, “Authentication and Authorization for the front-end web developer,” 2020.
2. S. Pérez, J. Quispe, F. Mullicundo, and D. Lamas, “Herramientas Y Tecnologías Para El Desarrollo Web Desde El Frontend Al Backend,” XXIII Work. Investig. en Ciencias la Comput., pp. 347–350, 2021, [Online]. Available: http://sedici.unlp.edu.ar/handle/10915/120476.

3. H. Jordan et al., “Rich Internet Applications w/HTML and Javascript,” Rich Internet Appl. w/HTML Javascript, p. 25, 2017.
4. W3C, “Web Services Glossary,” 2004. https://www.w3.org/TR/2004/NOTE-ws-gloss-20040211/.
5. Fielding Roy Thomas, “Architectural Styles and the Design of Network-based Software Architectures,” University of California, 2000.

УДК 004.4

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

РАСПРЕДЕЛЕНИЕ АППАРАТНЫХ РЕСУРСОВ НА ОСНОВЕ СИСТЕМЫ МАССОВОГО ОБСЛУЖИВАНИЯ С ПРИОРИТЕТНЫМИ ОЧЕРЕДЯМИ

Настоящая работа посвящена разработке и анализу подхода к построению инструмента на основе системы массового обслуживания (СМО) с ожиданием для координации использования аппаратных средств (кластеров) в рамках тестирования программного обеспечения на рабочем, действующем оборудовании. Дано краткое описание предметной области и проблемы, связанные с ней. Главный интерес в данной статье представляет алгоритм построения очереди в системе массового обслуживания, реализованный в рамках вышеуказанного инструмента.

Тестирование систем хранения данных (СХД) является трудоёмким и долгим процессом[1], поскольку архитектура СХД представляет собой сложный механизм. Для обеспечения максимального высокого уровня качества продукта необходимо производить тестирование, максимально приближенное к реальным условиям. Именно поэтому проведение набора тестов требуется осуществлять на рабочем, действующем оборудовании, то есть кластере с подключенными серверами нагрузки, развернутой инфраструктурой и большим количеством данных[2].

Для каждого набора тестов необходима определенная инфраструктура, которая должна работать на кластере, однако количество кластеров на команду инженеров по тестированию ограничено. Поэтому могут возникать ситуации, при которых инженер длительное время не может получить доступ к кластеру с нужным наполнением и вынужден ждать, пока он освободиться другим инженером. Чтобы качество продукта оставалось на высоком уровне, требуется, чтобы каждый инженер мог получить необходимую по параметрам систему в нужное время.

Для решения данной проблемы был разработан инструмент на основе системы массового обслуживания с приоритетными очередями, позволяющий прозрачно координировать использование кластеров внутри команд инженеров по тестированию. Данный инструмент позволяет регламентировать очередность использования аппаратных ресурсов, предоставляя инженерам возможность планировать свои рабочие задачи исходя из расписания использования каждого кластера.

Системы массового обслуживания с ожиданием являются наиболее распространенными и часто встречающимися на практике СМО[3], поэтому для разработки алгоритма, описанного в этой статье, был выбран именно этот тип как наиболее популярный.

Рассматриваемая система массового обслуживания является одноканальной с неограниченной очередью. Предполагается, что для системы такого типа в разработанном инструменте должны выполняться следующие условия [4]:

1. Появление каждой заявки не зависит от других заявок, то есть, является независимым событием;
2. Заявки поступают из неограниченного множества;
3. Каждая заявка в рамках одного приоритета обслуживается по принципу FIFO («first in – first out», то есть, «первый пришел – первый обслужен»);

4. Темп распределения заявок по очередям всегда выше темпа поступления заявок.

Основополагающей частью рассматриваемой СМО является алгоритм, регулирующий очереди на кластеры в зависимости от приоритетов заявок пользователей. На данный момент представлено три типа приоритетов заявок – P0, P1, P2 (от самого высокого приоритета к самому низкому соответственно). На Рисунке 1 представлен один из возможных вариантов поведения алгоритма. Пусть существуют три пользователя и три кластера, на которые эти пользователи отправляют свои заявки с определенным приоритетом. Пусть также в одной из очередей на кластер (Cluster 1 queue) уже имеется заявка с приоритетом P1. По рисунку видно, что первый пользователь отправляет заявку на первый кластер, второй – также на первый, а третий – на первый и третий. После применения алгоритма, регулирующего очереди на кластерах в соответствии с приоритетом заявок, ситуация A поменяется на ситуацию B.

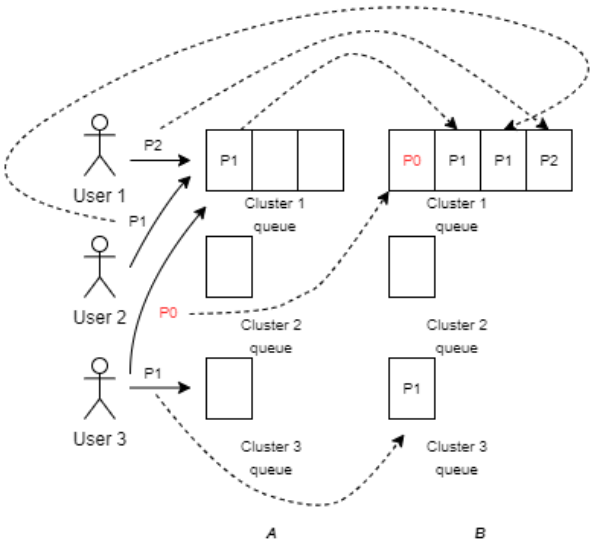


Рисунок 1 – Работа алгоритма регулирования очередей на кластере.

В систему, реализованную в рамках данной работы, поступает большое количество заявок в случайные моменты времени, длительность обслуживания также случайна и зависит от состояния системы в данный момент времени, количества заявок, загруженности процессов обработки заявок.

Для оптимизации процесса обслуживания необходимо обеспечить такое соотношение скорости поступления заявок и скорости их обслуживания, при котором затраты времени на обслуживание одной заявки минимальны [5]. Для достижения этого условия в разработанном алгоритме (схема 1) используется параллелизация процессов, распределяющих поступившие заявки по приоритетам, а также процессов, отвечающих за распределение обработанных заявок по очередям на кластеры. Вычисление очередности для поступления заявки на кластер основывается на приоритете заявки и времени поступления заявки в систему массового обслуживания. Чем приоритетнее заявка, тем быстрее она будет поставлена в очередь, и, соответственно, обслужена. Такая параллелизация позволила сократить время на обслуживания заявки в сравнении с поочередным обслуживанием заявок, при этом не усложняя саму архитектуру СМО.

ЛИТЕРАТУРА

1. Зарубин А. А. и др. Подходы к моделированию нагрузки на распределенную систему хранения данных //Системы синхронизации, формирования и обработки сигналов. – 2018. – Т. 9. – №. 3. – С. 90-95.
2. Karipzhanova A. Z. Testing of Data Storage System Using Multidimensional Parity Algorithms Resistant to Partial Loss of Storage Locations. – 2019.K. Elissa.
3. Юшина Т. С. Имитационное моделирование и управление сложым бизнес-процессом транспортной компании. Моделирующий алгоритм // Фундаментальные и прикладные научные исследования. 2019.

4. Осипов Г. С. Оптимизация одноканальных систем массового обслуживания с неограниченной очередью // Бюллетень науки и практики. 2016. №9 (10).M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.
5. Ageed Z. et al. Cloud computing resources impacts on heavy-load parallel processing approaches //IOSR Journal of Computer Engineering (IOSR-JCE). – 2020. – Т. 22. – №. 3. – С. 30-41.

УДК 004.415.2

Чучин Д.Ю. (4 курс бакалавриата),
Шмаков В.Э., к.т.н., доцент

РАЗРАБОТКА СЕРВЕРНОЙ ЧАСТИ ПРИЛОЖЕНИЯ ДЛЯ ПРЕДОСТАВЛЕНИЯ ИНТЕРАКТИВНОЙ СРЕДЫ КЛИЕНТАМ HTTP API

Цель данной работы – реализовать прототип серверной части приложения, которое будет предоставлять разработчикам HTTP API возможность интегрировать интерактивную среду (консоль) на их веб-ресурс с минимальными временными затратами.

Интерактивные среды, встроенные в документацию, помогают пользователю быстрее освоить продукт, так как можно сразу на практике испытать функциональность сервиса. Однако на данный момент не существует удобного решения по интеграции интерактивных сред в документацию HTTP API. Главная проблема заключается в том, что в существующих средах, нет функциональности, позволяющей обращаться к обработчикам API без личного API ключа, для получения которого нужно зарегистрироваться и возможно предоставить платежные данные.

Разрабатываемое приложение позволит разработчикам HTTP API встраивать в свою документацию интерактивную среду, при этом пользователь не должен будет регистрироваться в сервисе. Пользователи смогут получить доступ к программному продукту почти без потери его функциональности и опробовать его, чтобы понять подходит он или нет и принять правильное коммерческое решение. Таким образом, доступность сервиса для новых пользователей возрастет, и, как следствие, увеличится количество посетителей, которые потенциально станут клиентами программного продукта.

В рамках разрабатываемого прототипа интерактивные среды будут подобны среде IPython, поскольку для оценки возможностей API вполне хватает функциональности языка Python. Язык прост в освоении и знаком специалистам многих областей IT, что расширяет круг потенциальных пользователей.

Клиент-серверная архитектура разрабатываемого приложения предполагает два типа клиентов. Первый тип – Разработчик HTTP API, внедривший интерактивную среду на свой веб-ресурс, второй – Пользователь, посетитель веб-ресурса Разработчика, который необходимо испытать функциональность сервиса перед решением использовать его.

Предполагается, что на фронтенде разрабатываемого приложения (эта часть в рамках проекта не разрабатывается) будет расположена ознакомительная информация и документация приложения. Также здесь Разработчик сможет зарегистрировать свой сервис и создать для него интерактивную среду. При создании среды Разработчик указывает набор API ключей, которые будут применяться при исполнении команд Пользователя.

С фронтенда веб-ресурса Разработчика Пользователь будет отправлять команды разрабатываемому приложению.

Непосредственно реализованная в рамках работы часть (бэкенд) имеет микросервисную архитектуру (Рисунок 1) и состоит из следующих компонентов:

- Бизнес-логика для Разработчиков. Микросервис предоставляет API для фронтенда приложения, где Разработчик может зарегистрироваться, и создавать интерактивные среды.

- База данных. Этот компонент хранит данные о разработчиках, их сервисах, средах, которые они сконфигурировали, а также данные о том, как пользователи взаимодействуют с системой Разработчика.

- Бизнес-логика исполнения команд. Микросервис отвечает за обработку команд и передачу их на рабочие узлы.

- Менеджер сессий. Данный компонент осуществляет управление сессиями.

- База сессий. База данных, используемая менеджером сессий. Во время работы пользователя в интерактивной среде, необходимо хранить информацию о том какой рабочий узел какому пользователю соответствует. Данные о пользовательской сессии должны иметь очень высокую скорость чтения, иначе станет не комфортно работать в интерактивной среде из-за увеличения задержки при исполнении команды. Поэтому сессии будут храниться в базе данных ключ-значение.

- Рабочие узлы. На рабочих узах запускается Python-интерпретатор, который исполняет пользовательские команды и отдает результат в компонент бизнес-логики исполнения команд.

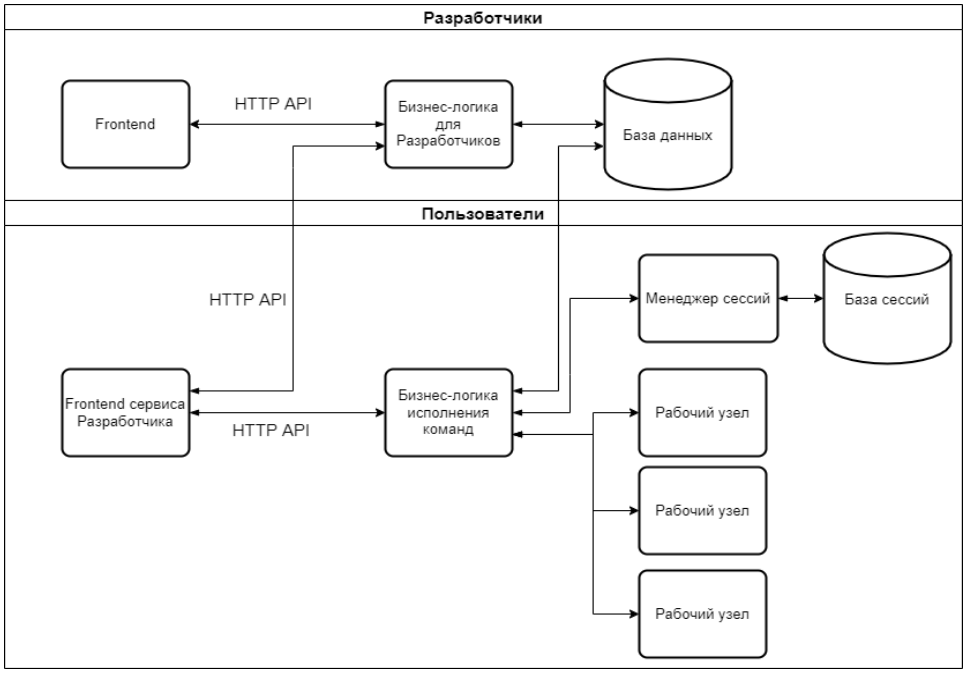


Рисунок 1 – Архитектура приложения.

Для разработки компонентов бизнес-логики и менеджера сессий использовался язык Python и фреймворк FastAPI. Микросервисы в основном отправляют данные другим микросервисам и ожидают ответа по сети, поэтому низкая производительность языка Python не будет сильно влиять на быстродействие сервиса. Кроме того, FastAPI позволяет создавать асинхронные веб-приложения, что повышает эффективность компонентов с учетом описанных особенностей.

Для хранения данных выбрана свободная объектно-реляционная СУБД PostgreSQL. Для данной СУБД существует асинхронный драйвер для языка Python – asyncpg. Без него невозможно асинхронно исполнять запросы к СУБД, поэтому наличие асинхронного драйвера является важным фактором при выборе СУБД с учетом того, что используется асинхронный фреймворк для компонентов бизнес-логики.

Данные о сессиях хранятся в резидентной СУБД Redis, работающей со структурами данных типа ключ-значение. Она обеспечивает высокую скорость чтения и отлично подходит для хранения пользовательских сессий.

Компоненты системы будут работать в Docker контейнерах, хорошо приспособленных для реализации микросервисной архитектуры и позволяющих легко масштабировать любой из компонентов.

Оркестрироваться контейнеры будут с помощью Kubernetes. На данный момент это одна из самых популярных платформ управления контейнеризованными рабочими нагрузками и сервисами, она облегчает как декларативную настройку, так и автоматизацию.

ЛИТЕРАТУРА

1. Fast API [Электронный ресурс] URL: https://fastapi.tiangolo.com/
2. What is Kubernetes? [Электронный ресурс] URL: https://kubernetes.io/docs/concepts/overview/what-is-kubernetes/

УДК 004.4

Шахмин Е.Н. (2 курс магистратуры),
Воинов Н.В., к.т.н., доцент

МЕТОДИКА НАСТРОЙКИ ПАРАМЕТРОВ КОНФИГУРАЦИИ APACHE SPARK ДЛЯ ПОВЫШЕНИЯ ПРОИЗВОДИТЕЛЬНОСТИ ETL-ПРОЦЕССОВ

В последнее время вычислительная система Apache Spark привлекла к себе большое внимание благодаря своей повышенной производительности при обработке больших объемов данных. Поскольку Spark обрабатывает данные в памяти, то для достижения оптимальной производительности крайне важно настроить его для выполнения конкретной рабочей нагрузки. В отличие от традиционных кластерных систем, например, MapReduce/Hadoop, выполняющих вычисления на дисках, Apache Spark имеет намного больше критических для производительности параметров конфигурации, и его производительность более чувствительна к размерам входных данных [1]. Поэтому нахождение оптимальных Spark параметров для заданной рабочей нагрузки является нетривиальной задачей. Предыдущие исследования по этой проблеме можно разделить на четыре категории: лучшие практики, подходы на основе анализа программ, подходы на основе поиска и подходы на основе обучения.

Применяя лучшие практики [2,3], разработчики способны настроить только наиболее очевидные параметры, связанные с производительностью, например, количество исполнителей или объем оперативной памяти на исполнителя, но из-за неоднородности рабочих нагрузок оптимальная настройка для одной может привести к непредсказуемым результатам для другой.

Метод конфигурации на основе анализа программ [4,5] фиксирует характеристики производительности, используя тонкий анализ состояния программы во время выполнения. Эти подходы опираются в основном на статистические заключения и предназначены для оптимизации узких мест производительности определенной рабочей нагрузки, которые зачастую не отражают общую производительность программы.

Подходы на основе поиска [6,7] рассматривают проблему конфигурации как задачу оптимизации «черного ящика» и используют алгоритмы поиска для ее решения. Для применения этих подходов нет необходимости в соответствующих знаний высокого уровня о внутреннем устройстве системы, но они требуют значительного времени для итеративного поиска в пространстве параметров.

Методы на основе обучения [1,8,9] пытаются построить модели прогнозирования производительности с использованием обучающих данных, а затем применяют некоторые алгоритмы для нахождения оптимальной конфигурации на основе этой модели. Результаты данных подходов зависят от точности модели прогнозирования, для обучения которой необходимо собрать значительное количество обучающих данных для каждой рассматриваемой рабочей нагрузки.

Для текущей работы наиболее актуальным подходом является метод на основе обучения, но в рассмотренных подходах строили модели для тривиальных рабочих нагрузок, например,

WordCount, PageRank, KMeans, а также не учитывали одновременную работу на кластере нескольких рабочих нагрузок, что зачастую встречается при работе с ETL-процессами. Поэтому целью данной работы является разработка подхода определения оптимальных конфигураций ETL-процессов в пространстве параметров высокой размерности фреймворка Apache Spark для уменьшения их времени работы.

Поставленную цель предполагается решать в несколько этапов. Во-первых, требуется отобрать из всего множества Spark параметров параметры, которые могут оказывать большое влияние на производительность ETL-процессов, каждый из которых может представлять собой отдельную рабочую нагрузку. Данный этап позволит уменьшить объем обучающих данных для построения точной модели прогнозирования. Для сбора этих данных, которые являются результатом второго этапа, необходимо учитывать возможность одновременной работы на кластере нескольких ETL-процессов. На третьем этапе для обучения модели прогнозирования планируется использовать нейронные сети или алгоритмы машинного обучения, выбор которого будет зависеть от объема обучающих данных. Имея модель прогнозирования можно будет использовать эвристический алгоритм поиска для нахождения оптимальных конфигураций рассматриваемых ETL-процессов.

Чтобы максимально приблизить обучающие данные к реальным рабочим нагрузкам, запуск ETL-процессов планируется производить на тестовом кластере Cloudera, где для одновременного запуска процессов будет использоваться планировщик рабочих процессов Apache Oozie. Сами ETL-процессы будут разработаны с помощью Spark Dataset API на языке Java 8. Для генерации данных, приближенных к реальным, с которыми работают ETL-процессы, будет использована готовая нейронная сеть, доступная в рамках компании. Для создания и обучения модели прогнозирования и эвристических алгоритмов будет применены специальные Python библиотеки для машинного обучения: skikit-learn и PyTorch.

ЛИТЕРАТУРА

1. Yu Zhibin, Bei Zhendong, Qian Xuehai. Datasize-Aware High Dimensional Configurations Auto-Tuning of In-Memory Cluster Computing // SIGPLAN Not. – 2018. – mar. – Vol. 53, no. 2. – P. 564–577.
2. Gupta Preeti, Sharma Arun, Jindal Rajni. An Approachfor Optimizing the Performance for Apache Spark Applications // 2018 4th International Conference on Computing Communication and Automation (ICCCA). – 2018. – P. 1–4.
3. Research on Parallel Task Optimization of High Performance Computing Cluster / Jiandong Shang, Dongpu Sheng, Runjie Liu et al. // 2020 IEEE International Conference on Artificial Intelligence and Information Systems (ICAIIS). – 2020. – P. 777–780.
4. Zhao Yao, Hu Fei, Chen Haopeng. An adaptive tuning strategy on spark based on in-memory computation characteristics // 2016 18th International Conference on Advanced Communication Technology (ICACT). – 2016. – P. 484–488.
5. Chiba Tatsuhiro, Onodera Tamiya. Workload characterization and optimization of TPC-H queries on Apache Spark // 2016 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS). – 2016. – P. 112–121.
6. Bestconfig: tapping the performance potential of systems via automatic configuration tuning / Yuqing Zhu, Jianxun Liu, Mengying Guo et al. // Proceedings of the 2017 Symposium on Cloud Computing. – 2017. – P. 338–350.
7. Using bad learners to find good configurations / Vivek Nair, Tim Menzies, Norbert Siegmund, Sven Apel // Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering. – 2017. – P. 257–267.
8. AutoConfig: Automatic Configuration Tuning for Distributed Message Systems / Liang Bao, Xin Liu, Ziheng Xu, Baoyin Fang // 2018 33rd IEEE/ACM International Conference on Automated Software Engineering (ASE). – 2018. – P. 29–40.
9. ATCS: Auto-Tuning Configurations of Big Data Frameworks Based on Generative Adversarial Nets / Mingyu Li, Zhiqiang Liu, Xuanhua Shi, Hai Jin // IEEE Access. – 2020. – Vol. 8. – P. 50485–50496.

УДК 004.42

Шевелев Р.Р. (4 курс бакалавриата),
Самочадина Т.Н., старший преподаватель,
Самочадин А.В., к.т.н., доцент

СРЕДСТВА ГЕНЕРАЦИИ ТЕСТОВЫХ СЦЕНАРИЕВ НА ОСНОВЕ ВАРИАНТОВ ИСПОЛЬЗОВАНИЯ

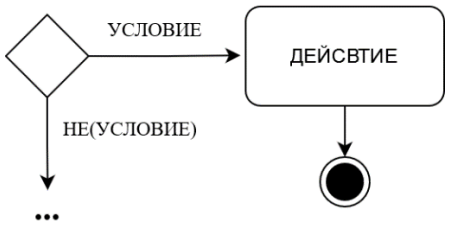
Целью работы является реализация программных средств, которые на основе вариантов использования будут генерировать тестовые сценарии. Сценарии вариантов использования являются одним из распространенных способов формирования требований к программным продуктам. Даже для небольших проектов их количество составляет несколько десятков, а их структура может быть сложной и включать в себя основной и несколько альтернативных сценариев. Задача построения тестовых сценариев на основе вариантов использования вручную в таких условиях является трудоемкой [1, 2, 3]. Таким образом, мотивацией разработки средств генерации тестовых сценариев является снижение трудозатрат на проектирование тестов, что в свою очередь позволит значительно сократить расходы компаний-разработчиков, сэкономить время и ресурсы, снизить риск выпуска на рынок некачественного продукта.

Предлагаемое решение реализуется в несколько этапов.

Входные данные представляются в виде UML диаграммы вариантов использования. При этом для каждого варианта использования предоставлен детальный сценарий. Сценарии вариантов использования представлены на структурированном естественном языке в соответствии с шаблоном А. Коберна [4]. Такой вид входных данных обусловлен тем, что этот шаблон достаточно широко используются для документирования функциональных требований к программным системам.

На основе входных данных для каждого сценария строится модель поведения, представленная в виде UML диаграммы деятельности. Такой вид поведенческой модели обуславливается возможностью четко выразить взаимодействия между действующими лицами и системой, различать виды действий (внешние или системные), а также выделять различные потоки выполнения сценария [5]. Построение модели производится с помощью набора алгоритмов поиска языковых шаблонов в тексте и трансляции этих шаблонов в элементы диаграммы [6]. Пример преобразования представлен в Таблице 1.

Таблица 1 – Пример соответствия шаблона и элементов диаграммы

Шаблон	Элементы диаграммы
Если [УСЛОВИЕ], то [ДЕЙСТВИЕ] и сценарий завершается	

Затем находятся все пути в диаграмме деятельности с помощью классических критериев покрытия графов (покрытие вершин, покрытие ребер, полное покрытие путей) [7]. Каждый путь в диаграмме деятельности является тестовым сценарием.

Таким образом, на выходе получается набор тестовых сценариев, представленных в виде последовательности действий в системе и ожидаемого поведения.

На Рисунке 1 представлена схема предлагаемого решения.

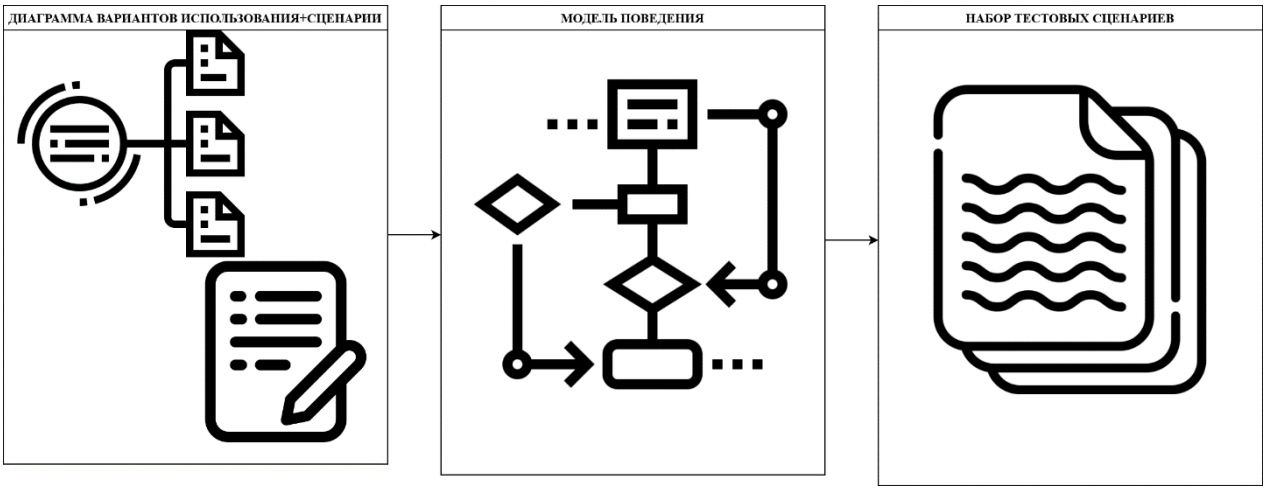


Рисунок 1 – Схема предлагаемого решения

Особенности реализации средств генерации:

Средства реализованы на языке программирования Java с использованием встроенных библиотек.

В качестве формата входных и выходных данных используется XML. Для работы с этой формой представления данных используется DOM XML Parser, входящий в состав стандартных библиотек Java. В качестве схемы XML используется стандарт XMI, получивший широкое распространение как формат обмена UML моделями.

ЛИТЕРАТУРА

1. Портал специалистов по тестированию и обеспечению качества ПО. — Режим доступа: https://software-testing.ru/
2. Noraida Ismail, Rosziati Ibrahim, Noraini Ibrahim. Automatic generation of test cases from Use-case diagram // Proceedings of the international conference on electrical engineering and informatics (Institut Teknologi Bandung, Indonesia June 17-19, 2007) — Bandung, 2007 — C. 699-702
3. Arjinder Singh, Er. Sumit Sharma. Functional test cases generation based on automated generated use case diagram // International journal of innovative research in advanced engineering. — 2015. — Issue 8, volume 2. C. 105–110
4. А. Коберн. Современные методы описания функциональных требований к системам — М.: «Лори», 2002 — 263 с.
5. Gutiérrez J.J., Escalona M.J., Mejías M. Derivation of test objectives automatically // Fifteenth International Conference On Information Systems Development (Budapest, Hungary, 31 August – 2 September, 2006) — Seville, 2006. — 13 с.
6. Б. Смит. Методы и алгоритмы вычислений на строках — М.: «Вильямс», 2006. — 496 с.
7. В.П.Котляров, Т.В.Коликова. Основы современного тестирования программного обеспечения, разработанного на C#. Учебное пособие. — Санкт-Петербург, 2004 — 170 с

Секция «Программная инженерия: методы и алгоритмы теории программирования»

УДК 621.319

Алейников П. И., Погребной А. С. (2 курс магистратуры),
Сараджишвили С. Э., к.т.н., доцент

ИССЛЕДОВАНИЕ ПОДХОДОВ СЕГМЕНТАЦИИ ОБЛАКОВ НА ИЗОБРАЖЕНИЯХ ПОЛНОГО НЕБА

Целью работы является исследование существующих алгоритмов сегментации облаков на изображениях, получаемых с камеры полного неба. Задача сегментации облаков актуальная при разработке систем краткосрочного прогнозирования глобальной горизонтальной освещённости. При построении таких систем на основе подхода с явным отслеживанием перемещения облачного покрова точность сегментации оказывает значительное влияние на достоверность прогноза.

В работе решаются следующие задачи: изучить принцип работы существующих алгоритмов, для каждого выявить проблемные облачные сцены, оценить сложность реализации, реализовать и сравнить некоторые из них на отобранных изображениях с различными типами неба.

Существующие алгоритмы сегментации облаков можно разбить на следующие классы: алгоритмы с пороговой сегментацией, алгоритмы, основанные на морфологических методах, алгоритмы с использованием методов глубоко машинного обучения. Однако распространенными в контексте систем прогнозирования являются алгоритмы пороговой сегментации поэтому в этой работе мы концентрируемся на первый класс методов.

Самыми простыми в реализации являются алгоритмы сегментации с фиксированным порогом. Такие алгоритмы работают с представлением изображения основанном на соотношении цветовых каналов RGB, например, красного и синего - Read Blue Ration (RBR) [1]. Работая с RBR, мы убедились, что для различных видов облачной картины требуются свой порог сегментации. Так же была выявлено, что невозможно подобрать оптимальный порог для некоторых смешанных облачных сцен и проблема сегментации высококучевых, пресытых облаков в околосолнечной зоне.

Более точным алгоритмом, использующим нормализованное соотношение синего и красного каналов, является Hybrid Thresholding Algorithm (HYTA) [2]. Этот алгоритм

разбивает исхожены изображения на два типа: одномодальный (изображение чистого неба или пасмурного) и бимодальные (небо частично покрыто облаками). Для одномодальных изображений используется фиксированный порог, эмпирически подобранный при анализе вручную сегментированных изображений. Для бимодальных изображений используется адаптивный порог. Порог выбирается, сводя к минимуму перекрестную энтропию между исходным изображением и его сегментированным вариантом. При тестировании алгоритма мы стабильно наблюдали ошибки в околосолнечной области. Так же не все изображения чистого неба хорошо обрабатываются фиксированным порогом. Но точность результата однозначно превосходит RBR с фиксированным порогом.

В работе [3] описана улучшенная версия HYTA алгоритма задача которого побороть проблемы ошибочной сегментации в околосолнечной области. Этот алгоритм дополнительно различает изображения на изображена с солнцем и без. Далее для изображений с солнцем применяется дополнительный адаптивный порог в околосолнечной области. Авторы демонстрируют впечатляющие результаты, однако алгоритм достаточно сложен в реализации и провести эксперименты с ним нам не удалось.

Интересным оказалась сегментация представления изображения, получаемая суммированием соотношений синего к красному и синего к зелёному каналов (BRBG) [4]. На отобранных нами изображениях удалось достаточно точно сегментировать облака используя фиксированных порог. Но алгоритм также испытывает проблемы в околосолнечной области ошибочно классифицируя, пиксели чистого неба как облачные.

Так же был рассмотрен подход, основанный на библиотеке чистого неба. Его идея состоит в формировании исторической многомерной базы данных изображений чистого неба с координатами солнца и другими параметрами, уточная погодные условии, туманность и так далее. При сегментации изображения из базы выбирается ближайшее схожее изображение чистого неба и применяется для коррекции анализируемого представлений оригинального изображения. Мы реализовали простую вариацию [5] этого метода и столкнулись с проблемой снижения качества сегментации при несовпадении координат солнца. Для достижения хорошего результата необходим анализ околосолнечной области и типа облачной ситуации оригинального изображения, что наряду с необходимостью ведения базы данных значительно усложняет алгоритм.

По результатам проделанной работы наиболее позитивный результат был получен при сегментации BRBG представление и работе с алгоритмом HYTA. Несмотря на то, что эти методы испытываются сложности в околосолнечной области, точность сегментации изображений со смешанной облачностью впечатляет. Учитывая то, что сегментация на основе BRBG даёт хорошие результаты при фиксированном пороге, а сегментация в околосолнечной области однозначно требует улучшений и адаптивной коррекции BRBG, мы приступили к созданию собственного алгоритма сегментации облаков на основе BRBG представления.

ЛИТЕРАТУРА

1. Hasenbalg M. и др. Benchmarking of six cloud segmentation algorithms for ground-based all-sky imagers // Sol. Energy. 2020. Т. 201.
2. Li Q., Lu W., Yang J. A hybrid thresholding algorithm for cloud detection on ground-based color images // J. Atmos. Ocean. Technol. 2011. Т. 28. № 10.
3. Li X. и др. A Cloud Detection Algorithm with Reduction of Sunlight Interference in Ground-Based Sky Images // Atmos. 2019, Vol. 10, Page 640. 2019. Т. 10. № 11. С. 640.
4. Tang J. и др. An improved cloud recognition and classification method for photovoltaic power prediction based on total-sky-images // J. Eng. 2019. Т. 2019. № 18.
5. Magnone L. и др. Cloud Motion Identification Algorithms Based on All-Sky Images to Support Solar Irradiance Forecast // IEEE PVSC-44, Washington. 2018.

УДК 004.051

Алтынова А.Ю. (4 курс бакалавриата),
Григорьев Д.А., к.ф.-м.н., доцент,
Колычева В.А., к.э.н., доцент,
Семенов А.В., доцент

МЕТОДИКА АВТОМАТИЧЕСКОЙ ГЕНЕРАЦИИ НАЗВАНИЙ КАРТИН НА ОСНОВЕ ГЛУБОКОГО ОБУЧЕНИЯ

В настоящее время ощущается дефицит работ по применению машинного обучения для изучения неоднозначных аспектов искусствоведения среди участников арт-рынка, не имеющих соответствующего образования и компетенций. Прежде всего, трудность вызывают субъективные названия произведений искусства, а также разъясняющие (на практике же нередко усложняющие понимание и восприятие) подписи самих авторов к ним. Таким образом, задача машинного накопления знаний в области искусства тесно связана с задачей открытия доступа к арт-рынку экономистам, инвесторам, бизнес-аналитикам, ученым-исследователям.

Целью работы является разработка программного обеспечения, способного по изображению картины сгенерировать для неё подходящее название, максимально отражающее ее содержание и смысловую ценность.

Для достижения поставленной задачи и оценки полученных результатов были выбраны следующие методы: датасет WikiArt, нейронная сеть с архитектурой Transformer, метрики BLEU, CIDEr, ROUGE-L и METEOR.

WikiArt датасет состоит из более чем 80.000 изображений изобразительного искусства. датированных от 15 века до настоящего времени, содержит 27 стилей и 45 различных жанров. Он составлен на основе содержания сайта wikiart.

Для генерации названий была использована модель, построенная на основе модели трансформера [1]. Нейросети с таким строением успешно используются для генерации подписей к фотографиям с момента публикации работы, однако для получения подписей к картинам ранее не использовались.

Модель состоит из двух блоков: шифровщика и дешифровщика(encoder-decoder), которые работают и взаимодействуют друг с другом на механизме внимания Предшествующие архитектуры не опирались на этот механизм и использовали рекуррентные нейронные сети, обладающие рядом недостатков, таких как “забывчивость” и невозможность генерировать слова в предложении параллельно. Эксперименты, проведенные исследователями на многих датасетах, подтверждают приемлемое качество модели трансформера, то есть примерное соответствие подписи изображению.

Качество полученных подписей будем измерять с помощью метрики BLEU [2], ROUGE-L [3], METEOR [4] и CIDEr[5], используемых как в задаче генерации подписей, так и, например, в оценке качества машинного перевода.

Результаты автоматической оценки представлены в таблице:

Bleu1	5.2
Bleu-2	1.5
Bleu-3	0.4
Bleu-4	0.0000198
Rouge-L	4.2
Cider	1.9
Meteor	2.6

Было отмечено, что в данной задаче стандартные метрики и особенно метрика BLEU не всегда точно отражают качество полученного названия. Чтобы понять, вызван такой разрыв

особенностью метрик, датасета или же низким качеством подписей, необходима экспертная оценка качества сгенерированных названий и последующее ее сравнение с машинной оценкой.

Примеры:

	
Оригинал: the beach at trouville at low tide Предсказание: the fields of the thames	Оригинал: bridge astrakhan Предсказание: sad woman harbor the temple

ЛИТЕРАТУРА

1. Vaswani A. et al. Attention is all you need //Advances in neural information processing systems. – 2017. – Т. 30.
2. Papineni K. et al. Bleu: a method for automatic evaluation of machine translation //Proceedings of the 40th annual meeting of the Association for Computational Linguistics. – 2002. – С. 311-318.
3. Lin C. Y. Rouge: A package for automatic evaluation of summaries //Text summarization branches out. – 2004. – С. 74-81.
4. Banerjee S., Lavie A. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments //Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization. – 2005. – С. 65-72.
5. Vedantam R., Lawrence Zitnick C., Parikh D. Cider: Consensus-based image description evaluation //Proceedings of the IEEE conference on computer vision and pattern recognition. – 2015. – С. 4566-4575.

УДК 004.023

Василевский А.Н. (4 курс бакалавриата),
Черноруцкий И.Г., д.т.н., профессор

ИССЛЕДОВАНИЕ АЛГОРИТМА КОЛЛЕКТИВНОГО ИММУНИТЕТА ОТ КОРОНАВИРУСА ДЛЯ РЕШЕНИЯ ЗАДАЧ ОПТИМИЗАЦИИ

Целью работы является исследование алгоритма оптимизации, основанный на природе человека, который называется оптимизатором коллективного иммунитета от коронавируса.

Алгоритм имитирует стратегию коллективного иммунитета, а также концепции социального дистанцирования как способов борьбы с пандемией коронавируса. Данные концепции смоделированы в терминах оптимизации. Для коллективного иммунитета используются три типа индивидуальных случаев: восприимчивые, инфицированные и иммунизированные. Это необходимо для определения того, как вновь созданное решение обновляется с помощью стратегий социального дистанцирования.

В процессе работы алгоритма происходит следующее:

– сначала осуществляется генерация популяции случайным образом, некоторые особи помечаются как восприимчивые, некоторая немногая часть как инфицированные;

– в соответствии с базовым коэффициентом воспроизводства коллективный иммунитет формируется с использованием трех типов индивидуальных случаев, описанных выше;

– далее происходит преобразование членов популяции в соответствии с порогом коллективного иммунитета;

– алгоритм закончит работу, когда популяция достигнет состояния коллективного иммунитета.

В качестве средств для реализации алгоритма и его последующего тестирования был выбран язык программирования MATLAB и среда разработки MATLAB R2021b. Алгоритм был реализован в соответствии со следующей блок-схемой:

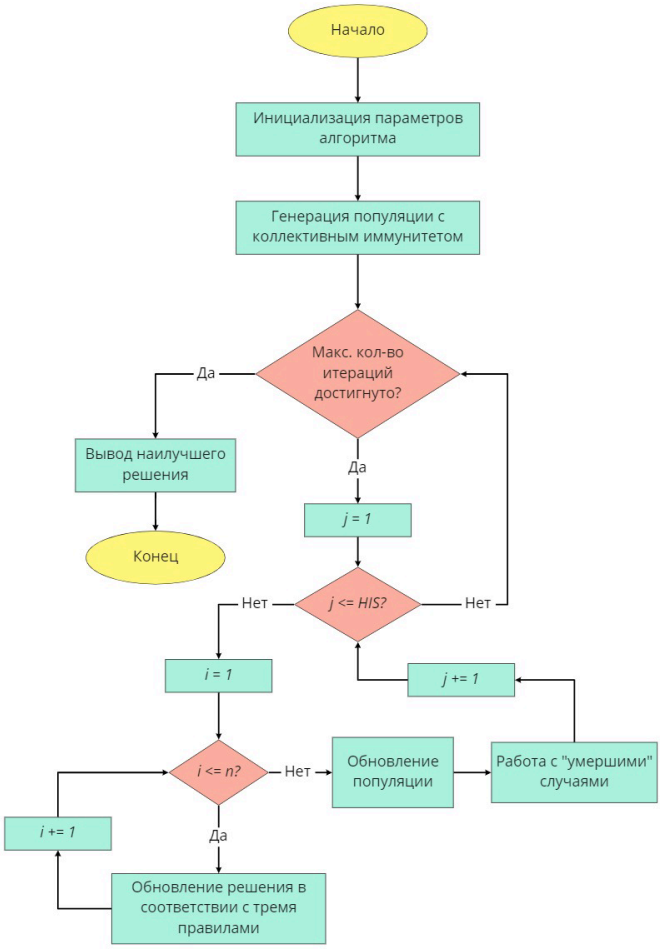


Рисунок 1 – блок-схема основной процедуры алгоритма коллективного иммунитета от коронавируса

В качестве проверки работоспособности алгоритма коллективного иммунитета от коронавируса было выбрано несколько тестовых функций разной модальности для поиска глобального минимума, а также была произведена оценка производительности алгоритма путем выбора различных сценариев работы для исследования чувствительности параметров.

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 и получение новой версии уже без параметров.

ЛИТЕРАТУРА

1. Mohammed Azmi Al-Betar, Zaid Abdi Alkareem Alyasseri, Mohammed A. Awadallah, Iyad Abu Doush // Coronavirus herd immunity optimizer (CHIO) // 2020 // URL: https://link.springer.com/content/pdf/10.1007/s00521-020-05296-6.pdf // Дата обращения: 25.11.2021
2. И. Г. Черноруцкий, Н. В. Воинов, Л. П. Котлярова // Методы оптимизации. Учебное пособие // Санкт-Петербург 2020 // Дата обращения: 29.11.2021

УДК 004.9

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

СИСТЕМА ОБРАБОТКИ РЕЗУЛЬТАТОВ АВТОМАТИЗИРОВАННЫХ ТЕСТОВ НА ОСНОВЕ МЕТОДА ПЕРВИЧНОГО АНАЛИЗА

Целью работы является разработка метода первичного анализа и реализация его принципов в специализированном программном продукте.

Метод первичного анализа основан на базовой классификации проблем. Вместо того, чтобы до конца разбираться в причинах неуспешного выполнения автоматизированных тестов (т.е. сразу отвечать на вопрос «почему это произошло?»), первым шагом будет ответ на вопрос «что произошло?». Просматривая и объединяя каждое падение подобным образом, тестировщик или разработчик, выполняющий анализ, получает полную картину результатов выполнения автоматизированных тестов и может распределять дальнейшее исследование причин падений или действия по их исправлению между участниками команды. Тем самым, метод первичного анализа больше подходит для работы в больших командах, так как гарантирует отсутствие двойной работы (если, например, два инженера одновременно смотрят падения с одинаковым проявлением), а также позволяет распараллелить активности по поиску причины падения между разными командами – если проявление техническое, такое как недоступный сервер, то подобный анализ уходит в команду инфраструктуры, а если проблема похожа на изменение бизнес-логики приложения – в команду обеспечения качества программного продукта. Дополнительно, метод первичного анализа подходит для команд, активно нанимающих новичков – более опытный коллега подсвечивает, куда и что смотреть, а менее опытные отталкиваясь от этого могут завершить анализ, тем самым сокращается время, тратящееся на изучение особенностей реализации или обнаружение известных проблем.

Программный продукт, основанный на методе первичного анализа, представляет собой веб-приложение, написанное с использованием фреймворка ASP.NET MVC [1] на языке C#. Отладка приложения производилась с помощью IIS Express [2] и браузера Google Chrome. База данных приложения управляется Microsoft SQL Server. Связь между программным кодом и базой данных производится при помощи Entity Framework [3] и LINQ [4]. Для разработки фронтенда приложения был использован фреймворк Bootstrap [5]. Импорт и экспорт данных для анализа происходит в .csv формате, выбор подобного формата обоснован универсальностью и поддержкой во многих языках программирования.

