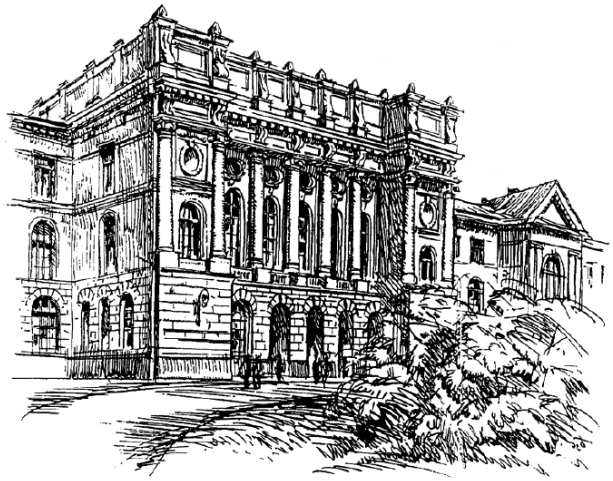
Министерство науки и высшего образования Российской Федерации

САНКТ-ПЕТЕРБУРГСКИЙ
ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО

СОВРЕМЕННЫЕ ТЕХНОЛОГИИ В ТЕОРИИ И ПРАКТИКЕ ПРОГРАММИРОВАНИЯ

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

24–25 апреля 2024 года



ПОЛИТЕХ-ПРЕСС
Санкт-Петербургский
политехнический университет
Петра Великого

Санкт-Петербург
2024

УДК 004
ББК 32.973
С56

Современные технологии в теории и практике программирования : сборник материалов научно-практической конференции студентов, аспирантов и молодых ученых, 24–25 апреля 2024 г. – СПб. : ПОЛИТЕХ-ПРЕСС, 2024. – 311 с.

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

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

ISBN 978-5-7422-8530-4

Приветствие от компании YADRO

Уважаемые участники!

Компания YADRO уже традиционно выступает партнером конференции «Современные технологии в теории и практике программирования».
Наше сотрудничество с Политехническим университетом началось в 2023 году. Мы видим, что студенты активно интересуются деятельностью компании и участвуют в образовательных программах YADRO. Наши сотрудники, бывшие выпускники университета, также хотят внести свой вклад в совместные учебные и научные проекты, курируют будущих инженеров. Многие участники конференций прошлых лет сделали успешную карьеру, поднявшись от инженера до руководителя в YADRO. Мы ценим уровень подготовки выпускников Политеха, качество преподавания профильных дисциплин в области разработки ПО и создания систем на кристалле.

Цель этой конференции, в том числе и секции YADRO, — это поддержка молодых учёных, их научных исследований в области программной инженерии и обсуждение передовых технологических трендов в индустрии. Для устойчивого развития компаниям, создающим сложные технологические продукты, важно создавать и развивать партнёрские отношения с университетами.

Желаю вам плодотворной работы на конференции, профессионального роста и новых свершений!

С уважением,
Евгений Викторович Максимов
Директор по развитию экосистемы и образовательных инициатив YADRO

Тезисы докладов конкурса-конференции

Секция «Программная инженерия: приложения, продукты и системы»

УДК 004.421.6+510.635

Н. Н. Алексеев (2 курс магистратуры),
А. С. Герасимов, к.ф.-м.н., доцент

РЕАЛИЗАЦИЯ АЛГОРИТМОВ УНИФИКАЦИИ РОБИНСОНА И ПАТЕРСОНА-ВЕГМАНА НА ЯЗЫКЕ JAVA

В логике унификатором термов s и t называют такую подстановку (термов вместо переменных), что результаты применения этой подстановки к термам s и t совпадают; см., например, [1, разд. 4.3.1]. Говорят, что пара термов s и t унифицируема, если существует унификатор термов s и t. Задача унификации — определить, унифицируема ли пара термов, и если они унифицируемы, то построить унификатор этих термов.

Существует множество различных алгоритмов унификации. Среди них исторически первым является алгоритм унификации конечного множества термов, описанный Робинсоном [2]. Этот алгоритм является легко реализуемым на практике. Но он имеет экспоненциальную временную сложность.

В работе [3] описан вариант алгоритма Робинсона для унификации пары термов. Термы на вход алгоритму подаются в виде синтаксических деревьев. Алгоритм одновременно обходит оба дерева, используя метод рекурсивного спуска.

Алгоритм унификации Робинсона имеет полиномиальный вариант, описанный, например, в [3]. В нем используется представление пары термов в виде ориентированного ациклического графа. Особенность такого представления термов заключается в том, что все переменные при таком представлении являются *разделяемыми,* то есть для представления каждой переменной в обоих термах используется единственный узел. В ходе работы полиномиального варианта алгоритма Робинсона некоторые узлы графа изолируются, что значительно снижает количество рекурсивных вызовов процедуры унификации. Этот алгоритм широко распространен на практике и является достаточно эффективным, несмотря на существование более эффективных в теории алгоритмов (подробнее см. [4]).

Существуют также алгоритмы унификации, имеющие линейную временную сложность. Одним из таких алгоритмов является алгоритм Патерсона–Вегмана [5]. Так же, как и в полиномиальном варианте алгоритма Робинсона, здесь используется представление термов в виде ориентированных ациклических графов. Основная идея алгоритма заключается в построении классов эквивалентности на множестве узлов графа, представляющего пару термов. В процессе выполнения алгоритма проверяется соответствие построенных классов эквивалентности определенным свойствам. Особенность, которая делает этот алгоритм линейным, заключается в особом порядке проверки классов эквивалентности, при котором классы эквивалентности, содержащие корневые узлы, обрабатываются в первую очередь. В оригинальном описании содержались опечатки и неточности, исправленные в работах [6] и [7].

Целью данной работы является разработка библиотеки на языке Java, в которой реализованы следующие алгоритмы унификации:

– алгоритм Робинсона для унификации пары термов;

– полиномиальный вариант алгоритма Робинсона;

– линейный алгоритм Патерсона–Вегмана для унификации пары термов.

Для достижения данной цели были поставлены следующие задачи:

– детализировать описание всех вышеупомянутых алгоритмов унификации в виде псевдокода;

– реализовать представление терма в виде ориентированного ациклического графа;

– реализовать синтаксический анализатор, который преобразовывает термы в виде строк в представление терма в виде ориентированного ациклического графа;

– реализовать алгоритмы унификации;

– провести тестирование реализованных алгоритмов унификации и сравнить их эффективность между собой.

Данная библиотека предоставляет API (Application Programming Interface, интерфейс программирования приложений), который можно использовать для встраивания этой библиотеки в другие проекты.

ЛИТЕРАТУРА

1. Герасимов А. С. Курс математической логики и теории вычислимости: Учебное пособие. – 4-е изд., перераб. и доп. – СПб.: Издательство «Лань», 2014. – ISBN 978-5-8114-1666-0
2. Robinson J. A. A Machine-Oriented Logic Based on the Resolution Principle // J. ACM. – 1965.– Vol. 12. – P. 23–41. – DOI: 10.1145/321250.321253.
3. Baader F., Snyder W. Unification Theory // Handbook of Automated Reasoning: Volume 1 / ed. by J. Robinson, A. Voronkov. – Cambridge, MA, USA : MIT Press, 2001. – P. 445–533. – ISBN 978-0-262-18221-8.
4. Hoder K., Voronkov A. Comparing Unification Algorithms in First-Order Theorem Proving // KI 2009: Advances in Artificial Intelligence / ed. by B. Mertsching, M. Hund, Z. Aziz. – Berlin, Heidelberg : Springer Berlin Heidelberg, 2009. – P. 435–443. – ISBN 978-3-642-04617-9.
5. Paterson M. S., Wegman M. N. Linear unification // Journal of Computer and System Sciences. – 1978. – Vol. 16, no. 2. – P. 158–167. – DOI: 10.1016/0022-0000(78)90043-0.
6. de Champeaux D. About the Paterson-Wegman linear unification algorithm // Journal of Computer and System Sciences. – 1986. – Vol. 32, no. 1. – P. 79–90. – DOI: 10.1016/0022-0000(86)90003-6.
7. Motroi V., Ciobâcă Ş. A Typo in the Paterson-Wegman-de Champeaux algorithm // CoRR. – 2020. – arXiv: 2007.00304.

УДК 004.514

Д. И Астафьева, И. Д. Берсудский (4 курс бакалавриата),
А. П. Маслаков, ст. преподаватель

РАЗРАБОТКА ANDROID ПРИЛОЖЕНИЯ ПОМОЩНИКА ПО НАСТОЛЬНОЙ РОЛЕВОЙ ИГРЕ

Настольные ролевые игры, появившиеся в 1974 году, завоевали популярность во всём мире и пользуются спросом среди всех слоёв и возрастов населения [8]. В них играют как студенты, так и известные актёры, дети в школах и бизнесмены. Данное хобби уже превратилось в настоящий культурный феномен, который помогает развивать в человеке свои навыки социализации, креативное и критическое мышление [1]. С ростом популярности появляется всё больше разнообразных представителей этого жанра настольных игр, которые требуют использовать не только классический подход с бумажными носителями для записи происходящего во время игры, но и информационные технологии.

Одним из таких средств, упрощающих игру, служат различные Android-приложения. Разрабатываемое приложение может выступать в роли помощника для игроков и мастера для в ходе игровой сессии. Мастер – это человек, который придумает мир, создает сценарий и направляет других игроков в своей истории [2]. Приложение позволяет создавать своих персонажей с необходимыми характеристиками, иметь быстрый доступ к справочнику по

правилам игры конкретной ролевой системы, а также создавать лобби для текущей компании. Лобби создаёт и полностью контролирует мастер, в нём может просматривать все характеристики персонажей игроков и изменять их в соответствии с происходящими событиями. К примеру, начислять опыт, изменять количество здоровья. А другие игроки смогут просматривать профили своих коллег по игре. Теперь не нужно будет запоминать или записывать что-то, ведь всё необходимое уже будет автоматически сохранено в приложении.

Архитектура разрабатываемого приложения представлена на рисунке 1.

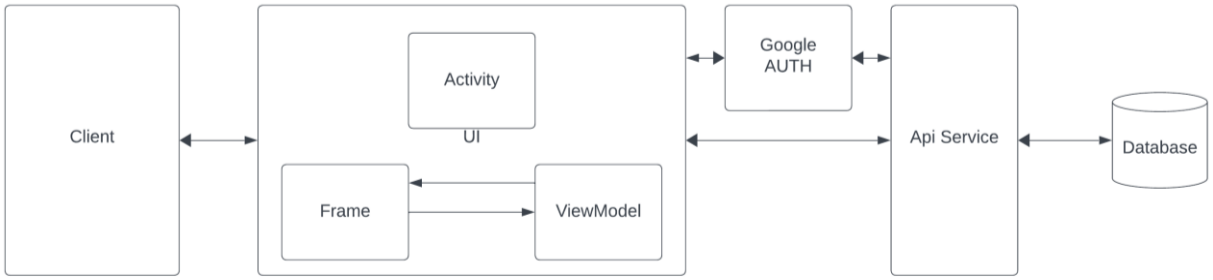


Рисунок 1 – Архитектура системы

Приложение состоит из серверной и клиентской частей, которые взаимодействую между собой посредством REST API используя JSON в качестве формата представления данных (см. рисунок 2). В отдельном модуле реализована Google-авторизация через Gmail как один из способов авторизации в самом приложении, так как она доступна для всех пользователей Android [7] и не требует ввода и запоминания дополнительной информации.

- Метод: Post
- URL: /auth/registration
- Тело JSON запроса:

```
{
"email": "email@email.com",
"password": "password",
"avatar": "avatar_data",
"nickname": "exapmle"
}
```

Рисунок 2 – Пример эндпоинта API

Для серверной части было принято решение использовать язык программирование GoLang, так как это компилируемый язык, который обеспечивает высокую производительность [3]. Для реализации маршрутизации HTTP-запросов в Go используется библиотека gorilla/mux. В качестве базы данных была выбрана MySQL, так как она бесплатна, с открытым исходным кодом, стабильна и есть широкая поддержка языка SQL. Для взаимодействия с ней реализовано API позволяющее выполнять все CRUD-операции с сущностями, содержащими в базе. Кроме того, API обеспечивает безопасность и аутентификацию пользователей при доступе к данным.

Для разработки Android-приложения был выбран Kotlin, так как он является официальном языком программирования для Android [4]. По аналогичной причине в качестве IDE была выбрана Android Studio [5]. Также для реализации интерфейса будет использован Material Design. Это дизайн-система, в которой описаны принципы для создания красочного и функционального пользовательского интерфейса. Туда входят стили и готовые компоненты, помогающие сделать визуальный стиль единообразным [6].

Для мгновенного обновления данных в лобби была выбрана технология WebSocket. WebSocket — это протокол связи, обеспечивающий одновременную двустороннюю передачу данных между клиентом и сервером через одно установленное соединение. Он позволяет мгновенно передавать информацию от сервера клиентам при изменении характеристик персонажа, входе или выходе игрока из лобби. Это связано с тем, что при получении сообщения сервер может сразу обработать его и выполнить необходимое действие и отправить необходимую информацию клиентам через уже установленное соединение, без необходимости устанавливать новое.

В процессе реализации приложение было покрыто модульными тесты для обеспечения надежности и стабильности функционала, на момент написания статьи покрыто 81% кода. Модульные тесты включают в себя проверку обработки HTTP запросов, бизнес-логику и взаимодействие с базой данных.

В клиентской части реализованы модули с фрагментами, такими как: авторизация, аккаунт, лобби, персонаж, справочник

В результате было разработано приложение, содержащее справочник по настольной ролевой игре с удобным поиском, позволяющее пользователям хранить информацию о персонажах, а также объединяться в лобби, что позволит упростить взаимодействие и отслеживание характеристик персонажей во время игровых сессий.

ЛИТЕРАТУРА

1. Каткова, А. Л., Булычева, Е. С., & Каткова, А. А. (2022). Влияние настольных игр на социализацию подростков. Наука о человеке: гуманитарные исследования, 16(2), 137-143. DOI: 10.17238/issn1998-5320.2022.16.2.137
2. Филологические науки. Вопросы теории и практики / Philology. Theory & Practice, 16(1), 97-103. ISSN 2782-4543 (online), ISSN 1997-2911 (print). Материалы журнала доступны на сайте: philology-journal.ru
3. Golang programming language. [Электронный ресурс]. URL: https://go.dev/
4. Vasic, M. (2019). Mastering Android Development with Kotlin: Deep dive into the world of Android to create robust applications with Kotlin. Издательство: Packt Publishing.
5. Phillips, B., & Hard, B. (2020). Android Programming: The Big Nerd Ranch Guide. Издательство: Big Nerd Ranch.
6. Material Design [Электронный ресурс]. URL:. https://m3.material.io/
7. Google Developers. [Электронный ресурс]. URL: https://developers.google.com/identity?hl=ru.
8. Цыганкова П.В., Суворова Е.Ю. Ролевые игры живого действия и ролевые онлайн-игры: психологические функции в современном социокультурном контексте // Вестник Пермского университета. Философия. Психология. Социология. 2020. Вып. 3. С. 459-474. DOI: 10.17072/2078-7898/2020-3-459-474

УДК 004.94

А. А. Абдрахманов,
И. В. Никифоров, к.т.н., доцент,
Су Ли

РЕАЛИЗАЦИЯ МЕТОДОВ ГЕНЕРАЦИИ И АНАЛИЗА РАСПРЕДЕЛЕНИЙ СЛУЧАЙНЫХ ВЕЛИЧИН В СИСТЕМЕ ДИСКРЕТНО-СОБЫТИЙНОГО МОДЕЛИРОВАНИЯ

Генерация случайных величин является важной составляющей для всех систем дискретно-событийного моделирования [1]. Необходимость данного функционала обусловлена тем, что многие процессы и параметры при моделировании могут иметь стохастический характер. Например, время прибытия, характеристики агентов, условия переходов между узлами [2].

В данной работе рассматривается реализация математического сервиса на языке Java внутри системы дискретно-событийного моделирования, позволяющего работать со случайными величинами, их генерацией и анализом [3]. Показана архитектура и функциональность сервиса, описаны некоторые ключевые шаги работы над сервисом, а также представлено описание отличий и преимуществ конкретной реализации.