На Рисунке 1 показана архитектура приложения, реализующая популярный шаблон проектирования MVC [6] (Model-View-Controller, Модель-Представление-Контроллер).

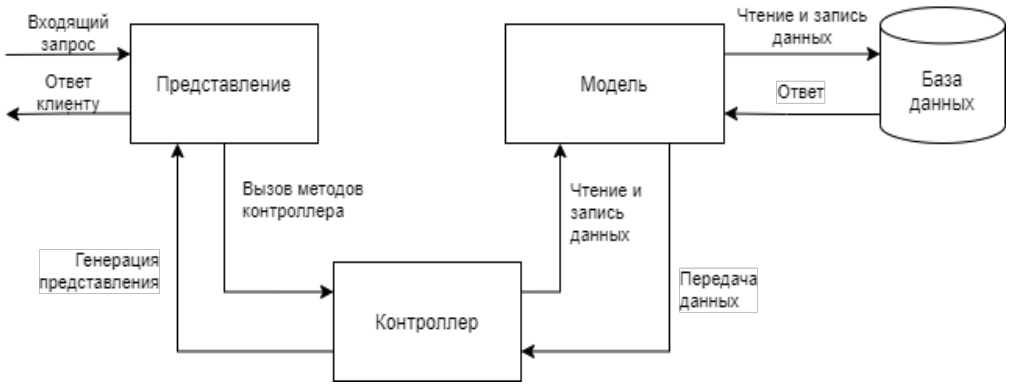


Рисунок 1 – Архитектура разрабатываемого приложения

Архитектура MVC позволяет работать с визуальным представлением и бизнес-логикой приложения отдельно. Модель предоставляет данные и реагирует на команды контроллера,

изменяя своё состояние; Представление отвечает за отображение данных модели пользователю, реагируя на изменения модели; Контроллер интерпретирует действия пользователя, оповещая модель о необходимости изменений.

На Рисунке 2 показана диаграмма прецедентов, отображающая взаимодействие пользователей с системой.

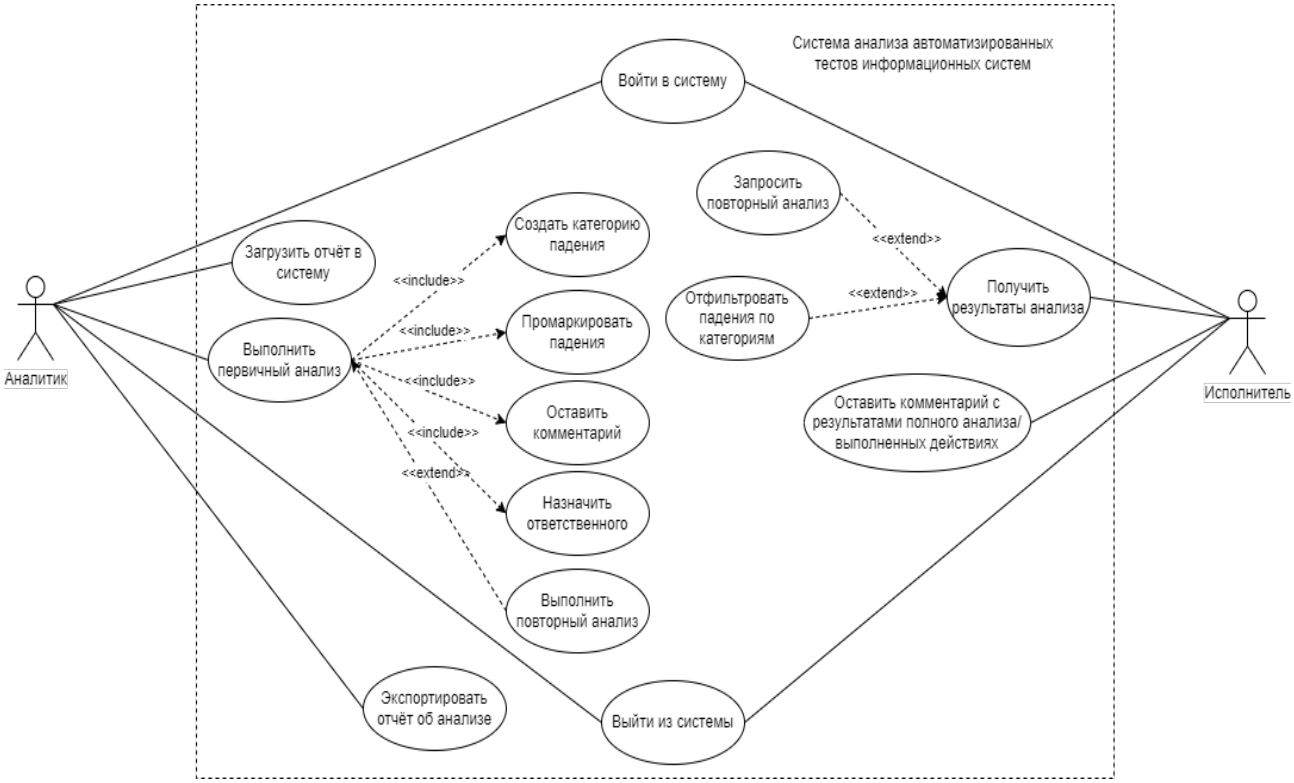


Рисунок 2 – Диаграмма прецедентов

Как видно из диаграммы, главные особенности метода первичного анализа были реализованы, а именно: группировка и просмотра кластеров при помощи маркеров; комментирование по результатам первичного анализа; перенаправление анализа на ответственного пользователя.

Активное использование разработанного приложения позволит перейти на следующую фазу реализации метода первичного анализа – автоматическую кластеризацию с помощью алгоритмов машинного обучения, однако для этого должна быть собрана обширная база реальных прецедентов, сгруппированных по признакам падения.

ЛИТЕРАТУРА

1. ASP.NET MVC Pattern // Microsoft [Электронный ресурс]. Режим доступа: https://dotnet.microsoft.com/en-us/apps/aspnet/mvc
2. Поддержка служб IIS во время разработки в Visual Studio для ASP.NET Core // Microsoft [Электронный ресурс]. Режим доступа: https://docs.microsoft.com/ru-ru/aspnet/core/host-and-deploy/iis/development-time-iis-support?view=aspnetcore-6.0
3. Документация по Entity Framework // Microsoft [Электронный ресурс]. Режим доступа: https://docs.microsoft.com/ru-ru/ef/
4. Общие сведения о LINQ // Microsoft [Электронный ресурс]. Режим доступа: https://docs.microsoft.com/ru-ru/dotnet/standard/linq/
5. Документация Bootstrap – Настройка // Bootstrap [Электронный ресурс]. Режим доступа: https://bootstrap-4.ru/docs/5.1/customize/overview/
6. Sarcar V. Design Patterns in C# / V. Sarcar. — Berkeley: Apress, 2020. — С. 495-519.

УДК 004.652

Зайцева Е.А. (4 курс бакалавриата),
Котлярова Л. П., старший преподаватель

ПОСТРОЕНИЕ ОПТИМАЛЬНОЙ СТРУКТУРЫ ДОКУМЕНТНОЙ БАЗЫ ДАННЫХ ПО МЕТАДАННЫМ РЕЛЯЦИОННОЙ БАЗЫ ДАННЫХ

На данный момент самым популярным способом хранения данных является использование реляционных баз данных [1]. Однако у реляционных баз данных есть ряд недостатков, например, проблема горизонтального масштабирования базы данных, жесткая структура таблиц. Этих недостатков лишены так называемые NoSQL [3] базы данных.

Реляционные базы данных удерживают устойчивое положение по популярности на протяжении многих десятилетий. В связи с этим во многих организациях на момент появления первых NoSQL баз данных уже существовали системы, основанные на реляционных базах данных. Поэтому одной из актуальных проблем при переходе к NoSQL базам данных является трансляция данных из формата реляционной базы данных в формат NoSQL базы данных.

В свою очередь, ключевой проблемой при переводе данных из реляционной базы данных в NoSQL является различие между методами хранения данных. Например, в документных базах данных [4] данные хранятся в коллекциях документов с произвольной структурой и отсутствием хороших механизмов для организации связи между коллекциями, а в реляционных БД данные хранятся в виде таблиц с возможностью извлечения связанных данных из нескольких таблиц.

Производительность и скорость обработки запросов в NoSQL базах данных очень зависит от способа организации данных. Чем эффективнее данные распределены по коллекциям (документные базы данных), тем быстрее система обработает запрос и найдет нужные данные.

Наиболее распространенными способами являются:

- «прямая» трансляция: каждой таблице реляционной базы данных ставится в соответствие отельная коллекция документов в документной базе данных;
- «обобщенная» трансляция: из всех таблиц реляционной базы данных создается одна коллекция документов в документной базе данных;

Данные способы не используют метаданные реляционной базы данных для построения новой структуры.

Однако, при переводе данных из реляционной базы данных в NoSQL, важно правильно сгруппировать данные в коллекции так, чтобы при выполнении запроса избежать операций объединения между коллекциями. При этом чтобы не потерять полноту информации об объектах необходимо учитывать отношения между исходными таблицами, а также включать поля, к которым обращается запрос, в одну коллекцию.

Этим требованиям отвечает метод выбора числа и состава коллекций для базы данных типа ключ-документ, основанный на теории множеств. Метод представлен в статье [2] в виде теоретического описания, однако его программная реализация авторами не предоставлена.

В рамках данной работы предполагается создание программного модуля на языке Java, который на основе изложенного метода позволит создавать новую структуру коллекций в документной базе данных MongoDB [6]. Для построения новой структуры модулю необходимо подключиться к произвольной базе данных PostgreSQL [5] и считать её метаданные, такие как параметры таблиц, полей, ключей. Объединение полей таблиц в множества по запросам является одним из ключевых шагов алгоритма - поэтому на вход программе также необходимо передать список запросов к выбранной базе данных.

В рамках исследования предлагается создать базу данных в PostgreSQL, заполнить её (больше 10 000 записей в каждой из таблиц), с помощью разработанной программы получить оптимальную структуру коллекций в MongoDB и перенести данные из PostgreSQL базы данных в новую MongoDB базу данных. Сравнив скорость запросов к PostgreSQL базе данных

со скоростью запросов к MongoDB базе данных, можно сделать выводы об эффективности полученного из программы способа структурирования коллекций данных.

ЛИТЕРАТУРА

1. Кириллов, В.В. Введение в реляционные базы данных / В.В. Кириллов, Г.Ю. Громов [Электронный ресурс]. URL: https://picloud.pw/media/resources/posts/2018/02/20/Кириллов_В.В_Громов_Г.Ю._-_Введение_в_реляционные_базы_данных_-_2008.pdf (дата обращения 20.03.2022)
2. Muon Ha and Yulia Shichkina, Creating collections without embedded documents for document databases taking into account the queries // Computation 2020, 8(2), 45. – URL: https://www.mdpi.com/2079-3197/8/2/45 (дата обращения: 22.01.2022).
3. NoSQL [Электронный ресурс]. URL: https://en.wikipedia.org/wiki/NoSQL
4. Документная база данных [Электронный ресурс]. URL: https://aws.amazon.com/ru/nosql/document/ (дата обращения: 20.03.2022).
5. PostgreSQL [Электронный ресурс]. URL: https://www.postgresql.org/ (дата обращения: 22.01.2022)
6. MongoDB [Электронный ресурс]. URL: https://www.mongodb.com/ (дата обращения: 22.01.2022).

УДК 004.42

Иванов Д. А. (2 курс магистратуры),
Молодяков С.А., д.т.н., доцент

РАЗРАБОТКА СИСТЕМЫ АВТОМАТИЧЕСКОЙ ТРАНСКРИПЦИИ МУЗЫКАЛЬНЫХ ФРАГМЕНТОВ

Возможность транскрипции музыкальных фрагментов в нотное представление – это выдающийся пример человеческих способностей, который включает процесс восприятия музыки (анализ сложных звуковых композиций), формирование умозаключений и выводов и их анализ, а также реализацию знаний теории музыки за счёт создания композиции музыкальных структур. Автоматическая транскрипция музыки, подразумевающая под собой вычислительные алгоритмы для преобразования аудио-сигналов в какую-либо из форм музыкального представления, представляется комплексной задачей цифровой обработки сигналов и машинного обучения. Данный процесс включает в себя такие подзадачи как определение высоты звука/звуков в определённый момент времени, определение начала и конца звучания, отслеживание ритма, определение музыкального инструмента, интерпретацию динамики звука, размера и даже тональности [1]. Учитывая такое количество трудоёмких подзадач и широкий спектр применимости, автоматическая транскрипция музыки воспринимается как фундаментальная задача в рамках процесса воссоздания и воспроизведения музыкальной информации. Из-за самой природы звука, которая в большинстве случаев подразумевает несколько источников (например, множество музыкальных инструментов или многоголосие), которые часто коррелируют между собой по частоте в каждый момент времени, автоматическая транскрипция музыки всё ещё считается нерешённой задачей [2].

Цели и задачи. Целью данной работы является реализация системы автоматической транскрипции музыки. Данная цель достигается за счёт решения следующих задач:

1. Определение начала и конца звучания звукового сигнала в определённый момент времени (onset/offset detection)
2. Определение высотности звуков, одновременно звучащих в один момент времени (multi-pitch detection)
3. Трансформация полученных результатов в нотное представление
4. Оценка эффективности разработанной системы и определение последующих шагов и мер для улучшения качества её работы

Отслеживание начала звучания сводится к определению дискретных отсчётов цифрового представления, в котором аудио-сигнал берет своё начало. Определение высотности звуков заключается в оценке предполагаемых частот звуков, звучащих в определённый момент времени одновременно. Так, полученные данные после отслеживания начала и конца звучания могут послужить основой для воссоздания ритма музыкального фрагмента, а извлечённые показатели высоты звуков в каждый момент времени соответственно будут использованы в процессе отслеживания мелодии и гармонии. На Рисунке 1 изображена основная последовательность действий в рамках каждой из двух подзадач. Также из рисунка видно, что данные подзадачи играют фундаментальную роль в контексте других комплексных прикладных задач из области обработки музыкальных фрагментов.

Обе задачи имеют прямое отношение к временно-частотному анализу. Большинство существующих методов временно-частотного анализа имеют ряд ограничений, заключающихся только в линейном масштабе представления (присуще методу временно-частотного распределения А. Коуэна), либо только в логарифмическом (характерно для Constant-Q анализа) [3], что не соответствует специфике устройства человеческого уха. В связи с этим был разработан гибкий алгоритм, формирующий спектрограммы на более широком и адаптивном временно-частотном представлении и предусматривающий зависимость оконной функции от анализируемой частоты [4]. Данный алгоритм реализован с использованием набора БИХ-фильтров, является вычислительно эффективным и служит основой для решения последующих задач.

Рисунок 1 – Детальное описание подзадач определения начала и конца звучания нот и отслеживания высотности звуков и их фундаментальная роль в прикладных задачах обработки музыкальных фрагментов.

Так как смена ноты в музыкальном фрагменте далеко не всегда характеризуется резким изменением энергии сигнала, определение начала звучания нот на основе изменения этой характеристики представляется в корне неверным. Для решения данной задачи был реализован алгоритм, базирующийся на определении высотности звука в каждый момент времени звучания и формирующий адаптивное представление энергетического спектра сигнала, в котором плавные, неакцентированные и оттого незаметные смены нот воспринимаются более явно. Реализация данного метода уменьшила количество ложно позитивных распознаваний, отчего итоговая эффективность распознавания ритма увеличилась на величину порядка 10–15%.

Для отслеживания высотности звуков в музыкальных фрагментах с многоголосием был реализован алгоритм с использованием методов цифровой обработки сигналов и машинного обучения [5]. Основная идея алгоритма заключается в комбинировании временно-частотного анализа с переменной разрешающей способностью, упомянутого выше, с методом опорных векторов из области машинного обучения, что фактически сводит задачу к процессу распознавания образов на основе векторов, содержащих извлечённые предшествующим

модулем значения спектральных пиков звукового сигнала. Подробная схема описанного алгоритма приведена на Рисунке 2.

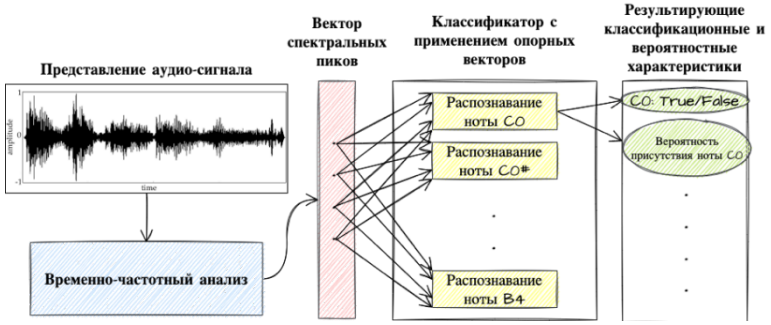


Рисунок 2 – Схема алгоритма для отслеживания высотности звука в музыкальных фрагментах с многоголосием.

Используя описанные алгоритмы в рамках единой системы, удалось провести базовый анализ эффективности автоматической транскрипции на основе результатов распознавания многоголосных произведений Моцарта, Бетховена и Чайковского, показатель которой достиг 60%. К будущим шагам в рамках решения данной задачи можно причислить повышение общей эффективности транскрипции, разработку методов определения музыкального размера и тональности, более точное определение пауз в музыкальных фрагментах, а также возможность отслеживания конкретных инструментов и голосов для разбиения нотного представления по отдельным партиям.

ЛИТЕРАТУРА

1. Benetos, Emmanouil & Dixon, Simon & Duan, Zhiyao & Ewert, Sebastian. (2019). Automatic Music Transcription: An Overview. IEEE Signal Processing Magazine. 36. 20-30. 10.1109/MSP.2018.2869928.
2. Wu, Yu-Te & Luo, Yin-Jyun & Chen, Tsung-Ping & Wei, I-Chieh & Hsu, Jui-Yang & Chuang, Yi-Chin & Su, Li. (2021). Omnizart: A General Toolbox for Automatic Music Transcription. Journal of Open Source Software. 6. 3391. 10.21105/joss.03391.
3. Boashash, Boualem & Touati, S. & Auger, Francois & Flandrin, Patrick & Chassande-Mottin, Eric & Stankovic, Ljubisa & Sucic, Victor & Khan, N. & Sejdić, E. & Saulig, Nicoletta & Shafi, Imran & Jamil, Ahmad & Shah, S.I.. (2015). Measures, Performance Assessment, and Enhancement of TFDs. 10.1016/B978-0-12-398499-9.00007-8.
4. Zhou, Ruohua & Mattavelli, Marco. (2007). A new time-frequency representation for music signal analysis: Resonator time-frequency image. 1 - 4. 10.1109/ISSPA.2007.4555594.
5. Voinov, N.V., Ivanov, D.A., Leontieva, T.V., Molodyakov, S.A. Implementation and Analysis of Algorithms for Pitch Estimation in Musical Fragments Proceedings 24th International Conference on Soft Computing and Measurements, SCM 2021, 2021, P. 113–116.

УДК 004.8

Кобышев К.С. (аспирант),
Молодяков С.А., д.т.н., доцент

ПРИМЕНЕНИЕ МОДЕЛЕЙ ОБРАБОТКИ ЕСТЕСТВЕННЫХ ЯЗЫКОВ ДЛЯ ПОЛУЧЕНИЯ АВТОМАТИЗИРОВАННЫХ ТЕСТОВ

В настоящее время разработка крупных промышленных проектов практически невозможна без процессов автоматизированного тестирования. Требуется покрытие функциональности автоматизированными тестами ввиду того, что крупное приложение может иметь огромное множество различных сценариев использования, которых невозможно проверить в короткие сроки. Нередко и сама тестовая система требует задания структуры, так как она так же может представлять собой сложную программную систему, как и тестируемое приложение. В этом случае тестовая система разбивается на две части: автоматические тесты и тестовый фреймворк. Разработка тестовой системы может приводить к существенным

расходам временных ресурсов, хоть и облегчает тестирование существующей функциональности.

В данном исследовании был предложен способ для частичной автоматизации построения тестовой системы. По сравнению с существующими другими подходами по автоматизации тестирования (ручная разработка автоматических тестов, Behavior-Driven Development [1], методы формальной верификации [2], построение тестовой системы при помощи нейронных сетей [3]) предложенный способ позволяет избежать таких проблем автоматизации тестирования, как: отсутствие структуры тестовой системы, отдельная работа инженеров обеспечения качества и аналитиков, резкое падение производительности тестовой системы при усложнении программы, низкая степень надежности проводимого тестирования.

Предложенный способ заключается в том, чтобы автоматизировать процесс построения интерфейсов тестового фреймворка и тестов из спецификации, составленной на естественном языке [4]. Аналитик составляет спецификацию на естественном языке, которая преобразуется предлагаемым алгоритмом в автоматические тесты. В полученных автоматических тестах используются сгенерированные методы интерфейсов, которые предлагается реализовать инженерам по обеспечению качества на языке Kotlin.

Рассмотрим подробнее алгоритм генерации тестов и интерфейсов тестового фреймворка, показанный на Рисунке 1. Первый этап алгоритма – это получение синтаксического дерева при помощи предобученной модели Open IE. Синтаксическое дерево имеет в своих узлах: субъект, отношение, объект. Объект представляет собой группу параметров. Параметры могут так же представлять собой группы параметров, либо значения. Значения составляют листья полученного синтаксического дерева.

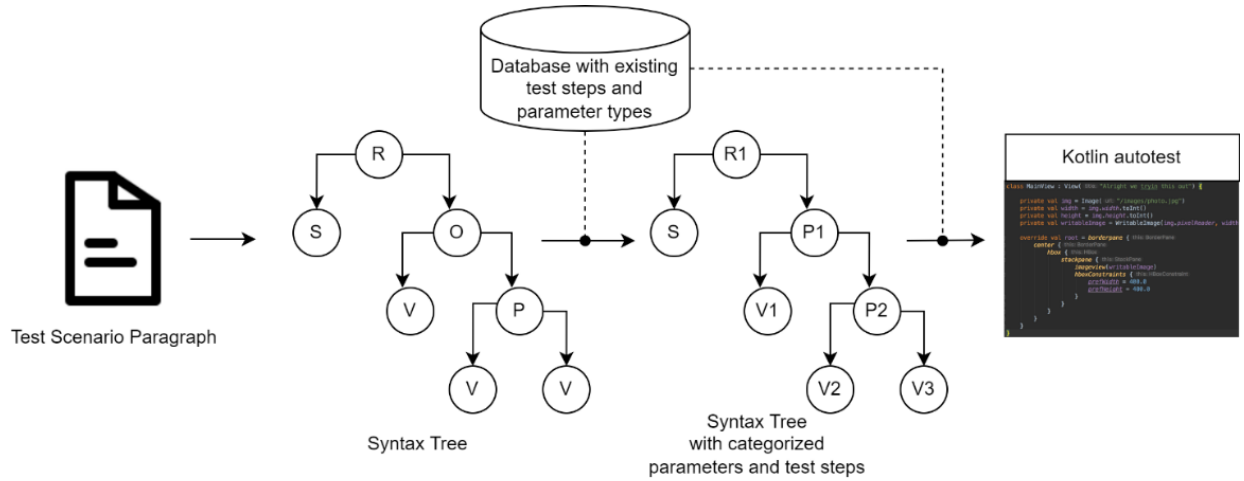


Рисунок 1 – Шаги предлагаемого решения для получения автоматических тестов

Затем узлы полученного синтаксического деревья подлежат кластеризации при помощи координат, полученных из модели GloVe. В процессе построения каждого нового теста динамически формируются новые кластеры отношений, субъектов, групп параметров. Каждый кластер ассоциируется с первым определенным для данного кластера элементом. Также каждый элемент ассоциируется с набором из слов, полученных из исходной спецификации на естественном языке.

Последним этапом является преобразование полученного дерева с типизированными элементами в код на языке Kotlin. Название пункта из спецификации преобразуется в имя тестового метода. Субъект преобразуется в объект языка Kotlin, в контексте которого (при помощи лямбда-выражения с контекстом) выполняются тестовые шаги. Отношения преобразуются тестовые шаги. Объект, представляющий собой иерархию групп параметров и значений, преобразуется во вложенные вызовы конструкторов, и используется, как параметр тестового шага.

В будущем планируется полностью завершить реализацию предлагаемого алгоритма для получения автоматизированных тестов, а также апробация и внедрение алгоритма на промышленном проекте.

ЛИТЕРАТУРА

1. M. Irshad, R. Britto and K. Petersen. “Adapting Behavior Driven Development (BDD) for large-scale software systems”. Journal of Systems and Software. 2021. №17. 20 p.
2. W. Wasira. “Existing Tools for Formal Verification and Formal Methods”. MS Computer Science, Lewis University. 2020.
3. А. Д. Данилов, В. М. Мугатина. Верификация и тестирования сложных программных продуктов на основе моделей нейронных сетей. Вестник ВГТУ. 2016. Том 12. №6. сс. 62–67.
4. M. A. Fountoura. “Systematic Approach for Framework Development”. Rio de Janeiro, 1999. 165 p.

УДК 519.6

Сергеева Е. И. (аспирант),
Лисс А. Р., д.т.н., профессор

«БЫСТРЫЙ» АЛГОРИТМ ФОРМИРОВАНИЯ ПРОСТРАНСТВЕННЫХ КАНАЛОВ В ШИРОКОМ СЕКТОРЕ ОБЗОРА С РАЗРЕШЕНИЕМ ПО ДАЛЬНОСТИ

В работе рассматривается задача извлечения в реальном времени информации о пространственном положении источника гидроакустического сигнала по данным линейной антенной решётки (ЛАР).

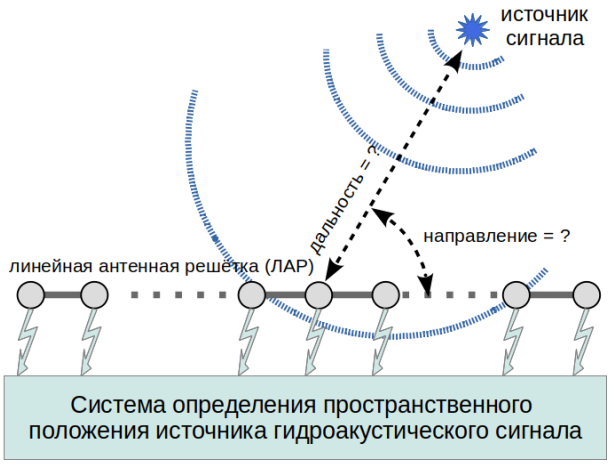


Рисунок 1 – Постановка задачи определения пространственного положения источника гидроакустического сигнала

Большие апертуры современных ЛАР дают возможность проводить обработку не только для традиционного плосковолнового приближения в дальней зоне, но и с учётом кривизны волнового фронта в зоне Френеля. Обработка в зоне Френеля позволяет извлекать информацию не только о направлении, но и о дальности до источника сигнала в пассивном режиме [1,2].

Современная обработка гидроакустических сигналов выполняется программно на специализированных многопроцессорных системах, которые обычно строятся на базе высокопроизводительных цифровых сигнальных процессоров (ЦСП), связанных между собой при помощи коммуникационной среды. К таким многопроцессорным системам предъявляется ряд специфических требований характерных как для высокопроизводительных, так и для бортовых вычислительных систем [3].

Актуальным является повышение производительности программного обеспечения для решения задач обработки гидроакустической информации в реальном времени, путём

сокращения вычислительной сложности алгоритмов решения задач обработки информации ЛАР (разработки «быстрых» вычислительных алгоритмов).

Вопросы синтеза систем пространственно-частотно-временной обработки гидроакустических сигналов хорошо изучены в литературе, в частности в работах [4, 5]. Одним из вычислительно трудоёмких этапов такой обработки является формирование пространственных каналов (ФПК) в широком секторе обзора с разрешением по дальности.

Авторами предложен [6; 7] вычислительный алгоритм ФПК ЛАР, который имеет квазилинейную вычислительную сложность вместо квадратичной. «Быстрый» алгоритм ФПК ЛАР в каждом частотном канале сводится к выполнению двух быстрых преобразований Фурье (БПФ) и трёх поэлементных комплексных умножений векторов, и является универсальным для случаев дальней зоны и зоны Френеля.

На Рисунке 2 продемонстрированы результаты его работы на примере широкополосных сигналов двух источников, находящихся на разных дальностях и разных направлениях.

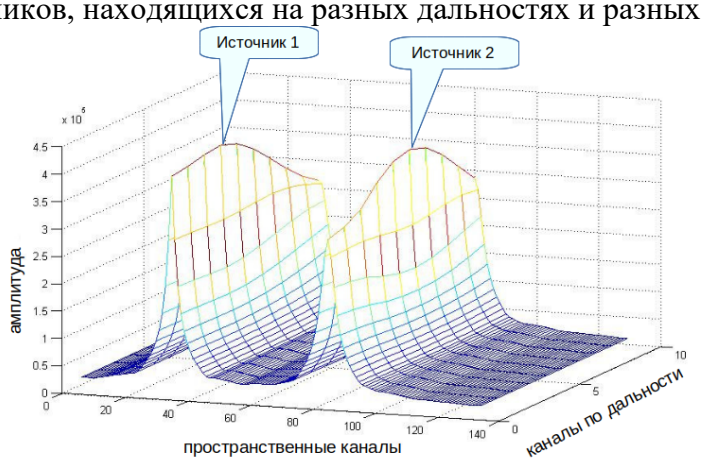


Рисунок 2 – Отклик широкополосных сигналов 2-х источников на выходе системы обработки

На Рисунке 3 в виде двумерной яркостной картины представлено изменение относительного выигрыша в производительности при независимом изменении количества приёмных элементов ЛАР (L) и количества пространственных каналов (Q). Максимальное ускорение достигается если сумма количества приёмников ЛАР и количества пространственных каналов равна степени двух. Учёт представленных результатов анализа производительности при выборе значений параметров обработки (количества пространственных каналов в секторе обзора) в технических системах позволяет достигать оптимальной вычислительной эффективности от применения «быстрого» алгоритма ФПК ЛАР.

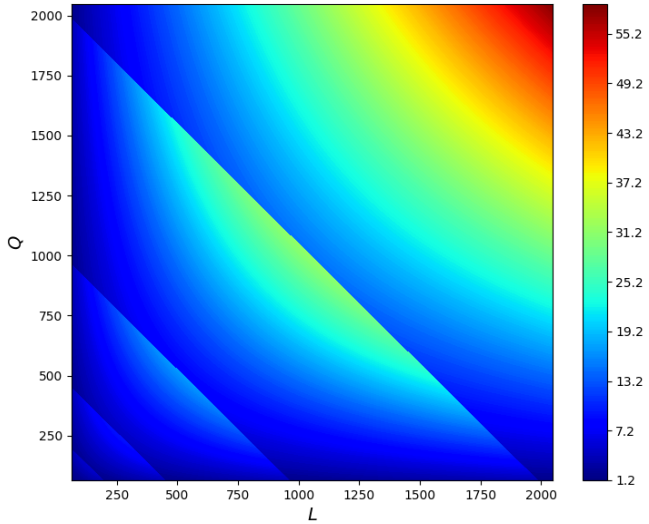


Рисунок 3 − Яркостная картина зависимости величины выигрыша в производительности от количества проёмных элементов ЛАР и количества пространственных каналов

Предложенный вычислительный алгоритм реализован в составе программного обеспечения системы определения пространственного положения источника гидроакустического сигнала на вычислительных средствах семейства «Комдив». Вычислительные эксперименты на ЦСП «Комдив-128» [8] показали, что выигрыш в производительности для «быстрого» алгоритма ФПК ЛАР качественно сопоставим с теоретическим и позволяет решать поставленные задачи обработки в реальном времени на данных вычислительных средствах.

ЛИТЕРАТУРА

1. Кремер И. Я., Кремер А. И., Петров В. М. Пространственно-временная обработка сигналов / под ред. И. Я. Кремера. М.: Радио и связь, 1984.
2. Рудько И. М. Определение дистанции до цели, находящейся в зоне Френеля антенны // Проблемы управления. 2006. Вып. 6. С. 79-82.
3. Лисс А. Р., Рыжиков А. В. Системы обработки сигналов в гидроакустических станциях и комплексах // Гидроакустика. 2016. Вып. 27 (3). С. 38-47.
4. Гусев В. Г. Системы пространственно-временной обработки гидроакустической информации. Л.: Судостроение, 1988.
5. Лоскутова Г. В., Полканов К. И. Пространственно-частотные и частотно-волновые методы описания и обработки гидроакустических полей. СПб.: Наука, 2007.
6. Пуеров Г. Ю., Сергеева Е. И. Применение быстрого преобразования Фурье при формировании характеристик направленности линейной антенной решётки // Гидроакустика. 2015. Вып. 24(4). С. 70-74.
7. Пуеров Г. Ю., Сергеева Е. И. Об использовании «быстрой свертки» для формирования характеристик направленности линейной антенной решётки при работе в зоне Френеля // Гидроакустика. 2019. Вып. 40 (4). С. 56-65.
8. Микросхема интегральная 1890ВМ7Я. Указания по применению, ЮКСУ.431281.104Д4. М., 2014.

УДК 004.627

Мурзаканов И.М., Почернин В.С. (2 курс бакалавриата),
Эйзенах Д.С. (1 курс аспирантуры),
Маслаков А.П., старший преподаватель

ИССЛЕДОВАНИЕ ЭФФЕКТИВНОСТИ СЖАТИЯ РЕДКО ИСПОЛЬЗУЕМЫХ ДАННЫХ

При работе с большими данными нередко возникают различные проблемы, связанные с их хранением. Например, может быть излишне затратным хранение редко используемых данных, которые занимают полезное пространство на носителе информации. Возможным решением является сжатие таких данных и распаковка непосредственно при обращении к ним. При правильном выборе алгоритмов сжатия и распаковки сохраненная на носителе память может являться более приоритетным ресурсом, чем время, затраченное на работу алгоритма. При успешных результатах работы алгоритма данную технологию можно применять в корпоративной среде для снижения затрат на покупку и обслуживание носителей информации.

Целью данной работы является исследование эффективности использования алгоритма сжатия и распаковки применительно к базам данных, содержащим большие объемы информации. Работа проводилась с данными, четко привязанными к определенному моменту времени. Это было необходимо для того, чтобы можно было однозначно установить, какую информацию нужно распаковать в каждом случае.

Тестирующая программа извлекает необходимые для сжатия данные, применяет к ним исследуемый алгоритм и замещает соответствующие ячейки в таблице. При этом вычисляется размер информации до и после сжатия и высчитывается разница и часть сэкономленной памяти.

Отдельное внимание стоит уделить процессу извлечения информации из базы данных для последующего сжатия. Обрабатываемые значения следует извлекать группами, при этом необходимо выбрать оптимальное число записей в одной группе. Если выбрать его слишком маленьким – количество запросов к базе данных будет большим и это негативно скажется на производительности. В противном случае, при выборе большого количества записей в одну группу, также может возникнуть проблема производительности. Так как во время обработки данные будут храниться в оперативной памяти, при её переполнении будет задействована гораздо более медленная память диска, а в крайних случаях процесс может аварийно завершиться.

На Рисунке 1 приведена схема исполнения программы:

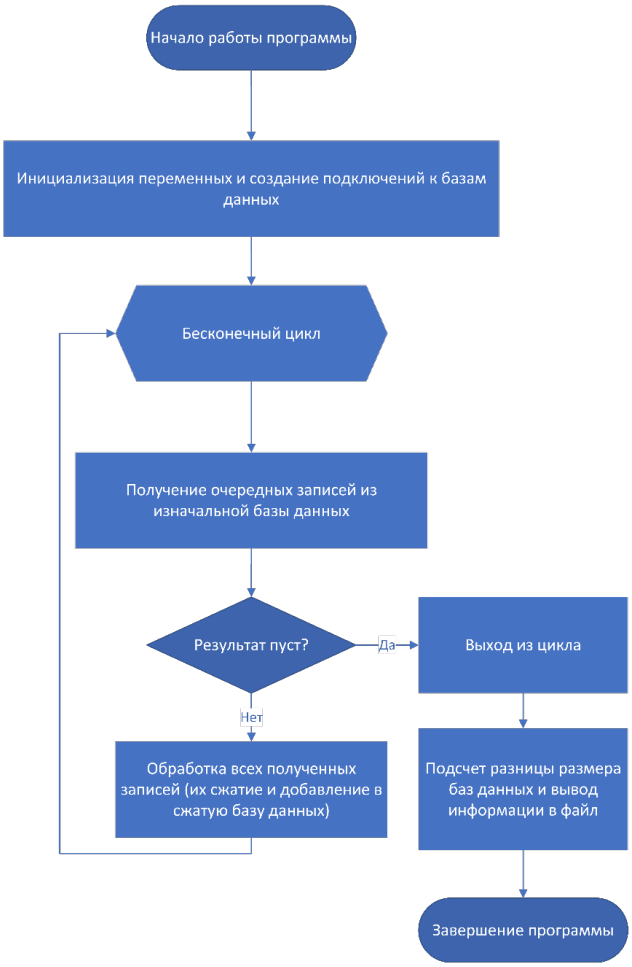


Рисунок 1 – Блок-схема работы программы

В качестве инструментария при разработке программы были выбраны следующие решения:

– В рамках исследования был выбран Deflate – это алгоритм сжатия без потерь, который используется во многих open-source библиотеках. В нашем случае этой библиотекой является Zlib [1].

– В качестве системы хранения информации использовалась объектно-реляционная СУБД с открытым исходным кодом – PostgreSQL [2].

– Языком программирования был выбран C++, поскольку он обладает высокой скоростью исполнения кода, что является неоспоримым преимуществом для сложных вычислительных алгоритмов.

– Для работы с PostgreSQL в среде C++ была использована библиотека libpqxx [3] – официальный C++ клиент API для PostgreSQL.

Экспериментальные данные были получены на персональном компьютере со следующими характеристиками:

– Процессор: AMD Ryzen 7 3700x 3.6 GHz

– Оперативная память: 16 ГБ DDR4 3200 MHz

– Носитель: SSD M.2 R1800/W1200 МБ/с

По итогу проведения эксперимента на больших объемах данных, был получен вывод:

Всего памяти сэкономлено	1.6724 Гигабайт
Память до сжатия	2.9001 Гигабайт
Память после сжатия	1.2276 Гигабайт
Часть сэкономленной памяти	57.7%

Согласно полученным данным, удалось освободить 57.7% памяти, что является удовлетворительным результатом и может помочь сэкономить средства при закупке систем хранения данных. При этом необходимо учитывать, что доступ к информации уже не будет предоставляться моментально: при каждом обращении к ячейке таблицы потребуется производить распаковку.

ЛИТЕРАТУРА

1. Жан-Лу Гайи, Марк Адлер, «Zlib,» [В Интернете]. Available: http://www.zlib.net/. [Дата обращения: 20 Март 2022].
2. «Documentation - PostgreSQL,» [В Интернете]. Available: https://www.postgresql.org/docs/. [Дата обращения: 19 Март 2022].
3. «libpqxx: the official C++ language binding for PostgreSQL,» [В Интернете]. Available: http://pqxx.org/development/libpqxx/. [Дата обращения: 19 Март 2022].

УДК 519.6

Шестакова А.Ю. (4 курс бакалавриата),
Черноруцкий И.Г., д.т.н., профессор

ИССЛЕДОВАНИЕ УЛУЧШЕННОГО АЛГОРИТМА ЧЕРНОЙ ДЫРЫ ДЛЯ РЕШЕНИЯ ЗАДАЧ ОПТИМИЗАЦИИ

Целью работы является разработка и исследование улучшенного алгоритма Черной дыры при решении жестких (овражных) задач.

Алгоритм Черной дыры – вдохновленный природой алгоритм, относящийся к группе метаэвристических алгоритмов. Алгоритм вдохновлен феноменом черной дыры и пытается имитировать ее свойство притягивать другие звезды в космосе. Для его улучшения применяется несколько модификаций, главная цель которых – решение проблемы локальных оптимумов, присущая базовому алгоритму.

В программе задаются следующие параметры: целевой функционал, его размерность, область поиска решений, требуемое значение погрешности, максимальное количество итераций, размер популяции возможных решений (количество звезд).

Рассмотрим этапы алгоритма:

- Инициализация популяции: в области поиска генерируем заданное число начальных приближений к искомому решению задачи – инициализируем популяцию агентов (звезд);
- Миграция агентов популяции: выбранная черная дыра применяет гравитационную силу, чтобы притянуть другие звезды, и когда какая-либо звезда приближается достаточно близко к черной дыре, она исчезает, а в исследовательском пространстве возрождается новая звезда. Таким образом, с помощью набора миграционных операторов перемещаем агенты в области поиска, чтобы в конечном счете приблизиться к искомой черной дыре – экстремуму оптимизируемой функции;
- Выбор текущего лучшего решения: популяция оценивается с помощью целевой функции, и звезда с наилучшей стоимостью становится текущей черной дырой;
- Завершение поиска: проверяем выполнение условий окончания итераций, и если они выполнены (достигнута искомая точность), завершаем вычисления, принимая лучшее из найденных положений агентов популяции за приближенное решение задачи. Если указанные условия не выполнены, возвращаемся к выполнению второго этапа.

Для решения проблемы локальных оптимумов в алгоритм внесен ряд модификаций. Звезды в улучшенном алгоритме Черной дыры движутся под случайным углом к лучшей звезде, что позволяет им более эффективно исследовать пространство вокруг текущей черной дыры. Также улучшение точности достигается за счет управления процессом регенерации после поглощения звезд черной дырой. Стратегия основана на увеличении возможностей

исследования (разведки) на одних этапах и акцентировании внимания на уточнении ранее найденных решений (эксплуатации) на других, что добавляет больше гибкости движению звезд в исследовательском пространстве.

Как упоминалось выше, в ходе работы алгоритма звезды приближаются к черной дыре, и, если какая-либо звезда проходит через горизонт событий (форма, подобная сфере, которая существует вокруг черной дыры и является критерием близости звезды к черной дыре), она поглощается черной дырой, что приводит к ее удалению из текущей популяции. Всякий раз, когда черная дыра поглощает звезду, новая звезда генерируется с использованием одной из следующих стратегий:

- Генерация новой звезды случайным образом с использованием равномерного распределения;
- Генерация новой звезды путем выбора двух звезд из популяции случайным образом и рекомбинация их атрибутов. Лучшая звезда, полученная в результате рекомбинации, будет добавлена к популяции.

Вероятности использования стратегий регенерации [0,75, 0,25] чередуются в ходе выполнения алгоритма. Таким образом, благодаря чередованию вышеперечисленных стратегий можно чередовать акценты на исследовании и эксплуатации в ходе эволюции алгоритма и достигать наилучшего баланса между ними для поиска оптимального решения заданного функционала, а также решения поставленной задачи.

Для тестирования алгоритма выбрана «банановая долина» Розенброка.

$$F = 100\,(x_1^2 - x_2)^2 + (1 - x_1)^2$$
$$x^* = (1;\ 1);\ F(x^*) = 0$$

Линии уровня функции Розенброка имеют ярко выраженную овражную структуру с криволинейным дном оврага. Несмотря на то, что степень овражности в этом случае не очень высока ($\eta \approx 2500$), работа многих алгоритмов затруднена из-за значительного уменьшения скорости сходимости. В частности, непригодными оказываются алгоритмы покоординатного спуска, а также классические методы спуска по антиградиенту.

Полученные результаты работы улучшенного алгоритма Черной дыры приведены в таблице ниже.

Таблица 1 - Результаты работы алгоритма для функции Розенброка

	x		F(x)	Количество итераций
	x_1	x_2		
Искомый результат	1	1	0	
1	1.0017	1.0033	0.000006	23
2	0.9971	0.9943	0.000008	72
3	1.0028	1.0056	0.000007	41
4	1.0026	1.0054	0.000008	151
5	0.9971	0.9943	0.000007	161
6	1.0028	1.0056	0.000008	147
7	0.9991	0.9982	0.000007	90
8	0.9968	0.9937	0.000009	56
9	1.0024	1.0048	0.000005	107
10	0.9970	0.9941	0.000008	89
11	1.0024	1.0048	0.000009	5
12	0.9968	0.9937	0.000009	102
13	0.9982	0.9965	0.000005	56
14	0.9994	0.9989	0.000004	23
15	1.0028	1.0057	0.000008	49

Опираясь на полученные результаты, можно сделать вывод, что алгоритм успешно справляется с поиском глобального минимума и получает искомое решение с заданной погрешностью 0.00001 достаточно быстро, при этом избегая ситуацию заклинивания и проблему попадания в локальные оптимумы. Более того алгоритм предоставляет нам возможность регулировать параметры, такие как размер популяции и область поиска значений, что делает его более гибким инструментом оптимизации. При увеличении размера популяции и уменьшении области поиска алгоритм будет получать искомые решения быстрее. Возможность регулировать эти параметры крайне важна для решения практических задач с более жесткими ограничениями и меньшими знаниями о природе рассматриваемого целевого функционала.

ЛИТЕРАТУРА

1. Черноруцкий И. Г., Воинов Н. В., Котлярова Л. П. Методы оптимизации. - СПб.: СПбПУ, 2020. - 167 с;
2. Черноруцкий И.Г. Методы принятия решений. – СПб.: БХВ Петербург, 2005. – 417 с.:
3. H.A. Abdulwahab, A. Noraziah, A.A. Alsewari, S.Q. Salih An enhanced version of black hole algorithm via levy flight for optimization and data clustering problems**.** IEEE Access 7, 2019. URL: 10.1109/ACCESS.2019.2937021.

УДК 004

Томилин И. С. (1 курс магистратуры),
Фёдоров С. А., старший преподаватель

ОПТИМИЗАЦИЯ МАШИННОГО КОДА ПРИЛОЖЕНИЙ C++

Целью работы является демонстрация одного из способов улучшения производительности приложений, написанных на языке C++, с помощью методов оптимизации машинного кода.

Для выполнения данной работы использовался компилятор g++ серии 8. В качестве тестируемого приложения был взят микросервис, обрабатывающий лидарные данные с помощью алгоритмов кластеризации и сегментации.

Компилятор g++ имеет большое количество методов оптимизации машинного кода. Применяя те или иные комбинации методов, можно достичь как улучшения производительности приложения, так и его ухудшения. Чем агрессивнее методы оптимизации, тем больше шансов, что производительность приложения ухудшится.

В рассматриваемом случае основной упор был сделан на оптимизацию с обратной связью **Profile-guided optimization** (PGO)[1]. Основной смысл этой оптимизации заключается в том, что при ее использовании компилятор не выполняет предсказания, а основывается на результатах профиля ***.gcda**, который порождается во время инструментального запуска приложения. Однако у этого подхода есть несколько недостатков:

1. Требуется большее время на компиляцию, поскольку теперь вместо одного этапа, компиляция делится на три (инструментальная компиляция и компоновка, инструментальный запуск, компиляция и компоновка с обратной связью[2]);
2. Нет гарантий, что данная оптимизация улучшит производительность лучше, чем стандартные методы оптимизации;
3. Необходимо иметь в наличии синтетический журнал для инструментального запуска и порождения профиля ***.gcda**.

На Рисунке 1 показан поэтапный[3] процесс оптимизации с обратной связью.

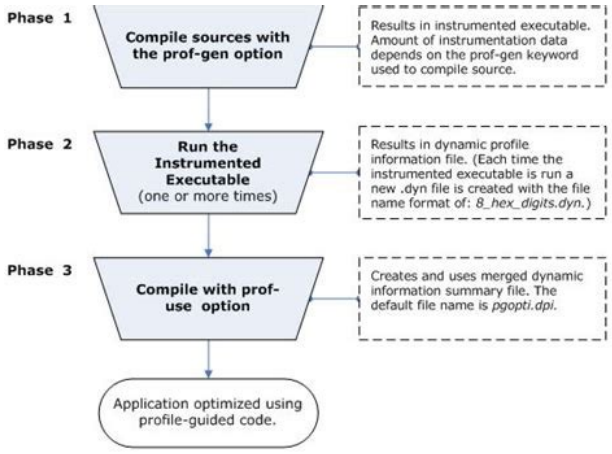


Рисунок 1 – Этапы оптимизации с обратной связью.

В результате подбора различных методов оптимизации с обратной связью получилась таблица, в которой показано сравнение размера исполняемого файла и скорости обработки данных приложения относительно базовых результатов без применения оптимизаций.

Базовые результаты скомпилированного микросервиса без оптимизаций:

27947 мс – время обработки данных;

724 Кб – размер исполняемого файла.

Таблица 1 – Результаты оптимизации с использованием разных методов

Методы	Время обработки данных, мс	Разница в скорости работы, %	Размер исполняемого файла, Кб	Разница в размере исполняемого файла, %
-O2 -flto -march=native	**28629**	**+2**	**428**	**41**
-O3 -march=native	**28364**	**+1**	**800**	**+10**
-O2 -march=native	**27953**	**0**	**724**	**0**
-fprofile-generate/-fprofile-use	206057	+636	10048	+1287
-fprofile-generate/-fprofile-use -O2 -march=native	**25086**	-10	**684**	**-6**
-fprofile-generate/-fprofile-use -O3 -march=native	**25890**	-7	**688**	**-5**
-fprofile-generate/-fprofile-use -flto -O2 -march=native	**25986**	**-7**	**480**	**-34**

Согласно полученной таблице можно выделить два наиболее оптимальных метода. Используя метод «-fprofile-generate/-fprofile-use -O2 -march=native», удалось уменьшить размер исполняемого кода приложения на 6% и улучшить скорость обработки данных на 10%. При добавлении к этому методу оптимизации времени компоновки -flto, удалось уменьшить размер исполняемого кода еще на 28%, однако при этом скорость обработки снизилась на 3% и составила 7%.

В ближайших планах сделать интеграцию автоматического подбора наиболее оптимального метода оптимизации кода в систему CI(continious integration) для улучшения производительности компилируемых микросервисов.

ЛИТЕРАТУРА

1. [Электронный ресурс] https://gcc.gnu.org/onlinedocs/gcc-8.5.0/gcc/Instrumentation-Options.html
2. [Электронный ресурс] https://developer.ibm.com/articles/gcc-profile-guided-optimization-to-accelerate-aix-applications/

3. [Электронный ресурс] https://www.intel.com/content/www/us/en/develop/documentation/cpp-compiler-developer-guide-and-reference/top/optimization-and-programming-guide/profile-guided-optimization-pgo.html

УДК 004.023

Ходырева В.А. (4 курс бакалавриата),
Черноруцкий И.Г., д.т.н., профессор

ИССЛЕДОВАНИЕ АЛГОРИТМА JAYA ДЛЯ РЕШЕНИЯ ЗАДАЧ ОПТИМИЗАЦИИ

Целью работы является исследование алгоритма оптимизации, основанного на принципе выживания сильнейших, который называется JAYA. За основу была выбрана статья «An Intensive and Comprehensive Overview of JAYA Algorithm, its Versions and Applications» [1], опубликованная 27 мая 2021 года в онлайн-журнале Springer.

Алгоритм JAYA был предложен в 2016 году. Данный метод оптимизации относится к группе алгоритмов роевого интеллекта, но он был изначально вдохновлён эволюционными алгоритмами и принципом «выживает самый приспособленный». В процессе поиска оптимальных решений алгоритм пытается приблизиться к лучшим решениям и пренебрегает худшими.

JAYA имеет ряд плюсов:

– простота в использовании и реализации. Необходимо установить всего два основных параметра: размер популяции и максимальное количество итераций;

– лёгкая модификация алгоритма. JAYA можно легко гибридизировать с другими алгоритмами оптимизации для решения задач разной сложности.

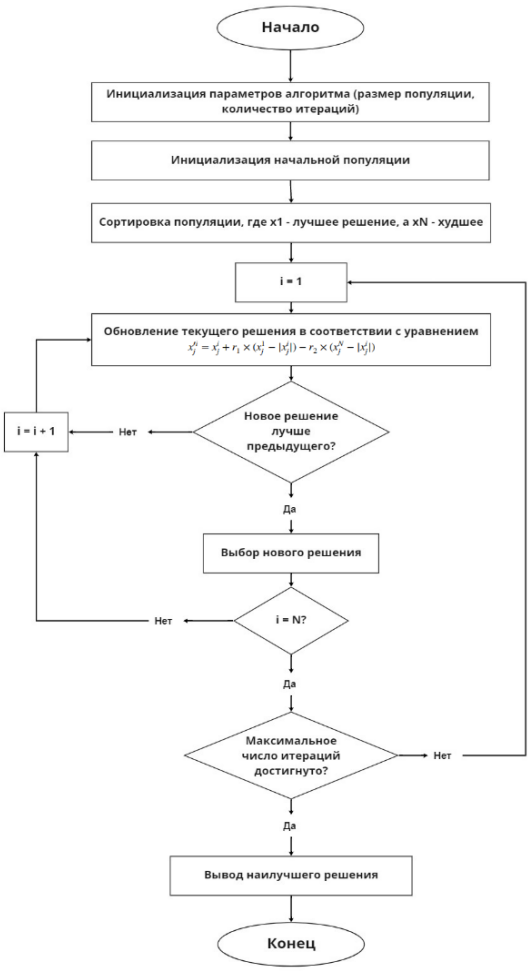


Рисунок 1 – блок-схема алгоритма JAYA

Алгоритм состоит из пяти шагов:

– инициализация. На данном этапе задаются два алгоритмических параметра: размер популяции N и количество итераций T;

– построение начальной популяции. Начальные решения алгоритма конструируются и сохраняются в памяти JAYA, которая представляет собой расширенную матрицу размером N × D, где N — количество решений, а D — размерность;

– процесс эволюции. Итерация за итерацией, каждая переменная решения из памяти JAYA подвергается изменениям;

– обновление решений. На данном этапе происходит сравнение значений целевой функции нового и текущего решения. Текущее решение будет заменено, если значение целевой функции при новом решении меньше. Этот процесс будет повторяться N раз;

– проверка условия завершения. Алгоритм JAYA повторяет шаги 3 и 4 до тех пор, пока не будет достигнуто правило остановки, которое является максимальным числом итераций T, заданным на шаге 1.

Для проверки работоспособности алгоритм был протестирован на пяти тестовых функциях: Розенброка, трёхгорбого верблюда, Изома, МакКормика, Шаффера N2.

По результатам эксперимента, проведённого на тестовых функциях, можно сказать, что алгоритм показывает достойный результат. Соответственно, его можно успешно использовать для решения некоторых видов задач оптимизации. Стоит отметить, что алгоритм также прост в модернизации. Его легко можно объединить с другим алгоритмом (например, с алгоритмом локального поиска) для увеличения производительности и устранения недостатков.

ЛИТЕРАТУРА

1. Raed Abu Zitar, Mohammed Azmi Al-Betar, Mohammed A. Awadallah, Iyad Abu Doush, Khaled Assaleh // An Intensive and Comprehensive Overview of JAYA Algorithm, its Versions and Applications // 2021 //Springer. URL: https://link.springer.com/article/10.1007/s11831-021-09585-8 // Дата обращения: 07.12.2021
2. И. Г. Черноруцкий, Н. В. Воинов, Л. П. Котлярова // Методы оптимизации. Учебное пособие // Санкт-Петербург 2020 // Дата обращения: 25.11.2021

Секция «Технологии искусственного интеллекта»

УДК 621.319

Абунагимова А.Р. (2 курс магистратуры),
Глазунов В.В., к.т.н., доцент

РАЗРАБОТКА ПАРАМЕТРИЧЕСКИХ МОДЕЛЕЙ ПОВТОРЯЕМЫХ СБОРОК В ИЗОЛИРОВАННОЙ СРЕДЕ DOCKER

Целью работы является создание воспроизводимых сборок образов программного обеспечения Docker. Задача особенно актуальна в сфере информационной безопасности, так как повторяемые сборки образов позволяют подтвердить, что исполняемый файл не подвергался изменениям, содержащим потенциальные уязвимости или не протестированные функции.

Docker – один из самых популярных инструментов для автоматизации, развертывании и управлении приложений в средах с поддержкой контейнеризации [1]. Контейнеризация исторически появилась после виртуализации, которая позволила не только увеличивать количество приложений на одном сервере, но и изолировать их друг от друга. Независимость на уровне контейнеров осуществляется средствами операционной системы – пространством имён. Таким образом, контейнеры позволяют разработчикам избавиться от проблем с взаимными зависимостями приложений, упрощают перенос приложения на новую инфраструктуру и его масштабирование.

Для того, чтобы запустить приложение в контейнере необходимо написать Dockerfile. Это простой текстовый файл, который содержит в себе пошаговые инструкции запуска приложения. В файле обязательно указать образ, на основе которого приложение будет запущено, это может образ операционной системы, пустой образ или же образ с предустановленным компилятором выбранного языка программирования. Далее могут следовать инструкции по копированию файлов с исходным кодом, инструкции для выполнения различных команд, скриптов, выполняющих сборку приложения. Сборка Dockerfile представляет собой исполняемый файл – Docker-образ [2], который содержит в себе код, зависимости, библиотеки, переменные окружения и файлы конфигурации приложения.

Современную разработку тяжело представить без использований контейниризированных приложений, именно поэтому воспроизводимость является одним из наиболее важных критериев для Docker-образов. К сожалению, текущее состояние программного обеспечения Docker не позволяет гарантировать воспроизводимость сборок: причиной может стать смена тега базового (родительского) образа или тег базового образа в удаленном хранилище может начать указывать на другой образ. Таким образом, контейнер на сервере с закэшированным образом может собираться, на новых окружениях могут возникнуть проблемы. Кроме того, нередко для сборки приложения требуется установить дополнительные модули или прочие пакеты, которые имеют скрытые зависимости. В таком случае также необходим механизм строгого тегирования, что вызывает сложности в процессе написания Dockerfile.

Каждая инструкция Dockerfile представляет собой слой Docker образа – набора изменения данных, которые накладываются друг на друга и максимально переиспользуются между различными образами благодаря механизму cope-on-write. Каждый слой в файловой системе имеет уникальный номер, который зависит от предыдущих слоев и измененных на текущем шаге данных [3]. Таким образом, для того, что гарантировать повторяемость сборки необходимо обеспечить повторяемость ID всех слоев образа, то есть зафиксировать изменения данных на каждом этапе сборки: файлы кода приложения, конфигурации, скачиваемые утилиты и пакеты, включая скрытые зависимости. Также для обеспечения одинаковых номеров сборок потребуется синхронизировать время создания Docker-образов и унифицировать номер дополнительного контейнера, в котором собирается текущий образ.

В рамках решения поставленной задачи предложено построить информационную систему, алгоритм которой схематически представлен на Рисунке 1. На вход программы поступает написанный Dockerfile и все файлы, которые необходимы для запуска приложения в контейнере, ПО обращается к текущему базовому образу и сохраняет версии скачиваемых утилит и пакетов. Далее производится сборка образа, в результате которой сохраняются ID слоев, а итоговый образ собирается с заранее известными параметрами времени и даты, для последующей унификации. Для того, чтобы убедиться в воспроизводимости сборки тот же алгоритм применяется на новом окружении, верифицирование ID слоев производится с помощью расчёта уникального номера для каждого слоя по следующей формуле [5]:

$$ChainID(layerN) = \text{SHA256hex}\big(ChainID(layerN-1)\big) + \text{" "} + DiffID(layerN),$$

где ChainID(layerN) – ID слоя N, SHA256hex – криптографическая 256-битная хэш-функция, DiffID(layerN) – криптографическая 256-битная хэш-функция от несжатого архива изменений файловой системы формата tar. Далее необходимо унифицировать ID образа, используя приведение времени создание сборки к единому значению [4].

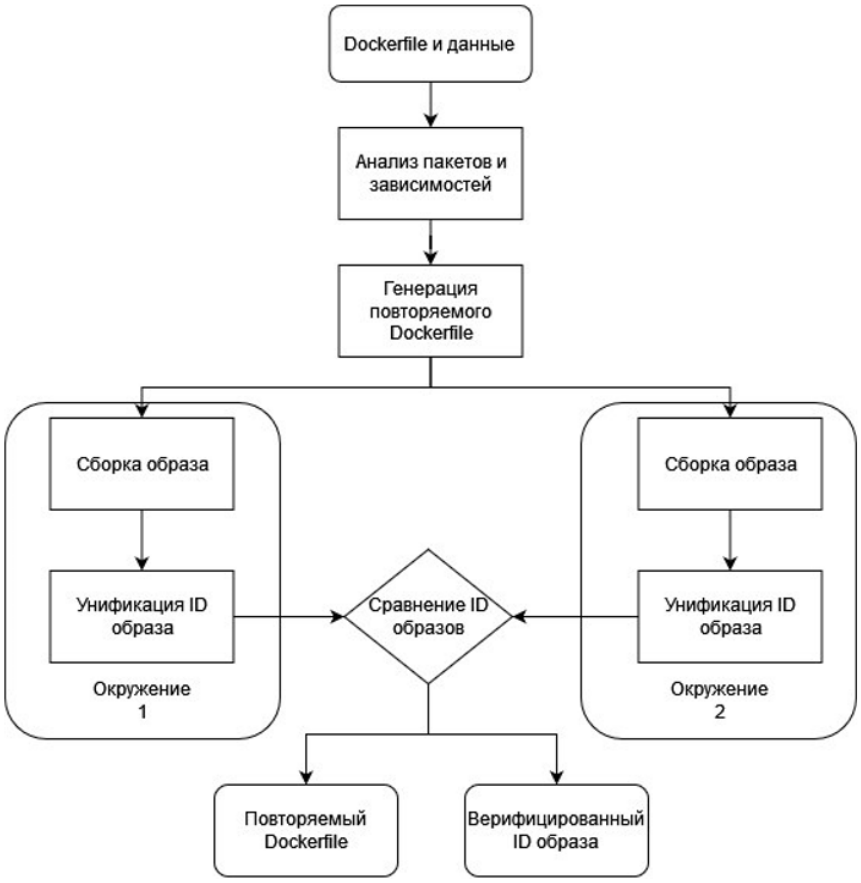


Рисунок 1 – Структурная схема алгоритма разработки параметрической модели повторяемых Docker образов.

В дальнейшем для воспроизведения повторяемой сборки необходимо передавать на вход разработанного программного обеспечения не только сгенерированный Dockerfile и исходные данные разрабатываемого приложения, но и полученный верифицированный идентификационный номер образа. На целевом окружении предлагается произвести сборку и унификацию ID образа, а также провести сравнение полученного и входного ID образов. Таким образом, предлагаемый алгоритм позволит получить воспроизводимые конвейеризированные сборки приложений без необходимости внесения изменений в исходный набор данных.

В ходе выполнения поставленной задачи были выполнены:

1) анализ предметной области систем контейнеризации;
2) разработана структура параметрической модели для различных языков программирования;
3) разработан алгоритм генерации повторяемых сборок на целевом окружении.

ЛИТЕРАТУРА

1. Моуэт Э. Использование Docker. – ДМК-Пресс, 2017 г.
2. Ferme V. Using Docker Containers to Improve Reproducibility in Software and Web Engineering Research // Conference: International Conference on Web Engineering, 2016
3. Container principle (understand layerID, diffID, chainID, cache ID) [Электронный ресурс], 2021. URL: https://programmer.ink/think/container-principle-understand-layerid-diffid-chainid-cache-id.html (Дата обращения 10.03.2022)
4. How to Digest a Docker Image [Электронный ресурс], 2019. URL: https://maori.geek.nz/how-to-digest-a-docker-image-ca9fc7630b71 (Дата обращения 10.03.2022)
5. Docker Image Specification v1.2.0 [Электронный ресурс], 2021. URL: https://github.com/moby/moby/blob/master/image/spec/v1.2.md (Дата обращения 10.03.2022

УДК 004.415

Агаев А.Ф. (2 курс магистратуры),
Медведев Б.М., к.т.н., доцент

АЛГОРИТМ СВЯЗЫВАНИЯ ИМЕНОВАННЫХ СУЩНОСТЕЙ С ИСПОЛЬЗОВАНИЕМ КОНТЕКСТА ИСПОЛЬЗОВАНИЯ И МОРФОЛОГИЧЕСКИХ ПРИЗНАКОВ

Одной из актуальных задач обработки естественного языка (ОЕЯ) является связывание именованных сущностей (СИС). Задача СИС заключается в том, чтобы провести соответствие между именованной сущностью (ИС) и записью о ней в некоторой базе знаний. Цель операции связывания состоит в том, чтобы избавиться от неоднозначностей при анализе текстов. Решение этой задачи необходимо для улучшения качества машинного поиска, статистического анализа, структурирования документов.

Можно выделить два основных подхода к СИС. Графовый подход предполагает построение графов знаний и проведение связывания именованных сущностей с помощью анализа связей между узлами графа [1]. Графовые подходы довольно популярны, но для хороших результатов требуют построения графов знаний огромных размеров, что является недостатком. Другие подходы – основанные на векторных представлениях – позволяют использовать контекст, содержащийся в самом тексте, и опознавать именованную сущность, то есть соотносить ее с записью в базе знаний, анализируя контекст, в котором эта сущность упоминается [2] [3]. Подходы с векторными представлениями имеют большую популярность и показывают лучшие результаты [3], по сравнению с графовыми, однако их недостаток в том, что случаются ошибки связывания, когда речь идет о сущностях, которые упоминаются в текстах с похожей тематикой, из-за чего имеют схожий контекст упоминаемости. Качество работы алгоритмов связывания именованных сущностей чаще всего оценивают с помощью f-

меры [4], которая является средним гармоническим полноты и точности результатов, и может быть выражена, как

$$f = 2 \times \frac{p \times r}{p + r},$$
$$p = \frac{tp}{tp + fn},$$
$$r = \frac{tp}{tp + fp},$$

где p – полнота, r – точность, tp — количество истинно положительных решений, fn — количество ложноотрицательных решений, fp — количество ложноположительных решений.

Целью работы является модификация векторного метода СИС, учитывающего частоту совместного употребления слов, которая обеспечивает достижение большего значения f-меры для тестовых данных, содержащих сущности со схожими контекстами. Для достижения цели необходимо решить следующие задачи: разработка алгоритма СИС, в котором используются данные о совместном употреблении слов и отношениях между словами; разработка программных средств, включая обучение модели word2vec [5] и реализацию алгоритма СИС, исследование разработанного алгоритма на корпусе новостных текстов.

В качестве основы был использован алгоритм, общий для подходов к СИС, использующих векторные представления [5]. Он заключается в том, что для всех ИС создаются матрицы, заполненные значениями меры tf-idf для окружающих ИС слов, на основе этих матриц обучается модель word2vec. Мера tf-idf отражает частоту совместного употребления слов, при этом учитывая, что слова, употребляемые во многих текстах корпуса, являются общеупотребительными, и для них значение меры должно быть понижено.
ИС, выделенная из текста, поданного на вход системы, связывается с вектором модели, для которого конусное расстояние [5] с входным текстом наименьшее.

Разработан новый алгоритм расчета векторов, используемых для обучения модели, путем корректировки значений tf-idf. Учитывается не только совместное употребление слов, но и то, в каких отношениях слова находятся с ИС. Значения могут быть увеличены, если рассматриваемое слово соответствует одному из сформированных правил. Зная шаблоны морфологических признаков и видов подчинительных связей в употреблении слов, которые хорошо описывают ИС, можно сформировать правила, соответствие которым является поводом для увеличения веса слова. Тогда используемая формула расчета весов принимает вид:

$$weight(t, d, D) = tfidf(t, d, D) \times koff,$$

где tf-idf(t, d, D) — значение меры tf-idf для текущего слова, t — рассматриваемое слово, в — рассматриваемый документ, D — коллекция документов, koff >= 1 — коэффициент увеличения веса слова в зависимости от правила. Величины коэффициентов определены путем экспертной оценки. Примеры правил и коэффициентов, которые были использованы в работе:

– существительное перед ИС, в том же роде, числе, падеже, что и ИС, по отношению к которому ИС в подчинительном отношении, имеет коэффициент 1.3;

– аббревиатура перед имеет ИС коэффициент 1.1;

– существительное, за которым следует аббревиатура, а затем ИС, и которое имеет тот же падеж, род и число, что и ИС, имеет коэффициент 1.2.

Для экспериментов был использован корпус из 100 текстов новостных статей на русском языке со средней длиной текста 140 слов и содержащих в среднем 2 ИС. Тексты прошли предварительную обработку (удаление служебных частей речи, частотных слов, специальных символов), с помощью инструмента stanza [6] были выделены именованные сущности и морфологические признаки и построены деревья зависимостей предложений. Для каждого из описанных алгоритмов были обучены модели, проведено тестирование и расчет f-меры для определения результатов.

В рамках работы был разработан алгоритм связывания именованных сущностей, проведены расчеты f-меры для тестовой выборки текстов. Разработанный алгоритм, использующий меру tf-idf и правила, показал лучшие результаты по сравнению с методом, использующим только tf-idf: f-мера полноты и точности результатов увеличивается на 17% (0.75 против 0.64). В дальнейшем планируется автоматизировать процесс формирования правил и провести сравнение результатов с другими алгоритмами связывания именованных сущностей.

ЛИТЕРАТУРА

1. Huang, Hongzhao & Heck, Larry & Ji, Heng. (2015). Leveraging Deep Neural Networks and Knowledge Graphs for Entity Disambiguation.
2. Bordes A., Usunier N., Garcia-Duran A., Weston J., Yakhnenko O. Translating Embeddings for Modeling Multi-relational Data. In: Proceedings of the 26th International Conference on Neural Information Processing Systems, 2013, vol. 2, p. 2787–2795.
3. Yamada I., Shindo H., Takeda H. Joint Learning of the Embedding of Words and Entities for Named Entity Disambiguation. In Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning. 2016. p. 250–259, Berlin, Germany.
4. Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003, p. 142–147.
5. Mikolov, Tomas & Chen, Kai & Corrado, G.s & Dean, Jeffrey. Efficient Estimation of Word Representations in Vector Space. Proceedings of Workshop at ICLR. 2013.
6. Stanza. [Электронный ресурс]. Режим доступа: https://stanfordnlp.github.io/stanza/.

УДК 004.89

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

ОБУЧЕНИЯ НЕЙРОННЫХ СЕТЕЙ С ЧАСТИЧНЫМ ПРИВЛЕЧЕНИЕМ УЧИТЕЛЯ НА ПРИМЕРЕ ЗАДАЧИ РАСПОЗНАВАНИЯ НОМЕРНЫХ ЗНАКОВ

В данной работе рассмотрен алгоритм детектирования и распознавания номерных знаков автомобилей с частичным привлечением учителя(semi-supervised learning). Благодаря использованию такого метода можно минимизировать потраченные человеческие ресурсы на разметку выборки данных для обучения нейронных сетей. Описанный алгоритм в данной статье подходит для детектирования и распознавания не только номерных знаков автомобилей, но и для других различных комплексных надписей.

Для обучения свёрточных нейронных сетей всегда требуется большое количество данных. Во-первых, это помогает охватить все больше и больше различных ситуаций и соответственно, оптимальным образом обучить нейронную сеть. Во-вторых, маленький набор данных может привести к переобучению нейронной сети, то есть нейронная сеть не будет работать на других изображениях, кроме обучающей выборки. В-третьих, мы сможем получить более точные метрики на тестовой выборке, ведь размер тестовой выборки зависит от размера обучающей выборки. В рамках нашей задачи мы будем работать с российскими номерными знаками транспортных средств, но, к сожалению, в интернете отсутствуют доступные выборки данных для данной страны. Чтобы решить данную проблему было решено скачать изображения с номерными знаками из интернета с помощью веб-скрапинга. С помощью python и многопроцессности менее чем за 12 часов было скачано около 95 000 изображений с автомобилями с российскими номерными знаками. В итоге мы получили изображения с номерными знаками и сами символы номерных знаков, но без расположений номерных знаков на изображениях.

Чтобы разметить такой объем данных, нужно было бы потратить сотню часов, именно поэтому для решения данной проблем мы будем пользоваться методом pseudo-labeling [1]. Суть данного метода заключается в обучении тяжелой модели с небольшим количество данных, а затем с помощью данной обученной модели размечается остальная часть данных. Из-за большего количество данных, неточность такой разметки будет нивелировано во время обучения нашей основной модели для детектирования номерных знаков. На основе выше сказанных, наш алгоритм разметки выборки данных состоит из следующих этапов:

1. Размечаем 5% от общей выборки данных (4 750 изображений)
2. Обучаем тяжелую модель для детектирования объектов, в нашем случаем мы выбрали Faster-RCNN [2] с Resnet-101.
3. Размечаем оставшуюся часть нашей выборки данных (95%) с помощью обученной модели Faster-RCNN.

Готово. Теперь можно смешать данные, размеченные руками и с помощью модели и обучить нужно модель для нашего алгоритма, в нашем случае YoloV3 with Darknet-53 backbone [10]. Итоговый алгоритм подготовки данных для детектора объектов показана на Рисунке 1.

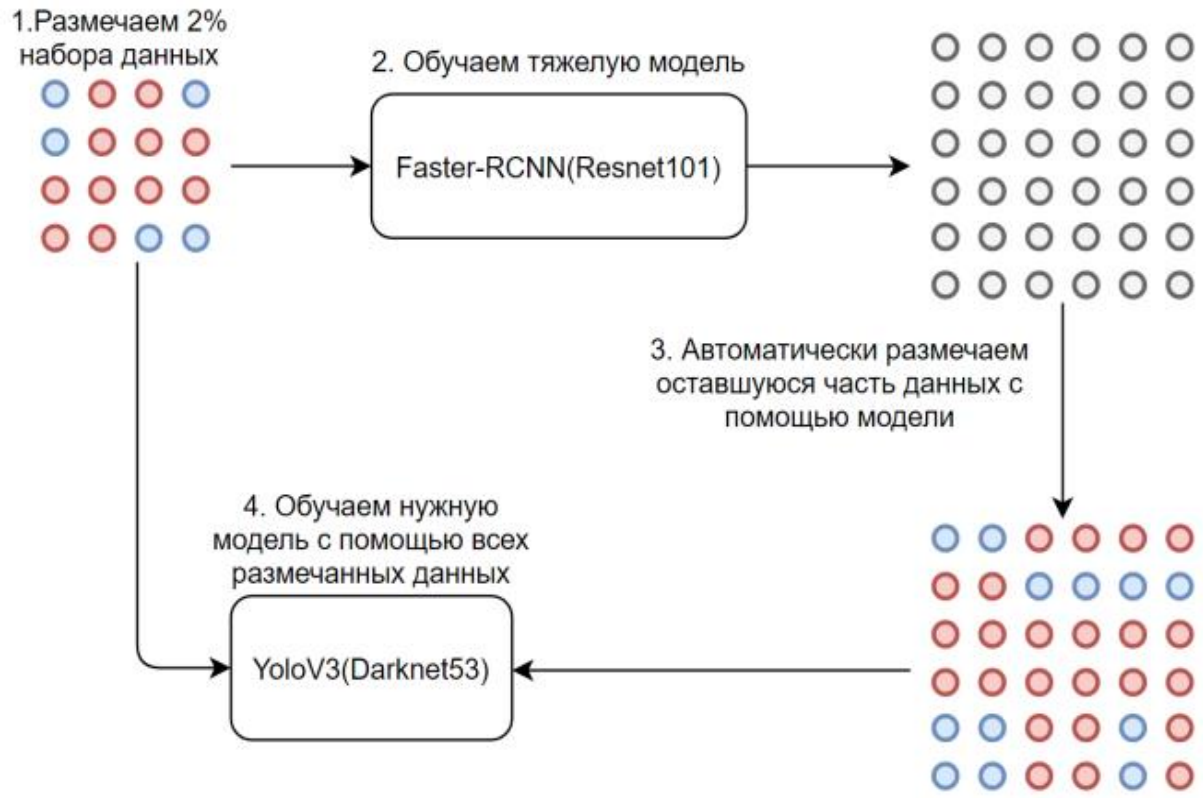


Рисунок 1 – Схема работы метода псевдо-разметки.

ЛИТЕРАТУРА

1. Arazo, E., Ortego, D., Albert, P., O'Connor, N. E., & McGuinness, K. (2020, July). Pseudo-labeling and confirmation bias in deep semi-supervised learning. In 2020 International Joint Conference on Neural Networks (IJCNN) (pp. 1-8). IEEE.
2. Ren, S., He, K., Girshick, R., & Sun, J. (2016). Faster R-CNN: towards real-time object detection with region proposal networks. IEEE transactions on pattern analysis and machine intelligence, 39(6), 1137-1149.
3. Redmon, J., & Farhadi, A. (2018). Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767.

УДК 004.896

Власенко Н. А., Дусаева А. И. (4 курс бакалавриата),
Преловский Д. С. (ассистент),
Никифоров И. В., к.т.н., доцент

РАЗРАБОТКА СИСТЕМЫ УПРАВЛЕНИЯ МАНИПУЛЯЦИОННЫМ РОБОТОТЕХНИЧЕСКИМ КОМПЛЕКСОМ НА ОСНОВЕ ПРОЦЕССОРА CORTEX-A72

За последнее время наблюдается увеличение объемов продаж электроавтомобилей в мире и в России в частности [1], что в свою очередь приводит к необходимости развития инфраструктуры для их обслуживания и зарядки [2]. Оптимизировать процесс зарядки можно при помощи манипуляционных роботизированных комплексов [3], которые способны обеспечить взаимодействие автомобиля и зарядочной станции без использования человеческих ресурсов [4]. Таким образом, целью работы является создание и разработка модели робота-манипулятора, который способен распознавать объекты – зарядочные коннекторы и двигаться по направлению к ним.

Описание предлагаемого программно-аппаратного комплекса робота-манипулятора приведено ниже.

Система включает в себя программную и аппаратную части (Рисунок 1).

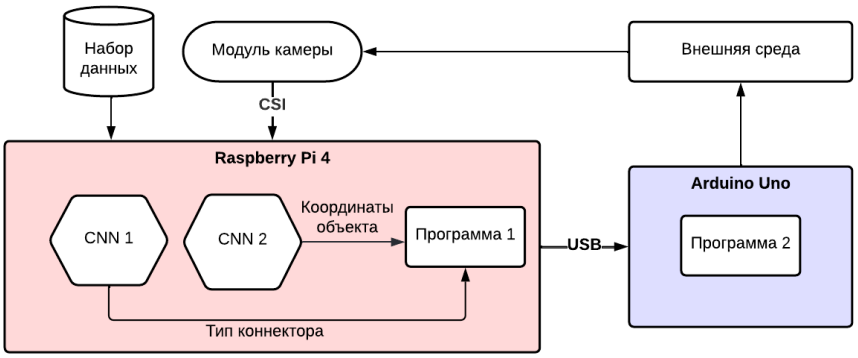


Рисунок 1 – Диаграмма развертывания разработанной системы

Аппаратная часть включает в себя:

–плату Raspberry Pi 4 [7], запускающую нейронную сеть, входными данными к которой являются фотографии, полученные с камеры Raspberry Pi v2 в режиме реального времени, а результатом - координаты необходимого объекта. Далее на их основе происходят основные вычисления, передача вектора движения на плату Arduino Uno;

–плату Arduino Uno [7], осуществляющую перемещение робота в пространстве. Она управляет пятью сервомоторами, которые составляют кинематические пары из основания и колонны, колонны и каретки, каретки и руки, руки и кисти, кисти и хвата);

К программной части относятся:

–нейронная сеть для вычисления координат объекта на полученном изображении («CNN2», Рисунок 1), реализованная при помощи открытых библиотек Tensorflow и Keras;

–нейронная сеть для определения типа коннектора на полученном изображении («CNN1», Рисунок 1);

–программный модуль для расчетов углов поворота сервомоторов («Программа 1», Рисунок 1), исполняемый на плате Raspberry Pi 4;

–модуль управления моторами («Программа 2», Рисунок 1), запускаемый на контроллере Arduino Uno;

Взаимодействие внутри программно-аппаратного комплекса происходит посредством интерфейса USB и Camera Serial Interface.

Входными данными к нейронным сетям являются фотографии, полученные с камеры, а выходными – массив координат объектов к каждому изображению и тип коннектора.

Для тренировки и тестирования нейронной сети было принято решение использовать синтетический датасет [6], сгенерированный автоматически с использованием CAD-моделей наиболее популярных зарядочных коннекторов, используемых на российском рынке электрокаров. На Рисунке 2 приведены примеры сгенерированных изображений.

Таким образом, в программе Blender был создан модуль на языке Python, который обрабатывает модели формата STL и OBJ. Выбор программы Blender обусловлен готовым API для взаимодействия с обширным внутренним функционалом (настройка положения объекта и сцены, добавление фона, изменение параметров света и т.д.).

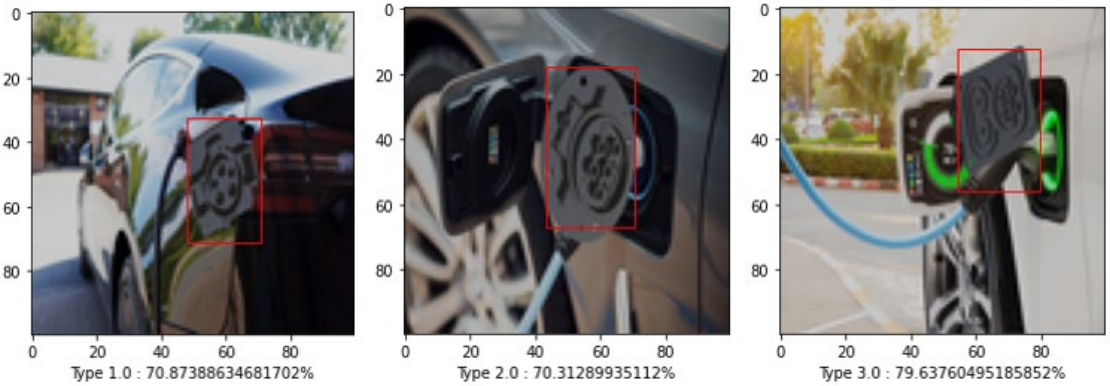


Рисунок 2 – Результаты распознавания

Результатом работы является модель робота-манипулятора, которая способна получать изображения из внешней среды, распознавать объекты и двигаться по направлению к ним. Алгоритм движения оптимизирован при помощи метода градиентного спуска [5], Точность выделения границ объектов составляет 96.41% при допустимой погрешности p = 0,05, с объемом обучающей выборки – 6000 изображений, точность классификации составляет 91.59%. На Рисунке 3 приведены результаты тренировки и работы нейронной сети по классификации сокетов.

Также была реализована программа для автоматического рендеринга датасета, включающего в себя три типа зарядочных сокетов в разных положениях и на разных фонах.

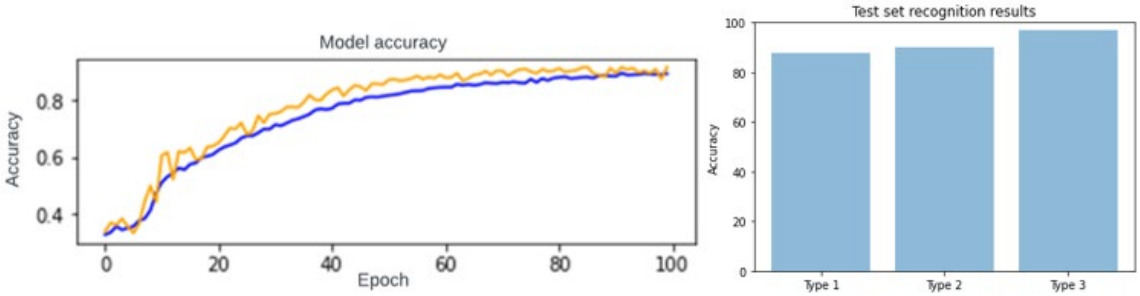


Рисунок 3 – Результаты тренировки и работы нейронной сети по классификации

ЛИТЕРАТУРА

1. Bagloee, S.A., Tavana, M., Asadi, M. et al. Autonomous vehicles: challenges, opportunities, and future implications for transportation policies. // Journal: журнал. — J. Mod. Transport, 2016. — № 24. — с. 284–303.
2. Pai Chuaychoo, Benjamas Panomruttanarug, Mario Hirz. Implementation of 3D Shape Matching for EV Charging Connector. Instrument and Control Engineers of Japan (SICE) 2021 // Conference: конференция. — 60th Annual Conference of the Society, 2021. — с. 498-501.
3. Walzel, Bernhard & Sturm, Christopher & Fabian, Jürgen & Hirz, Mario. Automated robot-based charging system for electric vehicles, Internationales Stuttgarter Symposium // статья. — 2016.
4. Harik, El H.C. Design and Implementation of an Autonomous Charging Station for Agricultural Electrical Vehicles // Conference: конференция. — Applied Sciences 11, 2021. — № 13.

5. Chernorutskiy, I., Drobintsev, P., Kotlyarov, V., Voinov, N. A New Approach to Generation and Analysis of Gradient Methods Based on Relaxation Function // Conference: конференция. — 19th International Conference on Modelling and Simulation, 2018. — с. 83-88.
6. 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.
7. П. А. Панков, И. В. Никифоров, Д. Ф. Дробинцев, Исследование эффективности использования одноплатных микрокомпьютеров в качестве обучающей платформы для работы с распределённой обработкой данных // 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. 120-121.

УДК 519.688

Демидов И. А. (2 курс магистратуры),
Ампилова Н. Б., к.ф.-м.н., доцент

АНАЛИЗ СТРУКТУРЫ ЦИФРОВЫХ ИЗОБРАЖЕНИЙ С ПОМОЩЬЮ ФРАКТАЛЬНЫХ МЕТОДОВ

Введение. Применение методов фрактального и мультифрактального анализа позволяет получать некоторые численные характеристики исследуемых изображений и использовать их как классификационные признаки. В данной работе рассматривается применение алгоритмов фрактального и мультифрактального анализа в задаче классификации снимков моря при различных погодных условиях.

Исследуемые изображения представляют собой снимки морской поверхности, сделанные под несколькими углами (10°–50°) при спокойной и ветреной погоде. В качестве методов получения характеристик рассматриваемых изображений были выбраны: вычисление спектра размерностей Реньи и метод расчета фрактальной сигнатуры.

Спектры Реньи. При вычислении спектра обобщенных размерностей Реньи рассматривается разбиение на $N(\varepsilon)$ ячеек размера ε. На данном разбиении определяется некоторая мера $\{p_i(\varepsilon)\}$, а также сумма моментов выбранной меры $S(q,\varepsilon) = \sum_{i=1}^{N(\varepsilon)} p_i^q\ (\varepsilon)$, где параметр $q\ \in R$. Спектр обобщенных размерностей Реньи определяется выражением:

$$D_q = \lim_{\varepsilon \to 0} \frac{1}{q-1} \frac{lnS(q,\varepsilon)}{ln\varepsilon}$$

В данной работе были использованы два способа определения меры. Первый способ задает меру ячейки как отношение суммы интенсивностей пикселей в ячейке к сумме интенсивностей пикселей всего изображения. Полученные результаты показали, что разделить исследуемые классы не удается, так как графики лежат в пересекающихся интервалах. При втором способе мера вычислялась как отношение суммы интенсивности пикселей заданного диапазона в ячейке разбиения к сумме интенсивностей пикселей в заданном диапазоне во всем изображении. Диапазон интенсивностей является параметром. Такой выбор меры основан на представлении сигнала с большей амплитудой пикселями большей интенсивности. Следовательно, чем больше светлых точек на изображении, тем более беспокойному состоянию поверхности данное изображение соответствует.

Метод фрактальной сигнатуры. Метод фрактальной сигнатуры [1] позволяет вычислить аналог емкостной размерности, а именно размерность Минковского, для полутоновых изображений. Он основан на технике "построения покрывала" (или построения так называемого δ-параллельного тела) для поверхности функции градации серого,

построенной над полутоновым изображением путем добавления третьей координаты для пикселя с координатами (i, j), значение которой равно его интенсивности [1-3].