В языке Java отсутствуют встроенные функции для генерации случайных величин и их статистических параметров (имеются генераторы только нормального и равномерного распределений). По этой причине был проведен анализ имеющихся библиотек и выбор наиболее подходящей. В таблице 1 представлены результаты проведенного анализа.

Таблица 1 – Таблица сравнения библиотек случайных распределений

Пакет	Кол-во поддерживаемых распределений	Среднее число методов в классе распределения	Вариативность параметров	Дополнительная функциональность
apache.math3 [4]	36	~12	+	+
Jaamsim [5]	23	~10	+	+
Colt [6]	26	~9	+	-

Исходя из проведенного анализа наиболее обширной реализацией являлась библиотека “apache.math3”. Однако, она не обладала полным необходимым функционалом. Из-за этого ряд вероятностных распределений был реализован самостоятельно, с помощью математических трансформаций случайных величин, полученных из имеющихся генераторов вероятностных распределений и их модификаций. К таким трансформациям относятся подсчет обратной функции кумулятивного распределения вероятности, для тех распределений, для которых это не представляет большой аналитической трудности, что дает возможность получать случайную величину, подчиняющуюся некоторому вероятностному закону θ, из равномерно распределенной величины. В некоторых случаях использовались более специфические методы, когда вычисление обратной функции было затруднительным.

Помимо генерации значений, сервис так же предоставляет возможность получения математических характеристик распределения, таких как подсчет функции плотности вероятности, функции кумулятивной вероятности распределения в точке, вероятность попадания в указанный интервал. Отличительной чертой является наличие упрощенного провайдера распределений, который является надстройкой над основным и позволяет получать случайное значение вызовом метода, совпадающим с названием распределения и передачей необходимых параметров, что упрощает взаимодействие с сервисом для конечного пользователя системы.

В части анализа распределений сервис покрывает следующую функциональность: определение случайного распределение по предоставленной выборке, генерация случайных величин, соответствующих распределению данной выборки, а также ряд статистических тестов для определения дискретности.

Аналитический сервис работает с объектами класса dataset, который предоставляет из себя отсортированный массив данных и набор константных полей, определяющих основные статистические характеристики набора данных. Объект dataset позволяет упрощенно и быстро получать доступ к необходимым статистическим показателям набора данных.

Функция идентификации распределения возвращает распределение, наиболее подходящее, для полученного набора данных, после проведения ряда статистических тестов и анализа их результатов. В случае, если ни одно из распределений не удовлетворило статистическим тестам, функция возвращает ответ “Custom”.

После нахождения закона распределения, можно построить распределение схожее с распределением исходных данных. В случае, когда имеющаяся выборка удовлетворила одному из известных распределений, генерация значений является тривиальной, с помощью классов распределений сервиса. В обратном случае, имеющаяся выборка сама выступает в качестве закона распределения. Сервис предоставляет ряд настроек, задающих характеристики custom-распределения и способ его генерации. В случае, когда набор данных является выборкой, подчиняющейся непрерывному закону распределения, по выборке стоится сглаженная кумулятивная функция вероятности. Используя полученную функцию можно генерировать случайные числа и получать статистические характеристики распределения.

На рисунке 1 представлена схема математического сервиса, показывающая взаимосвязи между классами, их методы и поля. На рисунке 2 представлен алгоритм определения вероятностного распределения.

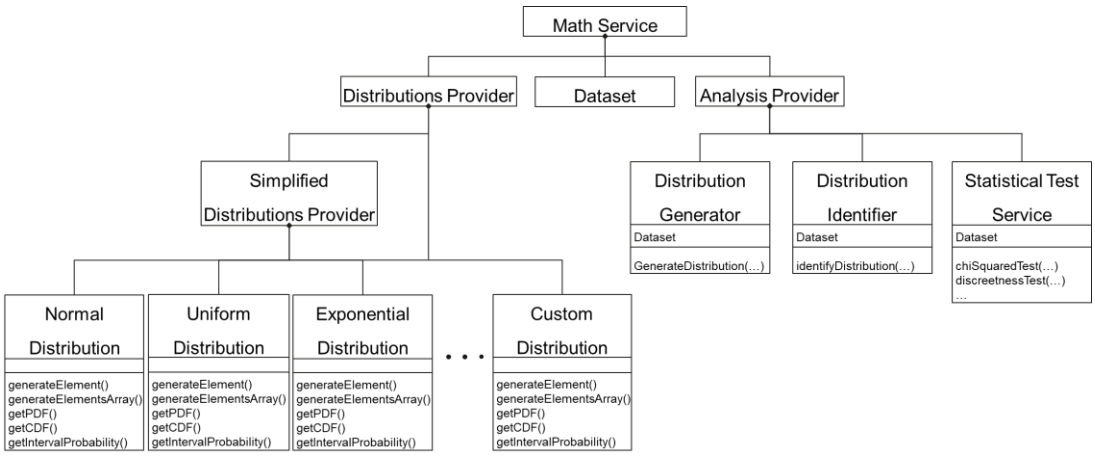


Рисунок 1 – Схема математического сервиса для CDES

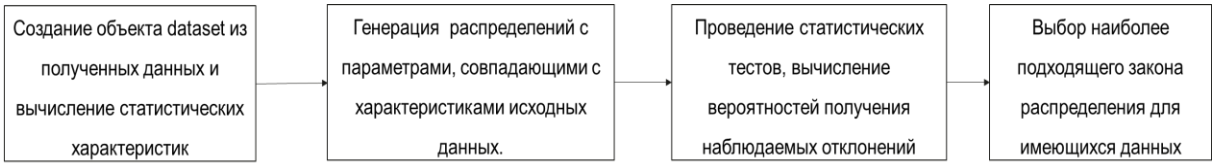


Рисунок 2 – Алгоритм для определения закона распределения полученных данных

В данной работе представлены этапы разработки математического сервиса для платформы дискретного моделирования. Данный сервис обладает широкой функциональностью, необходимой для моделирования и анализа результатов моделирования, а также достаточно прост для использования конечным пользователем, что является его основными преимуществами относительно других подобных продуктов или ненастроенных для специального использования общих библиотек математического назначения.

ЛИТЕРАТУРА

1. Имитационное моделирование поведения сложных многоагентных систем с использованием вероятностной модели / А. М. Сабуткевич, Д. А. Вихляев, И. В. Никифоров, А. В. Самочадин // Современные технологии в теории и практике программирования : Сборник материалов конференции, Санкт-Петербург, 26 апреля 2022 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2022. – С. 98-100. – EDN ACJXEE.
2. Решение проблемы комбинаторного взрыва при генерации траекторий поведения агентов в процессе имитационного моделирования / А. М. Сабуткевич, Д. А. Вихляев, И. В. Никифоров, А. В. Самочадин // Современные технологии в теории и практике программирования : Сборник материалов научно-практической конференции студентов, аспирантов и молодых ученых, Санкт-Петербург, 26–27 апреля 2023 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2023. – С. 213-215. – EDN MMMDUC.
3. Тянутов, М. В. Применение алгоритма расчета метрики TCO для центров обработки данных / М. В. Тянутов, И. В. Никифоров, Л. П. Котлярова // Современные технологии в теории и практике программирования : Сборник материалов научно-практической конференции студентов, аспирантов и молодых ученых, Санкт-Петербург, 26–27 апреля 2023 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2023. – С. 257-258. – EDN JQQCTV.
4. Библиотека Apache.Math3. [Электронный ресурс]. Режим доступа: https://commons.apache.org/proper/commons-math/javadocs/api-3.6.1/overview-summary.html
5. Business models for distributed-simulation orchestration and risk management / S. Gorecki, J. Possik, G. Zacharewicz [et al.] // Information (Switzerland). – 2021. – Vol. 12, No. 2. – P. 1-22. – DOI 10.3390/info12020071. – EDN ZJSVUF.
6. Библиотека на Colt. [Электронный ресурс]. Режим доступа: https://dst.lbl.gov/ACSSoftware/colt/

УДК 004.4

Е. З. Бакова (4 курс бакалавриата),
Д. В. Луцив, к.ф.-м.н, доцент

ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА ТРЕНАЖЁРА НА БАЗЕ ВЕБ-СЕРВИСА MIMINET

В Санкт-Петербургском государственном университете на математико-механическом факультете есть несколько предметов, касающихся темы компьютерных сетей. Однако эти курсы ограничиваются только теорией и для лучшего понимания материала студентам важна практика.

Средства эмуляции и визуализации упрощают процесс изучения сетей на практике, так как не требуют реального оборудования [1]. В СПбГУ на кафедре системного программирования на базе эмулятора Mininet [2] был создан веб-эмулятор компьютерной сети Miminet. Веб-эмулятор позволяет нарисовать и настроить сеть, а затем визуально продемонстрировать её работу: передачу, движение и получение пакетов.

Самостоятельная работа помогает обучающемуся лучше усвоить и закрепить материал [3]. Нужен был инструмент, где обучающиеся могли бы попрактиковаться в настройке сети, а также проверить и закрепить уже полученные знания и навыки, поэтому для веб-эмулятора Miminet разрабатывается и внедряется тренажер с практическими и теоретическими заданиями по компьютерным сетям.

Таким образом, цель данной работы — создание тренажера по компьютерным сетям для веб-эмулятора Miminet. Для этого были определены следующие задачи:

1. Провести обзор существующих решений.
2. Спроектировать тренажер.
3. Реализовать приложение.
4. Провести апробацию.
5. Доработать выявленные замечания.

Miminet разрабатывается с помощью фреймворка Flask для обработки HTTP-запросов и взаимодействия с базой данных, шаблонизатор Jinja2 который интегрируется с Flask для создания шаблонов страниц, язык JavaScript и библиотека jQuery используются для улучшения интерактивности веб-страниц, а Bootstrap предоставляет компоненты и стили для разработки интерфейса. В качестве СУБД выбран SQLite, а SQLAlchemy — для упрощения работы с базами данных через ORM. Так как проект на момент начала данной работы уже разрабатывался, технологии не менялись.

В ходе обзора были рассмотрены несколько образовательных платформ (Stepik, Moodle, Открытое образование), а также продукты, направленные на обучение компьютерным технологиям такие как GNS3 [4], Boson NetSim [5] и другие [6]. На основании обзора были выделены три варианта текстовых заданий: с вариантами ответов, на сортировку, на сопоставление. Также были составлены общие требования к тренажеру:

– Проверка ответов в автоматическом режиме.
– Отображение пользователю пояснения к ответу.
– Наличие возможности ограничить время прохождения теста.
– Наличие возможности запретить проходить тест несколько раз.
– Ограничение круга пользователей, которые могли бы менять, создавать и удалять тесты.

Далее был спроектирован макет интерфейса [7] с помощью инструмента Figma. Пользователь заходит на страницу тренажера, где представлены все доступные тесты (рисунок 1), каждый тест разбит по разделам. Для каждого раздела отображается общее число вопросов и количество минут, данное на прохождение. При тестировании отображается оставшееся время и сколько вопросов уже пройдено. После ответа на вопрос пользователь видит результат (верно или неверно) и пояснение к ответу (если есть).

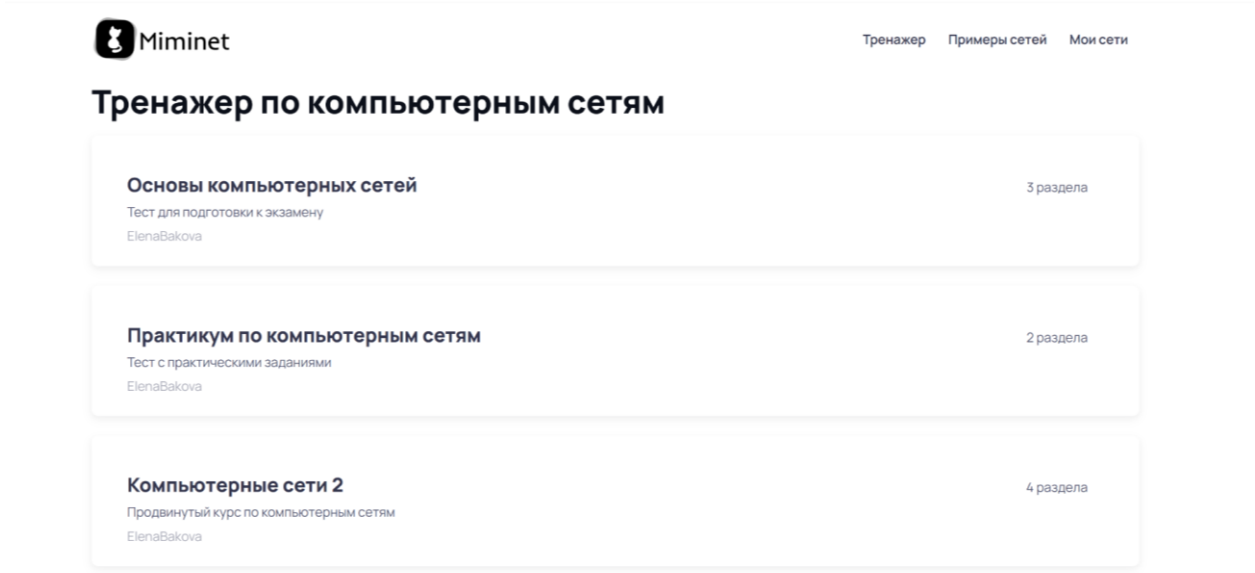


Рисунок 1 — Страница с тестами тренажера.

Теоретические вопросы преподаватель задаёт путем заполнения формы. Для практического задания преподаватель сначала задаёт конфигурацию сети, затем выбирает один из типов заданий и заполняет все необходимые параметры. Такой декларативный подход в создании задач был выбран ввиду его простоты как для использования, так и для разработки.

Проверка практического задания производится на сервере. После ответа пользователя, сеть, задаваемая как массивы вершин, рёбер и пакетов, отправляется на сервер, где эмулируется и производится проверка пакетов: отправитель, получатель, тип пакета. После чего ответ отправляется клиенту и результат отображается пользователю.

В будущем планируется внедрение тренажёра в Miminet и проведение его апробации путём предоставления студентам и преподавателям приложения для активного пользования в течение определенного периода, после чего они заполнят анкету по методике System usability scale для оценки удобства и эффективности тренажера.

По итогам работы выполнены следующие задачи:

1. Проведен обзор существующих решений.
2. Спроектирован и реализован тренажер.

ЛИТЕРАТУРА

1. Подсадников А. В., Розов К. В., Кратов С. В. Применение средств имитационного моделирования компьютерных сетей в учебном процессе //Информатика и образование. – 2021. – №. 1. – С. 47-56.
2. Mininet. [Электронный ресурс] Режим доступа: https://mininet.org/
3. Челнокова Е. А., Кузнецова С. Н. Роль самостоятельной работы студентов в образовательном процессе //Вестник Мининского университета. – 2017. – №. 1 (18). – С. 6.
4. Graphical Network Simulator. [Электронный ресурс] Режим доступа: https://www.gns3.com
5. Boson NetSim. [Электронный ресурс] Режим доступа: https://www.boson.com/netsim-cisco-network-simulator
6. Савельев Д. С., Абу-Абед Ф. Н. СРАВНИТЕЛЬНЫЙ АНАЛИЗ СЕТЕВЫХ СИМУЛЯТОРОВ, ПРИМЕНЯЕМЫХ ПРИ ПОДГОТОВКЕ IT-СПЕЦИАЛИСТОВ //ПРОБЛЕМЫ ИНФОРМАТИКИ В ОБРАЗОВАНИИ, УПРАВЛЕНИИ, ЭКОНОМИКЕ И ТЕХНИКЕ. – 2022. – С. 325-329.
7. Слива М. В. Прототипирование графического интерфейса пользователя как неотъемлемая часть процесса разработки программного обеспечения //Вестник Нижневартовского государственного университета. – 2013. – №. 1. – С. 74-76.