Эта поверхность вполне характеризует исходное изображение. Покрывала строятся итеративно для нескольких значений параметра $\delta = 0, 1, 2 \ldots$ Нулевое значение соответствует самой поверхности. Для каждого значения параметра δ вычисляется объем полученного тела, заключенного между верхним и нижним покрывалами, а затем приближенно вычисляется площадь поверхности A_δ. Размерность Минковского определяется через полученное значение площади. В прикладных задачах часто достаточно использовать только A_δ. В частности, удобным признаком является так называемый вектор характеристик, который строится в осях $(\ln \delta,\ \ln A_\delta)$ и отражает изменение площади в зависимости от итерации.

Результаты экспериментов. На приведенном ниже (Рисунок 1) представлены графики полученных спектров обобщенных размерностей Реньи.

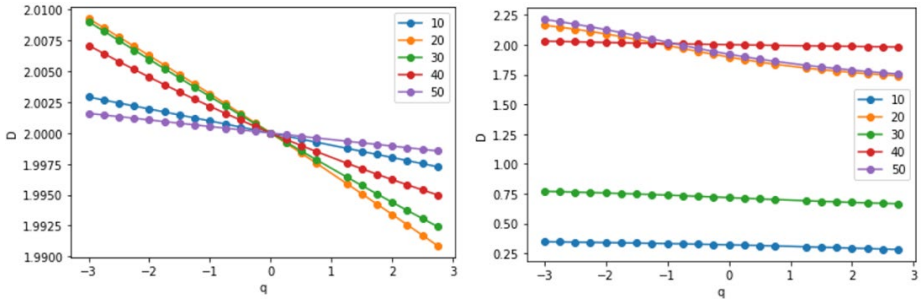


Рисунок 1 – Спектры Реньи для изображений в ветреную и безветренную погоду.

Можно заметить, что значения для двух разных категорий снимков (в ветреную и в спокойную погоду) лежат в различных диапазонах. Для изображений первого класса почти все спектры лежат в очень маленькой окрестности 2, диапазон значений для второго класса отличается на 2 порядка.

Наиболее явное разделение графиков заметно при применении второго метода. На Рисунке 2 представлены графики векторов характеристик для двух рассматриваемых категорий изображений. В отличии от спектра Реньи, в данном случае можно наблюдать более строгое и точное разделение исследуемых изображений на 2 класса.

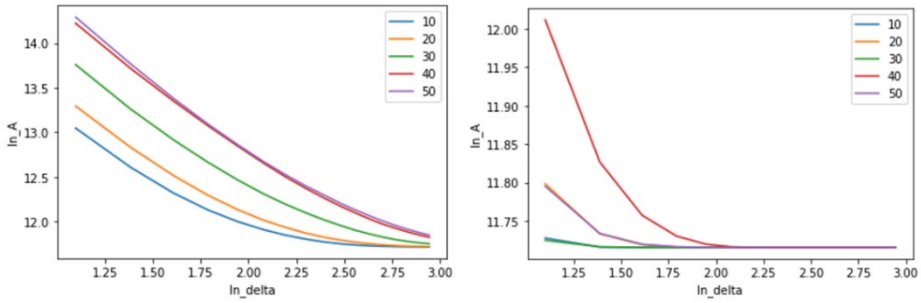


Рисунок 2 – Графики векторов характеристик в ветреную и безветренную погоду.

Заключение. Проведенные эксперименты показывают, что методы анализа, основанные на вычислении фрактальных характеристик, позволяют получить достаточно надежные классификационные признаки. Наиболее сложным вопросом является подбор меры на изображении. Наши эксперименты показывают, что естественным подходом к решению задачи является подходящая интерпретация физического процесса в терминах получаемых изображений. Для более качественной и точной классификации исследуемых изображений необходимо, в некоторых случаях, применять несколько методов расчета фрактальных характеристик с подбором соответствующих параметров.

ЛИТЕРАТУРА

1. Tang Y. Y., Hong Ma, Dihua Xi, Xiaogang Mao, C. Y. Suen. Modified Fractal Signature (MFS): A New Approach to Document Analysis for Automatic Knowledge Acquisition. — IEEE Transactions On Knowledge and Data Engineering, vol.9, no. 5, 1997, pp. 747-762.

2. Falconer, K.J. Fractal geometry: mathematical foundations and application. – John Wiley & Sons, 1990.
3. .Б. Ампилова, И.П. Соловьев. Алгоритмы фрактального анализа изображений. Компьютерные инструменты в образовании. N 2, 2012 с.19-24.

УДК 004.93

Зиянгиров А.И. (1 курс магистратуры),
Петров О.Н., к.т.н., доцент

ОСОБЕННОСТИ ПОДГОТОВКИ ДАННЫХ ПРИ РАСПОЗНАВАНИИ ТРЕХМЕРНЫХ ОБЪЕКТОВ

Целью работы является описание методов, который позволят представить цифровой трехмерный объект и пространственную карту в виде, удобном для последующей обработки.

Особый интерес использования трехмерных моделей представляют задачи распознавания. Сложность использования двумерных изображений, которые представлены в виде пикселей, обусловлена ракурсом и освещенностью. Трехмерные объекты не подвержены этим внешним факторам, а наличии объемности позволяет более эффективно использовать особенности габаритов модели.

Метод сканирования можно свести к задаче вычисления точек пересечения векторов с поверхностями сканируемого объекта. Если построить сферу вокруг модели, центр которой будет совпадать с геометрическим центром объекта, то вектора могут быть заданы с помощью двух точек:

1. Точка на поверхности сферы
2. Центр сферы

Направление будет задано от точки на сфере к центру сферы.

Для описания точек сферы следует выбрать систему координат. В данном случае лучший выбор – сферическая система координат, поскольку с помощью неё легко описать сферу, которая будет построена вокруг объекта. К тому же сферическая система координат легко приводится к декартовой и наоборот.

Сканирование пространственной карты будет аналогично сканированию объекта, однако учитывая, что пространственная карта представляет собой множество объектов, то следует учитывать угол обзора. Алгоритм сканирования тогда можно описать так:

1. Просчитываются сферические координаты точки, описывающие сферу, которая строится вокруг объекта на карте и центр которой совпадает с центром этого объекта.
2. Вычисляются декартовы координаты лишь тех векторов, которые противоположно направлены вектору обзора. Декартовы координаты точек сферы вычисляются так же, как и при сканировании трехмерных моделей.
3. Вычисляются координаты пересечения вектора, выходящего из центра сферы с поверхностью объекта
4. Алгоритм повторяется для всех векторов, начиная с пункта 3.

В пункте 2 вычисляются лишь те вектора, которые образуют тупой угол с направляющим вектором обзора.

Для реализации метода сканирования можно воспользоваться платформой Unity, которая имеет графический движок для визуальной демонстрации метода, имеет совместимость с языком C# и его библиотеками для работы с файловой системой для записи координат точек.

Демонстрация этапа сканирования представлена на Рисунке 1 и Рисунке 2. Здесь в качестве моделей выбраны голова человека и модель автомобиля. Множество шаров на поверхности объекта представляют собой визуализацию точек, который были получены в результате работы алгоритма.

Реализованные методы соответствуют цели работы и рассчитывают координаты точек пересечения. Программа сканирования является подготовительным этапом решения задачи классификации – сбор и обработка входных данных для построения обучающей выборки.

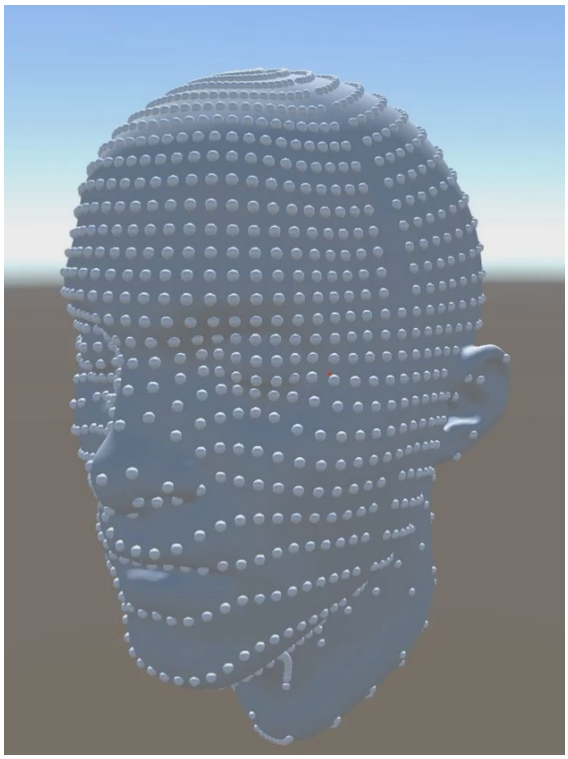


Рисунок 1 – Демонстрация сканирования лица человека.

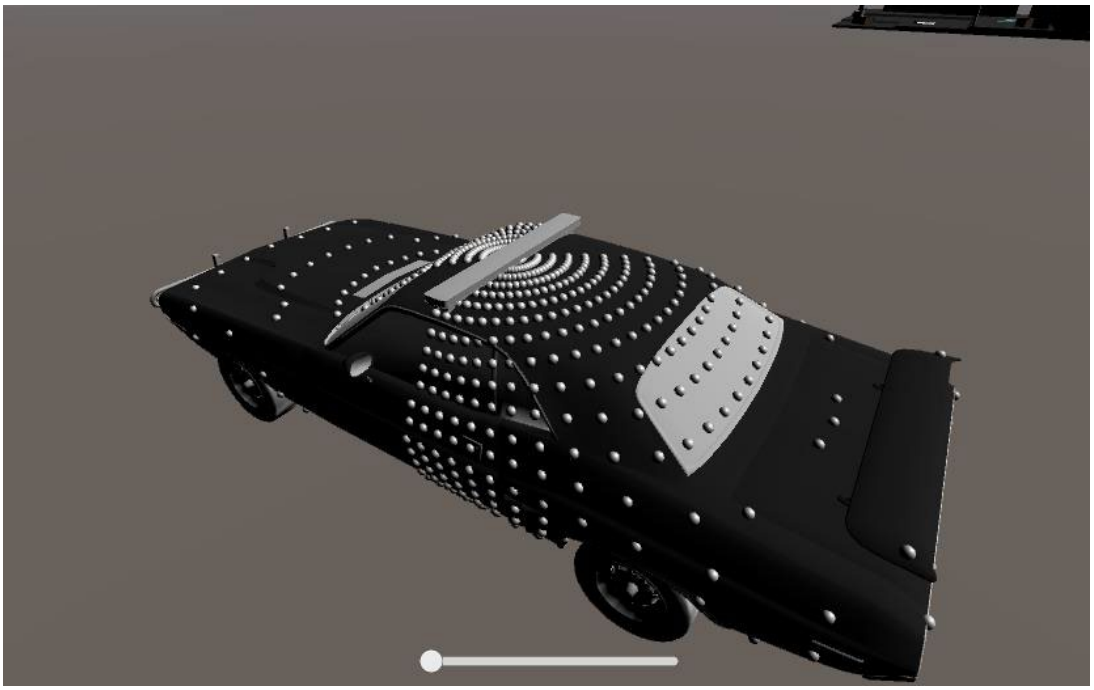


Рисунок 2 – Демонстрация сканирования модели автомобиля.

ЛИТЕРАТУРА

1. Бенджио, Гудфеллоу, Курвилль: Глубокое обучение. Издательство ДМК Пресс. Год выпуска : 2017
2. Официальная документация для разработчиков Unity [Электронный ресурс] :портал. – Режим доступа: https://unity.com/, свободный.

УДК 004.891

Калимуллина Р. Р. (2 курс магистратуры),
Большаков А. А., д.т.н., профессор,
Никитина М. А., д.т.н.

РАЗРАБОТКА СИСТЕМЫ ПОДДЕРЖКИ ПРИНЯТИЯ РЕШЕНИЙ ДЛЯ АВТОМАТИЗИРОВАННОГО ОПРЕДЕЛЕНИЯ НАЛИЧИЯ ФАЛЬСИФИКАЦИИ МЯСА

Качество питания и возникающие заболевания тесно связаны, а существующие технологии производства колбасных изделий и полуфабрикатов подразумевают дифференциацию компонентов растительного и животного происхождения, имеющие мало общего с потребностями человека. Эти фальсификаты требуют идентификации для дальнейшего улучшения качества потребляемых продуктов. Процесс распознавания заменителей или имитаторов мяса осуществляется технологами вручную, поэтому на ее результат непосредственно влияет человеческий фактор. Поэтому актуальна автоматизация соответствующей ГОСТ Р 53213 и ГОСТ Р 53222 ручной процедуры [1] определения наличия фальсификата по входному изображению срезов мясной продукции по ряду его характеристик: форма, цвет, размер и др.

Целью настоящего исследования является повышение результативности процесса идентификации наличия фальсификата в мясной продукции на основе автоматизации обработки изображений.

Для достижения этой цели необходимо решить следующие задачи:

– системный анализ процесса идентификации наличия фальсификата в мясной продукции;

– обзор методов и алгоритмов распознавания изображений;

– разработка алгоритма обработки изображений для автоматизированного определения наличия фальсификации мяса;

– построение системы поддержки принятия решений для определения наличия фальсификации мяса;

– разработка и программная реализация алгоритма задачи цветовой сегментации на основе оптимизации палитры;

– разработка и программная реализация алгоритма задачи аппроксимации границ выделенных на изображении областей;

– разработка и программная реализация алгоритма задачи определения размера областей фальсификата в мясе;

– разработка и программная реализация алгоритма задачи определения формы частиц фальсификата в мясе;

– разработка и программная реализация алгоритма задачи определения цвета частиц фальсификата в мясе.

Решен ряд из вышеперечисленных задач. Так, нами осуществлен системный анализ процесса идентификации наличия фальсификата, обзор методов и алгоритмов распознавания изображений, а также выполнена разработка и программная реализация алгоритма задачи цветовой сегментации на основе оптимизации палитры [2].

Кроме этого, на основе методологии построения экспертных систем предложен прототип системы поддержки принятия решений для встраивания в автоматизированную систему определения наличия фальсификата [3].

Предлагаемая архитектура системы поддержки принятия решений для идентификации наличия фальсификата в образцах мясной продукции (см. Рисунок 1) включает управляющий модуль, который управляет подсистемами, организует взаимодействие с использованием соответствующего интерфейса с администратором и авторизированным пользователем. Система включает информационную подсистему в виде экспертной системы, а также подсистемы для поддержки принятия решений (ППР).

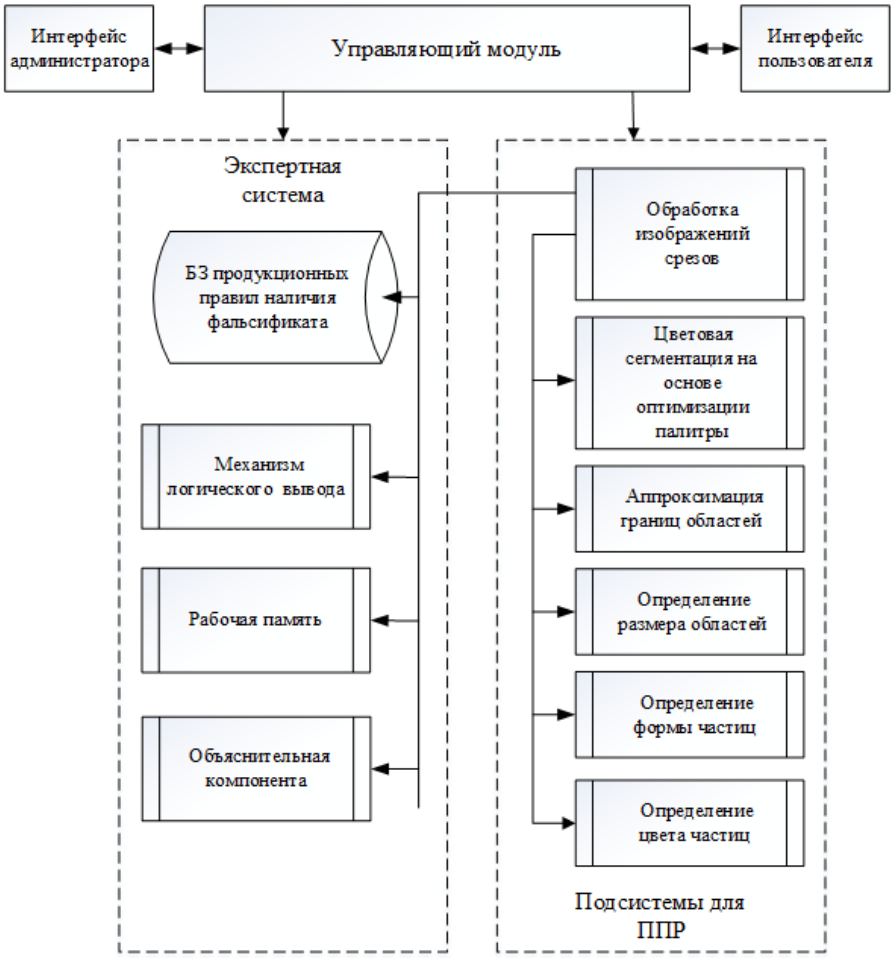


Рисунок 1 – Архитектура системы поддержки принятия решений по определению наличия фальсификации мяса.

Последовательное выполнение вышеуказанных этапов позволяет получить обоснованное решение о наличии определенных видов фальсификатов мясной продукции.

Разработка прототипа экспертной системы выполнена на языке программирования искусственного интеллекта Пролог. Для решения задачи оптимизации аппроксимации полихромного изображения разработана программа на языке программирования C++ [4]. С ее использованием проведен комплекс вычислительных экспериментов для исследования характера зависимости сходимости процесса оптимизации. Для дальнейших исследований также будет использоваться язык программирования C++.

ЛИТЕРАТУРА

1. Хвыля С. И., Пчелкина В. А., Бурлакова С. С. Стандартизованные гистологические методы оценки качества мяса и мясных продуктов // Все о мясе, 2011. – № 6. – С. 32-35.
2. Большаков А. А., Калимуллина Р. Р., Никитина М. А. Системный анализ и автоматизация процесса обнаружения фальсификата в мясе и мясной продукции на основе гистологического метода // Вестник технологического университета, 2021. – Т.24, №1. – С. 71-78. ISSN 1998-7072.
3. Bolshakov A. A., Nikitina M. A., Kalimullina R. R. Intelligent System for Determining the Presence of Falsification in Meat Products Based on Histological Methods // Studies in Systems, Decision and Control. Society 5.0: Cyberspace for Advanced Human-Centered Society. Springer Nature Switzerland AG 2021. 2021. Vol. 333. pp. 179-201. https://doi.org/10.1007/978-3-030-63563-3_12. ISSN 2198-4182. Scopus.
4. Калимуллина Р. Р., Большаков А. А. Программа минимизации палитры полихромного изображения // Свидетельство о государственной регистрации программы для ЭВМ № 2020617632. заявл. 25.06.2020 г.; зарег. 08.07.2020.

УДК 004.93

Клементьев М.М. (1 курс магистратуры),
Петров О.Н., к.т.н., доцент

ИНТЕЛЛЕКТУАЛЬНЫЙ АНАЛИЗ ПОКАЗАНИЙ ДАТЧИКОВ ПРИ ДЕФОРМАЦИОННОМ МОНИТОРИНГЕ

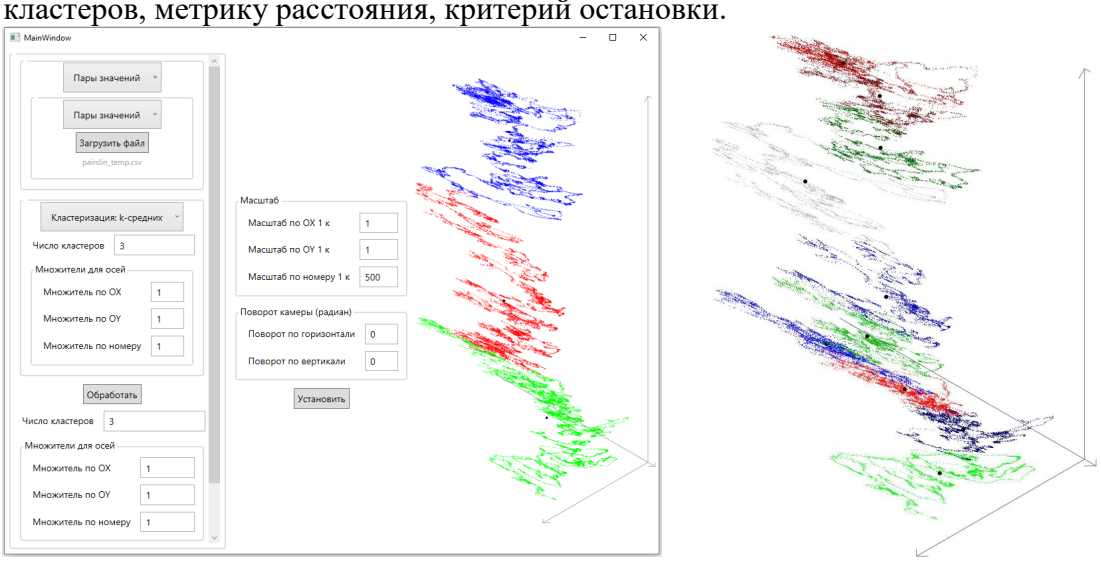
Целью работы является разработка программных средств оценки показаний датчиков и вывода полученных оценок для выявления зависимостей в входных данных.

Результат работы может использоваться при создании баз знаний о воздействии нагрузок на силовые конструкции техники и сооружений, а также при планировании их обслуживания, ремонта, других этапов жизненного цикла. Кроме того, возможно использование накопленных данных для повышения эксплуатационных качеств техники и сооружений на этапе проектирования оных.

Анализ получаемых данных сводится к выявлению закономерностей, связывающих деформации, нагрузки и время. С точки зрения математического аппарата, задача представляет собой кластеризацию трехмерных векторов (Рисунок 1). Применены 3 алгоритма кластеризации:

1. Метод k-средних
2. Кластеризация на основе нечеткой логики
3. Кластеризация при помощи искусственной нейронной сети Кохонена.

Метод k-средних представляет собой метод кластеризации, основанный на группировке точек относительно центров тяжести получаемых групп. Для этого метода требуется задать число кластеров, метрику расстояния, критерий остановки.



(а) (б)

Рисунок 1 – Примеры отображения разного количества кластеров с пользовательским интерфейсом (а) и без пользовательского интерфейса (б)

Кластеризация на основе нечеткой логики представляет собой метод кластеризации, основанный на определении центров кластеризации для последующей группировки точек вокруг них. Для этого метода требуется задать число кластеров, скорость обучения, метрику расстояния, критерий остановки.

Кластеризация при помощи искусственной нейронной сети Кохонена, метод кластеризации, основанный на использовании упомянутой нейронной сети. Сеть перемещает центры кластеров к скоплениям точек при помощи «слоя Кохонена». Метод предусматривает кластеризацию не по положению в пространстве, а по углу относительно начала координат. Для этого метода требуется задать число кластеров (часть из которых могут оказаться пустыми), скорость обучения, метрику расстояния, критерий остановки.

Реализация этих методов не имеет специальных требований к языкам программирования и платформам. В проделанной работе методы реализованы на языке программирования C#, разработан пользовательский графический интерфейс с использованием средств языка.

ЛИТЕРАТУРА

1. Трофимов, В. В. Алгоритмизация и программирование : учебник для вузов / В. В. Трофимов, Т. А. Павловская ; под редакцией В. В. Трофимова. — Москва : Издательство Юрайт, 2021. — 137 с. — (Высшее образование). — ISBN 978-5-534-07834-3. — Текст : электронный // ЭБС Юрайт [сайт]. — URL: https://urait.ru/bcode/471125 (дата обращения: 29.04.2021).
2. Кубенский, А. А. Функциональное программирование : учебник и практикум для вузов / А. А. Кубенский. — Москва : Издательство Юрайт, 2021. — 348 с. — (Высшее образование). — ISBN 978-5-9916-9242-7. — Текст : электронный // ЭБС Юрайт [сайт]. — URL: https://urait.ru/bcode/469863 (дата обращения: 29.04.2021).

УДК 004.93

Колесов А.Д. (1 курс магистратуры),
Петров О.Н., к.т.н., доцент

ПРИМЕНЕНИЕ МЕТОДОВ ИЗВЛЕЧЕНИЯ СКРЫТЫХ ЗНАНИЙ ПРИ АНАЛИЗЕ БЫСТРО ИЗМЕНЯЮЩИХСЯ РЫНКОВ

В настоящее время существует потребность в прогнозной аналитике быстро изменяющихся рынков на основе больших объемов данных в интернете. Эффективным инструментом для анализа информации с сайтов в сети являются программы-парсеры, которые автоматизирует сбор, обработку и накопление открытых данных. Целями парсинга с точки зрения аналитики являются исследования рынка, принятие решений по управлению ценообразованием, анализ и визуализация динамических ситуаций.

Целью работы является проектирование и реализация программы-парсера автоматического сбора и анализа бизнес-информации о товарах быстро изменяющихся рынков. В качестве сферы быстро изменяющихся рынков выбран сегмент смартфонов, поскольку лидирующие места на данном рынке постоянно занимают разные компании и модели телефонов. Рассматриваемыми интернет-площадками являются Ozon и DNS как одни из наиболее популярных в российском сегменте рынка мобильных устройств.

Программный комплекс включает в себя программу-парсер и программу аналитики. Общая структура программного комплекса приведена на Рисунке 1.

Рисунок 1 – Структура программного комплекса

Программа анализа предоставляет удобный инструмент для поиска, аналитики отдельных моделей и поиска зависимостей изменений на рынке при появлении новых смартфонов.

В качестве измененяемых параметров для анализа используются следующие характеристики: цена, рейтинг и количество отзывов. Для эффективного поиска зависимостей строятся графики изменения параметров моделей за последние дни.

Основной функцией программного комплекса является поиск новых смартфонов на рынке и исследование реакции рынка на выход новой модели. Программа находит похожие на новинку телефоны по техническим характеристикам и строит две линии тренда по анализируемому параметру: значения параметра до появления новинки и после ее появления.

Линия тренда является инструментом технического анализа, который позволяет предсказать тенденцию направления движения исследуемого параметра во времени. Таким образом, пользователь может определяет зависимости реакции рынка на выход новой модели, а также оценивать тенденции изменения решений компаний-производителей и ответной реакции пользователей продукта.

На Рисунке 2 тренд представлен в виде линейной зависимости исследуемого параметра от даты сбора информации.

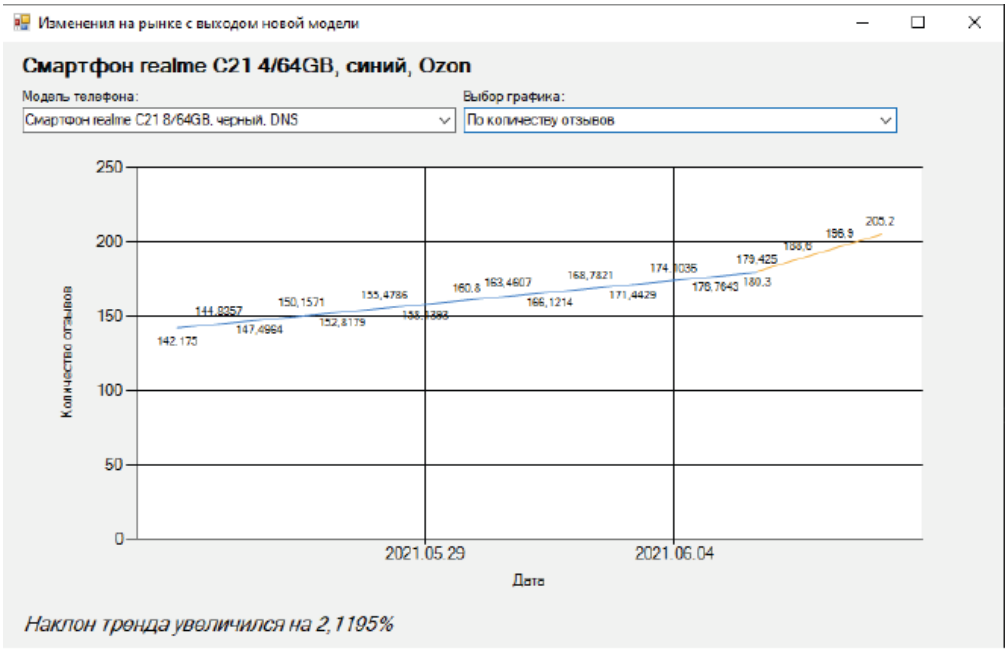


Рисунок 2 – Пример изменения линии тренда

Моменты времени, в которых существенно меняется наклон линии тренда, представляют особый интерес. К таким событиям относятся появления новых моделей конкурирующих компаний-производителей, скидки, анонсы или распродажи.

Практическое использование программы-парсера позволит выполнять автоматизированный сбор данных с разных категорий быстро изменяющихся рынков, представлять результаты в удобном структурированном виде для дальнейшей визуализации и проведении прогнозного анализа.

ЛИТЕРАТУРА

1. Суханов, А. А., Маратканов, А. С. Анализ способов сбора социальных данных из сети Интернет // International scientific review. – 2017. - № 1 (32). URL: https://cyberleninka.ru/article/n/analiz-sposbov-sbora-sotsialnyh-dannyh-iz-seti-internet.
2. Маккинни, У. Python и анализ данных / У. Маккинни ; пер. с англ. А. А. Слинкина. – 2-е изд., испр. и доп. – Москва: ДМК Пресс, 2020. – 540 с. – ISBN 978-5-97060-590-5.

УДК 004.051

Королев Д.О. (4 курс бакалавриата),
Малеев О.Г., к.т.н., доцент

ПРИМЕНЕНИЕ БИНАРНЫХ НЕЙРОННЫХ СЕТЕЙ ДЛЯ РАСПОЗНАВАНИЯ ИЗОБРАЖЕНИЙ

Целью работы является исследование эффективности применения бинарных нейронных сетей в задачах распознавания изображений и проектирование модели бинарной нейронной сети для распознавания изображений.

В бинарных нейронных сетях в качестве выходов нейронов и/или весов используются только двоичные значения (-1, 1). Такой подход уменьшает объем памяти, занимаемый весами, а также обеспечивает очень эффективную аппаратную реализацию, поскольку 32-битные операции умножения и накопления, которые вычисляют скалярное произведение между действительными выходными сигналами нейронов и весами синапсов, теперь могут быть заменены поразрядными логическими операциями XNOR и операциями подсчета накопления «popcount» (подсчета количества единиц) [1].

В областях классификации изображений и обнаружения объектов сверточные нейронные сети оказались надежными и достаточно эффективными и поэтому широко применяются на практике [4], однако наличие ресурсоемких вычислений и значительный объем памяти, занимаемый моделью, ограничивает их практическое применение на устройствах с ограниченными ресурсами. Применение бинарных сверточных нейронных сетей позволяет обойти это ограничение и получить большую эффективность на мобильных и встроенных устройствах.

Однако, несмотря на значительный прогресс, который был достигнут в развитии бинарных нейронных сетей, существующие методы бинаризации могут приводить к значительному падению точности [1], вызванному потерей информации как при прямом, так и при обратном распространении.

Для увеличения точности бинарных моделей при прямом распространении могут быть применены методы, основанные на увеличении разнообразия карт признаков сверточных слоев путем добавления ярлыков полной точности к двоичным активационным значениям [2], что позволяет достичь значительных улучшений в точности. Но из-за добавления операций с плавающей запятой эффективность такого метода неизбежно хуже, чем у ванильных бинарных нейронных сетей.

При обратном распространении для аппроксимации градиентов наиболее распространёнными в существующих решениях являются функции Identity и Hardtanh [1]. Функция Identity напрямую передает информацию о градиенте и полностью игнорирует эффект бинаризации. Функция Hardtanh может передавать информацию о градиенте только внутри интервала отсечения. Обе этих функции теряют информацию о градиенте и снижают точность на практике.

Особенностью метода, используемого в работе, является наличие операций балансировки и нормализации весов полной точности [3], изменяющих распределение данных перед бинаризацией для прямого распространения.

Для обратного распространения в предложенном методе используется «оценщик затухания ошибок», применяющий двухэтапный подход к аппроксимации градиентов [3]. На первом этапе функция аппроксимации имеет вид функции Identity. При переходе ко второму этапу значение отсечения постепенно уменьшается от большого числа до единицы. При этом форма функции аппроксимации изменяет свой вид от функции Identity к форме Hardtanh. Отсутствие отсечения информации о градиенте на первом этапе обеспечивает возможность обновления на ранней стадии обучения. На втором этапе значение интервала отсечения сохраняется равным единице, а форма функции аппроксимации постепенно принимает вид формы функции знака, что обеспечивает согласованность прямого и обратного

распространения. Функция аппроксимации g(x) для обратного распространения определяется как:

$$g(x) = k \tanh tx \tag{1}$$

Где k и t – управляющие переменные, изменяющиеся в процессе обучения, определяемые как:

$$t = T_{\min} 10^{\frac{i}{N} \times \log \frac{T_{\max}}{T_{\min}}} \tag{2}$$

$$k = \max(\frac{1}{t}, 1) \tag{3}$$

Где i – текущая эпоха;
N – количество эпох;
$T_{\min}$ и $T_{\max}$ – параметры, равные 10^{-1} и 10^{1}.

На Рисунке 1 представлено изменение формы функции аппроксимации для каждого этапа.

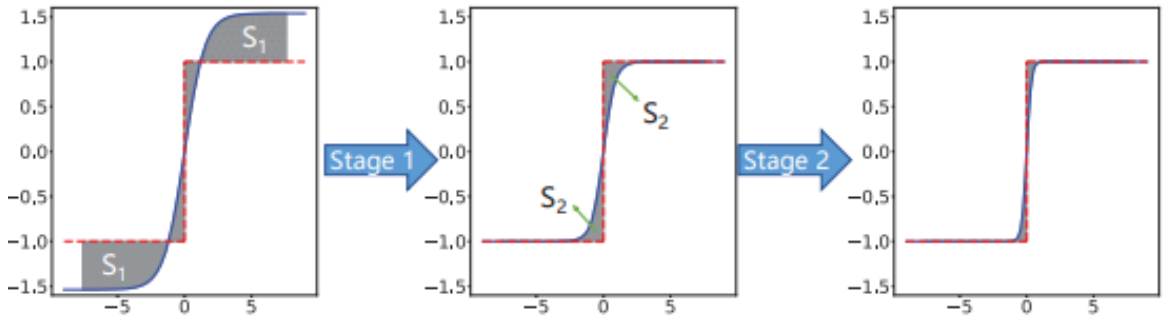


Рисунок 1 – Работа алгоритма обратного распространения.

В Таблице 1 представлены результаты сравнения модели сверточной нейронной сети полной точности (без применения бинаризации) и бинарной модели для задачи определения наличия лица человека на изображении, обучение производилось на датасетах CIFAR-10 и CELEBA. Тестирование производительности выполнялось на смартфоне с процессором MediaTek Helio P22 MT6762 (4 x Cortex-A53 2000 МГц, 4 x Cortex-A53 1500 МГц).

Таблица 1 – Результаты работы моделей.

	Точность, %	**Время выполнения, мс**	**Размер, Мб**
Модель полной точности	98,456	776	52,36
Бинарная модель	97,36	138	4,7

ЛИТЕРАТУРА

1. Binary neural networks: A survey / HaotongQin, RuihaoGong, XianglongLiu [и др.] – Текст : электронный // CoRR. – 2020. – Vol. abs/2004.03333. – URL: https://arxiv.org/pdf/2004.03333.pdf (дата обращения: 20.03.2021).
2. Bi-Real Net: Enhancing the Performance of 1-bit CNNs With Improved Representational Capability and Advanced Training Algorithm / Zechun Liu, Baoyuan Wu, Wenhan Luo [и др.] – Текст : электронный // CoRR. – 2019. – Vol. abs/1811.01335. – URL: https://arxiv.org/pdf/1811.01335.pdf (дата обращения: 20.03.2021).
3. Forward and Backward Information Retention for Accurate Binary Neural Networks / Haotong Qin, Ruihao Gong, Xianglong Liu [и др.] – Текст : электронный // CoRR. – 2020. – Vol. abs/1909.10788. – URL: https://arxiv.org/pdf/1909.10788.pdf (дата обращения: 20.03.2021).
4. Xiaofan Lin. Towards Accurate Binary Convolutional Neural Network / Xiaofan Lin, Cong Zhao, Wei Pan. – Текст : электронный // CoRR. – 2017. – Vol. abs/1711.11294. – URL: https://arxiv.org/pdf/1711.11294.pdf (дата обращения: 20.03.2021).

УДК 004.032.26

Кузнецова Ю.А. (4 курс бакалавриата),
Малеев О.Г., к.т.н., доцент

РАЗРАБОТКА И ПРИМЕНЕНИЕ НЕЙРОННОЙ СЕТИ ДЛЯ МЕТЕОПРОГНОЗА

Рассмотрена применимость нейронных сетей на основе LSTM-архитектуры для решения задач прогнозирования метеорологических элементов погоды.

Задача прогнозирования широко распространена в области метеорологии. Метеорологи активно используют системы определения состояния и прогноза погоды на основе спутниковых снимков. Данные исследования позволяют получить более точную информацию на территории большой протяженности, но для конкретной местности, не всегда точны. В связи с чем наиболее актуальным становится разработка математических моделей прогнозирования для портативных метеостанций [1].

В исследовании использовалась LSTM модель с 3 слоями, 100, 50 и 25 нейронами соответственно, в которой слои идут последовательно друг за другом. Нейронная сеть реализована с помощью фреймворка TensorFlow, библиотеки Pandas на языке программирования Python. Также, для повышения производительности был использован оптимизатор RMSprop.

Оптимизатор RMSprop похож на алгоритм градиентного спуска с импульсом. Оптимизатор ограничивает колебания в вертикальном направлении, таким образом, мы можем увеличить скорость обучения, и наш алгоритм может делать большие шаги в горизонтальном направлении, сходясь быстрее. Разница между RMSprop и градиентным спуском заключается в том, как рассчитываются градиенты [2].

Для обучения нейронной сети нам было необходимо использовать данные за несколько лет, в открытом доступе не нашлось подходящего набора данных, поэтому было принято решение сформировать его самостоятельно, используя VisualCrossing API [3].

Были выбраны следующие параметры: город (Санкт-Петербург); временной интервал (10 лет, наблюдения регистрируются каждый час); метеорологические элементы (температура воздуха, точка росы, относительная влажность воздуха, атмосферное давление, скорость ветра, осадки).

Данные уже представлены в числовом виде, поэтому нам не нужно векторизовать их, однако все временные последовательности в данных имеют разный масштаб (например, температура находится в диапазоне между –35 и +35, а атмосферное давление, измеряемое в миллибарах, изменяется в значениях, близких к 1000 и т.д.). Для нормализации временных последовательностей независимо друг от друга (чтобы все они состояли из небольших по величине значений примерно одинакового масштаба) была создана функция-генератор.

Количество прошедших дней, которые учитываются в прогнозе – 28, шаг отбора образцов из наблюдений – 1, количество эпох – 20, целью являются следующая неделя в будущем, т.е. предсказываем на 7 дней вперед (почасовой прогноз).

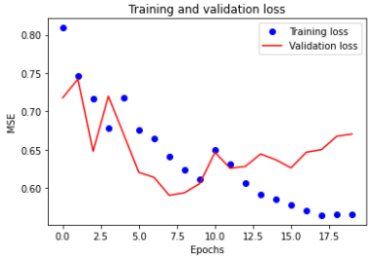


Рисунок 1 – График зависимости потерь обучающей и проверочной выборок от количества эпох.

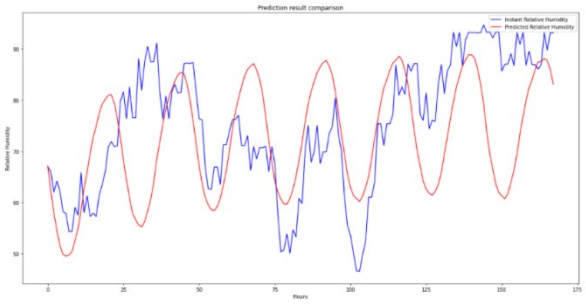


Рисунок 2 – Сравнение настоящей и предсказанной зависимости значения относительной влажности воздуха от времени.

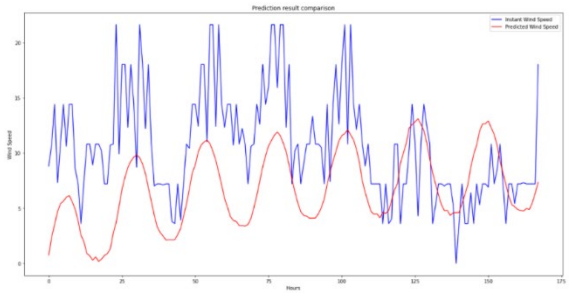


Рисунок 3 – Сравнение настоящей и предсказанной зависимости значения скорости ветра от времени.

Наша модель смогла выделить дневные циклы изменения силы ветра, влажности, точки росы и температуры. Значения осадков и давления оставляет практически без изменений.

Применяемые метрики качества и оптимизатор настраивают модель таким образом, чтобы уменьшать обобщенную ошибку по всем измерениям, а не по какому-то конкретному параметру (например, температура). Для примера рассмотрим предсказание только для значений температуры на день вперед.

Оценим результаты предсказаний:

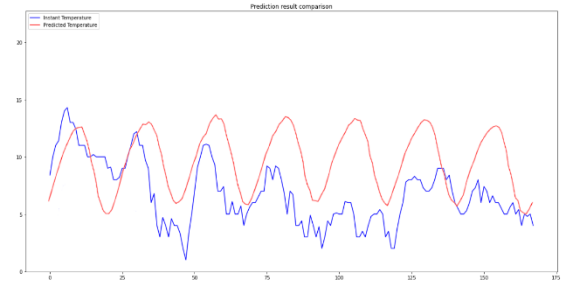


Рисунок 4 – Сравнение настоящей и предсказанной зависимости значений температуры воздуха от времени.

В результате, модель в среднем ошибается на 35-40% когда мы предсказываем все 6 показателей и если один, то на 20-25%. Модель также показала, что она умеет искать периодичность в данных и при этом умеет предсказывать сезонный тренд.

ЛИТЕРАТУРА

1. Гринченко Н.Н., Потапова В.Ю., Тарасов А.С. Алгоритмы прогнозирования погодных условий в системах сбора и обработки метеорологических данных. // Известие Тульского государственного университета. Технические науки. 2018. №2. – С. 113-119.
2. Gradient Descent and RMSprop Optimizers. [Электронный ресурс]. Режим доступа: https://towardsdatascience.com/a-look-at-gradient-descent-and-rmsprop-optimizers-f77d483ef08b
3. VisualCrossing API. [Электронный ресурс]. Режим доступа: https://www.visualcrossing.com/weather-api

УДК 004.85

Куксов Г.В. (4 курс бакалавриата),
Леонтьева Т.В., к.т.н., доцент

ПРИМЕНЕНИЕ ОБУЧЕНИЯ С ПОДКРЕПЛЕНИЕМ, ОСНОВАННЫМ НА ЛЮБОПЫТСТВЕ С ПОМОЩЬЮ САМОКОНТРОЛИРУЕМОГО ПРОГНОЗА В ACTION ИГРЕ

Целью работы является разработка отзывчивых интеллектуальных виртуальных персонажей в Action игре, используя обучение с подкреплением (Reinforcements Learning) с применением подхода, предложенным Диком Патаком и его коллегами из Беркли – «Curiosity-driven Exploration by Self-supervisored Prediction».

Когда дело доходит до обучения с подкреплением, основной сигнал обучения приходит в виде награды: скалярное значение, предоставляемое агенту после каждого принятого им решения. Эта награда обычно предоставляется самой средой и определяется создателем среды. Эти награды часто соответствуют таким вещам, как +1.0 для достижения цели, -1.0 для смерти и т. Д. Мы можем думать о таких наградах как о внешних, потому что они приходят извне агента.

Если есть внешние награды, то должны быть и внутренние. Вместо того, чтобы предоставляться средой, внутренние вознаграждения генерируются самим агентом на основе некоторых критериев, которые в итоге служат какой-то цели, например, изменение поведения агента таким образом, что он получит еще большие внешние награды в будущем, или что агент будет исследовать мир больше, чем мог бы иметь в противном случае.