УДК 004.4

А. О. Баранов (2 курс магистратуры),
Н. В. Воинов, к.т.н., доцент

РАЗРАБОТКА СИСТЕМЫ ПО ОТСЛЕЖИВАНИЮ ФИНАНСОВЫХ ОПЕРАЦИЙ ДЛЯ МЕЖВАЛЮТНЫХ СЧЕТОВ

В сфере международного бизнеса с каждым днем растет необходимость в управлении финансами компаний, оперирующих в различных странах и валютах [1]. Вследствие этого разработка эффективных систем для отслеживания финансовых операций на межвалютных счетах становится более актуальной. Несмотря на наличие широкого спектра программных решений для бухгалтерского учета, многие компании предпочитают индивидуальные подходы, отвечающие их специфическим требованиям. В данном контексте разрабатываемая система призвана обеспечить компаниям, управляющими множеством банковских счетов в различных валютах, надежное и эффективное решение для отслеживания и анализа финансовых операций. Она предназначена для тех, кто ведет основной валютный счет, но осуществляет транзакции в различных местных валютах. Система решает проблему точного учета операций в местных валютах относительно основного счета, что является критически важным для выявления финансовых потерь, обусловленных нестабильностью курсов валют [2]. Разрабатываемая система внедряется во внутреннюю инфраструктуру организации, что предоставляет полный контроль, конфиденциальность и гибкость в управлении данными.

Таким образом, целью данной работы является разработка полноценной настраиваемой системы, которая автоматизирует учет финансов, снижает трудоёмкость их отслеживания внутри организации, гарантирует конфиденциальность данных и гибкое развертывание.

Для достижения этой цели необходимо решение следующих задач:

- Обзор существующих программных решений бухгалтерского учёта.
- Проведение анализа существующих решений, определение проблем и ограничений, с последующим обоснованием необходимости разработки системы, решающей найденные проблемы.
- Определение стека технологий для разработки системы.
- Проектирование архитектуры системы.
- Разработка системы.
- Проведение тестирования и оптимизации системы. Статический и динамический анализ кода.
- Описание контейнеризации системы для гибкого развёртывания и масштабируемости [3].

Архитектура разрабатываемой системы представлена на рисунке 1.

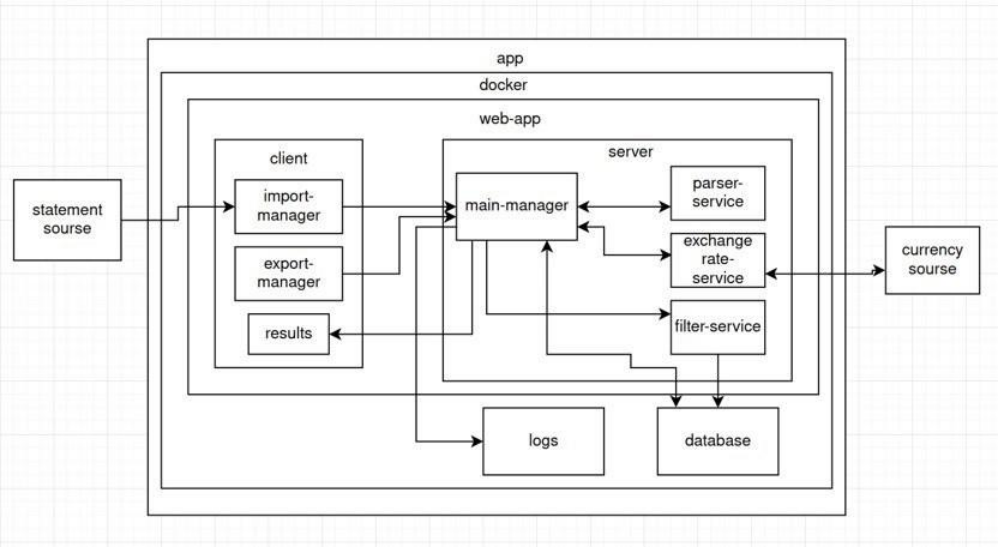


Рисунок 1 – Архитектура разрабатываемой системы

Система является монолитным Web-приложением и реализована с помощью языка Ruby и фреймворка Rails [4]. Клиентская часть служит для взаимодействия пользователя с данными в системе (загрузка банковских выписок, их экспорт, просмотр данных для их анализа). Серверная часть отвечает за обработку больших объемов данных (взаимодействие с внешними сервисами, такими как курсы валют, обработка загруженных выписок для последующего сохранения в базу данных, получение данных по API, фильтрация загруженных данных).

Каждый из элементов системы помещен в собственный Docker-контейнер: Web-приложение, база данных - PostgreSQL [5], хранилище - Redis [6], планировщик задач - Sidekiq, Logstash, ElasticSearch, Kibana.

Система спроектирована с акцентом на эффективность, масштабируемость и удобство использования. Используемый стек технологий, начиная от Ruby on Rails и заканчивая Window Functions [7,8] в PostgreSQL, стратегически выбран с целью обеспечения высокой производительности и гибкости при обработке больших объемов финансовых данных.

ЛИТЕРАТУРА

1. Accounting implication on Foreign Currency Transaction. [Электронный ресурс]. Режим доступа: https://www.researchgate.net/publication/359209390_Accounting_implication_on_Foreign_Currency_Transaction
2. Yildiz, Ferah et al. “The Effect of Foreign Exchange Differences on the Statement of Cash Flow Analytical Study”. Recep Tayyip Erdoğan Üniversitesi Sosyal Bilimler Dergisi 10/1 (June 2023), 144-162. https://doi.org/10.34086/rteusbe.1288822.
3. Документация Docker. [Электронный ресурс]. Режим доступа: https://www.docker.com/resources/what-container/
4. Язык программирования Ruby. [Электронный ресурс]. Режим доступа: https://ruby-doc.org/
5. Документация PosgreSQL. [Электронный ресурс]. Режим доступа: https://www.postgresql.org/docs/
6. Документация Redis. [Электронный ресурс]. Режим доступа: https://redis.io/docs/
7. Song, G., Ma, J., Wang, X., Jin, C., Cao, Y. (2017). Optimizing Window Aggregate Functions in Relational Database Systems. In: Candan, S., Chen, L., Pedersen, T., Chang, L., Hua, W. (eds) Database Systems for Advanced Applications. DASFAA 2017. Lecture Notes in Computer Science(), vol 10177. Springer, Cham. https://doi.org/10.1007/978-3-319-55753-3_22
8. Томилин, И. С. Развертывание микросервисной архитектуры с использованием consul, vault, nomad / И. С. Томилин, С. А. Федоров, В. Э. Шмаков // Современные технологии в теории и практике программирования : Сборник материалов научно-практической конференции студентов, аспирантов и молодых ученых, Санкт-Петербург, 26–27 апреля 2023 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2023. – С. 255-256. – EDN FUYZVW.

УДК 004.65

А. О. Барсегян (4 курс бакалавриата),
Е. Г. Михайлова, к.ф.-м.н., доцент

РАЗРАБОТКА ПРИЛОЖЕНИЯ ДЛЯ АНАЛИЗА АКТИВНОСТИ СЛУШАТЕЛЕЙ ОБРАЗОВАТЕЛЬНОЙ ПЛАТФОРМЫ «OPEN EDUCATION»: РАЗРАБОТКА МЕТРИК И ИХ ВИЗУАЛИЗАЦИЯ НА ФРОНТЕНДЕ

Онлайн-образование в последние годы стремительно набирает популярность, предоставляя студентам и преподавателям удобные и гибкие способы обучения и обмена знаниями. Платформа «Open edX» [1], одна из крупнейших и наиболее распространенных для онлайн-образования, занимает ключевую позицию в этой области.

«Open edX» предлагает множество инструментов для создания образовательных курсов и управления ими, делая ее идеальной для широкого круга образовательных учреждений [2].

Однако, несмотря на свою распространенность и универсальность, «Open edX» имеет ограничения, платформа генерирует детальные лог-файлы, которые содержат важную информацию об активности студентов и команды курса (администраторов, преподавателей), но не предоставляет достаточных инструментов для глубокого и эффективного анализа этих данных. Это становится проблемой, поскольку анализ успеваемости и образовательных трендов является ключевым для повышения качества и эффективности обучения [3].

«Open Education» [4], как одна из ведущих образовательных платформ, основанная на «Open edX», также сталкивается с этими проблемами. Отсутствие эффективных аналитических инструментов затрудняет для преподавателей контроль и оптимизацию образовательного процесса [5].

Целью данной работы является создание инструмента, который будет служить мостом между сырыми данными лог-файлов и практическими нуждами преподавателей и администраторов курсов. Это позволит преобразовать обширный массив данных в полезную информацию, способствующую более глубокому пониманию процессов обучения и вовлечения студентов, что, в свою очередь, сделает онлайн-обучение более эффективным и доступным.

Для достижения этой цели необходимо решение следующих *задач*:

– Обзор существующих инструментов для анализа лог-файлов платформы «Open Education».

– Исследование структуры и формата данных, используемых платформой «Open Education».

– Разработка метрик для анализа активности слушателей курсов.

– Реализация пользовательского интерфейса приложения.

Лог-файлы платформы «Open Education» хранятся в формате JSON, который используется для представления данных в виде пар ключ-значение. При каждом действии учащегося или команды курса генерируется JSON-документ. Все JSON-документы имеют общие поля, такие как логин пользователя, время инициации события, тип события и другие. Особый интерес представляют события, инициаторами которых являются студенты. Эти события могут быть классифицированы на несколько типов: взаимодействие с видеоконтентом, учебником, перемещение пользователя по различным разделам и ссылкам внутри обучающей системы, а также взаимодействие с вопросами или заданиями в рамках курса. Исследование данных событий позволяет осуществить сбор информации о содержании курса и учебной активности студентов. На основе этого анализа были разработаны метрики и реализованы с помощью языка SQL.

Реализованные метрики делятся на два типа: общие для курса и информация, специфичная для каждого студента.

К общим метрикам относится информация о количестве студентов, зарегистрировавшихся на курс, информация о том, сколько студентов приступили к прохождению курса, сколько студентов зарегистрировались, но не проявляли активности и количество студентов, завершивших курс. Кроме того, показывается активность взаимодействия с видеоматериалами и учебником (например, количество воспроизведений видеоматериалов по дням, количество уникальных просмотров глав учебника).

Метрики специфичные для каждого студента так же, как и общие, делятся на категории. Например, первая из них: взаимодействие с тестами и заданиями – здесь высчитывается метрика, сравнивающая количество попыток студента решить задачу и количество верно решённых задач. Эти события распределены по дням и визуализируются с помощью линейного графика (рисунок 1), в котором общее количество попыток решить задачу и количество верно решенных задач накладываются друг на друга. Если количество попыток решить задачу и количество верно решённых задач часто или абсолютно совпадает, то можно заподозрить учащегося в списывании.

Рисунок 1 – График корреляции количества попыток решить задачу и правильно решённых задач

Кроме того, был разработан пользовательский интерфейс, обеспечивающий более удобное и приятное взаимодействие пользователей с полученной информацией. Интерфейс был написан на языке программирования TypeScript с использованием фреймворка React. Для визуализации графиков и таблиц была использована библиотека React-компонентов Material UI [6].

ЛИТЕРАТУРА

1. Официальный сайт Open edX. [Электронный ресурс] Режим доступа: https://docs.edx.org/
2. Sriram M. Comparative Analysis of Massive Open Online Course (MOOC) Platforms. In Proc. of the 4th International Conference on Global Business, Economics, Finance and Social Sciences, 2015, pp. 1-7.
3. Krasnov S., Kalmykova S., Abushova E., Krasnov A. Problems of Quality of Education in the Implementation of Online Courses in the Educational Process. In Proc. of the International Conference on High Technology for Sustainable Development (HiTech), 2018, pp. 1-4.
4. Официальный сайт Open Education, о проекте. [Электронный ресурс] Режим доступа: http://npoed.ru/about
5. Mikhailova E., Boitsev A., Egorova O., Grafeeva N.G., Romanov A., Volchek D. Curriculum for Digital Culture at ITMO University//Advances in Intelligent Systems and Computing, 2020, Vol. 1161 AISC, pp. 235-244
6. Официальный сайт библиотеки React-компонентов Material UI. [Электронный ресурс] Режим доступа: https://mui.com/

УДК 004.416

Д. С. Беляев (1 курс магистратуры),
В. В. Амосов, к.т.н., доцент

РАЗРАБОТКА ВСТРАИВАЕМОГО ПРИЛОЖЕНИЯ ОПРЕДЕЛЕНИЯ ЭМОЦИОНАЛЬНОГО И ФИЗИЧЕСКОГО СОСТОЯНИЯ ВОДИТЕЛЯ В ТРАНСПОРТНЫХ СРЕДСТВАХ

В настоящее время современные технологии продолжают эволюционировать, встраиваясь в различные сферы нашей повседневной жизни. В контексте управления транспортными средствами появляется потребность в системе, способной определять эмоциональное и физическое состояние водителя для обеспечения безопасности на дороге[1]. Для этого предлагается разработать встроенное приложение, использующее видеокамеру для анализа лица водителя.

Функциональные требования:

- Распознавание лица водителя с помощью встроенной видеокамеры.
- Определение эмоционального состояния водителя на основе выражения лица, включая радость, грусть, злость и др.
- Анализ физического состояния водителя, включая усталость или сонливость.
- Вывод результата анализа на панель приборов или встроенный дисплей.
- Предупреждение водителя в случае обнаружения негативных эмоциональных или физических состояний.
- Возможность интеграции с другими системами безопасности транспортного средства.

Нефункциональные требования:

- Приложение должно обеспечивать точность распознавания лица и анализа эмоций не менее чем на 95%.
- Время обработки и анализа данных лица не должно превышать 100 миллисекунд.
- Система должна работать стабильно в различных условиях освещения и углах обзора камеры.
- Приложение должно соответствовать требованиям безопасности данных и обеспечивать защиту личной информации водителя.
- Интерфейс приложения должен быть интуитивно понятным и не отвлекать водителя от управления транспортным средством.

Для реализации предложенной системы можно использовать следующие технологии:

- Python для разработки алгоритмов обработки изображений и анализа лица.
- Библиотеки компьютерного зрения, такие как OpenCV, для распознавания лица и анализа выражений.
- Методы машинного обучения для обучения модели распознавания эмоций на изображении лица.
- Встроенные системы и платформы для интеграции приложения с оборудованием транспортного средства.
- Технологии защиты данных и шифрования для обеспечения безопасности информации о водителе.

Разработка такого приложения позволит повысить уровень безопасности на дороге, предупреждая водителей о возможных опасностях и снижая риски аварийных ситуаций вследствие эмоционального или физического перенапряжения.

Выбор языка программирования Python[2] обоснован его многочисленными преимуществами, идеально сочетающимися с требованиями для разработки встроенного приложения определения эмоционального и физического состояния водителя в транспортных средствах. Прежде всего, Python является универсальным языком программирования, предлагающим простоту и читаемость кода, что обеспечивает легкость в разработке и поддержке проекта. Благодаря своей широкой популярности и активному сообществу разработчиков, Python обладает обширной библиотекой инструментов и фреймворков, упрощающих создание комплексных систем, включая обработку изображений и анализ данных. Кроме того, Python является кроссплатформенным языком программирования, что позволяет без проблем запускать приложение на различных операционных системах, что критически важно для встраиваемого решения. Все эти факторы делают Python оптимальным выбором для разработки встроенного приложения, обеспечивая высокую производительность, гибкость и надежность системы.

Для разработки встроенного приложения определения эмоционального и физического состояния водителя в транспортных средствах, была выбрана среда разработки, которая сочетает в себе удобство, доступность и мощные возможности для работы с данными и алгоритмами обработки изображений. Наилучшим выбором в данном случае является PyCharm[3], интегрированная среда разработки для Python, разработанная компанией JetBrains. PyCharm обладает широким спектром функциональных возможностей, включая

поддержку инструментов для анализа кода, автоматическое завершение кода, отладку, интеграцию с системами управления версиями и многое другое. Благодаря своей интуитивно понятной пользовательской среде и удобной навигации, PyCharm позволит разработчикам эффективно создавать, тестировать и отлаживать код, обеспечивая высокую производительность и качество разработки. Кроме того, PyCharm поддерживает работу с различными операционными системами, что позволит команде разработчиков работать в удобной для них среде без ограничений.