В статье авторы предлагают обучать две отдельные нейронные сети: прямую и обратную модель. Обратная модель обучается принимать текущее и следующее наблюдение, полученное агентом, кодировать их оба с помощью одного кодера и использовать результат для прогнозирования действия, которое было предпринято между возникновением двух наблюдений. Затем прямая модель обучается принимать закодированное текущее наблюдение и действие и предсказывать закодированное следующее наблюдение. Разница между предсказанными и реальными кодировками затем используется в качестве внутренней награды и передается агенту. Большая разница означает большую внутреннюю награду.

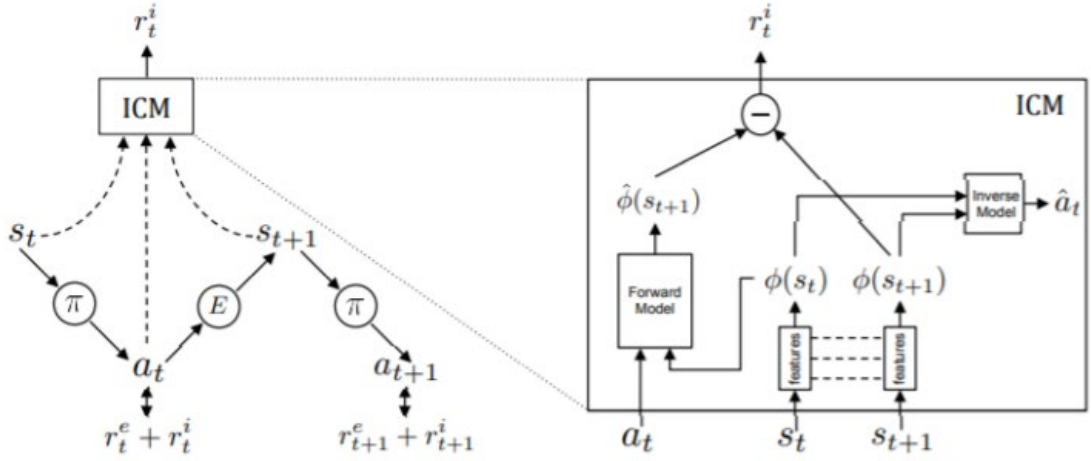


Рисунок 1 –Визуализация метода

Реализовать данный метод планируется в прототипе Action игры, используя межплатформенную среду разработки компьютерных игр Unity и его инструментарий ML-Agents в связи с его популярностью и гибкостью модульной системы при создании сцен и персонажей в игре.

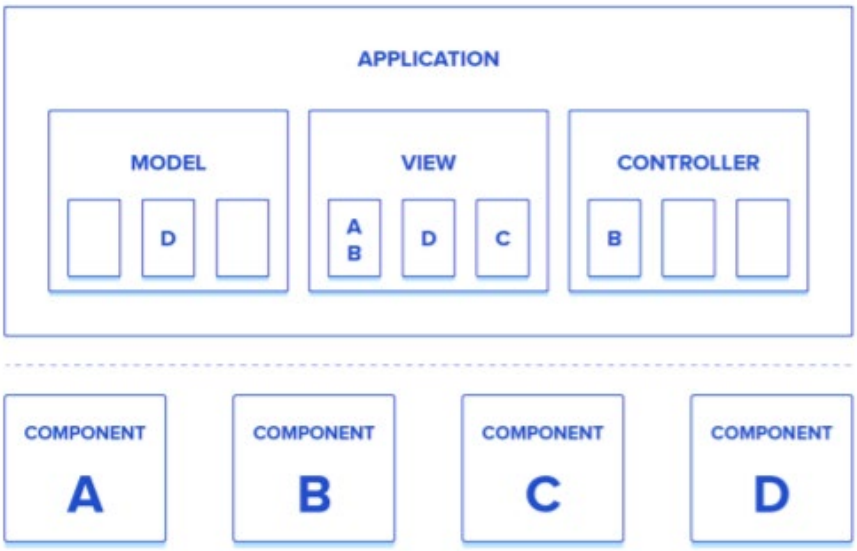


Рисунок 2 –Шаблон проектирования

Методы, основанные на любопытстве, не требуют предварительного проектирования задач, функций вознаграждения или чего-то еще. Они очень хорошо работают в средах, где основная цель - постоянно исследовать окружающую среду или оставаться в живых в течение длительного времени, всегда имея изменения состояния и решают проблему, когда внешнего вознаграждения недостаточно для желаемого результата обучения. В таких средах агент также может полностью тренироваться на внутреннем сигнале вознаграждения без необходимости во внешней функции вознаграждения. Однако методы, основанные на любопытстве, показывают слабые места в средах, которые не основаны на этих принципах, или где уже имеется плотная функция вознаграждения.

ЛИТЕРАТУРА

1. Solving sparse-reward tasks with Curiosity // Unity Blog URL: https://blog.unity.com/technology/solving-sparse-reward-tasks-with-curiosity (дата обращения: 20.03.2022).
2. Unity Machine Learning Agents // Unity URL: https://unity.com/ru/products/machine-learning-agents (дата обращения: 20.03.2022).
3. Curiosity-driven Exploration by Self-supervised Prediction // Github URL: https://pathak22.github.io/noreward-rl/resources/icml17.pdf (дата обращения: 20.03.2022).
4. Моделирование поведения агентов для реализации игрового искусственного интеллекта // Elibrary URL: https://elibrary.ru/item.asp?id=43908393 (дата обращения: 20.03.2022).
5. Unity и MVC: как прокачать разработку игры // Habr URL: https://habr.com/ru/post/281783/ (дата обращения: 20.03.2022).

УДК 004.45

Литвинов М.Б. (2 курс магистратуры),
Воинов Н.В., к.т.н., доцент,
Дробинцев Д. Ф., старший преподаватель

СИСТЕМА ПРОГНОЗИРОВАНИЯ ОШИБОК НА ОСНОВЕ ЖУРНАЛА СОБЫТИЙ

Целью работы является решение следующей проблемы. Компания “Deutsche Telekom” предоставляет своим клиентам телекоммуникационные сервисы [1], при возникновении проблемы клиент обращается в техподдержку, где оператор пытается решить проблему клиента, используя экспертную информационную систему. В случае непредвиденного сбоя в работе информационной системы происходит прекращение работы всей системы или ее

определенной части. Вне зависимости от масштаба возникшей неисправности подобная ситуация ведет к финансовым потерям, которые могут заключаться в затратах на исправление ошибки и ее последствий, а также в недовольных клиентах и их уходе [2].

Рисунок 1 – Система прогнозирования ошибок на основе журнала событий.

Предлагается разработать алгоритм, который позволит стабилизировать работу высоконагруженной системы благодаря интеллектуальному анализу генерируемых ей данных, а именно лог-файлов. Итоговая версия алгоритма представляет из себя функционирующую систему поиска аномалий в работе веб-приложения по лог-файлам.

Для реализации предложенного решения будут использованы рекуррентные нейронные сети [3,4]. В качестве математического аппарата алгоритма выбирается современная реализация рекуррентных нейронных сетей – сеть с долгой краткосрочной памятью.

Обычные рекуррентные нейронные сети (RNN) не предназначены для задач, в которых данные нужно запоминать на долгое время, т. к. влияние скрытого состояния или входа с определенного шага на последующие состояния рекуррентной сети экспоненциально затухает. Решением проблемы может быть использование рекуррентных сетей с долгой краткосрочной памятью (long short-term memory) с помощью введения специального LSTM-модуля, представленного на Рисунке 2. Данный модуль действует без использования функции активации внутри рекуррентных компонентов. Таким образом, запоминаемое значение не угасает со временем и градиент не пропадает при обучении рекуррентной нейронной сети методом обратного распространения ошибки.

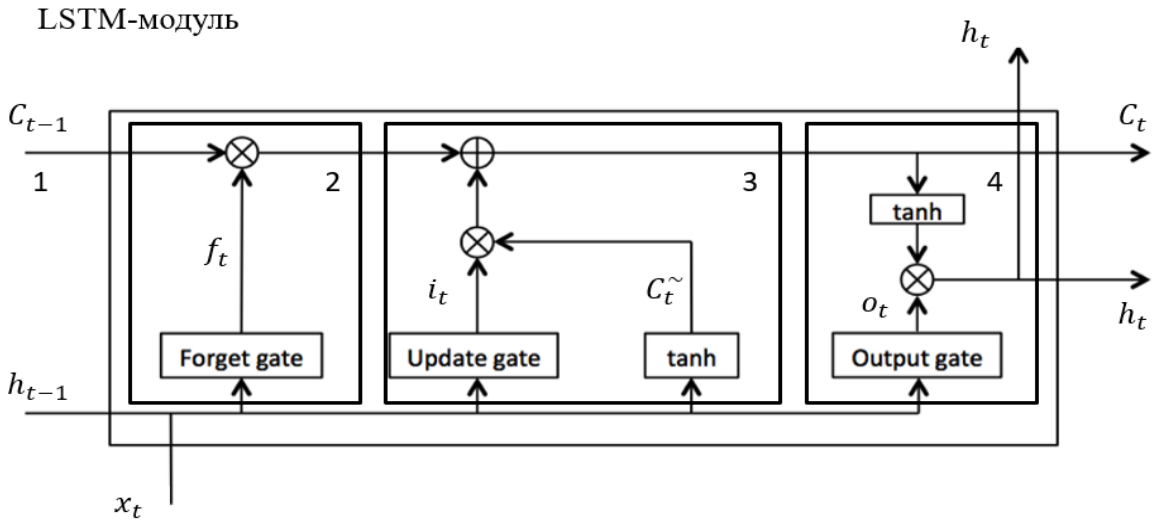


Рисунок 2 – Общий вид классического LSTM-модуля.

Главным компонентом LSTM-модуля является его состояние. Состояние ячейки проходит через весь LSTM-модуль и участвует только в нескольких линейных преобразованиях. Часть данных, хранимых состоянием, может быть удалена с помощью определенных параметров, называемых затворами. Представленный LSTM-модуль содержит в себе три затвора: затвор забывания, входной затвор и выходной затвор (цифры 2, 3 и 4 соответственно на Рисунке 2). Затворы исходя из определенных условий позволяют пропускать информацию и состоят из сигмоидального слоя и поточечного умножения.

Рекуррентная нейронная сеть с долгой краткосрочной памятью эффективно используется в задачах анализа последовательности данных, текста, временных рядов и показывает высокую точность. Свойство сохранения памяти определяет использование рекуррентной нейронной сети в разработке алгоритма прогнозирования ошибок на основе журнала событий с целью нахождения шаблонов в рабочем процессе экспертной системы. Отклонение от шаблона в рабочем процессе экспертной системы будет определяться как аномальное поведение, которое предположительно может спровоцировать сбой.

ЛИТЕРАТУРА

1. Литвинов, М. Б. Система интеллектуального анализа заявок пользователей телекоммуникационных услуг / М. Б. Литвинов, 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. 159.
2. Kovalev, A. Using the Doc2Vec Algorithm to Detect Semantically Similar Jira Issues in the Process of Resolving Customer Requests / A. Kovalev, N. Voinov, I. Nikiforov // Studies in Computational Intelligence. – 2020. – Vol. 868. – P. 96-101. – DOI 10.1007/978-3-030-32258-8_11.
3. Диаби, У. Разработка системы распознавания возгораний по видеопотоку на основе сверточных нейронных сетей / У. Диаби, 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. 51-53.
4. 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.

УДК 004.65

Маляренко М.Д. (4 курс бакалавриата),
Попов С.Г., к.т.н, доцент

ТЕХНОЛОГИЯ УПАКОВКИ БИТОВЫХ ПОЛЕЙ В ВЫСОКОУРОВНЕВОМ СИНТЕЗЕ АППАРАТНОГО УСКОРИТЕЛЯ

Технология высокоуровневого цифрового синтеза повышает уровень абстракции описания ИС путём отказа от временных ограничений и микроархитектурных особенностей на этапе проектирования и верификации схемы. HLS позволяет писать платформонезависимые, ориентированные на поток данных и управления спецификации специализированных процессоров, сопроцессоров, аппаратных ускорителей и другой специализированной цифровой аппаратуры.

В сочетании с технологией FPGA высокоуровневый цифровой синтез особенно полезен при прототипировании ИС ASIC, а также при программировании FPGA ускорителей в составе

гетерогенных вычислительных систем, где требуется частая реконфигурация аппаратных ускорителей при минимальном времени разработки.

Применение аннотаций типов переменных во входной спецификации алгоритма на ANSI C позволяет явно указать их минимальную допустимую битовую ширину. Основываясь на этой информации, при аппаратной реализации спецификации можно уменьшить ширину шин данных и регистров и, как следствие, уменьшить размер функциональных блоков, в которых переменная подаётся в качестве входного аргумента.

Оптимизация выделения и привязки аппаратных ресурсов при HLS трансляции стала предметом множества исследований с конца 90-х и до настоящего времени. В работах [1] и [2] предложен алгоритм планирования и привязки ресурсов при минимизации битовой ширины переменных, при этом в основе используется тот факт, что в более широкий регистр можно записать переменную меньшей ширины, но только единственную. Работы [3],[4],[5] предлагают математические модели и алгоритмы для решения NP-полной задачи минимизации количества регистров и мультиплексоров в аппаратной реализации входного алгоритма путём её сведения к проблемам раскраски графа и к задаче линейного программирования, а также предлагают эвристики.

В работе исследуется проблема эффективности использования ресурсов регистровой памяти FPGA при генерации микроархитектурного RTL описания специального аппаратного ускорителя, вычисляющего алгоритм по спецификации, написанной на языке ANSI C. К реализации предлагается технология минимизации количества регистров в контексте высокоуровневого цифрового синтеза специализированного аппаратного вычислителя функции, состоящей из одного базового блока. Цель достигается путём переиспользования регистра для хранения значения переменной со стандартной битовой шириной $n \in \{8,16,32,64\}$ и значений $m \in \{1,2,4,8\}$ переменных с битовой шириной $k=n/m \geq 8$. Задача решается созданием технологии упаковки битовых полей переменных меньшей битовой ширины в один регистр кратной ей большей битовой ширины с возможностью независимого записи и чтения этих битовых полей.

Предлагаемая технология реализует:

- эвристический алгоритм минимизации использования регистров при привязке к ним переменных с возможностью упаковки переменных в один регистр с кратной битовой шириной;
- микроархитектурную реализацию модифицированного регистра с раздельным доступом к битовым полям, метод коммутации модифицированных регистров и функциональных блоков и метод управления этими регистрами конечно-автоматным контроллером специализированного вычислителя.

Краткое описание алгоритма

В основе разрабатываемого алгоритма лежит жадный Left-Edge алгоритм привязки переменных к регистрам. Входными данными алгоритма является спланированный по тактам абстрактной машины состояний граф потока данных транслируемого базового блока (Data Flow Graph, DFG), множество используемых переменных, где для каждой переменной известна её битовая ширина $w(v_i) \in \{8,16,32,64\}$. В процессе работы алгоритма составляется таблица, строки которой определяют выделенный регистр, столбцы — такты машины состояний. Ячейки таблицы показывают в какой такт какие переменные привязаны к тому или иному регистру. Особенность разрабатываемого алгоритма, что в одной ячейке полученной таблицы могут в упакованном виде находиться больше одной переменной.

Пример аппаратной реализации технологии упаковки битовых полей.

Модифицированный регистр для аппаратной реализации предложенной технологии представляет собой кластер из m n-битовых регистров, входы и выходы которых объединены в шину шириной $m \times n$. Как и при постановке задачи, здесь $n \in \{8,16,32,64\}$ $m \in \{1,2,4,8\}$, $n \times m \leq 64$.

К каждому из регистров подведён разрешающий запись сигнал, тактовый сигнал и сигнал сброса. Разрешающие сигналы также объединены в шину, на которую подаётся

управляющее слово. На Рисунке 1 представлена RTL схема модифицированного регистрового кластера из 4-х регистров шириной 8 битов. На Рисунке 2 представлена RTL схема примера использования регистрового кластера в аппаратной реализации потока данных вычислителя простой функции foo:

```
uint16_t foo(uint8_t a, uint8_t b, uint16_t c) {
  uint16_t tmp = a * b;
  uint16_t y = tmp + c;
  return y;
}
```

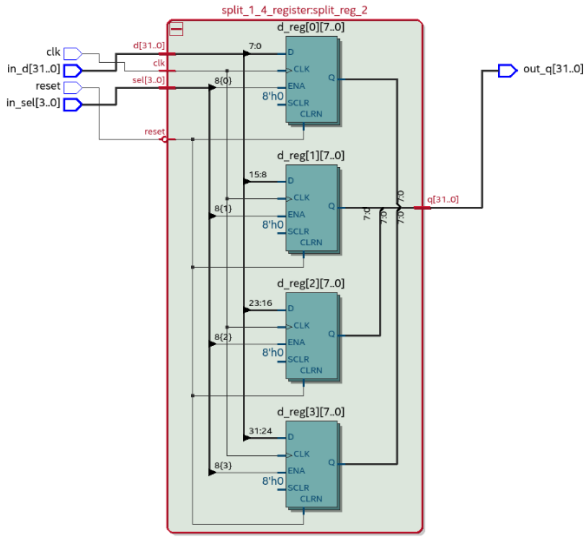


Рисунок 1 – RTL схема модифицированного регистрового кластера

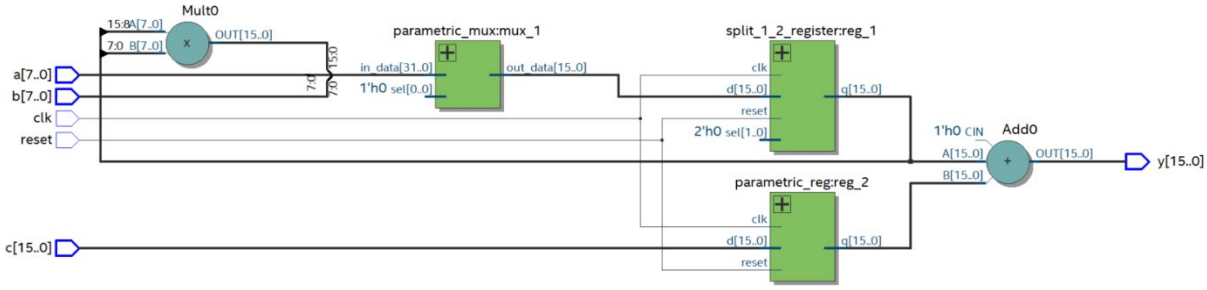


Рисунок 2 – RTL схема потока данных аппаратной реализации вычислителя простой функции

В данный момент технология находиться на этапе разработки прототипа. Разработка ведётся на основе академического HLS проекта с открытым исходным кодом ahaHLS (https://github.com/dillonhuff/ahaHLS). Данный HLS проект построен на базе фреймворка разработки компиляторов LLVM.

Предлагаемая технология должна встроиться в поток трансляции спецификации на ANSI C в Verilog HDL RTL описания схемы, а именно на следующих этапах:

1. Сбор информации о битовой ширине переменных на основании промежуточного представления LLVM (IR LLVM).
2. Привязка спланированного по тактам графа потока данных базового блока к ресурсам памяти в виде обычных регистров и модифицированных регистровых кластеров.
3. Генерация Verilog HDL на основе нетлиста сгенерированной схемы.

ЛИТЕРАТУРА

1. J. Cong et al., "Bitwidth-aware scheduling and binding in high-level synthesis," Proceedings of the ASP-DAC 2005. Asia and South Pacific Design Automation Conference, 2005., 2005, pp. 856-861 Vol. 2, doi: 10.1109/ASPDAC.2005.1466476.

2. N. Chabini and W. Wolf, "Register binding guided by the size of variables," 2007 25th International Conference on Computer Design, 2007, pp. 587-594, doi: 10.1109/ICCD.2007.4601957.
3. H. A. Atat and I. Ouaiss, "Register binding for FPGAs with embedded memory," 12th Annual IEEE Symposium on Field-Programmable Custom Computing Machines, 2004, pp. 165-175, doi: 10.1109/FCCM.2004.49.
4. Chu-Yi Huang, Yen-Shen Chen, Yan-Long Lin and Yu-Chin Hsu, "Data path allocation based on bipartite weighted matching," 27th ACM/IEEE Design Automation Conference, 1990, pp. 499-504, doi: 10.1109/DAC.1990.114907.
5. P. Sittel, M. Kumm, J. Oppermann, K. Möller, P. Zipf and A. Koch, "ILP-Based Modulo Scheduling and Binding for Register Minimization," 2018 28th International Conference on Field Programmable Logic and Applications (FPL), 2018, pp. 265-2656, doi: 10.1109/FPL.2018.00053.

УДК 004.75

Муратов С. Ю. (2 курс магистратуры),
Лукашин А. А., к.т.н., доцент

РАЗРАБОТКА АРХИТЕКТУРЫ ФРЕЙМВОРКА ЗАЩИЩЁННОГО ОЗЕРА БОЛЬШИХ ДАННЫХ

В современном мире объём всех данных неуклонно растёт. К 2025 году он составит 163 зеттабайт (Зб), что в 10 раз больше, чем общий объём данных, зафиксированный в 2016 году [1]. Прогнозируемый объём информации, который накопит население планеты менее чем за 10 лет, составит $163 * 10^{21}$ байтов. Данная величина больше на 2 порядка мирового объёма интернет-трафика за 2016 год (1 Зб) и на 4 порядка объёма произведённой информации за 2006 год (0.16 Зб).

С ростом объёма информации всё более значимым критерием становится качество при работе с данными, а темп увеличения количества защищенных данных отстает от темпа роста общего количества данных. Кроме того, допускается интенсивный рост метаданных, которые поступают с возрастающего количества умных устройств. Также количество данных, поступающих от одного человека, подвергается непрерывному росту вследствие распространения всевозможных сервисов взаимодействия с сетью Интернет.

Вышеприведённые факты являются основополагающими предпосылками для возникновения принципиально новых структур данных, таких как озёра данных. Хранение и обработка информации в подобных структурах не подвержены негативному воздействию как интенсивного роста, так и качественного изменения данных по ряду причин.

Необходимость в хранении и обработке возрастающего с течением времени количества информации способствует развитию альтернативных подходов для работы с данными, в частности, использования озёр данных.

Для полноценной и перспективной работы с озёрами данных однозначно определено его понятие, преимущества и недостатки данного подхода в сравнении с хранилищем данных и базой данных, а также изучены современные методы их построения, технологии обработки данных и реализованы на практике примеры с полезной нагрузкой.

Полученные знания и навыки обеспечивают фундамент для дальнейшего исследования архитектур озёр данных в контексте таких задач бизнес-аналитики, как поиск бизнес-проблем и возможные пути их устранения, разработка и модификация бизнес-процессов, оптимизация структуры предприятия, а также разработка и реализация новых концепций деятельности предприятия.

Анализ современных методов построения озёр данных позволил выявить как возможные модификации методов, так и положил начало разработки собственного метода построения архитектуры фреймворка, наполнения и обработки данных в контексте обеспечения их безопасности.

Обзор метрик и анализ актуальных сценариев атак на хранилища данных позволил выявить такие критические недостатки в фреймворках озёр данных, как отсутствие комплексного подхода к обеспечению безопасности и недостаточная степень проработанности уровней защиты данных от внешних и внутренних угроз.

За основу взят наиболее актуальный фреймворк – Data Lake Architecture Framework (DLAF) [2] как многофункциональная платформа для проектирования озёр данных с учётом всех аспектов. Архитектурные аспекты, включенные в DLAF, получены путём кластеризации результатов тщательного обзора современных примеров использования озёр данных. Девять полученных кластеров сформулированы в виде аспектов (см. Рисунок 1) и представлены в виде фиксированного набора: инфраструктура (A), хранилище данных (B), поток данных (C), моделирование данных (D), организация данных (E), обработка данных (F), управление метаданными (G), защита и конфиденциальность данных (H), качество данных (I).

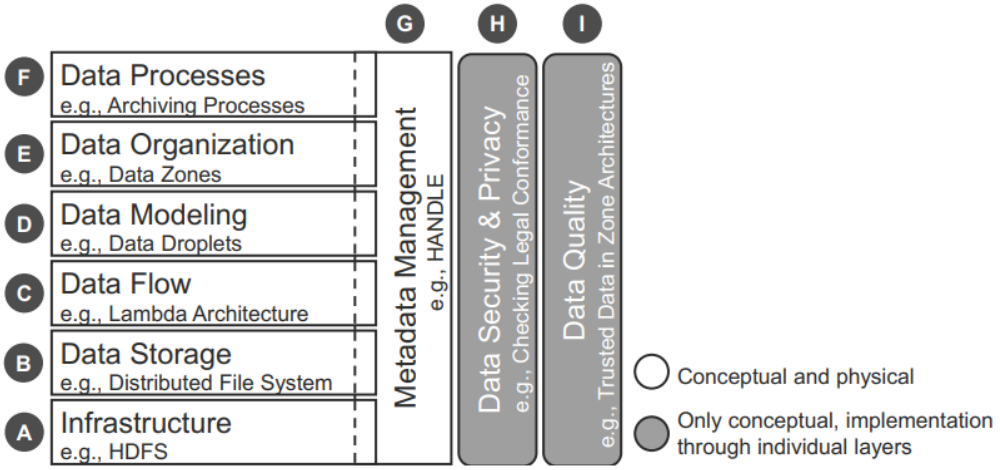


Рисунок 1 – Структура DLAF, состоящая из девяти аспектов озёр данных, которые необходимо учитывать при создании комплексной архитектуры озера данных.

Благодаря анализу моделей безопасностей и мер защиты произведён отбор наиболее подходящих для реализации аспектов, обеспечивающих защиту данных, и для модификации аспектов, косвенно влияющих на степень безопасности озера данных.

Произведён эксперимент с помощью средств Apache Airflow и Spark [3], состоящий из набора тестов и эмуляций как внешних атак: эксплуатация уязвимостей, внедрение самовоспроизводящихся данных, нелегальное форматирование и шифрование данных, внедрение вредоносного кода и/или мусорных данных, так и внутренних: нарушение ролевой модели доступа, генерация ложных атрибутов данных, утечка данных, нелегальное использование метаданных.

В результате проделанной работы разработана архитектура фреймворка озера больших данных, аспекты которого направлены на максимизацию степени защищенности данных от внутренних и внешних атак. Модифицированные модели безопасности могут использованы как в составе аспектов фреймворка, так и в роли самостоятельных мер обеспечения безопасности данных.

ЛИТЕРАТУРА

1. Прогноз объёма данных к 2025 году. [Электронный ресурс] URL: https://aboutdata.ru/2017/04/27/volume-of-data-by-2025/. – (дата обращения 21.06.2021).
2. Corinna Giebler, Christoph Gröger, Eva Hoos, Rebecca Eichler, Holger Schwarz, Bernhard Mitschang, The Data Lake Architecture Framework: A Foundation for Building a Comprehensive Data Lake Architecture, K.-U. Sattler et al. (Hrsg.): Datenbanksysteme für Business, Technologie und Web (BTW 2021), Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2021.
3. Bas Harenslak, Julian de Ruiter, Data Pipelines with Apache Airflow, Manning Publications, Shelter Island, 2021.

УДК 004.04

Хай Иен Нгуен (2 курс магистратуры),
Ковалев А. Д. (ассистент),
Никифоров И. В., к.т.н., доцент

ПРОГРАММНАЯ СИСТЕМА ОПРЕДЕЛЕНИЯ ВРЕДОНОСНОГО СОДЕРЖАНИЯ В ТЕКСТОВЫХ ДАННЫХ С ИСПОЛЬЗОВАНИЕМ АЛГОРИТМА DOC2VEC

На сегодняшний день в мире очень много информации, которая представлена в текстовом виде. Особое внимание заслуживает тот факт, что зачастую текст может содержать обманчивый, недостоверный контент. Например, рассылка писем с вредоносным содержанием, таким как xsl-файлы с макросами, doc-файлы с вирусами, встроенные в письма фишинговые страницы или ссылки на сторонние фишинговые сайты в сети Интернет. Актуальной является проблема автоматического определения таких ресурсов и предотвращения их использования. Ручное определение вредоносных ресурсов и писем является достаточно трудоемким процессом. Поэтому нужно разработать новый подход, который позволит определять вредоносное содержание в текстовых данных в автоматизированном режиме [1]. В основе предлагаемого подхода будет лежать алгоритм Doc2Vec [2,3].

Целью данной работы является разработка программного модуля, который определяет "вредоносности" в текстовых данных. Прикладная задача выглядит следующим образом и представлена на Рисунке 1:

- приходит входящее письмо;
- письмо поступает на вход модуля определения "вредоносности";
- модуль применяет разработанный алгоритм;
- на выходе модуля выдается ответ: "да", если письмо вредоносное; "нет", если письмо не вредоносное.

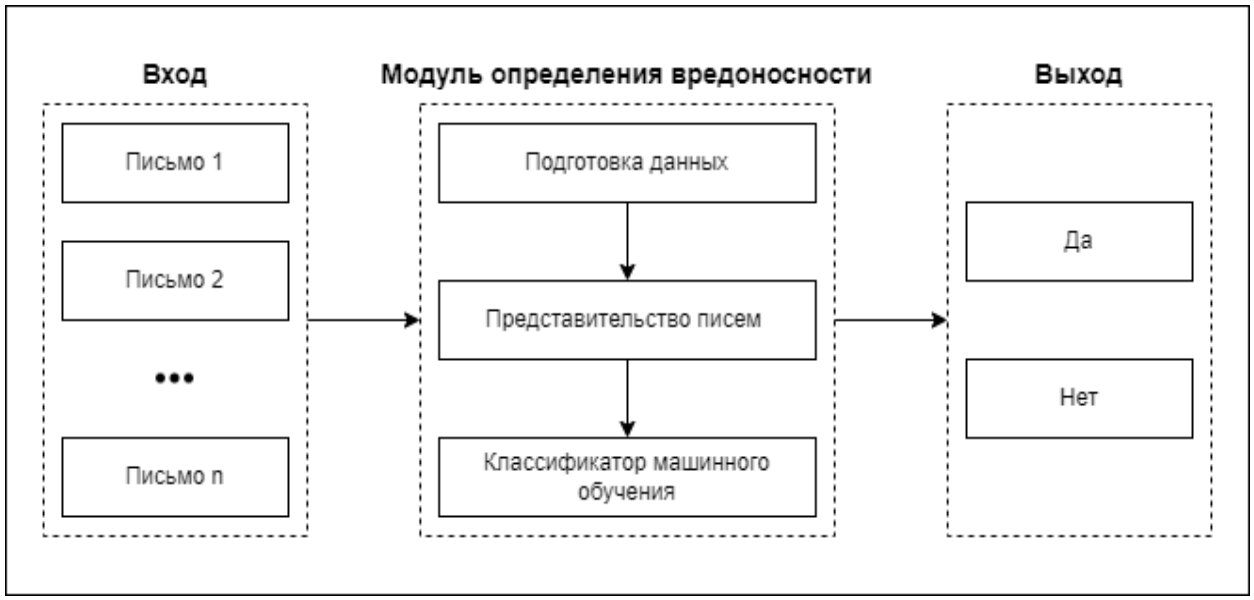


Рисунок 1 – Предлагаемая архитектура для определения вредоносности по электронной почте

Вначале необработанные данные очищаются и подготавливаются перед вводом в обучающие модели классификации. В следующем списке представлены реализованные шаги для получения предварительно обработанного набора текстовых данных для каждого электронного письма [4]:

1) удаление знаков препинания;
2) удаление URL-адресов, начинающихся с www.* или http://*;

3) удаление адресов электронной почты;

4) удаление стоп-слов;

5) токенизация текстов и лемматизация слов с помощью библиотеки Python NLTK (Natural Language Toolkit) [5];

6) семантическая реконструкция слов с ошибками, сопоставление эмодзи с их символьным выражением и замена сленга их первоначальным значением.

Было доказано, что в приложениях на основе технологий NLP (Natural Language Processing) использование простых функций, таких как «частота термина»-«инверсия частоты документа» (TF-IDF), менее эффективно, чем «вложения слов» (Word Embeddings) [6]. Поэтому для представления текста были предложены «вложения слов» путем преобразования слов в векторы фиксированной длины с действительными численными значениями. Doc2Vec [2] является одним из таких методов векторизации документов, при котором семантически похожие документы имеют минимальное значение косинуса угла между их векторами. Совокупность уникальных векторов документов представлена матрицей документов, а совокупность векторов слов представлена матрицей слов. В алгоритме Doc2Vec матрица векторов слов постоянна для разных документов, т. е. векторное представление одного и того же слова в разных документах будет одинаковое.

Кроме использования алгоритма Doc2Vec для решения поставленной задачи, также предлагается определять вредоносность электронных писем традиционными классификаторами на основе алгоритмов машинного обучения, например, Support Vector Machine, Naive Bayse, Logistic Regression, Random Forest и другие.

В рамках проекта предложена концепция подхода определения вредоносного содержания в текстовых данных с использованием алгоритма Doc2Vec и другими традиционными классификаторами. Будущими планами является реализация предложенного подхода в программной системе. Для хранения исходного кода будет использоваться платформа GitHub [7].

ЛИТЕРАТУРА

1. N. Voinov, I. Nikiforov, P. Drobintsev, V. Kotlyarov, A system prototype for real time automatic fraud detection in text data // Исследования по геоинформатике: труды Геофизического центра РАН. – 2017. – Vol. 5. – No 1. – P. 84. – DOI 10.2205/CODATA2017.
2. Q. Le and T. Mikolov, “Distributed representations of sentences and documents,” in ICML, 2014, pp. 1188–1196. [Online]. Available: http://proceedings.mlr.press/v32/le14.html
3. А. Д. Ковалев, И. В. Никифоров, П. Д. Дробинцев, 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
4. Jianqiang, Z.; Xiaolin, G. Comparison research on text pre-processing methods on twitter sentiment analysis. IEEE Access 2017, 5, 2870-2879.
5. 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.
6. Kulkarni, A.; Shivananda, A. Converting text to features. In Natural Language Processing Recipes; Apress: Berkeley, CA, USA, 2019; pp. 67–96.
7. N. Voinov, K. Rodriguez Garzon, I. Nikiforov, P. Drobintsev, 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.

УДК 004.65

Печеный Н.А. (4 курс бакалавриата),
Попов С.Г., к.т.н,, доцент

АЛГОРИТМ ПЕРЕРАСПРЕДЕЛЕНИЯ ДАННЫХ В РАСПРЕДЕЛЁННОЙ СУБД

В настоящее время работа крупных IT-компаний связана с обслуживанием клиентов, географически распределенных по очень большим территориям, а иногда и попросту находящихся в разных частях света. Это приводит к необходимости создания и использования распределенных средств хранения больших объемов данных, что технологически реализуется в форме распределенных баз данных. Основная проблема, возникающая при работе с распределенными БД — сложно прогнозируемое время выполнения запросов по, сравнению с классическими реляционными БД технологии клиент-сервер, что связано, в первую очередь, с необходимостью синхронизации выбора данных из нескольких территориально распределенных узлов в одном запросе. В случае, когда сохранена гомогенность схем базы данных, и её узлы реализованы в соответствии с одной схемой, возникает вопрос оптимальности распределения записей по узлам БД.

В таком случае возникает проблема исследования технологии оптимального распределения и перераспределения данных между узлами сети и разработки методов оптимизации времени выполнения запросов к гомогенной распределенной иерархической базе данных в условиях разделения данных на кластеры на уровне кортежей.

Решением этой задачи является применение технологии оптимизации времени выполнения запросов к БД, которая состоит из трёх этапов:

- оптимизация запросов [1, 2], состоящая в изменении непосредственно структуры запроса, его составных частей – подзапросов, и порядка их выполнения на основе анализа процесса выполнения запроса сервером;
- оптимизация структуры базы данных [3], то есть соблюдение правил построения архитектуры баз данных, нормализация данных, минимизация данных;
- оптимизация работы сервера получения запроса 1-го уровня иерархии при обращении к узлам (серверам) хранения данных на втором и последующих уровнях иерархии, для построения наилучшего маршрута получения данных и определение последовательности выполнения частей запроса с возможностью предварительной компоновки данных.

Исследованию подвергаются модели и методы оптимизации работы сервера, получившего запрос. Оптимизация выполнения запросов может быть выполнена путем выбора оптимального узла для хранения данных. Оптимальным будем считать узел, время выполнения запроса к которому минимально. Выбор оптимального узла зависит от двух параметров: времени исполнения запроса на запись данных и времени исполнения запроса на чтение данных. Формально критерий оптимальности при данных условиях формулируется следующим образом:

$$n_{opt} = n_i \Leftrightarrow \left(t_{in_i} + t_{out_i}\right) < \left(t_{in_j} + t_{out_j}\right) \forall j \neq i; i, j = 1,2, \dots, k \qquad (1)$$

где n_{opt} — оптимальный из k узлов, t_{in}, t_{out} — времена исполнения запросов на запись и чтение данных соответственно. Для простоты будем считать, что эти времена нам известны для любого конечного устройства, осуществляющего обращение к БД. Будем исходить из предположения о том, что запись и чтение из базы происходят с равной частотой. Таким образом, нельзя считать оптимальным узел, который оптимален только по одному из двух параметров.

Ввиду того, что количество запросов от конечных устройств, имеющих разное географическое положение, изменяется со временем, то и оптимальный узел для хранения конкретных данных постоянно изменяется.

Для оптимизации времени выполнения запросов необходимо в реальном времени определять количество обращений к тому или иному фрагменту данных с целью последующего перераспределения данных между узлами БД. Таким образом, при записи

новых данных их можно помещать в узел, время обращения на запись к которому на момент исполнения транзакции минимально. После чего начинать отслеживать количество обращений к вновь добавленным данным от серверов 2-го уровня иерархии, привязанных к другим узлам, запоминая откуда приходят запросы на получение данных.

Такой сбор статистики позволяет отслеживать потенциально оптимальные узлы для хранения конкретных данных. Когда количество обращений к данным с серверов, не относящихся к узлу, на котором эти данные хранятся, превышают количество обращений с сервера, связанного с узлом, необходимо признавать узел неоптимальным. В таком случае данные стираются с узла и записываются в тот, с которым связан сервер, от которого приходило наибольшее количество обращений на чтение. Тогда формальное определение оптимального узла для хранения конкретных данных принимает следующий вид:

$$n_{opt} = n_i \Leftrightarrow r_i > r_j \; \forall j \neq i; i, j = 1,2, \dots, k \qquad (2)$$

где r — количество обращений на чтение данных. Такое перераспределение данных происходит незаметно для конечного пользователя, однако уменьшает количество времени, необходимого для исполнения запроса на чтение из БД. Также, из-за того, что статистика по отдельным фрагментам данных, вплоть до строки таблицы, может собираться независимо, при перераспределении данных возможно избежать передачи больших массивов информации между узлами, пересылая данные минимально необходимого объема.

Однако простое отслеживание количества обращений к данным не может защитить систему от кратковременных всплесков активности, которые незаметны на долгих временных промежутках, но создают ситуацию, при которой некоторый отличный от текущего узел может оказаться предпочтительнее для хранения данных сиюминутно. Для защиты от таких выбросов можно собирать статистику в виде обращений за единицу времени, после чего, применяя метод квартилей, отбрасывать 50% значений — поровну 25% записей с наименьшим и 25% с наибольшим количеством обращений в единицу времени. После чего, просуммировав оставшиеся значения, можно получить оценку оптимальности узла, по которой и принимается решение о необходимости перераспределения данных.

Подобный алгоритм мониторинга и перераспределения данных позволяет разделить двухкритериальную задачу оптимизации (1) на две однокритериальных задачи — поиск одного оптимального узла при записи и другого при чтении данных.

Алгоритм может применяться в системах управления базами данных социальных сетей, как информационных систем с ярко выраженным распределением конечных пользователей по большим по площади территориям.

ЛИТЕРАТУРА

1. Matthias Jarke, Jurgen Koch. Query Optimization in Database Systems // Computing Surveys, Vol. 16, No. 2, 1984.
2. С. Г. Попов, А. А. Пурий, “Исследование алгоритмов декомпозиции и выполнения запросов на выборку в гетерогенных системах управления реляционными базами данных”, *Научно-технические ведомости СПбГПУ. Информатика. Телекоммуникации. Управление*, 12:2 (2019), стр. 50–67.
3. Christopher J. Date. An Introduction to Database Systems, 8th Edition. ISBN 0-321-18956-6, 2004.

УДК 004.896

Потапова М.А. (4 курс бакалавриата),
Черноруцкий И.Г., д.т.н., профессор

МЕТОД ОПОРНЫХ ВЕКТОРОВ НАИМЕНЬШИХ КВАДРАТОВ ДЛЯ ПРОГНОЗИРОВАНИЯ ОБЪЕМОВ ПЛАТЕЖЕЙ В ЭНЕРГОСБЫТОВЫХ КОМПАНИЯХ

Целью работы является построение математической модели, реализующей алгоритм метода опорных векторов наименьших квадратов (LSSVM), для прогнозирования объемов платежей в энергосбытовых компаниях в рамках разрабатываемой системы помощи принятия решений (СППР).

Построенная модель должна давать допустимую точность предсказания (5-12%), а также улучшить возможность модели прогнозировать пиковые значения, которые попадают в категорию выбросов.

Для анализа используются данные, полученные от электросбытовой компании за почти 10 лет (с 01.01.2012 до 24.11.2021). Данные представлены в виде таблицы, каждая строка представляет собой дату платежа и общую сумму платежей клиентов в этот день. Их можно представить в виде временного ряда и визуализировать, как представлено на Рисунке 1.

Рисунок 1 – Визуализация временного ряда.

LSSVM использует организованную функцию наименьших квадратов с ограничениями равенства, что приводит к линейной системе уравнений. Следовательно, задача может быть решена с помощью системы линейных уравнений.

Для реализации поставленной задачи использовалась регрессионная версия модели метода опорных векторов наименьших квадратов (LSSVR). Он предназначен для аппроксимации неизвестной функции с использованием заданной серии обучающих данных, содержащихся в окне, равном определенному периоду времени (в поставленной задачи применялись окна: 30, 60, 180 дней).

Выбор модели обусловлен тем, что в одном из изученных примеров реализации подобных задач была использована гибридная модель LSSVR и ARIMA. Авторами статьи была получена высокая точность в исследуемой задаче.

Такая гибридная модель не дала необходимых результатов, поэтому было принято решение использовать эти алгоритмы по отдельности. ARIMA прогнозировала с точностью порядка 30-40%, а LSSVR без различных модификаций – 12%, но с предсказанием пиков модель не справилась (использовалось окно в 180 дней). Результат представлен на Рисунке 2. Поэтому целесообразно анализировать ряд по частям, то есть сформировать отдельный временной ряд для предсказания выбросов, а затем после прогноза подменять предсказанные изначально значения новыми пиками. Лучший результат был получен, когда были отобраны данные, находящиеся выше 95% перцентиля каждого месяца.

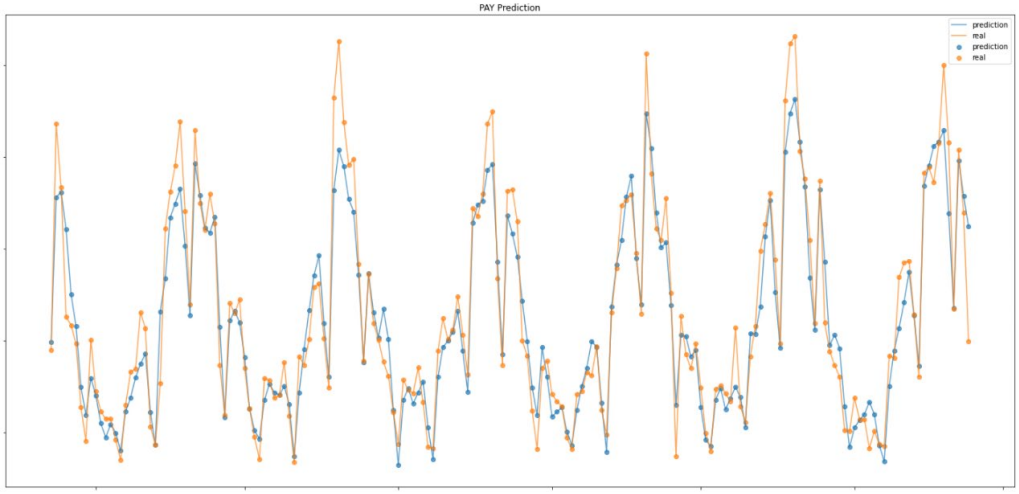


Рисунок 2 – Полученный результат LSSVR до выделения пиков.