Для анализа изображений и обработки лиц в реальном времени, была выбрана библиотека OpenCV[4] (Open Source Computer Vision Library). OpenCV является ведущей библиотекой компьютерного зрения с открытым исходным кодом, которая предоставляет обширный набор функций и алгоритмов для работы с изображениями и видео. Эта библиотека позволяет эффективно распознавать лица в реальном времени, анализировать их выражения и осуществлять обработку изображений для дальнейшего анализа. OpenCV широко используется в индустрии и научных исследованиях благодаря своей высокой производительности, гибкости и простоте использования. Ее активное сообщество разработчиков и постоянное обновление делают OpenCV оптимальным выбором для потребностей в обработке изображений и анализе лиц в реальном времени.

Для анализа эмоций на лицах водителей в реальном времени, праниуется использовать нейронные сети, обученные на большом объеме данных, содержащих различные эмоциональные выражения. Вот несколько рассматриваемых методов обучения нейронных сетей:

- Сверточные нейронные сети (CNN)[5]: CNN — это эффективный метод для анализа изображений, который хорошо подходит для распознавания лиц и выражений на них. Мы можем использовать предварительно обученные модели CNN, такие как VGG, ResNet или Inception, и дообучить их на нашем собственном наборе данных лиц с различными эмоциональными выражениями.
- Рекуррентные нейронные сети (RNN)[6]: RNN подходят для работы с последовательными данных, что может быть полезно для анализа последовательности выражений на лице водителя в течение времени. Мы можем использовать рекуррентные слои LSTM или GRU для обучения модели на последовательных данных изображений лиц.
- Сети глубокого обучения (Deep Learning)[7]: Мы также можем исследовать современные архитектуры сетей глубокого обучения, такие как Transformer, которые хорошо работают с последовательными данными и могут учитывать контекст изображений водителя для более точного анализа эмоционального состояния.
- Обучение с подкреплением (Reinforcement Learning)[8]: Этот метод может быть полезен для настройки параметров модели в реальном времени на основе обратной связи от системы безопасности транспортного средства. Модель может получать награду или штраф в зависимости от точности определения эмоционального состояния водителя и корректности его реакции на опасные ситуации на дороге.

Выбор конкретного метода обучения будет зависеть от характеристик набора данных, требований к производительности и степени сложности модели.

ЛИТЕРАТУРА

1. IEEE Transactions on Intelligent Transportation Systems "The Role of Advanced Driver Assistance Systems in Traffic Safety" [Электронный ресурс] Режим доступа: https://www.researchgate.net/publication/301763414_Advanced_Driver_Assistance_Systems
2. Python 3.12.1 Documentation / [Электронный ресурс] Режим доступа: https://docs.python.org/3/
3. PyCharm Documentation / [Электронный ресурс] Режим доступа: https://www.jetbrains.com/help/pycharm/getting-started.html

4. OpenCV Documentation / [Электронный ресурс] Режим доступа: https://docs.opencv.org/4.x/d6/d00/tutorial_py_root.html
5. K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks," IEEE Signal Processing Letters / [Электронный ресурс] Режим доступа: https://ieeexplore.ieee.org/document/7553523
6. Graves, A., Mohamed, A., & Hinton, G. (2013). Speech Recognition with Deep Recurrent Neural Networks. In 2013 IEEE International Conference on Acoustics, Speech and Signal Processing (pp. 6645–6649). IEEE / [Электронный ресурс] Режим доступа: https://doi.org/10.1109/icassp.2013.6638947
7. Y. LeCun, Y. Bengio, and G. Hinton, "Deep Learning," Nature / [Электронный ресурс] Режим доступа: https://www.researchgate.net/publication/277411157_Deep_Learning
8. Zoph, B., Vasudevan, V., Shlens, J., Le, Q.V. "Learning Transferable Architectures for Scalable Image Recognition." CVPR, 2018 / [Электронный ресурс] Режим доступа: https://arxiv.org/abs/1707.07012

УДК 004.42

Д. Е. Булатов, А. А. Вяземский, А. А. Шожат (4 курс бакалавриата),
Т. В. Коликова, ст. преподаватель

РАЗРАБОТКА РЕДАКТОРА ВОПРОСОВ ДЛЯ MOODLE

Для проведения тестирования среди сотрудников различных компаний, а также в учебных системах часто используется система Moodle[1], тесты для которой приходится составлять в редакторе сайта, где нет понятного интерфейса. Целью данной работы является разработка приложения с удобным интерфейсом и возможностью взаимодействия с существующими тестами.

Для достижения этой цели необходимо решение следующих задач:

1. Обзор существующих подходов составления тестов
2. Проведение сравнительного анализа существующих решений.
3. Определение ключевой функциональности приложения.
4. Реализация функциональности.

В качестве прямого аналога нашей системы существует встроенный редактор тестов системы Moodle[2], однако он обладает довольно высоким порогом вхождения в использование, требует продвинутых навыков владения системой в целом. Еще один аналог: Программа "Ассистент". Эта программа является плагином/расширением для Microsoft Word. Пользователь по специальным правилам оформляет вопросы, после чего они экспортируются в формат Moodle. У такого подхода есть несколько недостатков. Пользователь должен соблюдать определенные правила оформления, что бы программа смогла корректно сформировать файл с вопросами - это не удобно для пользователя. Второй недостаток - зависимость от проприетарного программного обеспечения.

Рисунок 1 – Диаграммы модели C4

Приложение должно поддерживать 4 типа вопросов – с текстовым и числовыми ответами, с сопоставлением и с выбором ответа – множественным или единственным. Для удобной сортировки вопросов приложение должно поддерживать работу с различными категориями, в которых вопросы будут иметь уникальные и автоматически формируемые имена. Для единообразности оформления тестов на портале все тексты приводятся к одному стилю. Поскольку приложение будет использоваться на разных компьютерах, мы выбрали один из популярнейших языков программирования для осуществления данной задачи — язык Java[3]. Парсинг данных производился с помощью библиотеки jaxb[4].

Первой причиной того почему мы выбрали этот язык программирования является то, что, благодаря высокой кроссплатформенности языка Java, разработанные на нём приложения одинаково запускаются и работают на различных десктопных операционных системах. Второй причиной выбора этого языка являлось то, что в составе JRE распространяется также GUI фреймворк Swing[5], использование которого позволяет существенно сократить вес приложения в сравнении с другими кроссплатформенными GUI-фреймворками, для использования которых пришлось бы производить компиляцию нативного кода под каждую платформу.

Внешний вид приложения представлен на рисунке 2.

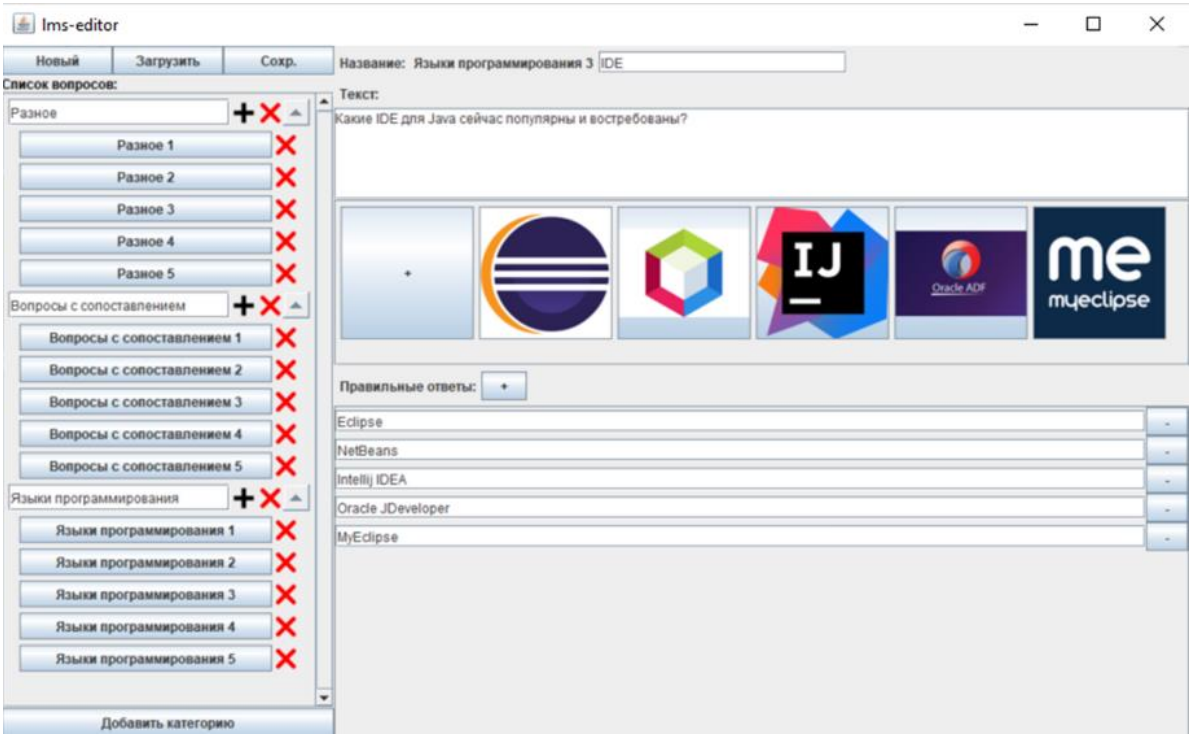


Рисунок 2 – Пример работы программы

ЛИТЕРАТУРА

1. Гильмутдинов А.Х., Ибрагимов Р.А., Цивильский И.В. Элекронное образование на платформе Moodle. – Казань, КГУ, 2008. – 170 с.
2. Андреев А.В., Андреева С.В, Доценко И.Б. Практика электронного обучения с использованием Moodle. – Таганрог: Изд-во. ТТИ ЮФУ, 2008. – 146 с.
3. Кей Хорстманн, Гари Корнелл "Java. Библиотека профессионала. Том 1". 11-е издание
4. JAXB. [Электронный ресурс] Режим доступа: https://www.oracle.com/java/technologies/jaxb.html
5. Swing. [Электронный ресурс] Режим доступа: https://docs.oracle.com/en/java/javase/17/docs/api/java.desktop/javax/swing/package-summary.html

УДК 004.4

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

РАЗРАБОТКА КОМПЛЕКСА ИНСТРУМЕНТОВ ДЛЯ АНАЛИЗА БИЗНЕС-ПРОЦЕССОВ В БУХГАЛТЕРСКОЙ СИСТЕМЕ

На сегодняшний день, бухгалтерия является одной из самых регламентированных областей. Каждый шаг в любом процессе строго задокументирован законами. Теми же законами регламентировано и обязательное участие человека в процессах, что делает невозможной полную автоматизацию всей сферы целиком. Помимо законодательного запрета, автоматизации мешает и то, что учитывать все возможные проводки, планы счетов, оформления документов и другие процессы – слишком трудоемкая задача. Поэтому бизнес пошел по пути разработки инструментов для упрощения ведения бухгалтерской деятельности. Самым популярным решением в России на данный момент, является 1С: Бухгалтерия. Однако пользователь заинтересован в дополнительной функциональности. Самый частый запрос от клиентов– средства мониторинга для повышения эффективности бизнес-процессов.

В данной работе поставлена задача, реализовать средства мониторинга и визуализации бизнес-процессов [1,3]. На рисунке 1 представлена схема взаимодействия различных элементов системы в процессе мониторинга.

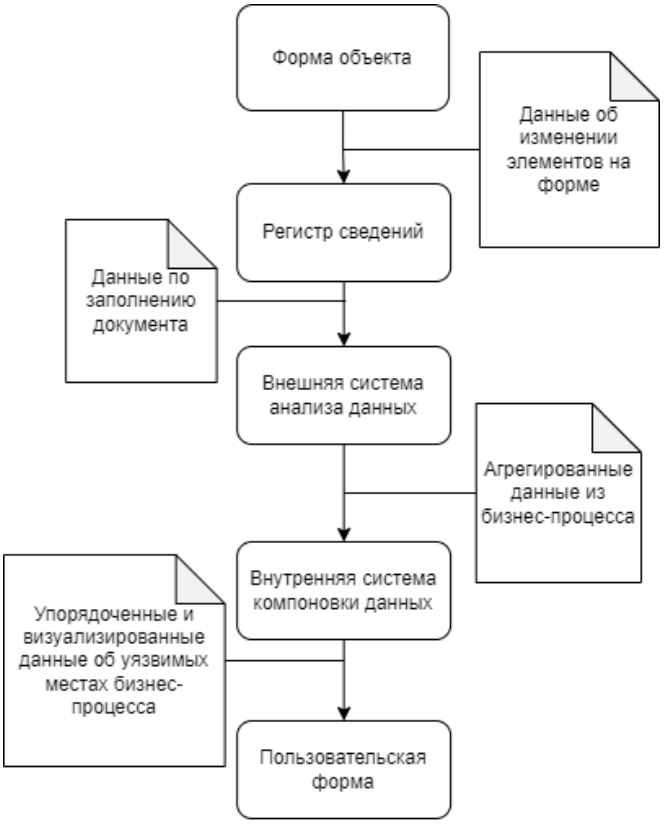


Рисунок 1 – Схема процесса мониторинга

Форма объекта предназначена для отображения и редактирования информации, содержащейся в базе данных. В данной работе расширяется функционал стандартной формы, чтобы отслеживать каждое изменение пользователем. После заполнения и проведения документа, данные об изменении записываются в регистр сведений.

Регистр сведений, позволяет хранить произвольные данные в разрезе нескольких измерений. В данном решении, такими измерениями будет ссылка на изменяемый документ и дата его последнего изменения. Таким образом обеспечивается целостность данных и

учитывается возможность редактирования уже проведенного документа. После того, как собрано достаточно данных по всем бизнес-процессам, информация из регистра сведений передается во внешнюю систему анализа данных, где для каждого процесса система выводит карту бизнес- процесса с помощью алгоритмов кластеризации и упорядочивает данные по сотрудникам и бизнес-процессам, в которых они участвовали. [2]

Данные из внешней системы передаются во внутреннюю систему для компоновки данных, чтобы упорядочить и визуализировать полученные данные из внешней системы. Система компоновки данных предназначен для построения отчетов, а также вывода информации, содержащий произвольный набор таблиц и диаграмм. Полученная визуализация отображается пользователю на форме, где он может самостоятельно увидеть слабые места в своей работе, чтобы повысить свою эффективность. [4,5]

На рисунке 2 изображена схема взаимодействия системы со средствами мониторинга

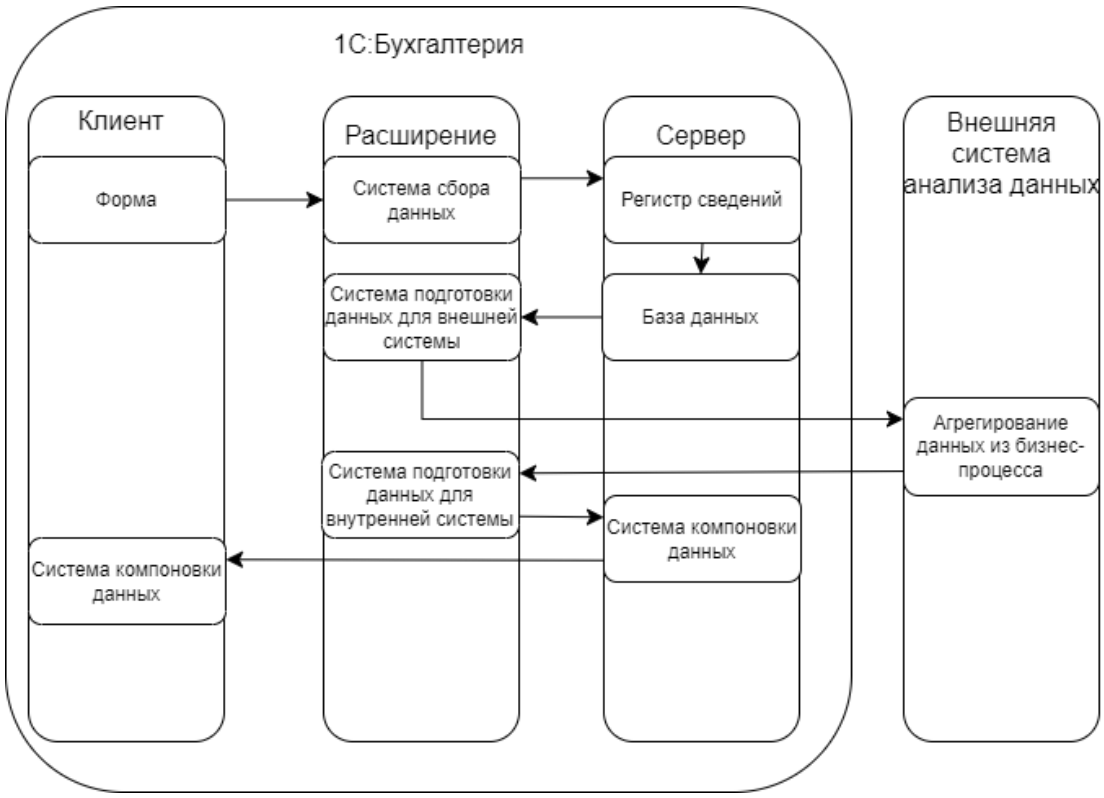


Рисунок 2 – Схема взаимодействия системы со средствами мониторинга

Вывод: в ходе работы были изучены основные подходы к реализации бизнес-процессов в 1С: Бухгалтерия. На основе полученных данных, была разработана архитектура и описаны модули системы мониторинга бизнес-процессов.

ЛИТЕРАТУРА

1. Коренюк, С. Программирование на языке запросов 1С:Предприятие 8 / С. Коренюк. - СПб.: БХВ-Петербург, 2018. - 256 с.
2. Голубев, И. Разработка интегрированных приложений с использованием 1С:Предприятие 8.3 / И. Голубев. - М.: Диалектика, 2020. - 416 с.
3. Ключарев, В. Методы анализа и проектирования информационных систем / В. 3. Ключарев. - М.: Издательство Национального открытого университета "ИНТУИТ", 2017. - 224 с.
4. Лавриненко, С. Бизнес-анализ и моделирование процессов с использованием BPMN / С. Лавриненко. - Киев: Видавництво Старого Лева, 2019. - 352 с.
5. Матвеев, А. Управление предприятием на основе процессов / А. Матвеев. - М.: ИНФРА-М, 2018. - 288 с.

УДК 004.42

Д. Е. Булатов, А. А. Вяземский, А. А. Шожат (4 курс бакалавриата),
Т. В. Коликова, ст. преподаватель

РАЗРАБОТКА TELEGRAM-БОТА ДЛЯ ОРГАНИЗАЦИИ ОЧЕРЕДЕЙ

В современном мире автоматизация процессов играет ключевую роль в обеспечении эффективной работы организаций и учебного процесса. Зачастую студентам приходится организовывать очереди для сдачи работ и экзаменов. В этом контексте создание программных решений, способных оптимизировать организацию очередей, становится актуальной задачей.

На личном опыте мы выявили функционал, который должен присутствовать в разрабатываемом проекте. Бот должен предоставлять весь необходимый функционал для работы с очередями. Каждый пользователь может задать отображаемое имя, которое будет указано в списке на сдачу. Бот позволяет создавать предметы, по которым в дальнейшем можно создать очередь. Бот представляет гибкий механизм записи в очередь. Пользователь может выбрать самый удобный для себя вариант: запись на первое свободное место, запись в конец очереди и запись на определенное место. Также должен присутствовать удобный механизм обмена местами, по обоюдному согласию пользователей они могут поменяться своими местами.

Для разработки проекта требуется выполнить следующие задачи:

- Разработка Telegram-бота;
- Настройка базы данных;
- Разработка ПО для связи базы данных и бота;
- Развертывание системы;
- Тестирование системы.

Нами была разработана архитектура проекта по принципу Interface Segregation Principle[1]

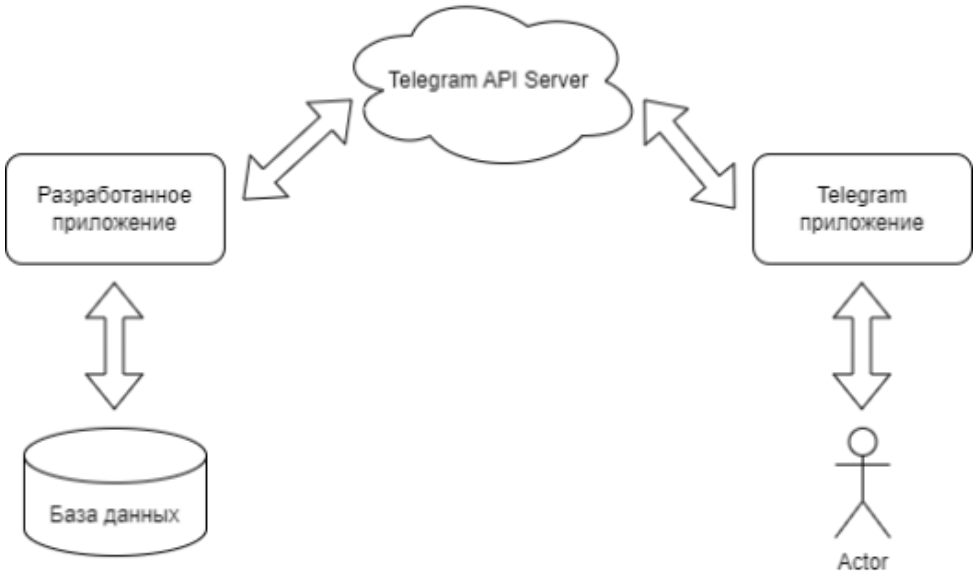


Рисунок 1 – Архитектура проекта

Преимущества разработанного Telegram-бота охватывают как архитектурные особенности, так и широкий спектр функциональных и нефункциональных требований, которые были успешно реализованы и покрыты тестами. В процессе разработки было уделено внимание высокому качеству кода, что привело к созданию чистой и легко читаемой программной системы. Разбиение на классы позволило логически группировать функционал, облегчая поддержку и расширение кодовой базы в будущем.

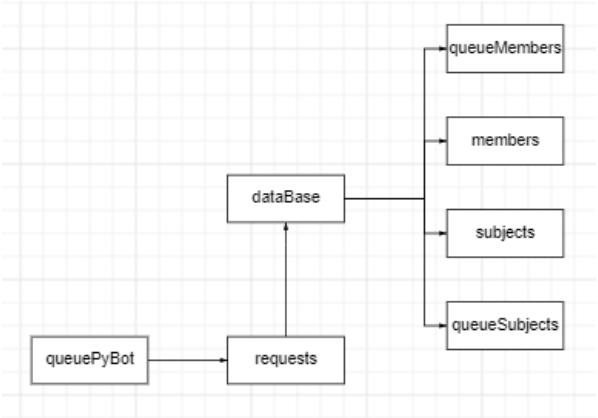


Рисунок 2 – Диаграмма классов

Особое внимание в данном проекте уделено тестированию[2] разработанного Telegram-бота[3]. Были проведены модульные, интеграционные и системные тесты, охватывающие проверку каждого модуля программы, каждого взаимодействия между модулями и базой данных, а также взаимодействия с Telegram API[4] при помощи библиотеки Pyrogram. Такой подход к тестированию позволяет обеспечить высокую надежность и стабильную работу разработанного программного решения.

Проект направлен на создание удобного и эффективного инструмента для организации очередей, который позволит пользователям быстро и удобно записываться на услуги, получать актуальную информацию о состоянии очередей и эффективно управлять своим временем. Результаты исследования и практическая реализация данного проекта могут быть полезны для различных сфер деятельности, включая общественные учреждения, медицинские организации, банки и торговые центры, где эффективное управление очередями играет важную роль в обеспечении качественного обслуживания клиентов и оптимизации рабочих процессов.

ЛИТЕРАТУРА

1. Роберт Мартин, Чистая архитектура. Искусство разработки программного обеспечения. 2022, - с. 232-235
2. Реализация системного тестирования для телеграм-ботов. [Электронный ресурс] Режим доступа: https://shallowdepth.online/posts/2021/12/end-to-end-tests-for-telegram-bots/
3. pyTelegramBotAPI. [Электронный ресурс] Режим доступа: https://pypi.org/project/pyTelegramBotAPI/
4. Modrzyk N. Building Telegram Bots: Develop Bots in 12 Programming Languages using the Telegram Bot API, 2019 – ISBN 978-1-4842-4196-7

УДК 004.42

Д. Е. Булатов (4 курс бакалавриата),
Т. В. Коликова, ст. преподаватель

РАЗРАБОТКА АРХИТКТУРЫ СИСТЕМЫ УМНЫЙ ДОМ

В современном мире технологий и быстрого развития интернета вопрос создания Умного дома[1] становится все более актуальным. Умные дома представляют собой системы, интегрирующие различные устройства и технологии для автоматизации и управления жилым пространством. Разработка архитектуры такой системы имеет огромное значение, поскольку от правильного выбора компонентов и их взаимодействия зависит эффективность и удобство использования технологии.

Целью данной работы является разработка функционирующей системы Умный дом, включающая в себя исполнительные устройства, а также приложение для управления.

Система должна быть разработана с возможностью горизонтального масштабирования и учитывать недостатки аналогичных, существующих решений, которые будут рассмотрены далее.

Таблица 1 – Сравнение аналогов

	Возможность взаимодействия	сервер	Обмен данными внутри системы	Стоимость (рубли)	надежность
Разрабатываемый проект	Свое приложение	есть, локальный	есть	100-1000(одно устройство)	средняя
Xiaomi	приложение экосистемы Xiaomi	есть, удаленный	нет	2000-6000(одно устройство)	низкая
MajorDoMo	свое приложение	есть, удаленный	есть	5000-10000(одно устройство)	средняя
Nanoleaf	свое приложение	есть, локальный	есть	1200000-2000000 (вся система)	высокая

Существенная разница между представленными решениями заключается в архитектуре, по которой они построены. Nanoleaf является специализированной компанией, проектирующей и внедряющей систему Умный дом в объекты на этапе ремонта. Особенность такого подхода заключается в высокой надежности и защищенности системы. Остальные же системы для обмена данными используют беспроводные технологии, что немного снижает надежность системы, но многократно снижает стоимость внедрения. Кроме способа передачи данных к существенным различиям между системами можно отнести расположение сервера. У Xiaomi он удалённый и вся обработка данных происходит на нем, из-за чего возникает дополнительная задержка, а при обрыве связи устройства становятся не работоспособными.

Рассмотрев существующие решения можем сделать вывод о том, что система должна иметь беспроводное соединение между компонентами и локальный сервер. Для достижение этих условий была разработана архитектурная схема, включающая в себя пользовательское приложение, основной роутер, подключенный к сети Интернет, локальный сервер с базой данных и MQTT брокером, дополнительный роутер для обеспечения внутренней связи между устройствами Умного дома.

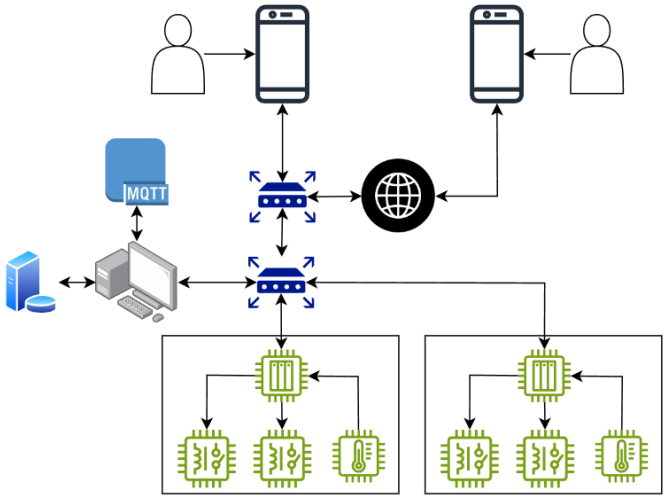


Рисунок 1 – Архитектурная схема разрабатываемой системы.

В качестве беспроводной сети для передачи информации выбран Wi-Fi. Он имеет ряд преимуществ по сравнению с Zigbee[2] и Bluetooth, которые так же являются наиболее распространенными для Умных устройств. Во-первых, Wi-Fi имеет большую дальность

действия, во-вторых, можно настроить прямой доступ к системе из глобальной сети без дополнительных устройств, в-третьих, микроконтроллеры, содержащие в себе чип с поддержкой данной технологии, имеют стоимость значительно ниже, чем дополнительные расширения для других протоколов.

Для распределения сообщений между устройствами Умного дома используется протокол MQTT, брокер которого развернут на локальном сервере при помощи инструмента с открытым исходным кодом mosquito[3]. Данный протокол использует технологию издатель-подписчик, что позволяет не обрабатывать ненужные сообщения устройствам. Которые они не предназначены, а кроме того, предоставляет возможность горизонтального расширения продукта и гибкой настройки.

В качестве контроллеров, обрабатывающих показания датчиков и управляющих конечными устройствами используются контроллеры семейства esp[4]. Данные модули имеют поддержку Wi-Fi, цифровые и аналоговые входы, низкую стоимость, поддержку энергосберегающего режима, возможность создания локальной сети. Все описанные преимущества делают данный тип контроллеров безальтернативным решением для подобного проекта.

Поскольку система управляется при помощи собственного приложения, в ней предусмотрены меры безопасности. Доступ к управлению осуществляется с помощью аутентификации, при этом пользовательские данные хранятся на локальном сервере, а сообщения, отправляемые на контроллеры, шифруются.

ЛИТЕРАТУРА

1. Черняк, А. А. Система «Умный дом» // Молодой ученый С. 51-53., 2020.
2. Закалюжный А.А., Кудряшев С.Б. ZIGBEE – ПРОТОКОЛ СОВРЕМЕННОЙ БЕСПРОВОДНОЙ ТЕХНОЛОГИИ ПЕРЕДАЧИ ДАННЫХ // Международный студенческий научный вестник С. 470-472, 2018.
3. https://mosquitto.org/
4. https://www.espressif.com/en/products/modules

УДК 519.6

Шань Ван (1 курс аспирантуры),
Ю. Б. Сениченков, д.т.н., профессор

КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ СОБЫТИЙНО-УПРАВЛЯЕМЫХ СИСТЕМ

Современные программные среды моделирования сложных динамических систем предоставляют возможность строить и исследовать событийно-управляемые динамические системы. Существуют достаточно подробные обзоры различных используемых на практике математических определений событийно-управляемых систем и соответствующих им графических языков, однако отсутствует сравнительный анализ результатов моделирования в различных средах.

Целью данной работы является разработка методики сравнения результатов моделирования событийно-управляемых систем.

Для достижения этой цели необходимо:

1. Выяснить, какие математические модели событийно-управляемых систем используются в современных средах моделирования.
2. Описать графические языки, соответствующие используемым определениям.
3. Провести сравнительный анализ численных методов, используемых для моделирования событийно-управляемых систем.
4. Построить тестовый набор для проведения вычислительных экспериментов.

Математические модели событийно-управляемых систем рассматривались во многих работах [1-4], однако появляются все новые среды, в том числе и универсальные (MapleSim, System Modeler), и этот вопрос требует уточнения.

Современной стандартной моделью можно считать машины состояний UML [19], допускающие непрерывное поведение в состояниях. Это подход восходит к работам [11-13,18], и был впервые реализован в средах MvStudium (https://www.mvstudium.com/) и AnyLogic (https://www.anylogic.ru/) [6]. Позже машины состояний UML появились в средах Simulink (Stateflow) (https://www.mathworks.com/products/stateflow.html) и OpenModelica [20]. Другой подход использован в среде Ptolemy II (https://ptolemy.berkeley.edu/ptolemyII/index.htm).

Событийно-управляемые системы реализованы в следующих средах: AnyLogic [31], Stateflow [32], Dymola [33], OpenModelica [37], Ptolemy II [34], HyTech [35], AnyDynamics [36], и они используют различные графические языки для представления событийно-управляемых систем [18-20,29-30].

В соответствии приведенным планом работ выполнен обзор литературы [1-37], посвященной использованию событийно-управляемых систем в современных средах моделирования. Проведено сравнение математических моделей, используемых для построения событийно-управляемых систем [31-36] и графических языков, с которыми работают пользователи при создании событийно-управляемых систем. В настоящее время ведется построение набора примеров для сравнения моделирования событийно-управляемых систем в средах AnyLogic, OpenModelica, AnyDynamics.