Такой алгоритм снизил ошибку до 11%, а также довольно точно предсказал пиковые значения. Результат представлен на Рисунке 3.

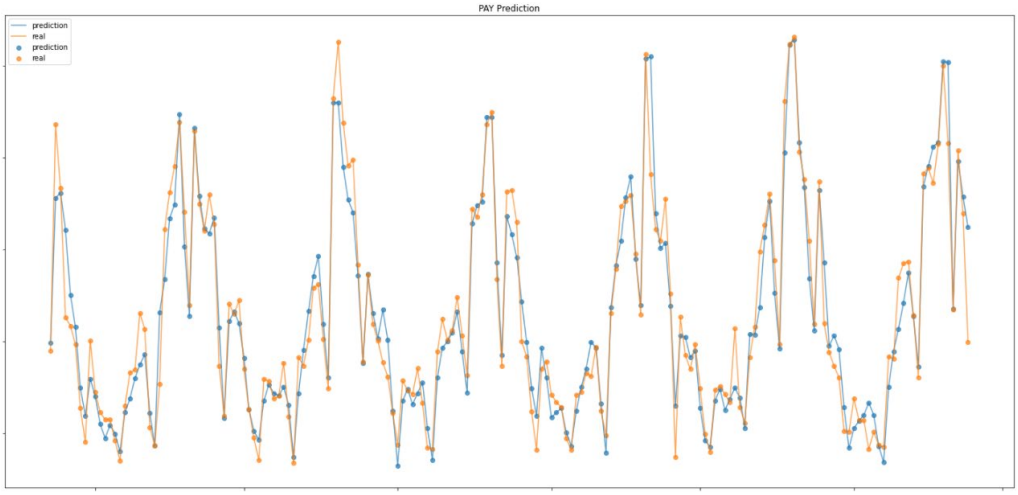


Рисунок 3 – Полученный результат LSSVR после выделения пиковых значений.

ЛИТЕРАТУРА

1. ScienceDirect. A hybrid approach based on autoregressive integrated moving average and least-square support vector machine for long-term forecasting of net electricity consumption. [Электронный ресурс]. Режим доступа: https://www.sciencedirect.com/science/article/pii/S0360544220303078
2. MQL5. ПРОГНОЗИРОВАНИЕ ВРЕМЕННЫХ РЯДОВ (ЧАСТЬ 2): МЕТОД НАИМЕНЬШИХ КВАДРАТОВ ОПОРНЫХ ВЕКТОРОВ (LS-SVM). [Электронный ресурс]. Режим доступа: https://www.mql5.com/ru/articles/7603

УДК 004.855.5

Просвирнин Е. Н. (2 курс магистратуры),
Лукашин А. А., к.т.н., доцент

МЕТОДЫ И МОДЕЛИ МАШИННОГО ОБУЧЕНИЯ ДЛЯ РЕШЕНИЯ ЗАДАЧИ РЕТРОСИНТЕЗА ХИМИЧЕСКИХ СОЕДИНЕНИЙ

Ретросинтез – это метод химического синтеза, который включает «деконструкцию» продукта химической реакции на легкодоступные простые исходные молекулы для определения наилучшего пути синтеза. В работе рассматривается обзор методов предсказания реагентов на основе продукта с помощью алгоритмов машинного обучения.

Классификация методов для решения задачи ретросинтеза.

Методы использующиеся для решения задачи ретросинтеза можно разделить на три группы на основе использования ими дополнительной информации о химических реакциях.

1. **Методы на основе шаблонов реакций**, автоматически извлекают шаблоны из базы данных химических реакций и выбирают наиболее подходящий шаблон для применения к продукту для восстановления реактантов. Главный недостаток таких методов, это невозможность обнаружения новых реакций, так как они способны работать только с ограниченным множеством химических реакций.
2. **Методы без использования химических шаблонов,** рассматривают как задачу перевода одной последовательности в другую используя SMILES репрезентацию молекул. Однако химическая недействительность предсказанных результатов, является главной проблемой данного подхода.
3. **Методы частично использующие шаблоны,** соединяют в себе преимущества описанных выше методов. Сначала происходит конвертация продукта в синтоны, а затем синтоны превращаются в реактанты.

Также используемые методы можно разделить на основе архитектурных особенностей применяемых моделей и используемых репрезентаций химических молекул.

- *Модели, работающие с последовательностями*. Такие модели используют SMILES репрезентацию химических реакций и архитектуру наиболее успешных моделей для обработки естественного языка.
- *Модели, работающие с графовыми структурами*. Используют графовые представления молекул и техники машинного обучения на графах.
- *Модели на основе энергии*. Этот раздел машинного обучения становится все более популярной темой и эти модели показывают наилучший результат.

В Таблице 1 представлено краткое описание наиболее популярных и методов, ретросинтеза.

Таблица 1 – Методы ретросинтеза с использование глубоких нейронных сетей.

Метод	**Описание**
Aug. Transformer[1]	Трансформерная архитектура модели. Обучение различным представлениям одной и той же реакции путем аугментации исходных данных устранило эффект запоминания и повысило обобщающую способность модели.
Graph truncated attention for retrosynthesis (GTA)[2]	Использует графовое и представление как последовательность. Также модель маскирует self-attention используя матрицу смежности в энкодере и функцию потерь используя сопоставление атомов для cross-attention в декодере.
Tied Two-Way Transformers with Latent Variables[3]	Пытается решить проблему химической валидности и разнообразия набора кандидатов реагентов, используя проверки согласованности циклов, совместное использование параметров и многочленные скрытые переменные.
Graph2SMILES[4]	Модель, сочетающая в себе мощность модели для генерации текста с декодером графов молекул робастным к перестановкам, что снижает потребность в аугментации входных данных.
Dual-TF[5]	Фреймворк, который объединяет sequence- и graph-based модели в виде модели на основе энергии с различными представлениям функций энергии.
GraphRetro[6]	Модель на основе графов, которая рассматривает задачу ретросинтеза как множество превращений графа в неполные молекулы – синтоны. Затем модель учится достраивать синтоны до полных молекул –реактантов.

Все три подхода показывают схожие результаты по метрике Top-N на датасете USPTO-50K. Модели Dual-TF (модель на основе энергии), Chemformer (sequence-based) и GraphRetro (graph-based) показывают наилучшие метрики.

Датасеты

Наиболее популярные датасеты для обучения моделей машинного обучения для решения задачи машинного обучения это USPTO датасет и его подмножества, Reaxys и Pistachio.

USPTO-50k – это наиболее популярный и находящийся в свободном доступе датасет для обучения и тестирования моделей машинного обучения, а Reaxys самый популярный проприетарный.

Метрики, применяемые для оценки методов предсказания ретросинтеза.

Описание наиболее часто используемых метрик представлено в Таблице 2.

Таблица 2 – Метрики

Метрика	**Описание**
Top-n (or Top-k)	Процент примеров в которых был найден набор исходных реагентов среди первых N прогнозов, сделанных моделью.
Round-trip accuracy	Определяет какой процент предсказанных реактантов является химически валидным.
Top-n validity	Измеряет количество химически валидных примеров среди N лучших предсказаний модели.

В результате работы сформулирована задача ретросинтеза молекулярных соединений, приведены способы оценки качества решения задачи: датасеты и метрики. Описаны различные подходы к решению, так же проанализированы главные особенности основных методов машинного обучения, применяемых для решения поставленной задачи. При анализе SOTA решений были выявлены недостатки каждого подхода и определены направления для возможных модификаций.

ЛИТЕРАТУРА

1. I. V. Tetko, P. Karpov, R. Van Deursen, and G. Godin, “State-of-the-art augmented NLP transformer models for direct and single-step retrosynthesis,” *Nat. Commun.*, vol. 11, no. 1, p. 5575, Nov. 2020.
2. S.-W. Seo *et al.*, “GTA: Graph truncated attention for retrosynthesis,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021, vol. 35, pp. 531–539.
3. E. Kim, D. Lee, Y. Kwon, M. S. Park, and Y.-S. Choi, “Valid, Plausible, and Diverse Retrosynthesis Using Tied Two-Way Transformers with Latent Variables,” *J. Chem. Inf. Model.*, vol. 61, no. 1, pp. 123–133, Jan. 2021.
4. Z. Tu and C. W. Coley, “Permutation invariant graph-to-sequence model for template-free retrosynthesis and reaction prediction,” *arXiv [cs.LG]*, Oct. 19, 2021. [Online]. Available: http://arxiv.org/abs/2110.09681
5. R. Sun, H. Dai, L. Li, S. Kearnes, and B. Dai, “Energy-based View of Retrosynthesis,” *arXiv [physics.chem-ph]*, Jul. 14, 2020. [Online]. Available: http://arxiv.org/abs/2007.13437
6. V. R. Somnath, C. Bunne, C. W. Coley, A. Krause, and R. Barzilay, “Learning Graph Models for Retrosynthesis Prediction,” *arXiv [cs.LG]*, Jun. 12, 2020. [Online]. Available: http://arxiv.org/abs/2006.07038

УДК 004.832.32

Скоков Н.С. (4 курс бакалавриата),
Пшеничная К.В., к.т.н., доцент

МЕТОДЫ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА В ЗАДАЧЕ АНАЛИЗА СЕТЕВОГО ТРАФИКА

В настоящее время активно развиваются технологии искусственного интеллекта (ИИ). Согласно отчету Стэнфордского университета [1] с каждым годом увеличивается уровень инвестиций в использование этих технологий в различных отраслях (Рисунок 1).

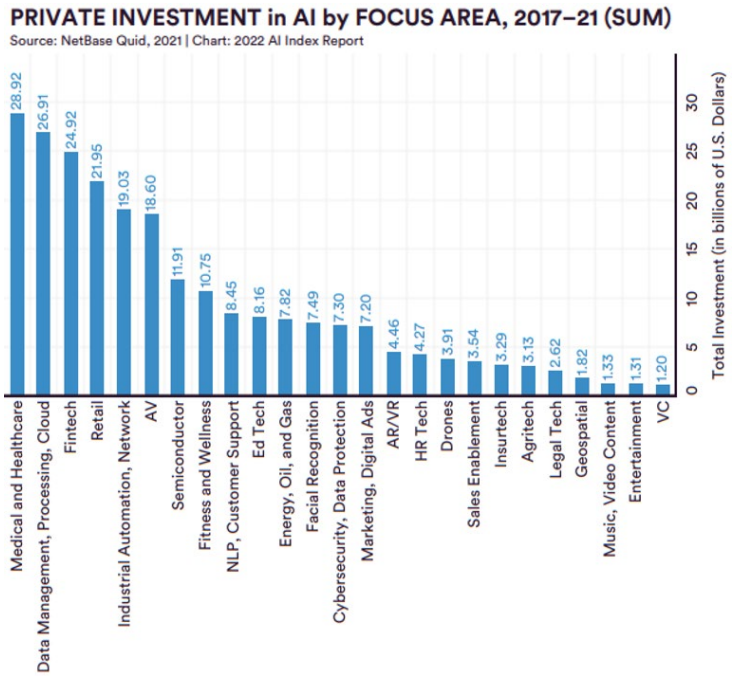


Рисунок 1 – Отчет Стэнфордского университета о привлечении новых инвестиций в области использования технологий ИИ

Защита информации (Cybersecurity, Data Protection) в данном рейтинге занимает 13 место по объему инвестирования. Больше всего данная технология популярна в продуктах антивирусной защиты, средствах обнаружения вторжений (СОВ), системах управления инцидентами и др. Почему же это так актуально? Эффективность использования ИИ в области информационной безопасности (ИБ) связана с тенденциями развития компьютерных угроз и цифровых технологий в целом. С помощью машинного обучения возможно обрабатывать огромные массивы данных с наименьшими погрешностями, анализировать различного рода аномалии и экстраполировать признаки неизвестных еще сигнатурам угроз, таких как атаки нулевого дня.

Целью работы является разработка системы исследования аномалий сетевого трафика и управления инцидентами ИБ на основе различных моделей ИИ. В задачи работы входит выбор набора данных для обучения, проведение статистической оценки признаков аномалий на основе исходных данных и анализ полученных результатов.

Для обучения системы обнаружения атак имеется ряд доступных публичных наборов данных: DARPA1998, KDD99, ISCX2012, ADFA2013 и др. В рамках данного проекта был выбран набор данных KDD99 [2], содержащий 140588 записей. Каждая запись представляет собой *feature-вектор*, идентифицированный в конце характером трафика: нормальный (normal) или аномальный (type of attack). Рассматриваются следующие признаки аномальности: logged_in, count, hot, flag, protocol type, land, src (dst) bytes, srv_count, service, serror_rate, rerror_rate и др. После предварительной обработки данных производится весовая оценка

признаков методами статистического анализа, с помощью построения матрицы корреляции (Рисунок 2).

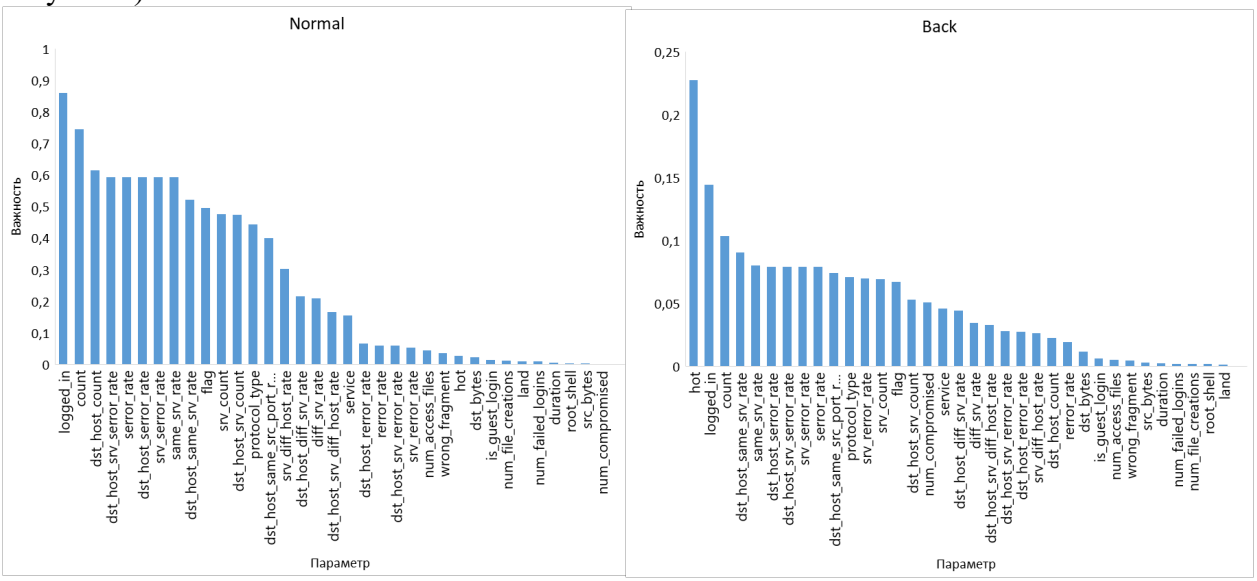


Рисунок 2 – Гистограмма важности параметров для трафиков типа «normal» и атаки «back»

Данный анализ позволяет выявить связи признаков с классами трафика. Это повышает качество модели ИИ, увеличивая эффективность принятия решений и снижая время выполнения алгоритма.

Предварительно обработанный набор данных использовался для обучения моделей ИИ различными методами (KNN, RF, AdaBoost, QDA и MLP). Использовалась библиотека Scikit-Learn для языка Python [3]. Оценка качества моделей производилась по ряду параметров:

1) Доля правильных ответов.
2) Точность.
3) Полнота.
4) F1-мера.

Результаты оценки качества моделей представлены в Таблице 1.

Таблица 1 – Оценка качества методов ИИ в задаче обнаружения инцидентов ИБ

Модель	**Доля правильных ответов**	**Точность**	**Полнота**	**F1**	**Время выполнения, с**
KNN	0,969	0,945	0,952	0,967	6,34
RF	0,971	0,973	0,936	0,965	1,89
AdaBoost	0,981	0,964	0,968	0,977	31,17
QDA	0,881	0,975	0,7	0,926	2,52
MLP	0,892	0,931	0,915	0,764	70,37

Результаты исследования позволяют выбрать наиболее эффективный алгоритм работы ИИ и на его основе реализовать систему ИБ. Правильное внедрение и обучение модели ИИ в области информационной безопасности позволяет выйти на новый уровень защиты данных с возможностью фиксировать не только уже реализованные угрозы, но и уязвимости, способствующие появлению новых возможностей реализации атаки.

ЛИТЕРАТУРА

1. Measuring trends in Artificial Intelligence (Измерение тенденций в области искусственного интеллекта) / Stanford University : [сайт]. — URL: https://aiindex.stanford.edu/report/ (дата обращения: 15.03.2022).
2. KDD99 : электронная база данных / MIT University. — URL: http://kdd.ics.uci.edu/databases/kddcup99/ (дата обращения: 17.02.2022).
3. Scikit-Learn: программная библиотека для машинного обучения на Python : [сайт]. — URL: https://scikit-learn.org/stable/index.html (дата обращения: 17.02.2022).

УДК 004.021

Сысоева О.М. (4 курс бакалавриата),
Селин И.А., старший преподаватель

РАЗРАБОТКА АЛГОРИТМА ДЕТЕКТИРОВАНИЯ ИСПОЛЬЗОВАНИЯ МОБИЛЬНОГО УСТРОЙСТВА В АВТОМОБИЛЕ С ПОМОЩЬЮ ДАННЫХ АКСЕЛЕРОМЕТРА МОБИЛЬНОГО УСТРОЙСТВА

В современном мире большое количество автолюбителей используют телефон во время вождения, что является важным фактором риска дорожно-транспортных происшествий. В результате использования мобильных устройств водители могут отвести взгляд от дороги и окружающей обстановки или убрать руки с руля.

Наиболее частым подходом к решению данной проблемы является использование дорожных камер или технологий ручного контроля. Но такие методы имеют ряд технических ограничений и некоторое количество водителей остается не охваченным. Предложенный алгоритм направлен на решение описанных проблем.

Целью работы является разработка и анализ алгоритма детектирования использования мобильного устройства в автомобиле с помощью данных акселерометра мобильного устройства.

Для достижения поставленной цели были сформулированы следующие задачи:

- Определить подход для разработки алгоритма детектирования использования мобильного устройства в автомобиле с помощью данных акселерометра мобильного устройства
- Разработать алгоритм детектирования использования мобильного устройства в автомобиле с помощью данных акселерометра мобильного устройства
- Реализовать программный модуль для детектирования использования мобильного устройства в автомобиле с использованием данных акселерометра мобильного устройства
- Оценить качество детектирования использования мобильного устройства в реализованном программном модуле

В процессе работы был проведён обзор существующих решений по распознаванию использования мобильного устройства в автомобиле. Были рассмотрены следующие подходы к решению этой проблемы:

- Определение использования телефона по датчику приближения [5]
- Определение использования телефона по акселерометру и гироскопу [2]
- Определение использования телефона по камере в телефоне [4]

В результате был определен подход для разработки алгоритма детектирования использования телефона на основе данных акселерометра. Выбор подхода обоснован тем, что данные с акселерометра получить проще и с их помощью можно выявить определенные паттерны использования мобильного устройства, чтобы отличить использование телефона от изменения ориентации, вызванных движением автомобиля.

Для реализации алгоритма детектирования использования телефона в автомобиле были собраны и разбиты на шаблоны тестовые данные.

Таблица 1 – Примеры шаблонов использования мобильного устройства

Начальные положения	Конечные положения
Телефон закреплен в портретной ориентации	Телефон около уха пользователя
Телефон лежит в машине экраном вверх	Телефон в руках в портретной ориентации
Телефон лежит в кармане экраном вверх	Телефон в руках в альбомной ориентации

Был совершен переход от фактических показаний акселерометра к ориентации устройства в пространстве с помощью матрицы поворота. Далее с помощью ориентации мобильного устройства были получены кватернионы, которые отражают изменение положения вектора ориентации устройства. Таким образом, данное преобразование позволяет исключить влияние абсолютного положения мобильного устройства и совершить переход к изменениям ориентации устройства за интервал времени.

Для классификации полученных данных был применен метод опорных векторов, который способен быстро определить принадлежность к классу и подходит для небольших объёмов данных. Задача классификации была направлена на определение 2 действий в машине:

- нет взаимодействия с телефоном
- взаимодействие с телефоном

В результате обученная модель была протестирована, и доля правильных предсказаний модели оказалась равной 98 %.

Была произведена оценка модели на реальной поездке. Всего было выполнено 32 варианта использования телефона. Доля ложноотрицательных предсказаний оказалась равной 3 %, а доля ложноположительный – 6 %.

В дальнейшем планируется внесение улучшений для сокращения числа ложноположительных срабатываний модели. Один из вариантов улучшения – добавление данных экранного времени для определения длительности использования мобильного устройства.

ЛИТЕРАТУРА

1. Muralidharan K, Anirudh Ramesh, Rithvik G. 1D Convolution approach to human activity recognition using sensor data and comparison with machine learning algorithms. – International Journal of Cognitive Computing in Engineering, 2021. – 130-143 с.
2. Xinhua Liu, Huafeng Mei. A Vehicle Steering Recognition System Based on Low-Cost Smartphone Sensors. – MDPI, 2017.
3. Mobile phone use: a growing problem of driver distraction. – World Health Organization, 2011. – 13 с.
4. Rushil Khurana, Mayank Goel. Eyes on the Road: Detecting Phone Usage by Drivers Using On-Device Cameras. – ACM, 2020.
5. Detection of in-progress phone calls using smartphone proximity and orientation sensors. – ACM, 2017.
6. Support Vector Machines: A Simple Tutorial. [Электронный ресурс]. Режим доступа: https://svmtutorial.online/

УДК 004.891.2

Чигасова М.А. (4 курс бакалавриата),
Малеев О.Г., к.т.н., доцент

ПРОГНОЗИРОВАНИЕ КОТИРОВОК АКЦИЙ С ИСПОЛЬЗОВАНИЕМ НЕЙРОННОЙ СЕТИ

Целью работы является построение эффективной модели прогнозирования финансовых инструментов, трендов и движения цен с использованием нейронной сети, а также разработка оптимальной стратегии торговли для демонстрации практического применения результатов прогнозирования. Стратегия торговли будет давать рекомендацию к покупке или продаже рассматриваемого финансового инструмента и рассчитывать полученный доход.

Предсказание финансовых временных рядов – это актуальная задача для инвестиционной деятельности. Решение о инвестициях в тот или иной инструмент основывается на прогнозе его будущих котировок. Любая задача, связанная с

манипулированием финансовыми инструментами, сопряжена с риском и требует тщательного расчета и прогнозирования.

Фондовый рынок – это место, для происходит торговля акциями, облигациями, валютами и прочими активами. В широком смысле анализ фондового рынка делится на две части – фундаментальный анализ и технический анализ. Фундаментальный анализ включает в себя анализ будущей прибыльности компании на основе ее текущей бизнес-среды и финансовых показателей. В основе технического анализа лежит выделение и изучение определенных закономерностей в движении графика котировок. В случае применения методов машинного обучения для обработки торговых данных, чаще используют именно метод технического анализа — цель заключается в том, чтобы понять, может ли алгоритм точно определять паттерны поведения акции во времени.

В последние годы у финансовых аналитиков стали вызывать большой интерес нейронные сети. Возможность обучения – одно из главных преимуществ нейронных сетей перед традиционными алгоритмами. В процессе обучения нейронная сеть способна выявлять сложные зависимости между входными и выходными данными. После обучения сеть способна предсказать будущее значение некой последовательности на основе нескольких предыдущих значений.

Для реализации модели прогнозирования была использована сеть долгой краткосрочной памяти (Long Short-Term Memory, LSTM) – разновидность архитектуры рекуррентной нейронной сети, созданная для более точного моделирования временных последовательностей и их долгосрочных зависимостей, чем традиционная рекуррентная сеть. Структура нейронной сети имеет входной слой, три скрытых слоя LSTM и выходной слой. В качестве функции потерь использован метод наименьших квадратов (Mean Square Error, MSE). Для каждой точки вычисляется квадрат отклонения, после чего полученные значения суммируются и делятся на общее количество точек. Чем ближе полученное значение к нулю, тем точнее модель. Цель – уменьшить среднеквадратичную ошибку для увеличения точности прогноза.

В данной работе рассматривается прогнозирование цены закрытия (Close) акции Apple. Данные в период с 2012 по 2022 год были разделены на тренировочную и тестовую выборки в отношении 85 к 15. Входные параметры для обучения модели содержат в себе ежедневные данные о цене открытия (Open), максимальной цене за день (High), минимальной цене за день (Low), цене закрытия (Close) и объеме продаж (Volume). Обучение производится на 150 эпохах. В результате нейронная сеть может предсказать цену закрытия на следующий день на основании данных предыдущих 50 дней. Среднеквадратичная ошибка MSE на тренировочной выборке составила 0.000585, на тестовой выборке – 0.000861.

Рисунок 1 – Прогнозирование цены закрытия на основе исторических данных

Также была разработана симуляция торговли на бирже с использованием разработанной стратегии. Стратегия ориентируется на прогноз нейронной сети на следующий день и сравнивает с ценой закрытия на сегодняшний день. Если прогнозируемая цена выше нынешней, тогда совершается покупка, иначе – продажа. При этом ввиду высокой

волатильности акции Apple было установлено дополнительное условие совершения продажи. Продажа совершается только при условии, что прогнозируемая цена на следующий день ниже нынешней цены закрытия, а также цена при продаже должна превышать цену покупки с комиссией, затраченной на покупку. Такое условие предотвращает множество покупок и продаж с незначительной разницей в цене, не покрывающей затраченную комиссию.

Изначально было взято 2000 долларов, для симуляции совершения сделок на бирже. При совершении каждой сделки происходит покупка или продажа 10 акций Apple. Установлена комиссия с коэффициентом 0.001 от каждой сделки. Итоговая сумма спустя полтора года использования стратегии составила 2769.48 долларов, что говорит о прибыли в 38,47% от изначальной суммы.

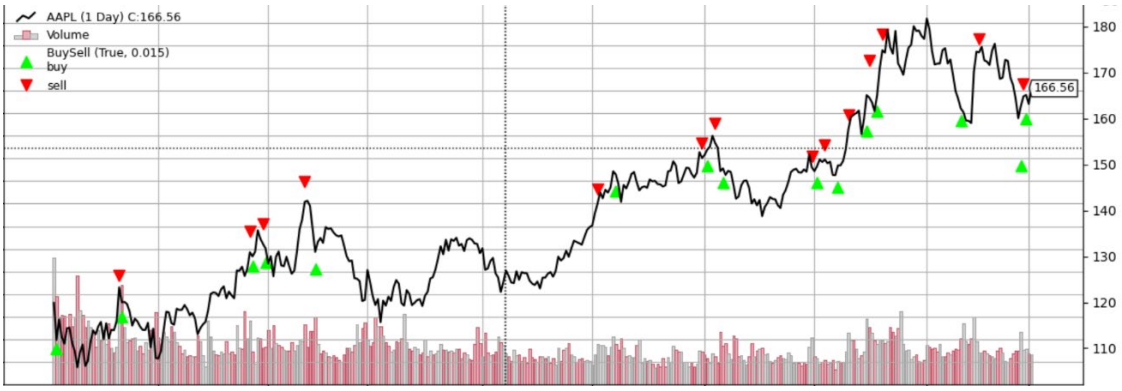


Рисунок 2 – Симуляция торговли на бирже

Применение машинного обучения и, в частности, искусственных нейронных сетей в прогнозировании котировок акций показывает высокий потенциал возможностей для финансовых аналитиков. Однако нейронная сеть не может предсказать динамику цен в зависимости от ситуаций, происходящих в мире. Примером служит сильное падение рынка в марте 2020 года из-за ограничений вследствие распространения Covid-19.

ЛИТЕРАТУРА

1. Iacomin R.: Stock market prediction. In: 19th International Conference on System Theory, Control and Computing (ICSTCC), Cheile Gradistei, pp. 200-205. (2015).
2. Сергиенко А. Г., Бугорский В. Н. Использование нейронных сетей для моделирования прогноза котировок ценных бумаг // Прикладная информатика, Вып. 3, – Изд.: М., 2008. – Стр. 3–11.
3. Backtrader. [Электронный ресурс]. Режим доступа: https://www.backtrader.com/docu/strategy/

УДК 004.032.26

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

ИСПОЛЬЗОВАНИЕ НЕЙРОННЫХ СЕТЕЙ ДЛЯ ИДЕНТИФИКАЦИИ ЛЮДЕЙ ПО ГОЛОСУ

Целью исследования было изучение и реализация алгоритма идентификации людей по голосу с использований нейронной сети. Допустим, имеется набор записей голоса без определённой связи между собой и с произвольным начитанным текстом. Длина записей также разнится, могут присутствовать звуковые искажения и междометия. Система должна однозначно определить, кто носитель голоса.

Для решения поставленной задачи была использована нейронная сеть, которая в качестве скрытых слоёв использует заранее обученную сеть-энкодер, основанный на архитектуре долгой краткосрочной памяти (LSTM) и преобразует звуковой сигнал в 256-мерный вектор. Процесс состоит из создания мел-спектрограммы и передачи её в энкодер для получения вектора. Этот вектор передаётся в полносвязную сеть без скрытых слоёв, что оценивает вектор

и выбирает оратора из известных. Итоговая сеть состоит из следующих слоёв: 40 входных нейронов, три скрытых слоя в 256 LSTM-модуля, промежуточный слой в 256 нейронов с функцией активации ReLU, выходной слой на N нейронов с логистической функцией активации (N – число известных ораторов).

Для обучения нейронной сети использовался набор данных с речью одиннадцати актёров. Аудиофайлы не проходили никакой предварительной обработки и не были проверены вручную и постфактум стало известно, что в части из них были эффекты эхо и радио (низкочастотный и высокочастотные фильтры в разных комбинациях). Максимальное число файлов с голосом одного актёра составило 2057, а минимальное – 90.

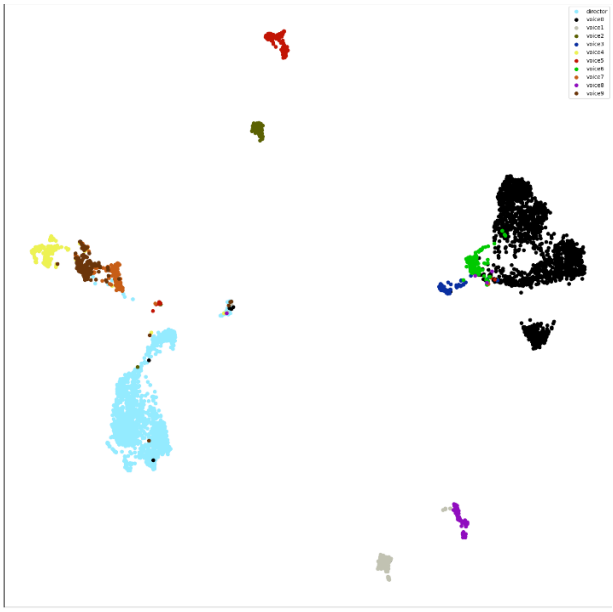


Рисунок 1 – Проекция полученных представлений на двумерное пространство.

Интересно то, что в полученной проекции можно наблюдать три разных кластера из чёрных точек. Возможно, в аудиозаписях одного актёра есть три разные эмоциональные окраски или интонации, что и создало эти кластеры. К тому же, полученную проекцию можно визуально разделить на две половины по диагонали и выделить мужские и женские голоса. Близко к центру находится кластер, где есть точки почти всех цветов. Возможно, там звуки кашля или междометия.

Лучшие результаты были достигнуты при обучении на выборке в 20 файлов на актёра и проверке на выборке в 60 файлов, точность была равна 93%.

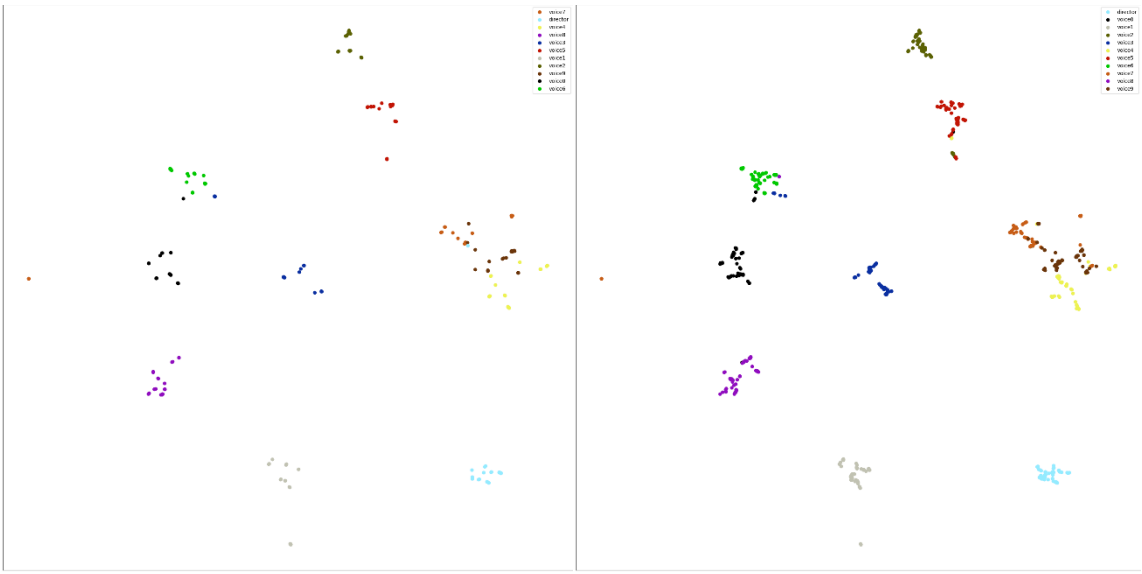


Рисунок 2 – Проекции обучающей (слева) и тестовой (справа) выборок.

Использование нейронных сетей сильно упрощают программное нахождение шаблонов и закономерностей, что хорошо себя показывает в том числе в распознавании голоса с целью идентификации пользователя. Предложенная в исследовании модель может быть уже использована в простой системе распознавания, но также имеет пространство для улучшения. Например, использование белого шума в аудиофайлах может положительно повлиять на качество обучения модели. Также есть возможность, что разделение аудиофайлов на короткие равные отрезки может увеличить обучающую и проверочную выборки и их равномерность, а также улучшить качество обучения.

ЛИТЕРАТУРА

1. Мел-купстральные коэффициенты (MFCC) и распознавание речи [Электронный ресурс]. Режим доступа: https://habr.com/ru/post/140828/
2. PyTorch Documentation [Электронный ресурс]. Режим доступа: https://pytorch.org/docs/stable/index.html
3. Энкодер [Электронный ресурс]. Режим доступа: https://github.com/CorentinJ/Real-Time-Voice-Cloning

УДК 621.319

Фомина Д. Д. (4 курс бакалавриата),
Римша А. С. (соискатель),
Большаков А. А., д.т.н., профессор

ОБРАБОТКА НЕПРИЕМЛЕМЫХ РИСКОВ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ В УСЛОВИЯХ ФИНАНСОВЫХ ОГРАНИЧЕНИЙ НА ОСНОВЕ КОМБИНИРОВАННЫХ ГЕНЕТИЧЕСКИХ АЛГОРИТМОВ

Важным фактором корректного функционирования АСУ ТП является их информационная безопасность (ИБ). Для отраслей, в которых защита АСУ ТП особенно критична, например, информационно-телекоммуникационных сетей и систем АСУ, функционирующих в сфере топливно-энергетических комплексов, возможен не только значительный материальный ущерб из-за выхода из строя дорогостоящего оборудования, но и негативные экономические последствия из-за нарушения поставки нефти или газа, приводящих к перебоям в электроснабжении, экологическим и/или гуманитарным катастрофам, а также человеческим жертвам.

Одним из перспективных подходов к достижению целенаправленного управления процессами ИБ, прежде всего АСУ ТП, является риск-ориентированный подход, основанный на моделях и методах управления функциональными рисками и ресурсами АСУ ТП, ориентированный на преимущественное использование упреждающих мер [1, 2].

Принимая во внимание вышеперечисленные факторы, обеспечение гарантированного уровня ИБ АСУ ТП требует совершенствования методик оценки и обработки рисков ИБ. Следовательно, проведение теоретических исследований и решение практических задач в области оценки и обработки рисков ИБ АСУ ТП является актуальной задачей.

Целью работы является исследование имеющихся подходов и разработка нового метода и алгоритмов оценки рисков ИБ АСУ ТП, а также их обработки в условиях многоуровневой иерархической структуры.

Достижение поставленной цели, т.е. обеспечение требуемого уровня ИБ в АСУ ТП, непосредственно связано с решением следующих научных и технических задач:

– разработка модели АСУ ТП для описания различных видов активов, подверженных угрозам ИБ, и особенностей их взаимодействия при решении задачи управления рисками ИБ;

– разработка методики и алгоритма для оценки рисков ИБ на основе модели АСУ ТП и подбора оптимальной конфигурации (набора) защитных мер для системы защиты информации;

– создание программного обеспечения для управления рисками ИБ АСУ ТП и поиска лучших конфигураций по снижению риска и по эффективности затрат.

Одной из задач при построении системы защиты информации является составление набора защитных мер. При этом необходимо учитывать не только требования безопасности, но и затраты на внедрение и поддержание системы защиты информации. Для этого осуществляется переход к задаче оптимизации [3]. Ее идея заключается в выборе оптимальной последовательности реализации мер обработки рисков для обеспечения минимального остаточного риска для множества угроз в общем случае на множестве мер обработки риска.

В рассматриваемой задаче применяется критерий Вальда (критерий «максимина») и имеются финансовые ограничения, поэтому предлагается решать ее методом генетического алгоритма, лучше гибридного.

В настоящей работе предлагается метод оценки рисков информационной безопасности для АСУ ТП на основе гибридного генетического алгоритма (Рисунок 1). С учетом характеристик АСУ ТП использование гибридного генетического алгоритма (ГА) позволяет идентифицировать и оценить активы и угрозы и в результате получить степень воздействия угроз безопасности на конкретную информационную безопасность АСУ ТП. Этот метод может служить основой для принятия решений по снижению рисков информационной безопасности в системе управления.

Генетический алгоритм способен быстро найти во всей области поиска хорошие решения, однако может испытывать трудности в получении из них наилучших. Обычный оптимизационный метод может быстро достичь локального максимума, но не может найти глобальный. Сочетание двух алгоритмов позволяет использовать преимущества обоих.

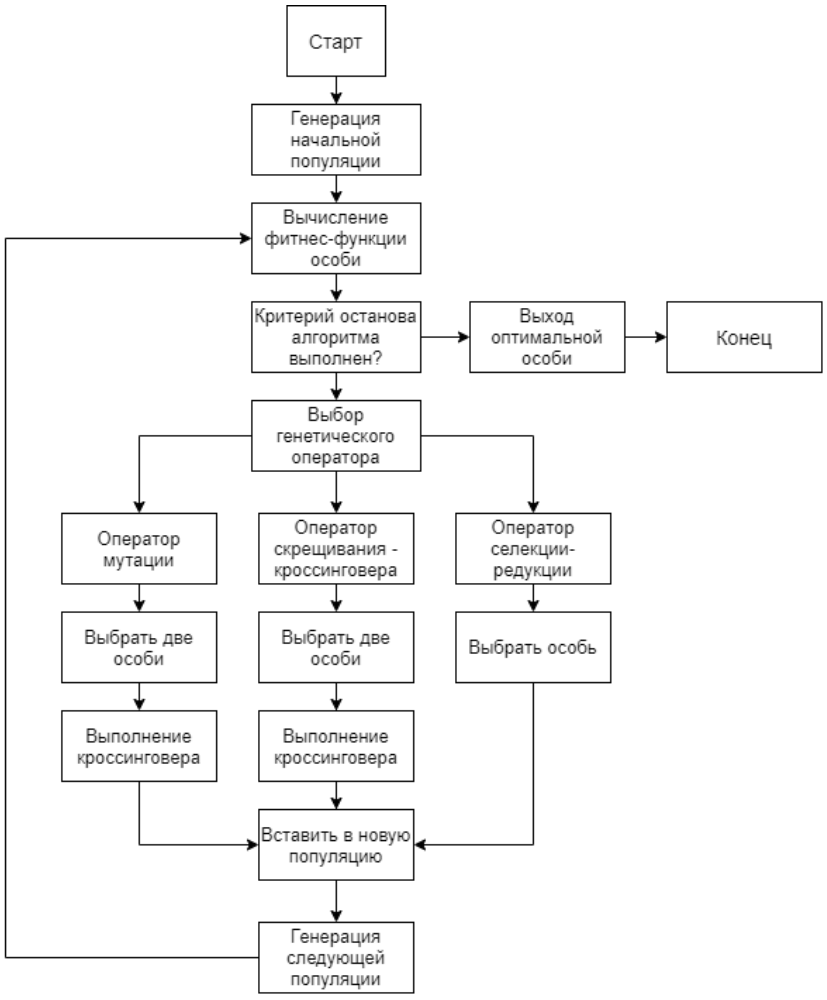


Рисунок 1 – Схема гибридного генетического алгоритма оценки рисков ИБ АСУ ТП.

Предложенный подход гибридизации алгоритма основан на включении локальной оптимизации наряду с генетическими операторами в классическую блок-схему простого генетического алгоритма. В этом случае локальная оптимизация применяется к каждому полученному потомку, чтобы "сдвинуть" его в сторону локального оптимума перед внедрением его в новую популяцию. Для кодирования допустимых решений используется метод десятичного кодирования, для операции кроссинговера выбирается метод дискретной реорганизации рекомбинации вещественных значений, а для выполнения операции мутации выбирается метод вариации вещественных значений. Хромосома искомой задачи содержит

два бита гена. Следовательно, операция кроссовера использует метод одноточечного кроссовера дискретной рекомбинации.

Для написания программы, реализующей комбинированный генетический алгоритм, выбран язык программирования C++. Применение гибридного ГА для оценки рисков ИБ АСУ ТП обеспечит приемлемый по скорости способ их вычисления.

ЛИТЕРАТУРА

1. Римша А. С., Римша К. С. Анализ средств обеспечения информационной безопасности АСУ ТП газодобывающих предприятий // Прикаспийский журнал: управление и высокие технологии, 2019. – № 3. – С. 102-121.
2. Rimsha A. S., Rimsha K. S. The Problem of Selecting APCS' Information Security Tools // Kravets A., Bolshakov A., Shcherbakov M. (eds) Cyber-Physical Systems: Industry 4.0 Challenges. Studies in Systems, Decision and Control, 2019. – vol. 260. – pp. 211-223. Scopus.
3. Большаков, А.А. Математическое обеспечение для анализа и управления информационными рисками в АСУ ТП газодобывающих предприятий / А.А. Большаков, А.С. Римша // Математические методы в технике и технологиях. – Санкт-Петербург, 2021. – № 10. – С. 73–78.

Секция «Подходы к разработке программного обеспечения на основе технологий Dell Technologies»

УДК 004.02

Агеев Д. Ю. (2 курс магистратуры),
Воинов Н. В., к.т.н, доцент

ПОВЫШЕНИЕ ОТКАЗОУСТОЙЧИВОСТИ ДЛЯ НАДЕЖНОГО СОЕДИНЕНИЯ МЕЖДУ КЛИЕНТОМ И СЕРВЕРОМ

Для поддержания отказоустойчивости в облачных приложениях используется подход failover, когда клиент при сбое сервера использует резервные серверы для продолжения работы. Этот подход реализуется благодаря методу Recovery Block [1] при обработке ошибок. В таких системах время простоя клиента и вероятность его сбоя зависят от вероятности сбоя сервера. В данной работе предлагается метод по улучшению стандартного Recovery Block, который будет использовать алгоритм принятия решения на основе ошибок, произошедших во время работы приложения. Для этого задействуются следующие элементы: клиент, который использует постоянное подключение только к одному серверу; группа серверов коммуникации (СК), расположенных в разных дата центрах по всему миру и обсуживающихся различными командами.