ЛИТЕРАТУРА

1. Сениченков Ю.Б. Численное моделирование гибридных систем. СПб.: Изд-во Политехн. ун-та, 2004. 206 с.
2. Новиков Е. А., Шорников Ю.В. Компьютерное моделирование жестких гибридных систем: монография: Новосибирск: Изд-во НГТУ, 2012. - 451 с.
3. Парийская Е. Ю. Сравнительный анализ математических моделей и подходов к моделированию и анализу непрерывно-дискретных систем / Е. Ю. Парийская // Дифференциальные уравнения и процессы управления. – 1997. – No 1. – С. 91–120.
4. Колесов, Ю. Б. Моделирование систем. Динамические и гибридные системы. Учебное пособие. / Ю.Б. Колесов, Ю.Б. Сениченков. – СПб.: БХВ-Петербург, 2012. – 224 с.
5. Шорников Ю. В., Уатай Б. У., Новиков Е. А. Методы решения жестких задач, гибридные системы и их приложения: монография: Изд-во КБТУ, г. Алматы, Казахстан, 2010. - 147 с.
6. Borshchev A Java engine for UML based hybrid state machines / A. Borshchev, Yu. Kolesov, Yu. Senichenkov / In Proceedings of Winter Simulation Conference, Orlando, California, USA, 2000. -p. 1888-1897
7. Yu. Kolesov, Yu. Senichenkov. Model Vision 3.0 for Windows 95/NT. The graphical environment for complex dynamic system design. ICI&C'97. International conference on Informatics and Control. St. Petersburg, 1997. pp. 764-768.
8. Yu. Kolesov, Yu. Senichenkov. Visual Specification language intended for event-driven hierarchical dynamical systems with variable structure / ICI&C'97. International conference on Informatics and Control. St. Petersburg, 1997. pp. 704-711.
9. Ю.Б Колесов, Ю.Б. Сениченков. Визуальное моделирование сложных динамических систем. С. Петербург, «Мир и семья и Интерлайн», 2000. ISBN: 5-9212-0018-2, 240 с.
10. Witsenhausen, Hans. "A class of hybrid-state continuous-time dynamic systems." IEEE Transactions on Automatic Control 11.2 (1966): 161-167/
11. Alur, Rajeev, et al. "Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems." International Hybrid Systems Workshop. Berlin, Heidelberg: Springer Berlin Heidelberg, 1991.
12. Henzinger, Thomas A. "The theory of hybrid automata." Proceedings 11th Annual IEEE Symposium on Logic in Computer Science. IEEE, 1996.
13. Lynch, Nancy, Roberto Segala, and Frits Vaandrager. "Hybrid i/o automata." Information and computation 185.1 (2003): 105-157.

14. Alla, Hassane, and René David. “Continuous and hybrid Petri nets.” Journal of Circuits, Systems, and Computers 8.01 (1998): 159-188.
15. Bemporad, Alberto, and Manfred Morari. “Control of systems integrating logic, dynamics, and constraints.” Automatica 35.3 (1999): 407-427.
16. Chaochen, Zhou, Charles Anthony Richard Hoare, and Anders P. Ravn. “A calculus of durations.” Information processing letters 40.5 (1991): 269-276.
17. 李晓山 and 周巢尘, “时段演算综述,” 计算机学报, vol. 17, no. 11, pp. 842–851, 1994
18. Y. Kesten, A. Pnueli, J. Sifakis, S. Yovine, "Integration Graphs: a Class of Decidable Hybrid Systems," in Robert L. Grossman, Anil Nerode, Anders P. Ravn, Hans Rischel, eds., *Hybrid Systems*, Springer-Verlag, 1993
19. Г. Буч, Д. Рамбо, И. Якобсон. The Unified Modeling Language Users Guide. — 2-е. — М.: ДМК Пресс, 2006. 496 с. — ISBN 5-94074-334-X
20. Hilding Elmqvist, Fabien Gaucher, Sven Erik Mattsson, Francois Dupont. State Machines in Modelica. Proceedings of the 9th International Modelica Conference September 3-5, 2012, Munich, Germany.
21. Андронов А.А. Качественная теория динамических систем второго порядка / А.А. Андронов, Е.А. Леонтович, И.И. Гордон, А.Г. Майер. М.: Наука, 1966. -568 с.
22. Уткин В.И. Скользящие режимы в задачах оптимизации и управления. М.: Наука, 1981.-368 с
23. Филиппов А.Ф. Дифференциальные уравнения с разрывной правой частью. - М.: Наука, 1985.-223 с
24. Borshchev A. Distributed Simulation of Hybrid Systems with AnyLogic and HLA / A. Borshchev, Yu. Karpov, V. Kharitonov // Future Generation Computer Systems. 2002. P. 829-839.
25. Maler O., Manna Z., Pnueli A. From timed to hybrid systems / O. Maler, Z. Manna, A. Pnueli / In Proceedings of the REX workshop "Real-Time: Theory in Practice", LNCS. Springer Verlag, New York, 1992.
26. Mosterman P. An overview of hybrid simulation phenomena and their support by simulation packages // Hybrid Systems: Computation and Control, vol. 1569 of Lecture Notes in Computer Science. Springer Verlag. 1999. P. 165-177.
27. Chinosi, M. and Trombetta, A., 2012. BPMN: An introduction to the standard. Computer Standards & Interfaces, 34(1), pp.124-134.
28. Chen, P.P.S., 1976. The entity-relationship model—toward a unified view of data. ACM transactions on database systems (TODS), 1(1), pp.9-36.
29. Gronback, R.C., 2009. Eclipse modeling project: a domain-specific language (DSL) toolkit. Pearson Education.
30. Friedenthal, S., Moore, A. and Steiner, R., 2014. A practical guide to SysML: the systems modeling language. Morgan Kaufmann.
31. Borshchev, A., 2014. Multi-method modelling: AnyLogic. Discrete-event simulation and system dynamics for management decision making, pp.248-279.
32. Hamon, G. and Rushby, J., 2004, March. An operational semantics for Stateflow. In International Conference on Fundamental Approaches to Software Engineering (pp. 229-243). Berlin, Heidelberg: Springer Berlin Heidelberg.
33. Brück, D., Elmqvist, H., Mattsson, S.E. and Olsson, H., 2002, March. Dymola for multi-engineering modeling and simulation. In Proceedings of modelica (Vol. 2002). Citeseer.
34. Ptolemaeus, C. ed., 2014. System design, modeling, and simulation: using Ptolemy II (Vol. 1). Berkeley: Ptolemy. org.
35. Henzinger, T.A., Ho, P.H. and Wong-Toi, H., 1997. HyTech: A model checker for hybrid systems. In Computer Aided Verification: 9th International Conference, CAV'97 Haifa, Israel, June 22–25, 1997 Proceedings 9 (pp. 460-463). Springer Berlin Heidelberg.
36. Ю.Б. Сениченков, Ю.Б. Колесов. Использование визуальной среды Rand Model Designer для разработки промышленных приложений. Университетский научный журнал, (8), 2014, p.112-123.
37. Modelica – A unified object-oriented Language for system modeling. Language specification. Version 3.4. 2017 (https://specification.modelica.org/v3.4/MLS.html)

УДК 004.415.2

В. В. Гладун (2 курс магистратуры),
Ю. В. Юсупов, к.т.н.

РАЗРАБОТКА СЕРВИСА ПО ПОИСКУ РАБОТЫ ДЛЯ IT-СПЕЦИАЛИСТОВ НА ОСНОВЕ АНАЛИЗА РЫНКА ТРУДА

Целью работы является разработка сервиса по поиску работы для IT-специалистов на основе анализа рынка труда. Данная задача является весьма актуальной. Так, по данным сервиса HeadHunter, наблюдается рост числа вакансий в России для IT-специалистов. Количество вакансий выросло в январе 2023 года на 63% по сравнению с тем же периодом 2022-го и достигло за неполный месяц 58700. Также по данным сервиса «Авито Работа», количество вакансий в январе 2023 года увеличилось на 52%, по итогам 2022 года - на 7%. Представители компании «МТС Диджитал» заявляют о росте найма сотрудников в 2022 году в 4 раза по сравнению с предыдущими годами.

Существует множество различных решений в области поиска и найма IT-специалистов в России [1]. Самыми популярными являются HeadHunter, GeekJob, ХабрКарьера, HeadHunter является лидером по популярности, сервис обладает наибольшей базой вакансий и резюме, а также располагает широким функционалом по аналитике как для работодателей, так и для соискателей. GeekJob позиционирует себя как анонимный сервис по поиску работу, данный ресурс располагает меньшим функционалом по сравнению с HeadHunter, но анонимность бывает важна для людей, недовольных своей нынешней работой и без возможности легко сменить работодателя. Tproger и Хабр - изначально сайты со статьями по программированию, агрегирование вакансий не является основным функционалом. Tproger располагает основным функционалом просмотра и отклика на вакансии.

Анализ существующих решений показал, что ни одно из них не предоставляет функционал поиска на основе применения искусственного интеллекта (ИИ) [2]. Данное нововведение поможет найти наиболее подходящую вакансию на основе текста и отзывов.

В рамках работы предполагается разработать собственный сервис-агрегатор вакансий с интеграцией ИИ. С помощью данного функционала пользователю будет проще скорректировать собственное резюме, а также точнее и выразительнее подготовить сопроводительное письмо для работодателя. Сервис позволит подобрать оптимальную вакансию из всех вакансий в базе. А при получении приглашения на собеседование поможет подготовиться к интервью [3].

Разработанное решение будет представлено в виде веб-сервиса [4] с клиент-серверной архитектурой [5]. Фронтенд [6] и бэкенд планируется реализовать на Python с использованием Django [7]. При разработке будет применяться IDE Pycharm в операционной системе MacOS.

Среди основных функций сервиса можно выделить:

- просмотр доступных вакансий по заданным параметрам;
- просмотр статистики по всем вакансиям в базе;
- функция корректировки резюме при помощи ИИ;
- умный подбор с использованием ИИ;
- функция помощи в подготовке к интервью с применением ИИ.

Функционал ИИ планируется реализовать посредством использования API [8], которыми пользуются в своих проектах одни из крупнейших мировых корпораций - Microsoft, Google, Delta Corp [9].

ЛИТЕРАТУРА

1. Логинов, А. В. Разработка веб-приложения сопровождения процесса отчетности сотрудников / А. В. Логинов, А. И. Тышкевич // 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. 42-44. – EDN DFYLXR.

2. Implementation and Analysis of Algorithms for Pitch Estimation in Musical Fragments / N. V. Voinov, D. A. Ivanov, T. V. Leontieva, S. A. Molodyakov // Proceedings of 2021 24th International Conference on Soft Computing and Measurements, SCM 2021 : 24, St. Petersburg, 26–28 мая 2021 года. – St. Petersburg, 2021. – P. 113-116. – DOI 10.1109/SCM52931.2021.9507134. – EDN CEIXTQ.
3. 4 ways to use AI in your job search [электронный ресурс]. Режим доступа: https://www.colorado.edu/career/2023/11/01/4-ways-use-ai-your-job-search
4. Орлов, Л. О. Управляющая система для умного дома / Л. О. Орлов, А. И. Тышкевич // 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. 53-55. – EDN PILPYY.
5. Зеленова, Д. Д. Автоматизация тестирования пользовательского интерфейса web-приложения / Д. Д. Зеленова, Н. В. Воинов, Т. В. Леонтьева // Современные технологии в теории и практике программирования: сборник материалов конференции, Санкт-Петербург, 19 апреля 2019 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2019. – С. 104-106. – EDN BVYPKM.
6. Шушарин, А. В. Клиентское приложение на базе Android для литературного портала / А. В. Шушарин, А. И. Тышкевич // Современные технологии в теории и практике программирования: Сборник материалов научно-практической конференции студентов, аспирантов и молодых ученых, Санкт-Петербург, 26–27 апреля 2023 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2023. – С. 187-188. – EDN MEGIAA.
7. Программные инструменты обработки и визуализации данных. Elasticsearch, Logstash, Kibana, Grafana, Prometheus / И. В. Никифоров, О. А. Юсупова, Н. В. Воинов [и др.]. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2023. – 140 с. – ISBN 978-5-7422-8075-0. – DOI 10.18720/SPBPU/2/id23-74. – EDN KPGDVS.
8. Чучин, Д. Ю. Разработка серверной части приложения для предоставления интерактивной среды клиентам HTTP API / Д. Ю. Чучин, В. Э. Шмаков // Современные технологии в теории и практике программирования: Сборник материалов конференции, Санкт-Петербург, 26 апреля 2022 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2022. – С. 112-114. – EDN EARKZS.
9. Цифровые технологии. Вычислительная техника. Программирование / В. Е. Баранов, Н. В. Ежова, Ю. В. Сотсков, В. Э. Шмаков. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2019. – 186 с. – ISBN 978-5-7422-6509-2. – EDN GKQIML.

УДК 004.42

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

РАЗРАБОТКА ПРИЛОЖЕНИЯ, ВЫПОЛНЯЮЩЕГО ОПТИМИЗАЦИЮ МОДЕЛИ ДЛЯ 3D ПЕЧАТИ

В настоящее время технологии 3D печати набирают все большую популярность. Детали, напечатанные на 3D принтере, уже внедряются в массовое производство.

Существует программное обеспечение, позволяющее проектировать изделия, подготавливать модели к 3D печати. Но для того, чтобы напечатать модель, недостаточно просто ее смоделировать. При проектировании нужно учитывать особенности моделирования под 3D печать, особенности определенной технологии 3D печати.

Это статья посвящена разработке приложения, которое упростит процесс проектирования 3D модели для ее дальнейшей печати на FFF 3D принтере [1]. Приложение позволит автоматизировать процесс построения оптимизаций модели для печати. Эти оптимизации дадут возможность экономить время и материал, улучшить качество изделия, упростить дальнейшую работу с ним. Существуют методы оптимизации, с помощью которых можно исправить или уменьшить некоторые дефекты печати. Также приложение поддерживает функции анализа созданной модели: указывает области, на которые инженеру стоит обратить внимание при проектировании.

Процесс производства изделия на FFF 3D принтере начинается с проектирования, создания 3D модели изделия в САПР [2]. На этом этапе инженеру необходимо учитывать технологию, по которой в дальнейшем будет изготовлено изделие. Построение оптимизаций проводится именно во время проектирования. Разрабатываемое приложение позволит автоматизировать этот процесс.

Далее созданная модель подготавливается к 3D печати в специальных программах слайсерах. Эти программы преобразовывают созданную на предыдущем этапе модель в последовательность команд, которые может выполнить 3D принтер.

Функционал приложения, на данный момент, состоит из двух больших блоков: автоматическое построение оптимизаций и анализ модели.

Функции анализа моделей, выделения областей модели, на которые стоит обратить внимание инженеру, существуют в большинстве современных слайсеров. Но этот анализ полезен не только на этапе подготовке к печати, но и на этапе проектирования. Заметив ошибку, с помощью такого анализа, на раннем этапе проектирования инженер сможет быстро ее исправить. Если же эта ошибка будет найдена на этапе подготовке к печати, то исправление модели может занять продолжительное время. Решений, которые позволяют добавить в САПР такой функционал мной найдено не было. На рисунке 1 представлен пример работы этой функции.

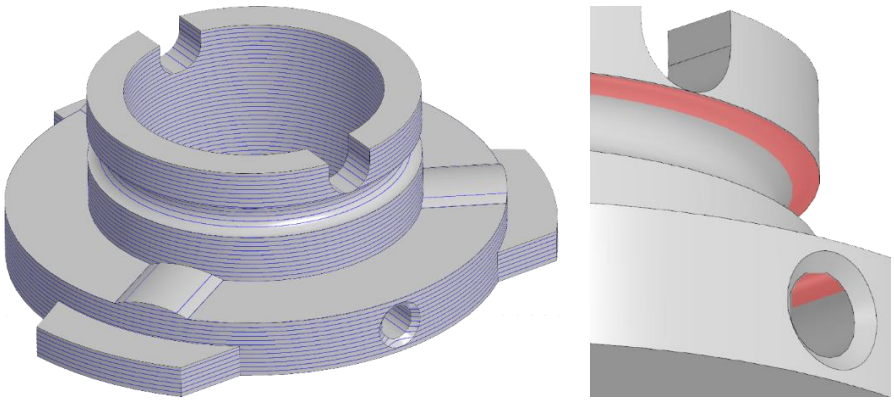


Рисунок 1 – Пример подсветки областей модели

Печать ведется по слоям, высоту слоев можно регулировать и от нее зависит качество полученного изделия. Синим цветом подсвечиваются границы слоев. Благодаря этому, можно сразу увидеть элементы, которые не получится качественно напечатать при заданной высоте слоя. Тогда инженер может перепроектировать эти элементы или изменить высоту слоя.