Чаще всего проблемы с надежностью связаны со стабильностью серверов: ошибки с DNS для конкретного региона, нарушения базы данных на определенном кластере, человеческий фактор, ошибки при обновлении серверов на новую версию [2]. Все это создает условия, при которых для непрерывной работы целой системы (связки СК и клиента) необходимо поддерживать методы восстановления на каждой из сторон. Данная работа предлагает специальный метод на основе Recovery Block, позволяющий уменьшить вероятность сбоя и время простоя.

Рассмотрим особенности предлагаемого решения. Если предположить, что сбои, которые происходят во время взаимодействия клиента и сервера, являются временными и их причина находится на стороне сервера, то можно попытаться выстраивать их закономерность исходя из результатов исполнения запросов. Эти факторы создают уникальную ситуацию, когда предоставляемое решение, основанное на статистике использования, может способствовать сокращению простоев и ошибок за счет выбора наилучших из доступных серверов.

Предлагаемый алгоритм (Рисунок 1) состоит из следующих шагов:

1. Создание нового события, для которого необходим запрос к серверу, определение типа для события.
2. Выбор самого первого сервера из предоставленного списка.
3. Установка соединения с сервером.

a. В случае успешной операции перейти к пункту 9.
b. В случае неуспешной операции перейти к пункту 4.

4. Определить тип ошибки от сервера на основе полученного ответа.
5. Увеличить счетчик типизированной ошибки для сервера, увеличить общий счетчик ошибок и обновить полученный файл.
6. Создать, если еще не существует, новый объект для сервера в соответствии с определенной ошибкой.
7. Выбрать новый сервер из списка в соответствии с наибольшей вероятностью успеха для определенной операции. Если были перепробованы все серверы, которые обладают информацией об ошибке такого типа, то начать обход других серверов в соответствии с их совокупной вероятностью.
8. В случае, когда ни одна операция не увенчалась успехом, система переводит событие в состояние неуспешного подключения, где клиент с периодичностью N будет повторять алгоритм до успешного подключения. При достижении N границы K событие удаляется.
9. Когда операция завершилась успешно, необходимо сохранить информацию о ней и записать её в файл. Сервер в таком случае не меняется до тех пор, пока не произойдет ошибка.
10. С периодичностью времени T ранжировать список серверов в соответствии с их вероятностями для этого клиента.

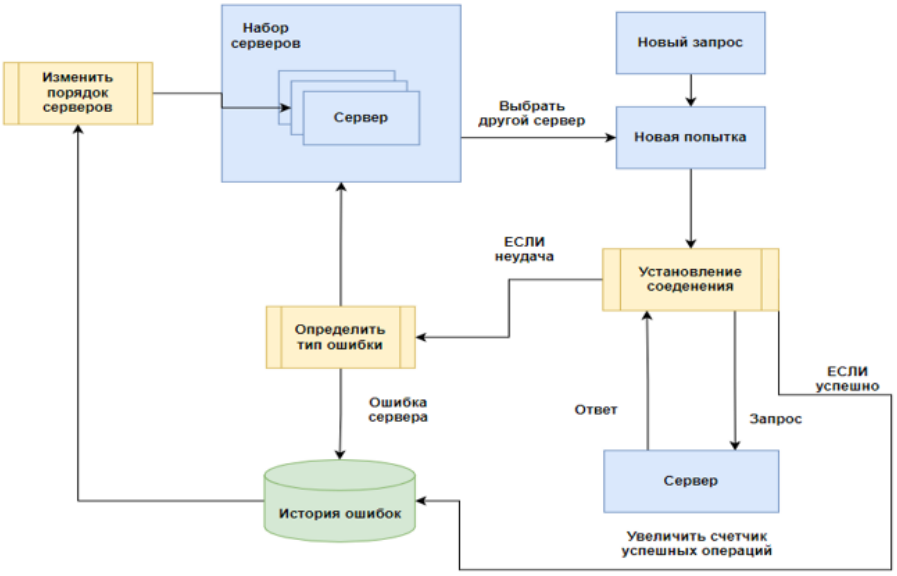


Рисунок 1 - Графическая схема алгоритма.

ЛИТЕРАТУРА

1. Distributed OAIS-Based digital preservation system with HDFS technology / N. Voinov, P. Drobintsev, V. Kotlyarov, I. Nikiforov // 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.Yu Zhibin, Bei Zhendong, Qian Xuehai. Datasize-Aware High Dimensional Configurations Auto-Tuning of In-Memory Cluster Computing // SIGPLAN Not. – 2018. – mar. – Vol. 53, no. 2. – P. 564–577.
2. Васильченко, А. В. Метод автоматической проверки лимитов ресурсов облачного провайдера AWS / А. В. Васильченко, 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. 146-147.

УДК 004.052.3

Варламов Д. А. (1 курс магистратуры),
Никифоров И. В., к.т.н., доцент,
Устинов С. М., д.т.н., профессор

ПОДХОД ДЛЯ ПОВЫШЕНИЯ СТАБИЛЬНОСТИ ИТ-ИНФРАСТРУКТУРЫ НА ОСНОВЕ ИНФОРМАЦИИ О ЕЕ СБОЯХ

Для компаний важна надежность и доступность внутренних сервисов, так как стабильность работы этих сервисов влияет на скорость и качество работы всех программистов, работающих в компаниях [1]. Любое время простоя этих сервисов влечет для компании убытки. Аналитика предыдущих отключений этих сервисов призвана найти наиболее критичные и уязвимые элементы инфраструктуры, склонные к поломке, чтобы заранее предсказывать, какие из элементов имеют большую вероятность сбоя [2]. Такой подход позволяет решать проблемы еще до их появления, а также позволяет сократить время поиска источника проблемы. Поэтому актуальной является необходимость создания, улучшения и автоматизации подходов для повышения стабильности ИТ-инфраструктуры.

Цель работы - повышение стабильности ИТ-инфраструктуры за счёт создания и реализации в программном средстве подхода к анализу сбоев, отличительной особенностью которого является учет статистики предыдущих отключений.

ИТ-инфраструктура (IT infrastructure) [3, 4] — это система аппаратного обеспечения, программного обеспечения, оборудования и сервисных компонентов, которая поддерживает доставку бизнес-систем и процессов с поддержкой информационных технологий. Основные компоненты ИТ-инфраструктуры, которые мы выделяем в нашей работе: программное обеспечение (software), аппаратное обеспечение (hardware) и компьютерные сети.

Надежность (Reliability) [5] – свойство ИТ-инфраструктуры, которое отвечает за её бесперебойную работу. Достигается за счет качественных компонентов, грамотной настройки и эффективного обслуживания.

Доступность (Availability) [5] – свойство ИТ-инфраструктуры, которое отвечает за возможность из любого места в любой момент времени иметь доступ к информационным и программно-сервисным службам (электронная почта, сетевой принтер, удаленный доступ к информации и др.).

Отключение (Outage) – период времени, в течение которого ИТ-инфраструктура не является доступной.

Деградация (Degradation) – период времени, в течение которого ИТ-инфраструктура не является надежной.

Для оценки эффективности предлагаемого подхода необходимо использовать численные показатели. На основе анализа существующих подходов к подсчёту ключевых показателей стабильности нами предлагается использовать два показателя для оценки эффективности подхода: доступность и надежность.

Доступность вычисляется по следующей формуле:

$$Availability = ((AST-OT)/AST)*100\%,$$

где AST (Agreed Service Time) – согласованное время обслуживания, а OT (Outage Time) – время отключения.

Надежность вычисляется по следующей формуле:

$$Reliability = ((AST-DT)/AST)*100\%,$$

где AST (Agreed Service Time) – согласованное время обслуживания, а DT (Degradation Time) – время деградации.

На Рисунке 1 представлена общая архитектура программного средства. Она состоит из нескольких модулей, работающих независимо. В верхней части рисунка изображена ИТ-инфраструктура, состоящая из трех компонентов: программное обеспечение (Software), аппаратное обеспечение (Hardware) и компьютерные сети (Networking). Затем идут модули

сбора данных мониторинга и логов. Они отвечают за сбор данных мониторинга и логов от разных компонентов ИТ-инфраструктуры: программного обеспечения, аппаратного обеспечения и компьютерных сетей [6]. Далее располагается модуль хранения конфигурации ИТ-инфраструктуры. Он отвечает за хранение внутренних зависимостей друг от друга компонентов ИТ-инфраструктуры. Затем расположен модуль хранения данных о текущем состоянии ИТ-инфраструктуры. Он отвечает за хранение данных о состояниях всех компонентов ИТ-инфраструктуры в данный момент времени и предоставляет API для изменения этих данных модулями сбора данных мониторинга и логов [7]. Он отправляет данные в модуль хранения исторических данных о состояниях ИТ-инфраструктуры. Также он отвечает за хранение исторических данных о состояниях компонентов ИТ-инфраструктуры. Далее располагается модуль анализа данных. Он отвечает за анализ отключений ИТ-инфраструктуры, поиск зависимостей между этими отключениями и выдачу рекомендаций по повышению стабильности. В самом конце расположен модуль визуализации данных. Он отвечает за визуализацию данных о состоянии ИТ-инфраструктуры.

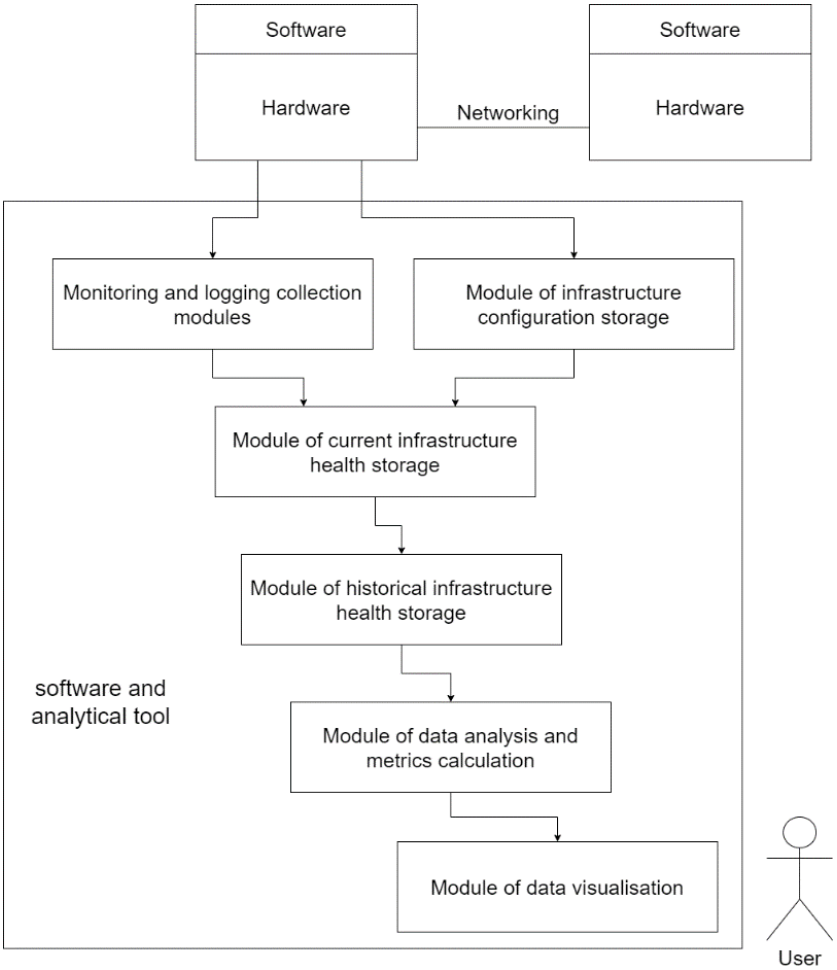


Рисунок 1 – Архитектура программного средства.

Данный подход был применен на практике в одной из производственных компаний. На основе информации, полученной при помощи созданного нами программного средства, удалось получить рекомендации по повышению стабильности одного из сервисов. Была найдена проблема, которая связана с одновременными каждодневными бэкапами, происходящими сразу на нескольких хостах. Сетевая файловая система не справлялась с большой одновременной нагрузкой, из-за чего сервис переходил в состояние отключения.

В конце января 2022 года были применены изменения: ежедневные бэкапы на каждом хосте стали выполняться в разное время, что уменьшило нагрузку на сетевую файловую систему.

До наших изменений показатели стабильности сервиса за январь 2022 года были следующими:

доступность 98.502%, надежность 92.584%.

После наших изменений показатели стабильности сервиса за февраль стали следующими:

доступность 98.696%, надежность 94.439%.

Таким образом, предложенный подход смог повысить доступность сервиса на 0.194%, а надежность на 1.885%

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 / С. Унагаев // Системный администратор. – 2013. – № 4(125). – С. 40-41.
4. Gartner Glossary [Электронный ресурс]. – URL: https://www.gartner.com/en/information-technology/glossary/it-infrastructure
5. BMC [Электронный ресурс]. – URL: https://www.bmc.com/blogs/reliability-vs-availability/
6. 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
7. 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.

УДК 004.05

Денисенко Б.А., Тянутов М.В., Кубов Н.А., Горшечников И.Д. (3 курс бакалавриата),
Никифоров И.В., к.т.н., доцент

АНАЛИЗ ОТКРЫТЫХ ИСТОЧНИКОВ ДАННЫХ ПО НАГРУЗКАМ НА ЦОД

Современная сфера услуг во всех областях жизнедеятельности человека опирается на использование многочисленных центров обработки данных (ЦОД) [1,2]. Благодаря этому количество ЦОД по всему миру кратно возрастает с каждым годом, а вместе с этим растет и энергопотребление, которое становится все большей проблемой [3]. Чтобы повысить эффективность использования ресурсов, необходимо понять характеристики рабочей нагрузки и использование машин в крупномасштабных облачных дата-центров [4]. Также необходимо выбрать правильные метрики оценки параметров ЦОД и осознавать их потенциальные ограничения [5].

Одной из наиболее простых и распространенных метрик является Server Compute Efficiency [6]. Эта метрика высчитывается как среднее значение использования машины (чаще

всего средняя нагрузка на CPU) за определенный промежуток времени. Есть также и расширенный вариант Data Center Compute Efficiency [6], распространяющийся на все машины сразу и позволяющий определить общую вычислительную эффективность ЦОДа. Используя DCcE и ScE, операторы центров обработки данных могут постоянно улучшать и повышать эффективность своих вычислительных ресурсов [6].

Ещё одним важным типом метрик являются энергетические метрики, однако в открытых источниках нам не удалось обнаружить информацию об энергопотреблении ЦОДов, поэтому в нашей работе мы сосредоточимся только на доступных нам данных.

Целью работы является исследование открытых источников данных на предмет расчета метрик производительности, анализ которых позволяет сравнить и повысить эффективность использования ресурсов. В ходе работы нами был проведен анализ ряда датасетов из открытых источников:

1) Google cluster management software and systems traces - один из самых популярных датасетов для исследования работы ЦОД. В ходе анализа данных из этого набора были рассчитаны метрики ScE и DCcE, по которым можно сделать вывод, что кластер большую часть времени простаивает без какой-либо нагрузки. Результаты вычисления метрик представлены на Рисунке 1.

На графике приведено сравнение нагрузки на CPU для двух машин из кластера.

Для них же рассчитаны ScE метрики:

Machine 1 ScE = 0.007871 %

Machine 2 ScE = 0.028543 %

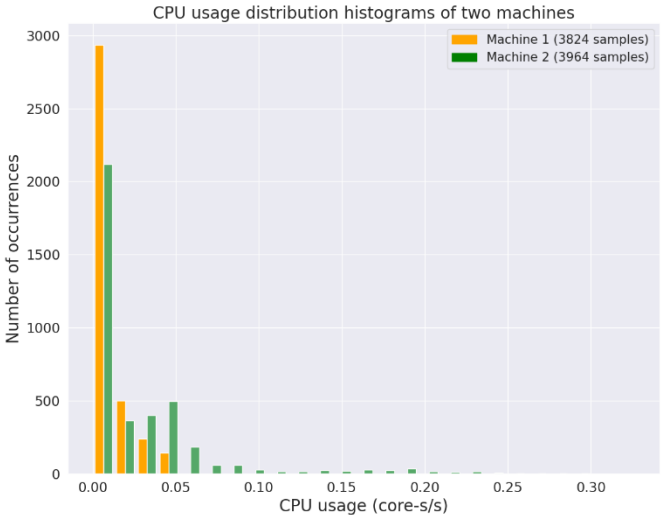


Рисунок 1 – нагрузка на CPU для двух серверов

2) OpenCloud Hadoop workload - это исследовательский кластер Университета Карнеги-Меллона (CMU), которым управляет Parallel Data Lab CMU. В результате анализа датасета от Hadoop, стало ясно, что он содержит только информацию о выполнении джобов и тасков, но не содержит информации о загруженности дата-центра. Таким образом этот датасет можно использовать для оптимизации выполнения джобов, но он не подходит для расчета метрик производительности.

3) GWA-T-12 Bitbrains - это распределенный ЦОД от Bitbrains, который является поставщиком услуг, специализирующимся на управляемом хостинге и бизнес-вычислениях для предприятий.

В результате анализа, были обработаны данные работе виртуальных машин и рассчитаны метрики ScE для каждой.

На Рисунке 2 отображен процент средней загрузки CPU (что также является метрикой ScE) и соответствующий этому проценту вес.

DCcE для данного датасета составляет около 7%, что значительно превосходит данную метрику для Гугл кластера и в целом является типичным значением для ЦОДа.

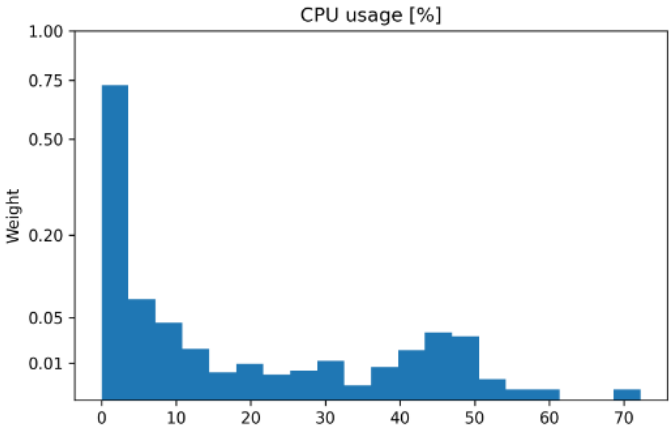


Рисунок 2 – средняя загрузка CPU

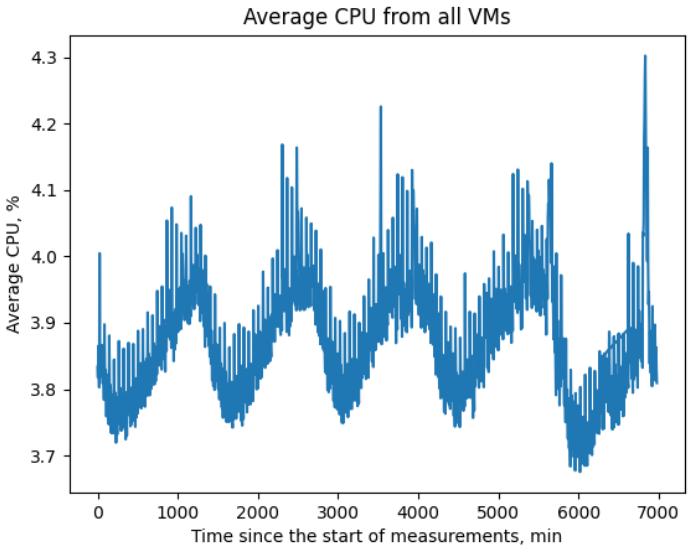


Рисунок 3 – медианное значение нагрузки на CPU

4) Azure Public Dataset содержит две репрезентативные трассировки рабочей нагрузки виртуальной машины (ВМ) Microsoft Azure, собранные в 2017 и 2019 годах. В результате анализа данного датасета был получен график медианных значений всех виртуальных машин по среднему ЦПУ.

По графику на Рисунке 3 можно увидеть, в какой промежуток времени происходит большая нагрузка на ЦОД (периодичность около 1500 минут, что соответствует суткам). Также можно увидеть по графику, что больше половины виртуальных машин используют менее 4% ЦПУ большую часть времени, что говорит о их малом использовании.

Таким образом, нами были рассмотрены открытые источники по центрам обработки данных. В результате исследования были рассчитаны некоторые метрики, которые позволяют сравнить работу дата-центров, но стоит отметить, что для полного анализа требуется расчет и других не менее важных метрик, например, энергетических, данными для расчета которых найденные открытые источники не обладают.

ЛИТЕРАТУРА

1. А. И. Акназарова, И. В. Никифоров, О. В. Прокофьев, Высокопроизводительная гибкомасштабируемая архитектура программного средства для обработки данных 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. 203-205.
2. Р. Д. Чусов, Н. В. Воинов, Разработка алгоритма и приложения для построения филогенетических деревьев по большим объемам данных // 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. 138-140.
3. Martijn Koot, Fons Wijnhoven, Usage impact on data center electricity needs: A system dynamic forecasting model, Applied Energy, Volume 291, 2021, 116798, ISSN 0306-2619, https://doi.org/10.1016/j.apenergy.2021.116798
4. C. Lu, K. Ye, G. Xu, C. -Z. Xu and T. Bai, "Imbalance in the cloud: An analysis on Alibaba cluster trace," 2017 IEEE International Conference on Big Data (Big Data), 2017, pp. 2884-2892, doi: 10.1109/BigData.2017.8258257.
5. V. Reddy, B. Setz, G. Rao, G. Gangadharan and M. Aiello, "Metrics for Sustainable Data Centers" in IEEE Transactions on Sustainable Computing, vol. 2, no. 03, pp. 290-303, 2017. doi: 10.1109/TSUSC.2017.2701883.
6. M. Blackburn, “The green grid data center compute efficiency metric: DCcE,” The Green Grid, White Paper-34, 2010.

УДК 004.051

Лялин Д.С. (4 курс бакалавриата),
Мамадалиев Ш.Р. (1 курс магистратуры),
Никифоров И.В., к.т.н., доцент,
Устинов С.М., д.т.н., профессор

ПРОЕКТИРОВАНИЕ СИСТЕМЫ МОНИТОРИНГА И РАСЧЕТА МЕТРИК ДЛЯ ЭФФЕКТИВНОГО ИСПОЛЬЗОВАНИЯ РЕСУРСОВ ЦЕНТРОВ ОБРАБОТКИ ДАННЫХ

За последние десять лет количество центров обработки данных выросло с 500 тысяч до 8 миллионов, а показатели потребляемой центром обработки данных электроэнергии удваиваются каждые четыре года [1]. В связи с этим растет необходимость в эффективной оценке работы ЦОД для снижения расходов на его эксплуатацию. Существует множество готовы решений, рещающих это задачу частично: Cacti, Zabbix, OpenNMS. Однако, в большинстве случаев, акцент этих систем мониторинга смещен в сторону сетевого мониторинга. Для решения задачи оценки эффективности работы ЦОД была спроектирована система мониторинга и расчета метрик центра обработки данных на базе инструментов Grafana [2] и Prometheus [3]. Система обладает свойствами гибкого масштабирования и обработки данных в режиме реального времени [4], а также позволяет на ранних этапах определить простои оборудования и предупредить администратора системы [5].

Инструмент Prometheus — это база данных временных рядов с возможностью присоединения экосистемы необходимых инструментов. Для описываемой системы мониторинга ЦОД одним из этих инструментов является процесс Node Exporter [6], обеспечивающий сбор и передачу серверу системных метрик центра. С помощью Node Exporter становиться возможным извлечение метрик через HTTP-вызовы к определенным конечным точкам, указанным в конфигурации Prometheus. Таким образом, может быть произведена настройка сбора (скрейпинга, англ. «scrapping» [7]) данных о нагрузке процессора, оперативной памяти и других показателей ЦОД через определенные интервалы времени.

Инструмент Grafana — это веб-приложение для визуализации, мониторинга и анализа данных. В системе выполняет функцию расчета метрик и их визуализации [8]. Приложение предоставляет возможность создания и тонкой настройки наборов панелей для отображения определенных метрик (дашбордов, англ. «dashboard» [9]). Универсальность каждой панели позволяет создавать актуальное представление необходимых для оценки ЦОД метрик. Также Grafana поддерживает настройку уведомлений, что может быть полезным при дальнейшем расширении предложенной системы.

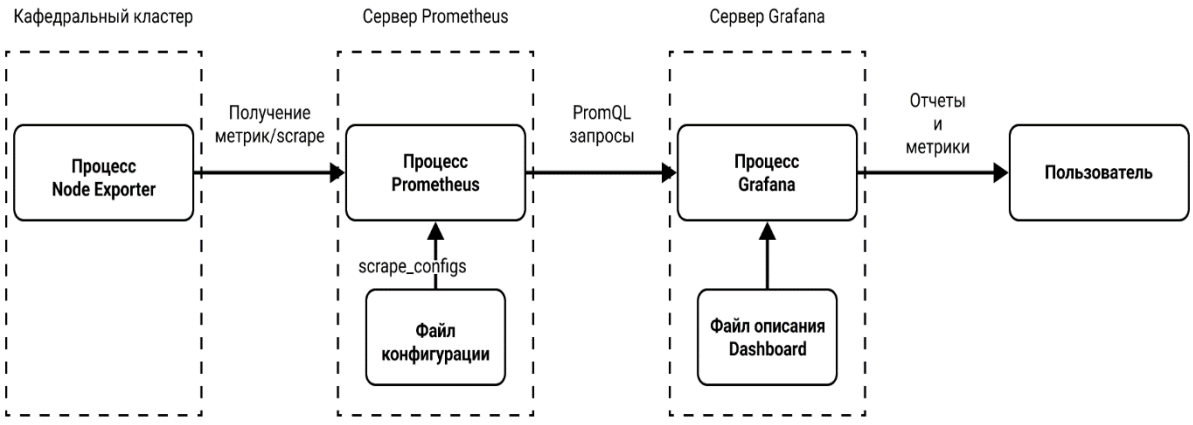


Рисунок 1 – Архитектура системы мониторинга и расчета метрик

Предложенная архитектура, изображенная на Рисунке 1, позволяет осуществить расчет метрик по актуальным данным ЦОД и представить их пользователю. В ней процесс Node

Exporter передает данные в базу данных Prometheus через определенные промежутки времени. Далее, по запросу пользователя, инструмент Grafana рассчитывает необходимые метрики и представляет их в дашборде, согласно его конфигурации.

Как уже было сказано ранее, альтернативные решения вроде Cacti и Zabbix больше подходят для мониторинга сети, что может сделать проблематичным расчет метрик эффективности ЦОД, зависящих от использования других ресурсов, таких как память или процессор. Спроектированная система мониторинга, наоборот, позволяет проводить более точную и гибкую оценку эффективности из-за высокой возможности настройки и может быть сконфигурирована под специфичные цели и метрики.

Для системы были сформулированы и описаны функциональные и нефункциональные требования, представленные в Таблице 1.

Таблица 1 – Функциональные и нефункциональные требования к системе.

Идентификатор	**Описание**
NNFUNC - 1	Данные о кафедральном кластере, собранные разработанным прототипом системы мониторинга, должны храниться в базе данных временных рядов Prometheus.
NNFUNC - 2	Веб-приложение Grafana следует использовать для визуализации результатов, рассчитанных разработанным прототипом системы мониторинга.
NNFUNC - 3	Сбор метрик и мониторинг кластера должны выполняться через процесс Node Exporter в Prometheus.
FUNC - 1	Полученный дистрибутив разработанного прототипа системы мониторинга должен представлять собой ZIP-архив, содержащий элементы из подразделов FUNC-1.
FUNC - 1.1	Полученный дистрибутив должен содержать необходимую документацию прототипа.
FUNC - 1.2	Полученный дистрибутив должен содержать скрипты запуска и установки прототипа.
FUNC - 1.3	Полученный дистрибутив должен содержать наборы данных или скрипт для создания указанных наборов данных для расчета метрик эффективности.
FUNC - 1.4	Полученный дистрибутив должен содержать результаты расчетов и их визуализацию.
FUNC - 2	Разработанный прототип системы мониторинга должен уметь собирать данные, описанные в подразделах FUNC-2.
FUNC - 2.1	Данные об использовании CPU должны быть собраны из кластера.
FUNC - 2.2	Данные об использовании RAM должны быть собраны из кластера.
FUNC - 2.3	Данные об использовании сети должны собираться из кластера.
FUNC - 2.4	Значение IOPS должно быть получено из кластера.
FUNC - 2.5	Значение задержки диска должно быть получено из кластера.
FUNC - 2.6	Данные об использовании дискового пространства должны быть получены из кластера.
FUNC - 2.7	Данные об энергопотреблении должны быть собраны из кластера.
FUNC - 2.8	Данные о температуре должны быть собраны из кластера.
….	….

Используемые технологии: Grafana, Prometheus.

Демонстрация варианта результата работы прототипа представлена на Рисунке 2.

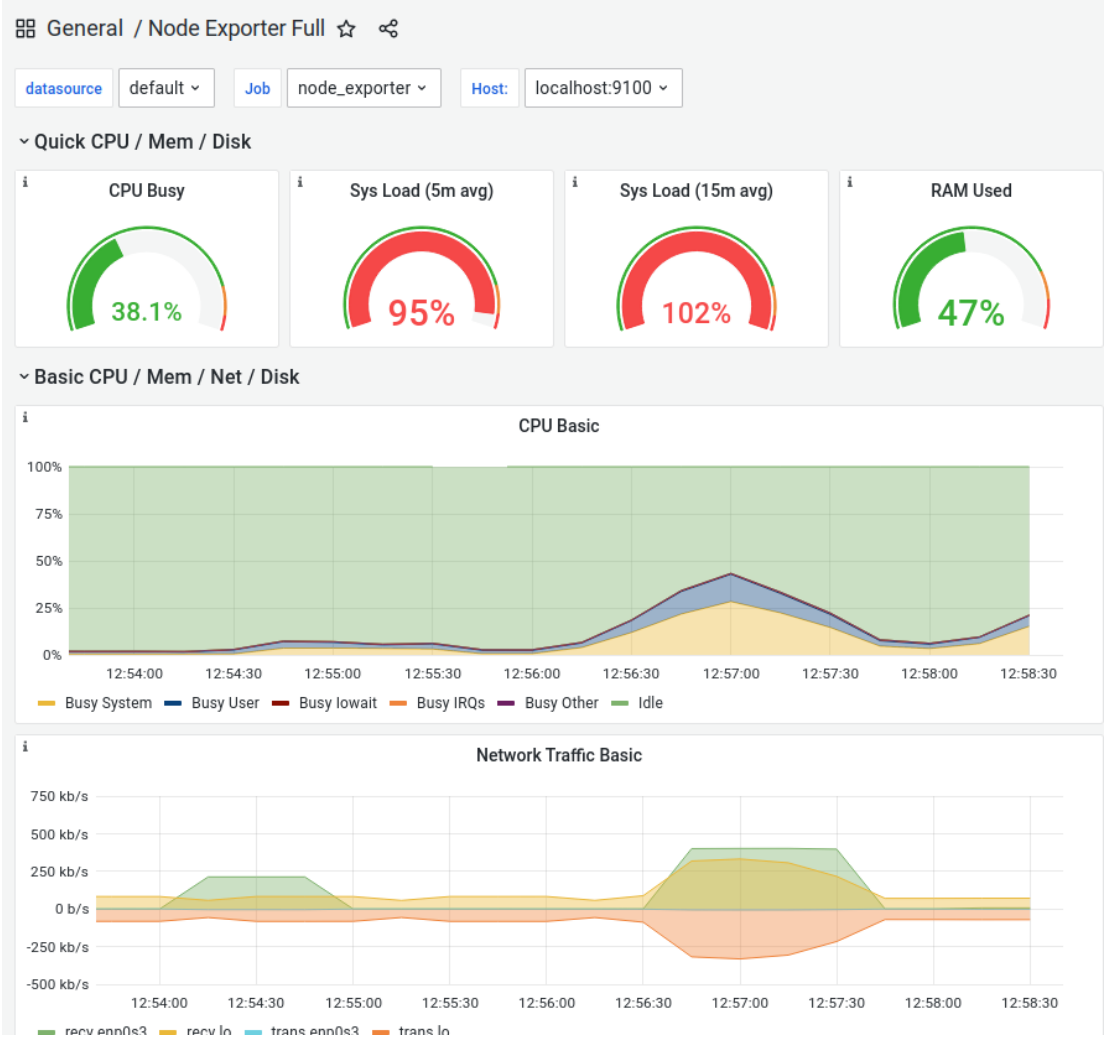


Рисунок 2 – Демонстрация визуализации базовых метрик

Таким образом, был разработан прототип системы мониторинга кластера с помощью инструментов Prometheus и Grafana. Использование подобной системы может решить задачу оценки эффективности работы ЦОД по необходимым для конкретного проекта метрикам и тем самым оптимизировать расходы на эксплуатацию. На основе предлагаемой системы мониторинга можно собирать статистику использования ресурсов кластера и осуществлять подсчет метрик (TCO, TCS, ScE, DCcE, PUE, DCPE, DPPE и т. д.) для эффективной оценки стоимости оборудования и экономии электроэнергии, что в свою очередь позволит снизить негативное влияние на окружающую среду.

Работа выполнена в рамках совместного проекта СПбПУ и компании «Dell Technologies» и лаборатории «Data Engineering Lab» Санкт-Петербургского политехнического университета Петра Великого.

ЛИТЕРАТУРА

1. The Data Center Dilemma: Is Our Data Destroying the Environment? [Электронный ресурс]. Дата обращения: 08.03.2022. https://www.datacenterknowledge.com/industry-perspectives/data-center-dilemma-our-data-destroying-environment
2. Grafana: The open observability platform | Grafana Labs [Электронный ресурс]. Дата обращения: 09.03.2022. https://grafana.com
3. Prometheus - Monitoring system & time series database [Электронный ресурс]. Дата обращения: 11.03.2022. https://prometheus.io
4. А. И. Акназарова, И. В. Никифоров, О. В. Прокофьев, Высокопроизводительная гибкомасштабируемая архитектура программного средства для обработки данных 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. 203-205.

5. К. А. Парахин, И. В. Никифоров, Разработка гибкой и высокопроизводительной платформы для анализа машинных данных // 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. 191-192.
6. Node exporter | Grafana Labs [Электронный ресурс]. Дата обращения: 09.03.2022. https://grafana.com/oss/prometheus/exporters/node-exporter [Электронный ресурс]. Дата обращения: 12.03.2022. https://www.octoparse.com/blog/web-scraping-introduction
7. D. S. Eyzenakh, A. S. Rameykov, I. V. Nikiforov, High performance distributed web-scraper // Proceedings of the Institute for System Programming of the RAS. – 2021. – Vol. 33. – No 3. – P. 87-100. – DOI 10.15514/ISPRAS-2021-33(3)-7.
8. А. В. Дымченко, Л. К. Птицына, Модельно-аналитический интеллект мультиагентных систем раннего предупреждения // Актуальные проблемы инфотелекоммуникаций в науке и образовании (АПИНО 2020): Сборник научных статей IX Международной научно-технической и научно-методической конференции. В 4-х т., Санкт-Петербург, 26–27 февраля 2020 года. – Санкт-Петербург: Санкт-Петербургский государственный университет телекоммуникаций им. проф. М.А. Бонч-Бруевича, 2020. – С. 291-294.
9. Dashboards | Grafana Labs [Электронный ресурс]. Дата обращения: 12.03.2022. https://grafana.com/docs/grafana/latest/dashboards/

УДК 004.052.3

Максимчук В. А. (2 курс магистратуры),
Устинов С. М., д.т.н., профессор

ПОВЫШЕНИЕ ПРОИЗВОДИТЕЛЬНОСТИ ВЗАИМОДЕЙСТВИЯ МИКРОСЕРВИСОВ С СХД PRAVEGA ДЛЯ АНАЛИТИЧЕСКОЙ ОБРАБОТКИ ДАННЫХ

В настоящее время наблюдается постоянный рост количества обрабатываемой информации в различных системах. Особую популярность набирают системы по аналитической обработке данных в режиме близкому к реальному времени [1]. Данные системы строятся на сервис-ориентированной или микросервисной архитектуре, которая также имеет весьма большую популярность в последнее время. В связи с этим в данных системах используются специальные инструменты, такие как СХД Pravega. Она позволяет обеспечить высокопроизводительную отказоустойчивую передачу данных без потерь между компонентами системы.

Система хранения данных Pravega представляет собой распределенную систему хранения и передачи данных классической архитектуры издатель-подписчик, основанной на потоках. Поток представляет собой высокопроизводительную неограниченную последовательность байтов, предназначенную только для добавления новой информации [2]. В данных условиях появляется задача реализации способов взаимодействия сервисов и микросервисов с данной системой.

При решении вышеупомянутой задачи следует учитывать, что от реализуемого подхода прямо зависит производительность системы в целом - скорость передачи данных, их объем, а также скорость выполнения аналитической обработки на стороне микросервиса. В современной индустрии существует множество различных форматов данных и способов организации обмена ими между сервисами.

Одним из них является Apache Arrow – это платформа разработки ПО для организации высокопроизводительной обработки и передачи данных, которая легла в основу нескольких инструментов экосистемы Apache. Главной особенностью данной платформы является столбчатый формат хранения данных в памяти, который позволяет стандартизировать данные и увеличить производительность аналитической обработки [3]. Также немаловажным компонентом платформы Apache Arrow является фреймворк Flight, который позволяет организовать обмен данными в данном формате между клиентом и сервером при помощи вызова удаленных процедур на основе протокола gRPC [4].

Целью данной работы является повышение производительности взаимодействия микросервисов с СХД Pravega, при помощи усовершенствования способа передачи данных, внедрением технологии Apache Arrow. Данная цель достигается за счет решения следующих задач:

– проведение исследования существующих подходов взаимодействия с СХД Pravega;
– внедрение технологии Apache Arrow;
– разработка способа взаимодействия с СХД Pravega;
– реализация предложенного подхода взаимодействия.

Исследование используемых подходов взаимодействия с СХД Pravega показало, что на момент написания работы уже существуют несколько реализованных способов, поддерживаемых Pravega, которые включают в себя Spark connector, Flink connector, Hadoop connector [5]. Однако их анализ показал, что присутствуют некоторые проблемы. Spark connector и Hadoop connector представляют собой узконаправленные решения, которые позволяют реализовать высокопроизводительную передачу и аналитику данных, однако ограничены использованием одноименных технологий, что далеко не всегда необходимо. Flink connector реализован подобным образом, однако, представляет более общее решение, благодаря используемой технологии, но при его использовании возникает необходимость реализации тяжеловесного клиента на стороне микросервиса, что непосредственно влияет на производительность последнего.

Разрабатываемый подход направлен на решение вышеупомянутых проблем. С целью обеспечения выдвигаемых требований к коннекторам СХД Pravega, было принято решение оставить в основе нативный клиент, и перейти на архитектуру клиент-сервер с минималистичным клиентом, что позволит снизить нагрузку на микросервис и оставит возможность реализации предобработки данных при чтении и записи. На стороне сервера необходимо реализовать основную логику взаимодействия со всеми интерфейсами СХД. Также необходимо стандартизировать формат данных для обеспечения возможности получения и передачи данных с различными компонентами системы, избегая излишнего переформатирования данных (Рисунок 1).

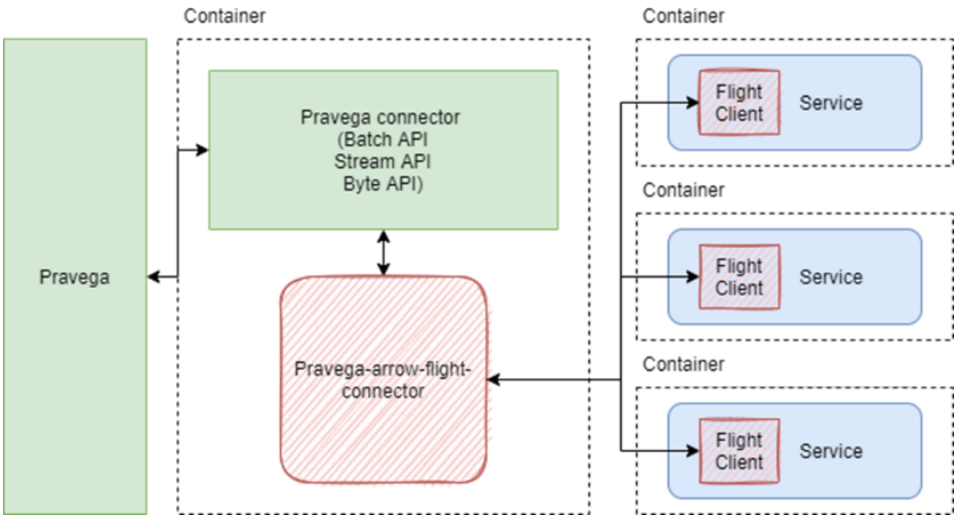


Рисунок 1 – Концептуальная схема Pravega-arrow-flight-connector

Для обеспечения стандартизации данных был внедрен столбчатый формат данных Apache Arrow. На стороне сервера был реализован модуль по обработке данных при чтении и записи согласно схеме, заданной микросервисом, что дает возможность передавать данные не

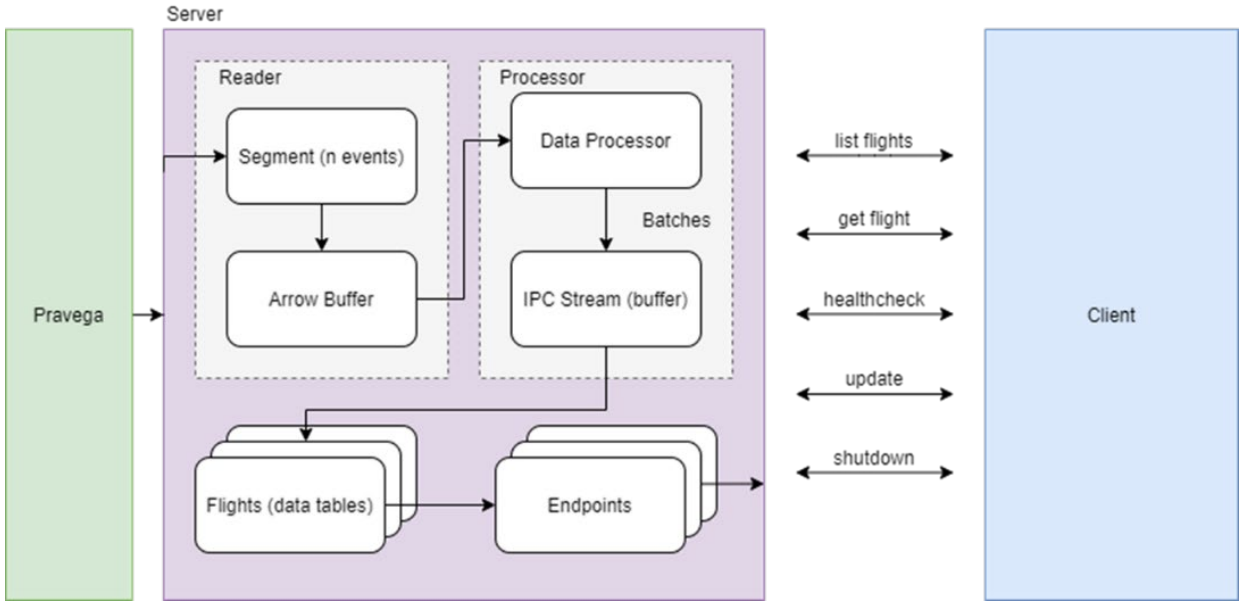


Рисунок 2 – Архитектурная схемы Pravega-arrow-flight-connector

отформатированные предварительно. Таким образом, данные при чтении из СХД записываются в буфер в памяти, а затем проходят обработку и, уже в столбчатом формате, записываются в таблицы данных, хранящиеся в памяти. Однако в данной реализации сохраняется возможность избежать данного процесса при передаче данных в столбчатом формате клиентом [6].

Для обеспечения высокопроизводительного обмена данными в столбчатом формате и реализации выбранной клиент-серверной архитектурой был использован фреймворк Apache Flight, который позволил выстроить гибкий способ взаимодействия клиента и сервера при помощи вызова удаленных процедур, а также обеспечить высокопроизводительную передачу данных между ними. Для таблиц (flights), хранящих в себе данные в столбчатом формате, создаются endpoints, к которым происходит подключение клиента для дальнейшей работы с данными (Рисунок 2). Также для одной таблицы может существовать несколько endpoint, благодаря чему появляется возможность различными микросервисам проводить манипуляции с одним и тем же набором данных [7]. В рамках данного подхода были разработаны несколько процедур - для чтения данных по параметрам, принудительного обновления данных на сервере, в случае необходимости такового, а также записи как в столбчатом формате, так и не отформатированных данных.

На данный момент система реализована в виде прототипа. Реализованный прототип позволил организовать производительное взаимодействие с СХД в одиночном режиме, сохранив легковесность и простоту достаточно гибкой реализации на стороне микросервиса. В дальнейшем планируется внедрить новые функциональные возможности как на стороне клиента, так и на стороне сервера, такие как взаимодействие с СХД в распределенном режиме.

ЛИТЕРАТУРА

1. 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
2. Г. Д. Тимошенский, Л. П. Котлярова, Исследование подходов и реализация автоматической конфигурации системы непрерывной интеграции для полностью облачных приложений //

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. 221-223.
3. Introducing Apache Arrow Flight: A Framework for Fast Data Transport, 2019, [Электронный ресурс]. URL: https://arrow.apache.org/blog/2019/10/13/introducing-arrow-flight/ (дата обращения 10.03.2022)
4. Introducing Apache Arrow Flight: A Framework for Fast Data Transport, 2019, [Электронный ресурс]. URL: https://arrow.apache.org/blog/2019/10/13/introducing-arrow-flight/ (дата обращения 10.03.2022)
5. N. Voinov, P. Drobintsev, V. Kotlyarov, I. Nikiforov, 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. T. Ahmad, N. Ahmed, J. Peltenburg and Z. Al-Ars, "ArrowSAM: In-Memory Genomics Data Processing Using Apache Arrow," *3rd International Conference on Computer Applications & Information Security (ICCAIS)*, 2020, P. 1-6, DOI 10.1109/ICCAIS48893.2020.
7. Lentner, Geoffrey. (2019). Shared Memory High Throughput Computing with Apache Arrow™. PEARC '19: Proceedings of the Practice and Experience in Advanced Research Computing on Rise of the Machines (learning). 1-2. 10.1145/3332186.3335197.

УДК 004.4

Петров А. А. (2 курс магистратуры),
Никифоров И. В., к.т.н., доцент

ПОДХОД ПЕРЕМЕЩЕНИЯ КОНФИГУРАЦИИ ESXI-КЛАСТЕРОВ МЕЖДУ VCENTER СЕРВЕРАМИ

На сегодняшний день гипервизоры и системы управления гипервизорами активно развиваются и внедряются в промышленности, так как позволяют эффективно использовать необходимые для работы информационных систем ресурсы, к которым относятся: вычислительные мощности процессоров, оперативная память, пропускная способность сети и другие [1]. При этом системы управления гипервизорами могут устаревать и/или выходить из строя, а также нуждаться в балансировке нагрузки, что влечет за собой проблемы, связанные с несовместимостью программного обеспечения с разными версиями систем управления или с уровнем интеграции приложений с ними. Эффективным решением представленных проблем является автоматизированная миграция конфигураций и ресурсов между экземплярами систем управления гипервизорами или восстановление такой конфигурации [2].

Целью работы являлось исследование существующих методов и алгоритмов миграции и восстановления конфигурации и на их основе предложение устойчивых к сбоям подходов автоматической миграции (и восстановления конфигурации) ESXi-кластеров между разными экземплярами vCenter серверов.

На сегодняшний день миграция ESXi-кластера наиболее хорошо и полно описана в работе компании VMware «*Moving a vSAN cluster from one vCenter Server to another*» [3]. Дополнительно есть ещё одна статья, описывающая процесс миграция ESX/ESXi-хостов, использующих Distributed Virtual Switch (DVS) [4]. Следует сказать, что в статье [5] предлагается использовать экспорт и импорт конфигурации DVS, что позволяет частично автоматизировать процесс. Остальные существующие работы, например, работа компании Nutanix [6] в основном пересказывают или уточняют статью [3] вместе со статьями [4, 5].

Существует также проект «Migrate-vCenter-with-vDS» [7], который автоматизирует процесс миграции ESXi-кластера между двумя экземплярами vCenter серверов. Но в случае

ошибки процесс миграции просто прервётся, что может привести к проблемам в работе сервера, развернутого на таком кластере.

Из вышесказанного следует, что имеет смысл разработать алгоритм автоматической миграции, который при своём завершении будет оставлять сервер, развернутый на ESXi-кластере, в правильно сконфигурированном виде, даже в случае возникновения ошибок. Для чего необходимо реализовать процедуру отката (англ. «rollback»).

При этом очевидно, что для процедуры восстановления конфигурации, которую следует применять в случае недоступности действующего vCenter сервера (на котором сейчас развернут ESXi-кластер), реализовать процедуру отката невозможно. В связи с этим алгоритм восстановления конфигурации должен иметь возможность перезапуска или иной способ запуска с действия, на котором произошла ошибка (англ. «roll forward»).

Таким образом, алгоритм миграции выглядит следующим образом. В первую очередь выполняются предварительные действия по миграции, которые включают проверку возможности осуществить миграцию и иные действия необходимые для конкретного сервера. Затем удаляются ESXi-хосты с текущего vCenter сервера и создаётся новый кластер на новом vCenter. В этот кластер необходимо добавить удаленные из предыдущего vCenter ESXi-хосты. Далее необходимо сконфигурировать параметры сети на новом vCenter, это делается в зависимости от того, как ESXi-хосты были подключены до этого, через DVS [8] или Standard Virtual Switch (SVS) [9]. Затем выполняются дополнительные действия по конфигурированию кластера и сети, после которых выполняются завершающие шаги, например, регистрирование «Storage Provider» и необходимые завершающие действия на сервере, после чего старый vCenter очищается. Шаги отката идут в обратном порядке. Но, если хосты добавлены в новый кластер, то перед тем, как его удалить, необходимо удалить оттуда хосты. Также в случае, если хосты должны быть подключены через DVS, алгоритм возврата сети на исходный vCenter сервер будет сильно меняться от того, на каком этапе произошла ошибка.

Алгоритм восстановления конфигурации в свою очередь не содержит механизмов отката, и не включает обращений на старый vCenter сервер. Но при этом аналогичен алгоритму миграции, за исключением того, что перед каждым шагом выполняются проверка необходимости выполнения данного шага, что позволяет перезапускать его после ошибок.

Результаты работы использованы в одной из систем хранения данных, реализованный модуль позволил сократить время миграции или восстановления конфигурации с 55 минут до 3 минут. В дальнейшем планируется увеличить количество восстанавливаемых модулем сущностей.

ЛИТЕРАТУРА

1. A. P. Nyrkov, Y. F. Katorin, V. D. Gaskarov, L. S. Brazhnikova and N. Vikhrov., Analysis of platform vulnerabilities for the virtualization process, 2018 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus). 2018. С. 94–97.
2. И. М. Сысоев, И. В. Никифоров, Е. В. Каплан, 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.
3. VMware // Moving a vSAN cluster from one vCenter Server to another (2151610). https://kb.vmware.com/s/article/2151610. Дата обращения: 17.12.2021.
4. VMware // Moving an ESX/ESXi host with vDS from one vCenter Server to another (1029498). https://kb.vmware.com/s/article/1029498. Дата обращения: 17.12.2021.
5. VMware // Exporting/importing/restoring Distributed Switch configs using vSphere WebClient (2034602). https://kb.vmware.com/s/article/2034602. Дата обращения: 17.12.2021.
6. Nutanix // Migrating a Nutanix Cluster from One vCenter Server to Another. https://portal.nutanix.com/page/documents/details?targetId=vSphere-Admin6-AOS-v6_1:vsp-cluster-migrate-vcenter-vsphere-t.html. Дата обращения: 18.03.2022

7. Vidanez // Migrate-vCenter-with-vDS. https://github.com/Vidanez/Migrate-vCenter-with-vDS. Дата обращения: 18.03.2022
8. VMware // vSphere Distributed Switch. https://www.vmware.com/products/vsphere/distributed-switch.html. Дата обращения: 18.03.2022.
9. Kamla, R. Z., Yahiya T., Mustafa N. B., Analysis of platform vulnerabilities for the virtualization process, 2018 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus). 2018. С. 94–97.

УДК 004.042

Сафронов Д. (2 курс магистратуры),
Никифоров И. В., к.т.н., доцент

АВТОМАТИЗАЦИЯ ПОДГОТОВКИ ОКРУЖЕНИЯ В СРЕДЕ KUBERNETES С ПОМОЩЬЮ РАСШИРЕНИЯ ИНСТРУМЕНТА KIND ДЛЯ ТЕСТИРОВАНИЯ ФУНКЦИОНАЛЬНОСТИ ГОРИЗОНТАЛЬНОГО МАСШТАБИРОВАНИЯ ПРИЛОЖЕНИЙ

Введение. На данный момент доля приложений на рынке, которые используют горизонтальное масштабирование в качестве инструмента повышения производительности, стремительно возрастает [1]. Для этого имеется несколько причин. Одна из них — это появления многофункциональных платформ, которые позволяют минимизировать временные затраты разработчиков для построения сервис-ориентированных приложений. В пример можно привести инструмент Kubernetes [2, 3]. В таких средах уже заложены функции обеспечения распараллеливания нагрузки, поддержки жизнеспособности важных сервисов, консистентного поведения заданного набора сервисов и других важных аспектов.

Для внедрения среды типа от разработчика требуется поддержать определенный набор API для интеграции. Также приложение должно самостоятельно отвечать за обработку заданной нагрузки.

То есть, если приложения взаимосвязано с другими сервисами, и либо оно не может предоставить им требуемую скорость обработки, либо они отвечают недостаточно оперативно, это будет сказываться на производительность отдельного компонента приложения. Общее понижение производительности отдельных компонент ввиду неправильной конфигурации или неправильного управления распределением входящего/исходящего траффика приводит к потере производительности системы в целом.

Если при увеличении числа реплик, критерий, описанный выше, приводит к недоступности сервисов всего приложения (выше какого-то уровня), то можно говорить о том, что оно не поддерживает горизонтальное масштабирование (выше этого уровня).

Нахождение и исправление уязвимостей и ошибок, которые приводят к уменьшению порога горизонтального масштабирования приложения, — это достаточно нетривиальная задача ввиду сложности локализации проблем подобного рода.

Поэтому одной из важнейших задач разработки является обеспечение и поддержка процесса непрерывного тестирования интеграции приложения [4] со средой Kubernetes и заданного в технической документации уровня горизонтального масштабирования, который бы позволял разработчикам в кратчайшие сроки локализовать и устранить возникающие при новых изменениях проблемы.

Сам процесс тестирования при этом достаточно трудозатратный, так как необходимо обеспечить постоянный доступ к железным ресурсам и закладывать дополнительное время работы инженеров, которые будут тестировать данный продукт.

Таким образом для оптимизации описанного процесса, *была поставлена цель* сократить время, которое требуется тратить разработчику и тестировщику программного продукта на реализацию функционала горизонтального масштабирования приложений в среде Kubernetes

и его поддержку, за счет автоматизации развертывания виртуального кластера, состоящего из большого количества узлов.

В качестве платформ для создания кластера, состоящего из большого количества виртуальных узлов, мной были выбраны две технологии «полной» виртуализации, а именно VmWare ESXi и Microsoft HyperV. И также одна технология контейнерной [5] виртуализации – инструмент Kind.

Сравнительный анализ показал, что первые две технологии имеют гораздо большую функциональности в плане безопасности и интерфейсов взаимодействия. А именно поддерживают протоколы vTPM, VBS, KMS и имеют специализированный сервис для развертывания, мониторинга и миграции виртуальных кластеров.

Kind в отличие от инструментов, описанных выше, может быть установлен с использованием только одного образа контейнера, а также его установка требует гораздо меньше времени по сравнению с аналогами. Еще в плюс данной технологии можно выделить упрощенный развертывания, для которого требуется только наличие утилиты на локальной машине пользователя, что значительно уменьшает CI/CD конвейер.

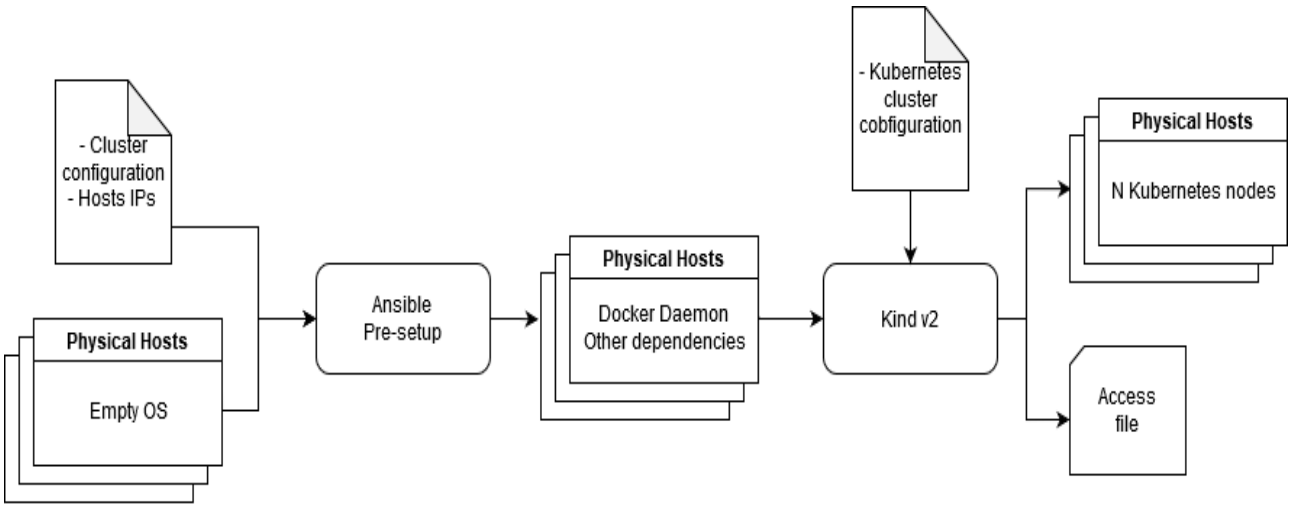


Рисунок 1 – Схема процесса подготовки окружения для тестирования системы.

Для начала разработки необходимо было сформулировать структурные требования к предлагаемому подходу. Конечный вариант был сформированы в виде пользовательской истории, которая представлена на Рисунке 1.

Здесь можно увидеть, что конвейер настройки системы состоит из двух частей:

- инициализация физических узлов и конфигурирование на них необходимых библиотек и пакетов;

- развертывание виртуального кластера, обеспечение сетевого взаимодействия между машинами кластера и выдача пользователь доступа к готовому решению.

Для решения первой задачи был взят инструмент Ansible, написанный на Python [6] и позволяющий проделывать идемпотентные операции над несколькими заданными узлами. Входом этого модуля можно считать подготовленные физические узлы и файл с предоставленными ключами доступа с необходимыми правами, по которым инструмент может установить все нужное программное обеспечение для следующего модуля. Взаимодействие с кластером на этом этапе осуществляется по протоколу SSH.

Второй модуль основан на описанном выше инструменте Kind и выполняет основную задачу конвейера, а именно развертывание виртуального кластера на нескольких физических узлах. Его входом является определенное количество физических узлов, предподготовленных на предыдущем шаге, а также файл с конфигурацией ожидаемого виртуального кластера. На выходе этого модуля пользователь получает физические узлы с запущенными на них

виртуальными узлами, а также файл доступа, по которому можно подключиться к готовому кластеру.

В результате проделанной работы, на кластере, состоящим из 8 физических узлов, удалось развернуть виртуальный кластер, состоящий из 48 виртуальных узлов. Виртуальный кластер удовлетворял техническим требованиям тестируемого продукта и позволил облегчить процесс валидации конечного продукта.

ЛИТЕРАТУРА

1. B. Kaitlyn, "CNCF Survey: Use of Cloud Native Technologies in Production Has Grown Over 200%," [Электронный ресурс]. Режим доступа: https://www.cncf.io/blog/2018/08/29/cncf-survey-use-of-cloud-native-technologies-in-production-has-grown-over-200-percent. [Доступ Январь 2022].
2. G. Velez, "Kubernetes vs. Docker: A Primer," [Электронный ресурс]. Режим доступа: https://containerjournal.com/topics/container-ecosystems/kubernetes-vs-docker-a-primer. [Доступ Январь 2022].
3. A. S. Shemyakinskaya, I. V. Nikiforov, 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. E. Wolff, A Practical Guide to Continuous Delivery, Addison-Wesley Professional, 2017.
5. W. G. Wong, "What’s the Difference Between Containers and Virtual Machines?," [Электронный ресурс]. Режим доступа: https://www.electronicdesign.com/technologies/dev-tools/article/21801722/whats-the-difference-between-containers-and-virtual-machines. [Доступ Январь 2022].
6. С. Э. Сараджишвили, В. В. Леонтьев, Н. В. Воинов, Введение в обработку изображений на языке Python // 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.

УДК 004.415.53

Соломонов М.С. (2 курс магистратуры),
Ковалев А.Д., ассистент

МЕТОД УПРАВЛЕНИЯ ВЫПОЛНЕНИЕМ АВТОМАТИЗИРОВАННЫХ ТЕСТОВ ДЛЯ СХД

Система хранения данных (СХД) – комплекс программных и аппаратных средств, предназначенный для хранения и обработки больших объемов информации [1]. СХД поддерживают различные сетевые протоколы и содержат множество различных архитектурных и инженерных решений [2]. Такие системы требуют высокие затраты на тестирование, в частности, человеческих ресурсов, поэтому автоматизация процесса тестирования является очень востребованной и актуальной, особенно на этапе системного тестирования. В таких крупных программных комплексах значительную часть времени при подготовке к тестированию отнимает процесс конфигурации окружения, поэтому возникает проблема эффективного распределения ресурсов.

Существует много различных инструментов, позволяющих управлять процессом тестирования, настраивать окружение в автоматическом режиме или выполняющих обе этих функции. К таким примерам можно отнести Ansible [3], Puppet, Test IT [4], TestRail. Такие инструменты в основном предоставляют лишь метрики результатов, анализировать которые предлагается инженерам. В них отсутствуют инструменты автоматического управления тестированием.

Предлагаемый в работе метод нацелен на снижение затрат на первичную конфигурацию и последующую переконфигурацию СХД. Общая архитектура подхода представлена на Рисунке 1. Использование информации о тестовых конфигурациях и резервах систем позволит объединить общие по конфигурациям тесты и построить последовательность тестов с минимальными издержками на пересоздание тестового окружения.

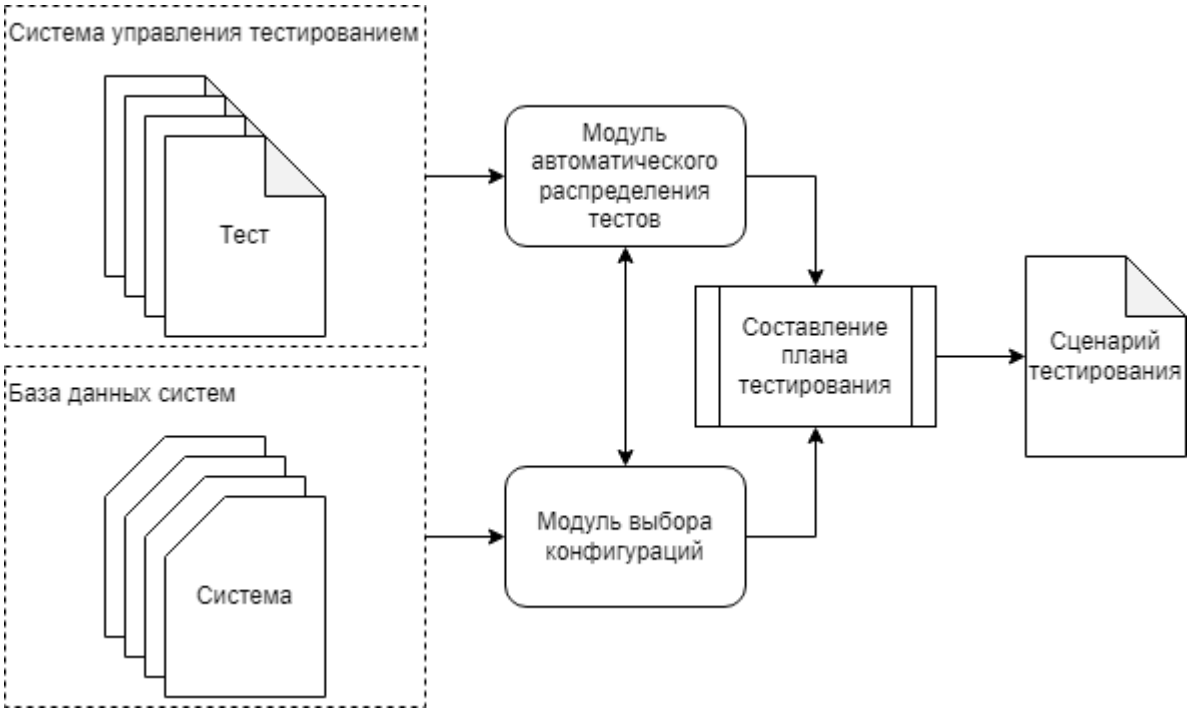


Рисунок 1 – Архитектура предлагаемого решения

Для выделения общих тестов предлагается использовать подход кластеризации [5]. Учет не только приоритетов тестов, но также и необходимой конфигурации позволит расширить размерность данных и поможет выявить латентные зависимости между ними. В качестве меры расстояний между конфигурациями предлагается использовать матрицу непохожести. В ней могут быть использованы различные метрики, но в данном исследовании предлагается использовать эмпирическую оценку времени, которое необходимо для перехода из одной конфигурации в другую. С другой стороны, анализ текущего положения в резерве систем позволяет выявить уже готовые конфигурации для тестирования, которые лишь необходимо проверить на готовность к тестированию.

В результате применения метода ожидается сокращение времени тестирования в среднем на 12,5%. Также метод способен повысить утилизацию ресурсов, что позволит снизить издержки на тестирование продукта. В будущем возможно добавление автоматического метода для построения матрицы непохожести, так как на данный момент ее получение является наиболее трудоемким этапом предлагаемого подхода.

ЛИТЕРАТУРА

1. Никифоров И.В., Курсовое проектирование по учебной дисциплине "Наука о данных и аналитика больших объемов информации", Учебное пособие / Санкт-Петербург, 2017
2. От хранения данных к управлению информацией. - М.: Питер, 2016. - 544 с.
3. Ansible [Электронный ресурс] – Режим доступа: https://www.ansible.com/
4. Test IT [Электронный ресурс] – Режим доступа: https://testit.software/
5. Тютин Б. В., Веселов А. О., Котляров В. П. Масштабирование выполнения тестового набора при автоматизированном тестировании // Информатика, телекоммуникации и управление. 2013. №3 (174).

УДК 004.453

Стоноженко К. М. (2 курс магистратуры),
Никифоров И. В., к.т.н., доцент

ПОДХОД АВТОМАТИЧЕСКОГО УПРАВЛЕНИЯ ОБЪЕКТНЫМ ХРАНИЛИЩЕМ С ПОМОЩЬЮ OPERATOR SDK И CUSTOM RESOURCE DEFINITION

Современные объектные системы хранения данных позволяют хранить колоссальные объёмы информации и совершать многочисленные непрерывные операции записи, чтения, удаления и модификации данных. Кодирование хранимой информации, распределение системы хранения по нескольким географически разнесённым узлам и наличие встроенных систем авторизации и аутентификации пользователей повышают бесперебойность и безопасность хранилища, однако возлагают определённые ограничения на аппаратное обеспечение [1].

Один из способов абстрагироваться от аппаратной части объектной системы хранения – упаковать программные сервисы в контейнеры [2] – легковесные виртуальные наборы изолированных ресурсов – и доверить управление ими системам оркестрации контейнеров. Уменьшив зависимость между программным обеспечением и платформой, мы упростим разработку, тестирование и обслуживание СХД, позволим развёртывать её на практически любом наборе аппаратных ресурсов и, следовательно, значительно сократим стоимость хранилища.

Одним из наиболее популярных средств оркестрации контейнеров является инструмент Kubernetes [3], предоставляющий возможность расширения собственного API, в частности добавления пользовательских ресурсов (Custom Resource Definition) и разработки средств управления пользовательскими ресурсами.

Таким образом, *целью* данной работы является снижение трудоёмкости управления объектной системы хранения за счёт создания подхода автоматического управления объектным хранилищем с помощью Operator SDK и Custom Resource Definition, который реализован в инструментальном средстве в среде Kubernetes.

Для достижения этой цели необходимо решение следующих *задач*:

1. Обзор существующих подходов реализации средства автоматического управления ресурсами Kubernetes.
2. Проведение сравнительного анализа найденных подходов.
3. Предложение автоматического подхода управления объектными хранилищами в среде k8s.
4. Реализация данного подхода.
5. Демонстрация снижения трудоёмкости.

Одним из основных компонентов разрабатываемого подхода управления объектным хранилищем в среде Kubernetes является оператор [4] – разновидность контроллера пользовательского ресурса. В ходе проведения сравнительного анализа в качестве инструмента разработки оператора ресурса объектного хранилища был выбран Operator SDK [5].

Архитектура системы управления объектной СХД представлена на Рисунке 1. Желаемое и текущее состояния компонентов хранилища описываются в специальном дескрипторе CRD, чтение и обработку которого осуществляет оператор при проведении различных сервисных процедур на кластере – масштабирование хранилища, удаление диска, включение режима обслуживания узла кластера и т. д.

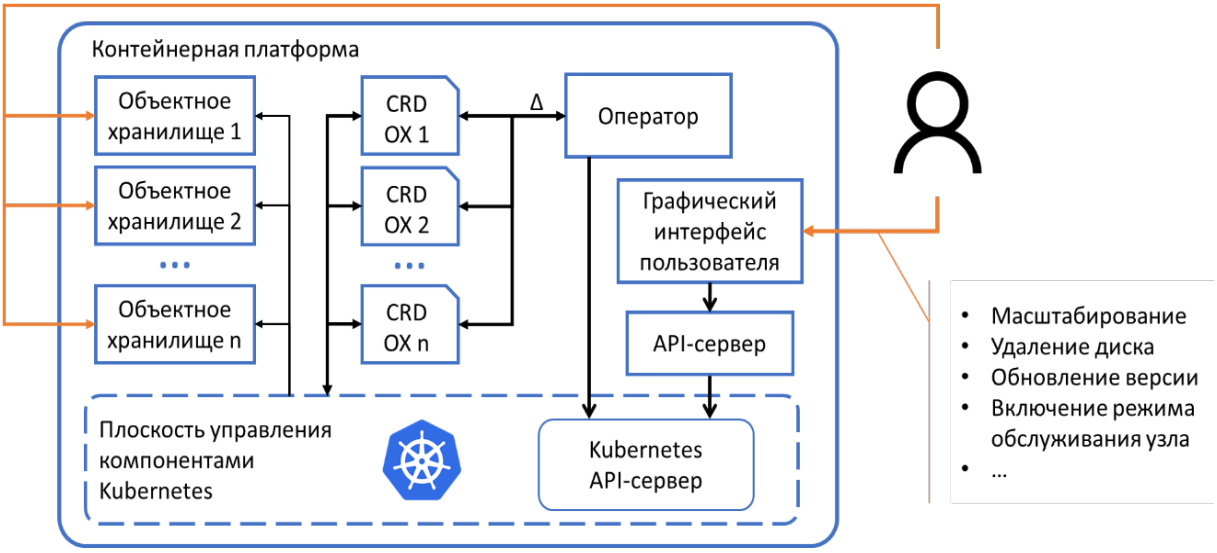


Рисунок 1 – Архитектура системы

Непосредственное управление компонентами СХД производится путём обращения к Kubernetes API-серверу при помощи запросов на создание, удаление и обновление встроенных ресурсов – разделов, подов, сервисов и др.

Подход *был реализован* в программном средстве на языке Golang [6] – родном языке разработки Kubernetes. Оператор осуществляет обработку пользовательского ресурса объектного хранилища, в котором описываются настройки работы всех компонентов. К поддерживаемым платформам развёртывания системы относятся vmWare vSphere [7] и OpenShift [8] от RedHat. Интеграция оператора с пользовательской системой мониторинга и UI платформы позволяет конечному пользователю следить за состоянием СХД в реальном времени.

Задачи по *дальнейшей* разработке оператора сводятся к реализации конвейеров автоматического тестирования ПО, использования дополнительных возможностей Operator SDK и добавления поддержки нескольких параллельных сервисных процедур.

ЛИТЕРАТУРА

1. В. А. Ивлев, И. В. Никифоров, Т. В. Леонтьева, 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. – 400 стр.
3. A. S. Shemyakinskaya, I. V. Nikiforov, 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. Operator pattern. [Электронный ресурс] Режим доступа: https://kubernetes.io/docs/concepts/extend-kubernetes/operator/
5. Operator SDK. [Электронный ресурс] Режим доступа: https://sdk.operatorframework.io/
6. Golang programming language. [Электронный ресурс] Режим доступа: https://go.dev/
7. vSphere. [Электронный ресурс] Режим доступа: https://www.vmware.com/ru/products/vsphere.html
8. Red Hat OpenShift. [Электронный ресурс] Режим доступа: https://www.redhat.com/en/technologies/cloud-computing/openshift

УДК 004.457

Шемякинская А.С. (1 курс магистратуры),
Ковалев А.Д., ассистент

АВТОМАТИЗАЦИЯ РАСШИРЕНИЯ ТОМА В СИСТЕМЕ ХРАНЕНИЯ ДАННЫХ С ПОМОЩЬЮ CSI И KUBERNETES

Системы оркестрации (CO) контейнеров набирают все большую популярность для развертывания приложений, значительно упрощая этот процесс. Kubernetes – это программное обеспечение с открытым исходным кодом, необходимое для управления и развертывания кластера Docker контейнеров [1]. Приложениям часто требуется хранить данные в СХД, например, базу данных, лог-файлы и другой неструктурированный контент.

По умолчанию, Kubernetes поддерживает интеграцию с ограниченным количеством СХД, поэтому появляется необходимость введения обобщенного интерфейса, называемого CSI (container storage interface). Производители СХД могут реализовать его на своей стороне, чтобы системы оркестрации могли использовать новые СХД. Интерфейс CSI разработан как стандарт для предоставления интеграции произвольных СХД с системами оркестрации контейнеров (CO), таких как Kubernetes [2, 3]. В настоящее время нет проекта CSI, полностью автоматизирующего управление жесткими дисками напрямую.

Целью работы является сокращение трудоемкости работы с дисковыми носителями в Kubernetes посредством CSI. Для достижения цели требуется решить задачи, перечисленные ниже.

1) Исследовать существующие инструменты CSI, которые позволяют автоматизировать работу с дисковыми носителями в Kubernetes и провести их сравнительный анализ.

2) Предложить подход к автоматизации работы с дисковыми носителями в CSI.

3) Реализовать предложенный подход.

4) Показать эффективность применения предложенного подхода и его реализации в CSI на реальных системах с Kubernetes.

На основе сравнительного анализа инструментов Baremeal CSI, Topolvm, CSI-driver-lvm, Minio можно сделать вывод, что проект Baremetal CSI – это наиболее эффективная реализация CSI. Но для повышения качества автоматизации данного средства необходимо добавить:

- поддержку работы с эфемерными томами (ephemeral volumes) в Kubernetes.
- процесс автоматической поддержки увеличения размера тома (volume expansion) в Kubernetes.
- поддержку автоматизации end-to-end тестирования.

В работе рассмотрена реализация расширения томов. Расширение тома — это функция, которая позволяет конечным пользователям Kubernetes расширять постоянный том (Persistent Volume) после создания, изменяя размер динамически подготовленного Persistent Volume Claim (PVC) [4]. Baremetal CSI поддерживает только расширения для lvm [5]. Архитектура решения представлена на Рисунке 1.

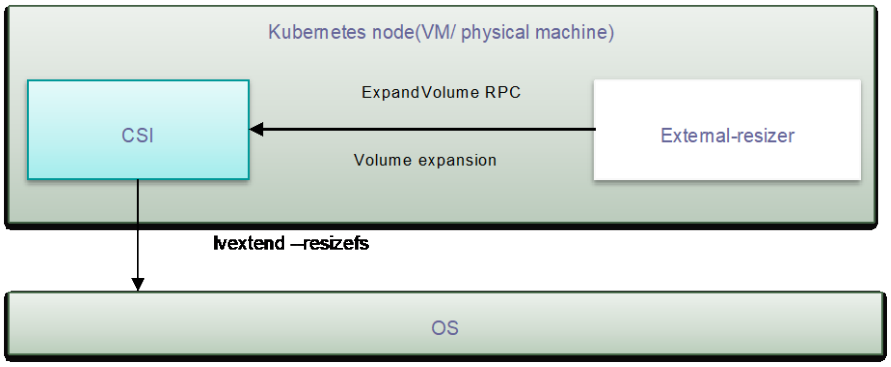


Рисунок 1 – Архитектура расширение тома в Baremetal CSI

Для достижения поставленных задач в работе были выполнены следующие шаги:

1) Реализован RPC запроса ExpandVolume, который приходит от стороннего компонента Resizer.

2) Для расширения LVM была выбрана команда lvextend –resizefs, потому что она позволяет расширять логический том и файловую систему на нем без использования дополнительных команд.

3) Расширение томов было реализовано таким образом, что оно работает как для монтируемых, так и для не монтируемых файловых систем.

В итоге, если рассмотреть сложность работы с расширением томов, то ее можно представить в виде простой формулы:

$$O(i + k + l)$$

где i – время ввода команды, k – время изучения документации Kubernetes, l – время изучения документации LVM. Так как в случае имплементации расширения тома в Kubernetes, необходимость в изучении документации LVM отпадает, то формула упрощается:

$$O(i + k)$$

Это говорит об уменьшении времени и сложности работы с расширенными томами.

В результате работы проведено исследование существующих реализаций CSI, сделаны выводы о наиболее эффективной реализации Baremetal CSI, в ходе работы была проведена автоматизация ее работы.

Реализован процесс автоматической поддержки увеличения размера тома (volume expansion) в Kubernetes, что позволило сократить трудоемкость расширения томов.

ЛИТЕРАТУРА

1. Kubernetes. Электронный ресурс [Режим доступа]: https://kubernetes.io/
2. В. А. Максимчук, И. В. Никифоров, Разработка системы масштабирования реплик приложений с целью увеличения отказоустойчивости узлов кластера 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. 160-161.
3. А. С. Шемякинская, И. В. Никифоров, Реализация паттерна "оператор" для отслеживания состояния дисков в системе оркестрации контейнеров 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. 167-169.
4. Kubernetes PVC/PV. Электронный ресурс. [Режим доступа]: https://kubernetes.io/docs/concepts/storage/persistent-volumes/, accessed 04.07.2021,
5. M. Bozic, M. Djukic, D. Narancic and I. Pap, "Comparison analysis of standard disk partitioning and LVM based disk partitioning on Linux systems," 2017 13th International Conference on Advanced Technologies, Systems and Services in Telecommunications (TELSIKS), 2017, pp. 415-418, doi: 10.1109/TELSKS.2017.8246313.

УДК 004.4`2

Фигловский К. С. (2 курс магистратуры),
Юсупова О. А., ассистент

АВТОМАТИЗАЦИЯ ПРОЦЕССА ГЕНЕРАЦИИ И ПУБЛИКАЦИИ ПРИМЕЧАНИЙ К ВЫПУСКУ МОБИЛЬНЫХ ПРИЛОЖЕНИЙ

Одним из важных этапов жизненного цикла разработки программного обеспечения является создание и подготовка дистрибутива продукта, который в итоге доставляется конечному пользователю [1]. Time To Market (TTM) — это время от начала разработки идеи до её конечной реализации, то есть «время до выхода на рынок». Ручная работа сильно увеличивает показатель TTM [2]. Внедрение CI\CD принципов — эффективный способ

уменьшения показателя TTM путем сокращения ручной работы [3]. Одним из наиболее важных артефактов являются примечания к выпуску (release notes). В них пользователю описываются изменения, внесенные в последнюю версию продукта [4]. Создание примечаний к выпуску вручную — трудоемкий и подверженный ошибкам процесс, требующий взаимодействия многих людей.

Несмотря на актуальность работы с примечаниями к выпуску, в открытом доступе их не так много, поэтому целью данной работы является внедрение нового подхода к снижению сложности и трудоемкости создания примечаний к выпуску за счет автоматизации процесса подготовки и хранения метаданных выпусков продуктов.

Целью работы является снижение трудоемкости создания release notes за счет автоматизации процесса подготовки и хранения метаданных для релизов продуктов, а также формирования и выдачи метаданных в формате, пригодном для непрерывного развертывания (Continuous Deployment). Для достижения целей необходимо решить следующие задачи: исследовать существующие подходы к автоматизации работы с Release notes, провести сравнительный анализ подходов автоматизации, предложить подход автоматизации процесса подготовки и хранения метаданных для релизов продуктов, реализовать предложенный подход в программном средстве, продемонстрировать эффективность и снижение трудоемкости от использования выбранного подхода.

Если говорить о существующих решениях, то можно выделить несколько подходов. Первый — использование плагинов систем сборки или CI/CD системы, которые автоматизируют публикацию приложений (Codemagic CI/CD, Buddy CI/CD). Второй — работа с различными системами генерации текста (Laura Moreno`s ARENA [5], Sebastian`s Klepper`s a semi-automatic method [6]). Третий — использование сервисов для хранения текста (Google docs, git). Так как каждый из подходов имеет определенные недостатки, требуется спроектировать новое решение.

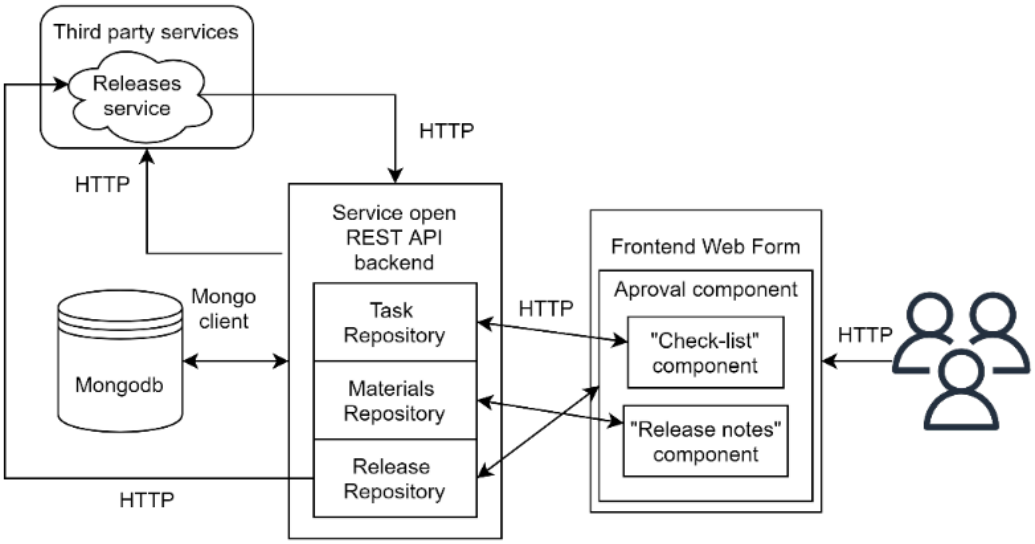


Рисунок 1 – Архитектура сервиса

Для повышения автоматизации предлагается разработать специальный сервис «метаданных», с которым будут взаимодействовать участники процесса. Архитектурно сервис предлагается разбить на 3 части: Frontend, с которым взаимодействуют пользователи через браузер, Backend, с которым по REST API взаимодействуют Frontend и сторонние сервисы, и база данных [7], с которой работает Backend (Рисунок 1). Описанная архитектура позволяет автоматизировать сбор и хранение данных о выпуске. Среди функций можно выделить отделение бизнес-логики от пользовательского интерфейса, что позволяет проводить параллельную разработку для нескольких человек, а также возможность расширения функциональности за счет наличия API для сторонних сервисов, таких как конвейер CI/CD.

Для программной реализации использованы следующие инструменты и технологии: Typescript 4.5.4, Node.js 16.1.0, Angular 13.1.2, RxJS 6.6.7, Mongoose 5.9.13, Express.js 4.17.1, Mocha 8.4.0, Jasmine 3.5.10, Karma 5.0.4, TSLint 6.1.3, MongoDB 4.4.3.

В результате работы разработан компонент для утверждения материалов релиза в сервисе метаданных. Компонент позволяет подготавливать и утверждать материалы к выпуску, предварительно заполнять материалы из готовых шаблонов, вести учет произвольного списка задач. Количество автоматизированных шагов было увеличено на 3 во всей цепочке. Количество шагов, выполняемых вручную, уменьшено на 1. При этом сокращение времени на заполнение release notes за счет использования разработанного компонента составило более 40%. Также разработанный сервис имеет перспективы дальнейшего развития, что позволит с помощью внедрения новых автоматизированных шагов сократить объем ручной работы и еще больше ускорить процесс релиза.

ЛИТЕРАТУРА

1. Conger, S. (2011). Software development life cycles and methodologies: Fixing the old and adopting the new. International Journal of Information Technologies and Systems Approach, 4(1), 1-22. doi:10.4018/jitsa.2011010101
2. “Time To Market: What it is, why it’s important, and five ways to reduce it | TCGen, [online] URL: https://www.tcgen.com/time-to-market
3. Drobintsev, P. D., Kotlyarov, V. P., Nikiforov, I. V., & Letichevsky, A. A. (2016). In-cremental approach to the technology of test design for industrial projects. Automatic Control and Computer Sciences, 50(7), 486-492. doi:10.3103/S014641161607004X
4. Abebe, S. L., Ali, N., & Hassan, A. E. (2016). An empirical study of software release notes. Empirical Software Engineering, 21(3), 1107-1142. doi:10.1007/s10664-015-9377-5.
5. Moreno, L., Bavota, G., Di Penta, M., Oliveto, R., Marcus, A., & Canfora, G. (2017). ARENA: An approach for the automated generation of release notes. IEEE Transac-tions on Software Engineering, 43(2), 106-127. doi:10.1109/TSE.2016.2591536.
6. Klepper, S., Krusche, S., & Bruegge, B. (2016). Semi-automatic generation of audi-ence-specific release notes. Paper presented at the Proceedings - International Work-shop on Continuous Software Evolution and Delivery, CSED 2016, 19-22. doi:10.1145/2896941.2896953
7. 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

СОДЕРЖАНИЕ

СОВРЕМЕННЫЕ ТЕХНОЛОГИИ В ТЕОРИИ И ПРАКТИКЕ ПРОГРАММИРОВАНИЯ

Сборник материалов конференции

26 апреля 2022 года

Налоговая льгота – Общероссийский классификатор продукции
ОК 005-93, т. 2; 95 3004 – научная и производственная литература

Подписано в печать 19.04.2022. Формат 60×84/16. Печать цифровая.
Усл. печ. л. 13,75. Тираж 200. Заказ 1787.

Отпечатано с готового оригинал-макета, предоставленного редколлегией,
в Издательско-полиграфическом центре Политехнического университета.
195251, Санкт-Петербург, Политехническая ул., 29.
Тел.: (812) 552-77-17; 550-40-14.