Слои при печати должны на что-то опираться, печать не может вестись по воздуху. Поэтому при моделировании нужно избегать нависающих элементов, такие элементы выделены красным цветом. Элемент подсвечивается, если угол между нормалью и вертикальной осью превышает угол, заданный пользователем.

Построить оптимизации вручную позволяет любой современный САПР. Но существующих решений, которые бы позволили автоматизировать этот процесс, мной также найдено не было. На рисунке 2 представлен пример построенной оптимизации. Измененная геометрия подсвечена зеленым цветом.

Проведенные в этом примере оптимизации позволяют уменьшить количество нависающих элементов. В горизонтальных отверстиях создаются уклоны, угол которых равен

максимальному, который может обеспечить принтер. Геометрия вертикального отверстия для гайки изменяется, чтобы все нависающие элементы могли напечататься «мостами» [3] (начало и конец каждой линии опираются на уже готовую часть).

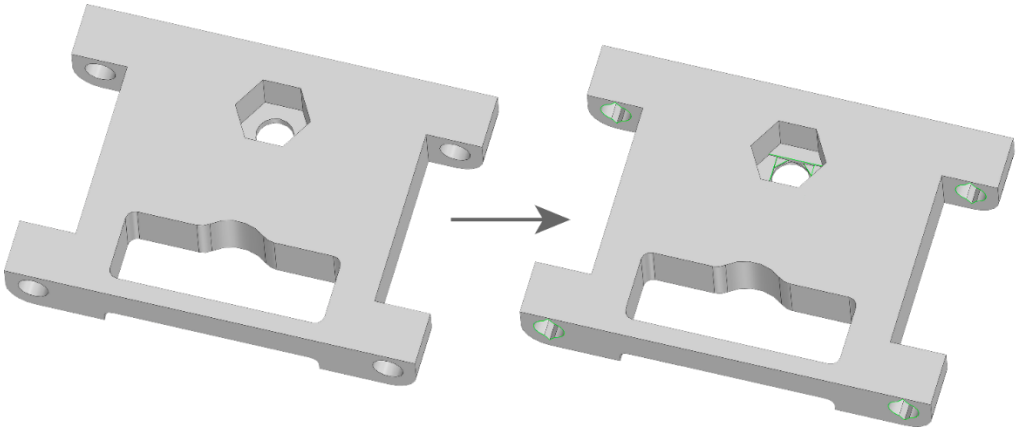


Рисунок 2 – Пример построенной оптимизации

Приложение является плагином-библиотекой для САПР КОМПАС-3D [4]. КОМПАС-3D – это российская импортонезависимая система трехмерного проектирования, которая широко используется для проектирования изделий в различных отраслях производства.

КОМПАС-3D предоставляет API, с использованием которого и разрабатывается приложение.

ЛИТЕРАТУРА

1. What is FFF 3D printing? An introduction to 'fused filament fabrication' technology – URL: https://ultimaker.com/learn/what-is-fff-3d-printing/ (дата обращения: 10.03.2024)
2. Система автоматизированного проектирования // Большая российская энциклопедия – URL: https://bigenc.ru/c/sistema-avtomatizirovannogo-proektirovaniia-82d30a (дата обращения: 10.03.2024)
3. Bridging in 3D Printing: Everything You Need to Know – URL: https://www.selfcad.com/blog/bridging-in-3d-printing (дата обращения: 10.03.2024)
4. Система трехмерного моделирования КОМПАС-3D: обзор программы – URL: https://ascon.ru/products/kompas-3d/ (дата обращения: 10.03.2024)
5. Голованов, Н. Н. Геометрическое моделирование : методическое пособие / Н. Н. Голованов. - Москва : ДМК Пресс, 2020. - 406 с. - ISBN 978-5-97060-806-7.

УДК 004.453

Е. С. Зайцев (4 курс бакалавриата),
Т. В. Леонтьева, к.т.н., доцент

ПРОГРАММНО-АППАРАТНЫЙ КОМПЛЕКС ДЛЯ СОРТИРОВКИ ЯНТАРЯ

На сегодняшний день идет четвертая промышленная революция, однако некоторые отрасли заметно отстают, реализуя подход второй промышленной революции, где все еще много ручного труда. Так складывается по ряду причин, в числе которых отсутствие автоматизированных решений или их высокая стоимость, делающая переход нерентабельным.

Процесс сортировки автоматизирован во многих отраслях и имеет разные подходы. Например, на алмазодобывающих предприятиях используются оптические методы и машинное зрение [1], в то время как на янтарном комбинате добытое сырье сортируется вручную по форме и цвету. Ежедневная работа по сортировке янтарных камней размером менее 7 мм занимает много времени и усилий, также, как и работа по сортировке янтаря «на 10 оттенков желтого», в отличии от сортировочной линии, в которой признаки сортировки могут оставаться независимыми от времени. Более точная сортировка может повысить качество выходных изделий и сократить время на их сборку. В данной работе описывается

процесс разработки программно-аппаратного комплекса для более эффективной сортировки больших объемов янтаря на предприятии.

Для классификации янтаря используется камера и машинное зрение [2]. В качестве основных признаков взяты форма и цвет. Для подачи собрана конвейерная линия и специальный механизм, подающий по одному камню на линию.

С помощью широких возможностей машинного обучения можно разбить камни на десятки различных типов, однако, как оказалось, бизнесу в этой отрасли достаточно всего 10 оттенков.

За все время поиска так и не удалось найти аппарат для сортировки янтаря, который можно приобрести. В БФУ им. И. Канта разрабатывают опытный образец [3] для янтарного комбината, но новостей о поступлении его в эксплуатацию до сих пор не было. Известно о промышленных сортировочных станках, которые производили в Литовской республике. Их поставляли в Россию по цене от 7000$, но после 2022 года компания прекратила их поставки и отказалась от обслуживания станков на территории России.

В связи с отсутствием аналогов было решено разработать программно-аппаратный комплекс для сортировки янтаря, используя общедоступные комплектующие и программные средства. В качестве управляющей платы был взят микрокомпьютер Raspberry pi 3 [4], являющийся бюджетным и проверенным устройством, и его мощностей достаточно для поставленной задачи. Для управления механикой используется плата на микроконтроллере Atmega 328.

Принцип действия комплекса простой: в бак загружается янтарь, затем через специальный узел камни по одному выпадают на конвейер, во время движения по которому, их положение фиксирует датчик и вызывает срабатывание затвора камеры. Снимок попадает на компьютер для дальнейшей обработки изображения. Первично с использованием библиотеки машинного зрения Open CV [5] кадр обрезается. К изображению применяются несколько фильтров и геометрических замеров, чтобы определить, что на картинку попал только один камень без каких-либо посторонних элементов, часто встречающихся с янтарем. Затем кадр подается в нейросеть, созданную с использованием Tensorflow [6], на выходе из которой получаем номер типа камня от 0 до 7. Камень продолжает двигаться по ленте и, достигнув номера нужного контейнера, попадает туда.

К наиболее очевидным проблемам, с которыми предстоит встретиться в процессе увеличения производительности комплекса, относится настройка работы камеры, поскольку с увеличением скорости подачи янтаря по конвейеру, камера должна быть способна создать четкий снимок в необходимый интервал времени. Важно понимать, что лента непрерывно движется, а камни должны располагаться на некотором расстоянии друг от друга, чтобы уменьшить вероятность случаев, когда в кадр попадает несколько камней и необходимо пропустить их. Таким образом, ограничение в производительности конвейера на данном этапе возникает из-за долгой выдержки камеры, на которую можно повлиять только через замену камеры на более дорогую.

В течение 6 месяцев планируется завершить первый станок и ознакомиться с патентными ограничениями и преимуществами. До конца 2024 года необходимо найти компанию готовую на взаимовыгодных условиях установить станок на своем предприятии для проведения испытаний.

ЛИТЕРАТУРА

1. Классификация алмазов [Электронный ресурс] Режим доступа: https://www.mallenom.ru/vnedrenia/pmz/diamondindustry1/
2. Что такое машинное зрение? [Электронный ресурс] Режим доступа: https://aws.amazon.com/ru/what-is/computer-vision/
3. Разработка оборудования для сортировки янтаря [Электронный ресурс] Режим доступа: https://vesti-kaliningrad.ru/sortirovat-yantar-budet-iskusstvennyj-intellekt-molodye-uchyonye-iz-kaliningrada-predstavili-svoyo-izobretenie/

4. Raspberry pi 3 [Электронный ресурс] Режим доступа: https://www.raspberrypi.com/products/raspberry-pi-3-model-b/
6. Библиотека Open CV [Электронный ресурс] Режим доступа: https://opencv.org/
7. Библиотека TensorFlow [Электронный ресурс] Режим доступа: https://www.tensorflow.org/?hl=ru

УДК 004.41

Г. Д. Голиков (2 курс магистратуры),
П. Д. Дробинцев, к.т.н., доцент

ВНЕДРЕНИЕ ЭЛЕКТРОННЫХ ТРАНСПОРТНЫХ НАКЛАДНЫХ В МОДУЛЬ ЭДО СИСТЕМЫ АВТОМАТИЗАЦИИ СНАБЖЕНИЯ С ПОМОЩЬЮ ИНТЕГРАЦИИ СО СТОРОННИМ СЕРВИСОМ

Статья посвящена вопросу внедрения электронных транспортных накладных (ЭТрН) в модуль системы электронного документооборота (ЭДО) для автоматизации процессов снабжения. Рассматривается возможность интеграции со сторонним сервисом для оптимизации учета и контроля транспортных процессов. Проанализированы основные проблемы, связанные с традиционными методами учета, и предложены пути их решения.

Современные требования к управлению логистическими процессами в снабжении предприятий подчеркивают важность внедрения эффективных инструментов для учета и контроля транспортных процессов [1]. Традиционные методы учета транспортных накладных часто связаны с бумажными документами, что приводит к задержкам, ошибкам и сложностям в контроле. Ручной ввод данных и их обработка являются источником недочетов и потерь в эффективности логистических процессов. Этот подход также ограничивает возможности мониторинга в режиме реального времени и усложняет взаимодействие с поставщиками и перевозчиками.

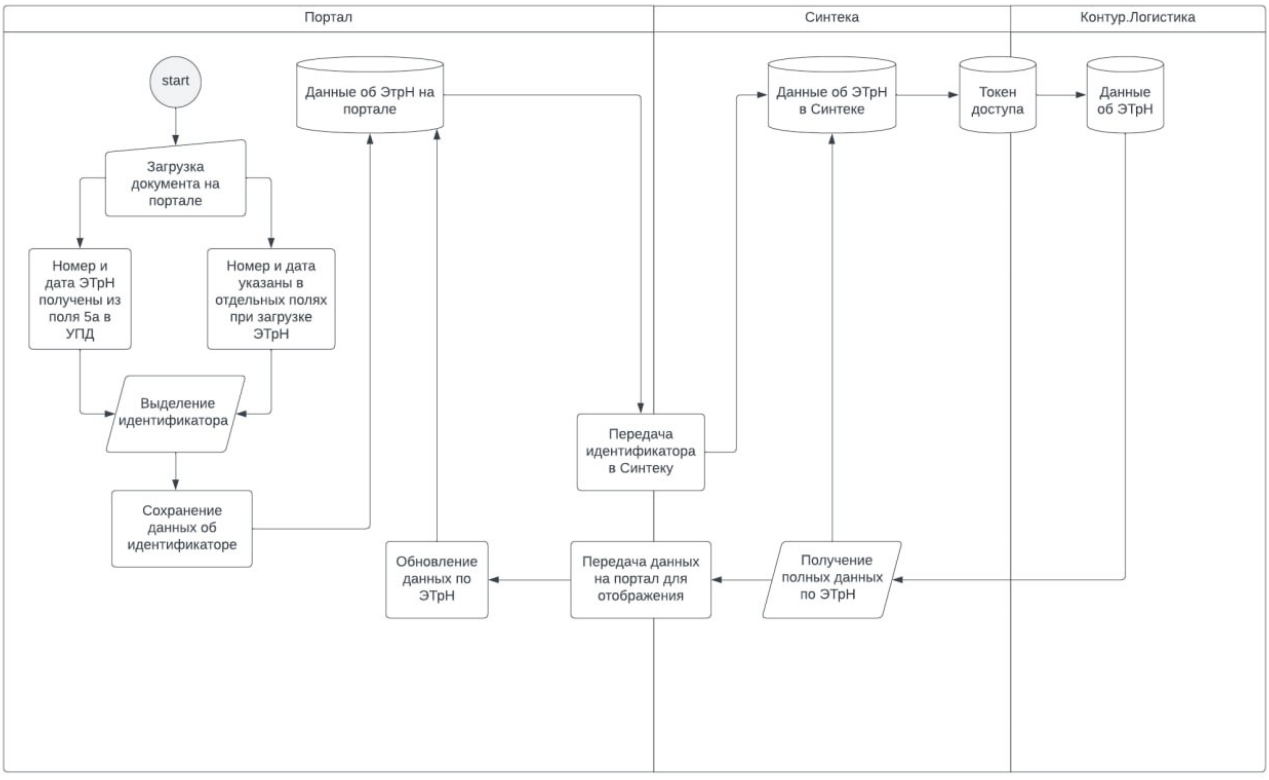


Рисунок 1 – Макет интеграции

Далее представлен список необходимых к выполнению задач, определяющих демоверсию разрабатываемого сервиса:

- Выбор способа интеграции с провайдером транспортных накладных [2].

Определение наилучшего способа интеграции с провайдером транспортных накладных является ключевым этапом. Задачи включают анализ API провайдера (проведение исследования API провайдера для определения его функциональных возможностей и ограничений) и выбор протокола обмена данными (определение наиболее подходящего протокола для обмена данными с провайдером, учитывая безопасность и эффективность передачи) [3].

- Декомпозиция таблиц реляционных баз данных для поддержания нового типа документа – электронных транспортных накладных.

Расширение базы данных для поддержки электронных транспортных накладных включает в себя следующие шаги: анализ текущей структуры базы данных [4] (изучение существующей базы данных для определения необходимых изменений), проектирование новых таблиц (создание структуры таблиц, учитывающей особенности хранения данных о электронных транспортных накладных) и определение связей (установление связей между новыми таблицами и уже существующими для обеспечения целостности данных).

- Реализация получения и обработки данных от провайдера накладных.

Разработка механизмов для эффективного получения и обработки данных от провайдера включает в себя следующие задачи: настройка точек подключения (определение точек подключения к API провайдера и настройка соединения) и разработка механизмов синхронизации [5] (создание механизмов, обеспечивающих регулярную синхронизацию данных электронных транспортных накладных с базой данных).

- Настройка передачи накладных на торговые площадки поставщикам.

Для эффективной передачи накладных на торговые площадки поставщикам необходимо выполнить следующие шаги: интеграция с торговыми площадками (определение форматов данных и механизмов передачи, соответствующих требованиям торговых площадок) и настройка автоматической передачи [5] (разработка механизмов автоматической передачи электронных транспортных накладных на торговые площадки с учетом расписания и обновлений).

- Индексирование таблиц, мониторинг и управление нагрузкой на сервер.

Для решения описанных задач используется микросервисная архитектура, языки программирования Java с фреймворком PlayFramework, база данных PostgreSQL для хранения информации о накладных.

Таким образом, разработка и внедрение новой системы для учета и контроля электронных транспортных накладных в системе автоматизации снабжения позволят обеспечить эффективную работу, оперативное взаимодействие с провайдерами и торговыми площадками, а также максимальную производительность базы данных.

ЛИТЕРАТУРА

1. Smith, J. (2019). "Digital Transformation in Supply Chain Management." Journal of Logistics Technology, 24(2), 45-60.
2. White, P. et al. (2021). "Integration of Electronic Transport Documents with Supply Chain Systems." Proceedings of the International Conference on Logistics and Transportation, 78-92.
3. Smith J., Johnson A. (2016). "Integration Patterns: Best Practices and Strategies for SOA, SaaS, and APIs". O'Reilly Media. 320 p.
4. Гусаров М. (2019). "Разработка и интеграция RESTful веб-сервисов". СПб.: БХВ-Петербург. 224 с.
5. Романов П. (2018). "Микросервисы: паттерны разработки и интеграции". М.: ДМК Пресс. 208 с.

УДК 004.654

В. В. Дмитриева (2 курс магистратуры),
П. Д. Дробинцев., к.т.н., доцент

ПРИМЕНЕНИЕ СОБЫТИЙНОГО ПОДХОДА В МИКРОСЕРВИСНОЙ АРХИТЕКТУРЕ ПРИ РАЗРАБОТКЕ ЗАЩИЩЕННОЙ СИСТЕМЫ УПРАВЛЕНИЯ ДОКУМЕНТАМИ

В ПАО “Сбербанк” существует бизнес-процесс по обмену моделями планирования корпоративно-инвестиционного бизнеса, которые находятся в файлах и должны передаваться между сотрудниками и подразделениями. По требованию заказчика необходимо обеспечить централизованную передачу этих моделей, отказавшись от использования почты и учитывая потенциально высокий размер файлов и требования безопасности. Дополнительно было выдвинуто требование о том, что реализация должна учитывать возможность простого подключения большого количества сторонних сервисов к данному решению, чтобы упростить документооборот на уровне всех потребителей сервисов планирования.

На основании имеющегося описания бизнес-процесса, выдвинутых требований и уже существующих внутри банка систем, в качестве основы будущего решения была выбрана событийно-ориентированная архитектура [1]. Событийно-ориентированный подход к разработке микросервисов предполагает проектирование и реализацию системы, в которой компоненты (микросервисы) общаются и взаимодействуют друг с другом посредством событий. В этой парадигме сервисы производят и потребляют события для достижения слабосвязанной и масштабируемой архитектуры. События - это сообщения, которые представляют собой изменения в состоянии или значительные события в системе.

Таким образом, *целью* данной работы является создание централизованной защищенной системы управления документами для корпоративно-инвестиционного бизнеса путем применения событийного подхода в микросервисной архитектуре.

Для достижения этой цели необходимо решение следующих *задач*:

1. Обзор внутренних и сторонних инструментов асинхронного взаимодействия, хранения и защиты файлов, доступных на уровне банка.
2. Сравнительный анализ инструментов асинхронного взаимодействия.
3. Сравнительный анализ требований к интеграциям с внутренними сервисами.
4. Проектирование событийно-ориентированной системы микросервисов защищенной системы управления документами
5. Программная реализация микросервисов; настройка окружения и интеграций.
6. Ввод в эксплуатацию, сбор метрик от потребителей сервисов планирования.

В банке есть 2 специальные системы, подходящие для использования в нашем решении: единая система управления документами ECM, непосредственно выступающая хранилищем и система IRM, позволяющая защитить файлы от несанкционированного доступа.

Для общения с ними от имени сервисов планирования мы вводим дополнительный сервис Outpost. В основе Outpost лежит паттерн Outbox [2] с основной разницей в том, что мы выбрали очередь сообщений в kafka, а не хранение их в базе. Для асинхронного взаимодействия с сервисом ECM используем Kafka [3] для передачи состояния доставки/получения файлов. Для размещения используем S3 [4] хранилище и еще одно внутреннее решение File Storage Gateway System.

Сервис ECM не поддерживает хранение защищенных файлов, поскольку обязательным требованием для его использования является сканирование безопасности файлов. Так как по требованиям бизнеса, при выгрузке файл должен быть защищен, мы обращаемся к системе IRM, файл отправляется на защиту, выгружается обратно и передается пользователю. IRM работает на базе Active Directory на основании ролевых моделей.

Дополнительно мы предоставляем пользователю собственный сервис Управления Документами, где он может напрямую загружать/выгружать файлы, привязывать их к другим доступным сервисам.

Схема взаимодействия при загрузке файлов представлена на рисунке 1.

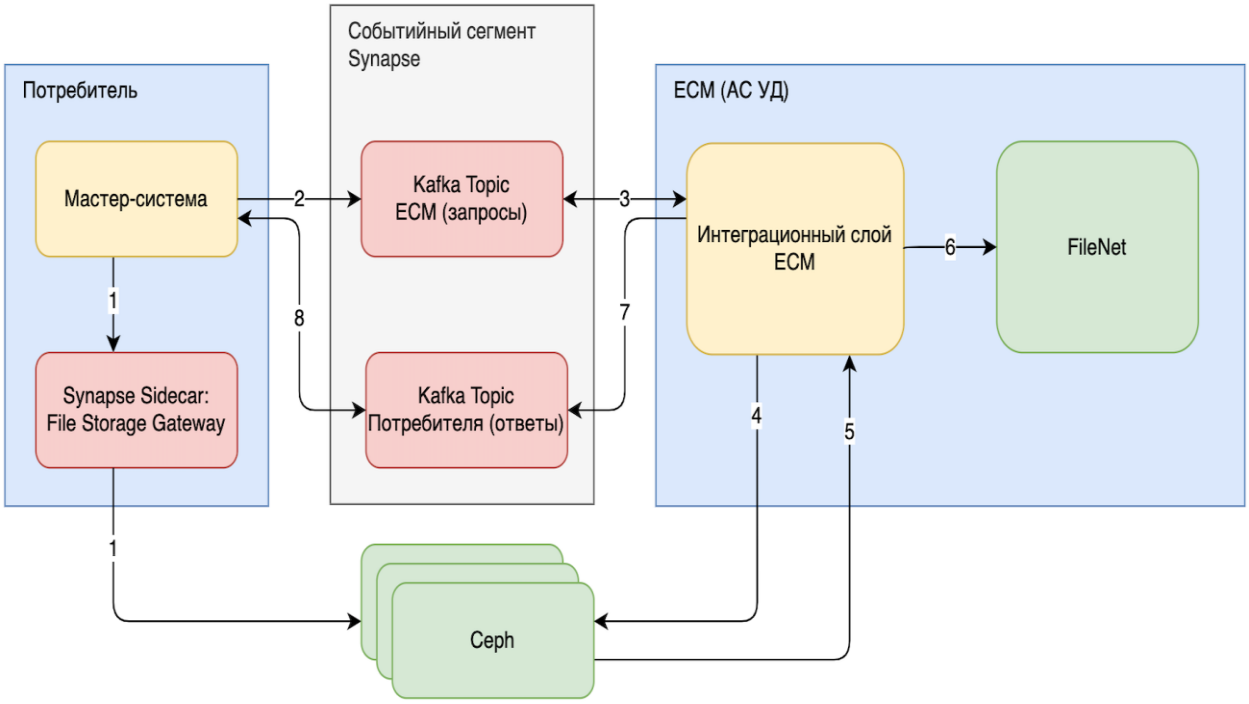


Рисунок 1 – Схема загрузки файла

Основным преимуществами организованного взаимодействия является то, что:

1. Функционал не описан структурой классов Java, а описан YAML файлами, которые потом на этапе компиляции трансформируются в Data Transfer Objects [5].
2. Для добавления каждого нового сервиса просто заводим типизированные топики Kafka и сертификат на доступ от имени сервиса через SSL.
3. Доступ между Outpost и нашим Управлением Документами работает на собственной Kafka без дополнительных ограничений.
4. Возможно реализовать подписание файлов ЭЦП и добавить другие дополнительные уровни защиты.

Спроектированная система покрыла все требования бизнеса и была успешно введена в эксплуатацию. На данный момент с ней взаимодействуют уже три сервиса-потребителя, существенно сокращая трудозатраты команд на документооборот в период подготовки бизнес-планирования.

ЛИТЕРАТУРА

1. Стопфорд Б. Проектирование событийно-ориентированных систем: Концепции и шаблоны проектирования сервисов потоковой обработки данных с использованием Apache Kafka: пер. с англ. 2-е изд., испр. – 2019. – С. 33-49.
2. Паттерн Outbox: как не растерять сообщения в микросервисной архитектуре. [Электронный ресурс] Режим доступа: https://habr. com/ru/companies/lamoda/articles/678932.
3. Thein K. M. M. Apache kafka: Next generation distributed messaging system //International Journal of Scientific Engineering and Technology Research. – 2014. – Т. 3. – №. 47. – С. 9478-9483.
4. Amazon S3. [Электронный ресурс] Режим доступа: https://aws.amazon.com/ru/s3/
5. Data Transfer Objects (DTOs). [Электронный ресурс] Режим доступа: https://docs.oracle.com/en/storage/tape-storage/sl4000/slkyp/data-transfer-objects-dtos.html

УДК 004.42

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

РАЗРАБОТКА ПЛАТФОРМЫ ДЛЯ АНАЛИЗА ТЕКСТОВЫХ ДАННЫХ В СЕТИ ИНТЕРНЕТ

Большая часть данных в современном мире представлена в виде текстовых документов: архивы научных статей, публикации на профильных сайтах, социальные сети. Многие специалисты – исследователи и научные сотрудники – занимаются анализом данных, который невозможно осуществлять вручную из-за их объёма. Существует множество инструментов для автоматизации этого процесса, но популярной платформы с открытым исходным кодом, которая объединяет разные инструменты в единую систему не существует [1,2,3].

В данной работе поставлена задача реализации подобной платформы, отвечающей следующим требованиям:

- Расширяемость и открытость кода.
- Визуальный язык программирования для настройки конвейера по обработке данных.
- Возможность визуализации, агрегации, фильтрации данных.
- Возможность совместной работы (поддержка пользователей, команд, проектов).

На рисунке 1 представлено схематичное описание возможного пользовательского опыта при работе с платформой.

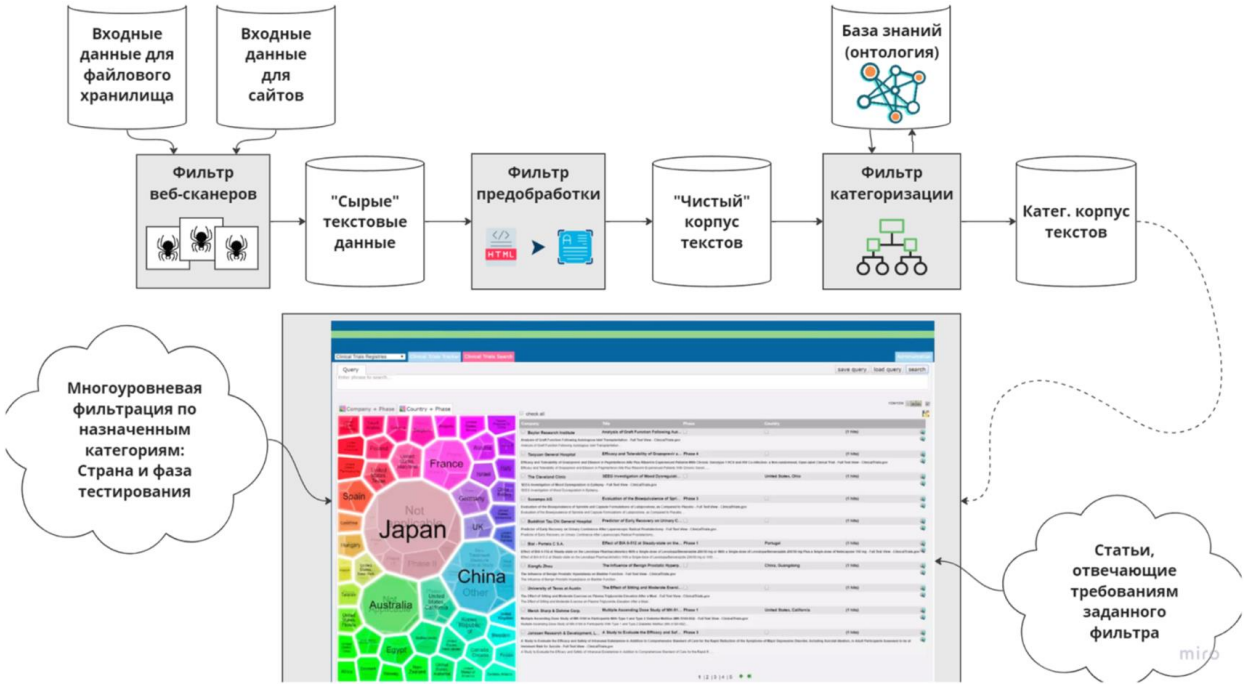


Рисунок 1 –Схема пользовательского опыта

В верхней части изображения представлен возможный вид визуального языка программирования для конфигурации по настройке конвейера. Язык состоит из двух верхнеуровневых блоков: хранилище данных и фильтры.

Хранилище данных (ХД) – это абстракция, под которой могут быть реализованы самые разные структуры и форматы данных. Индексированные тексты в ElasticSearch [4], граф знаний в Blazegraph [5], структура ключ-значение в Redis [6], или же обычная таблица в SQL. Пользователь сам должен выбрать подходящую реализацию под данные, которые он хочет сохранить.

Фильтр – это абстракция, обозначающая некоторый обработчик, «трансформер» данных. Фильтр может скрывать под собой реализацию самой разной функциональности – соединение данных по ключу, веб индексатор для извлечения текстов с интернет-ресурсов, предобработчик текстов для их нормализации, обучение машиной модели для классификации текстов.

Таким образом ХД и фильтры — это блоки, из которых пользователь может собрать свой собственный конвейер по обработке данных, соответствующий требованиям его проекта, а не следовать заранее разработанной схеме и методологии.

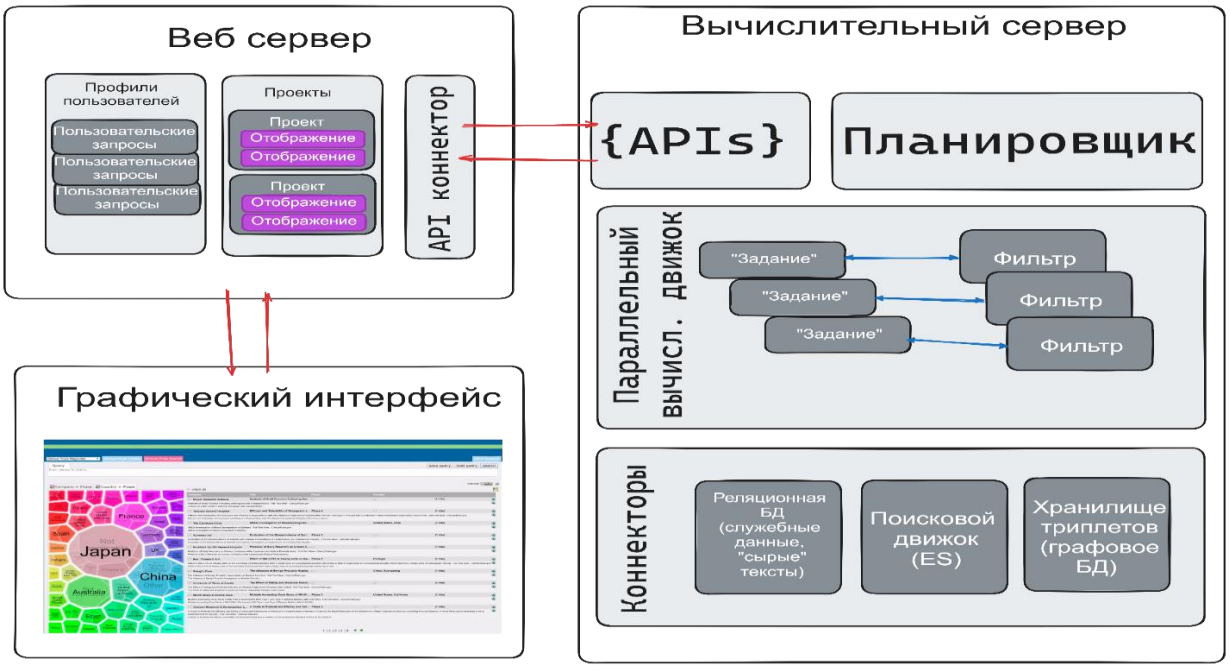


Рисунок 2 – Архитектура платформы

Вывод: в ходе работы были изучены основные подходы к анализу и обработке текстов, проанализированы современные инструменты, доступные специалистам. На основе анализа была разработана архитектура и описаны модули платформы для анализа текстовых данных.

ЛИТЕРАТУРА

1. Understanding the Use, Strengths and Limitations of Automated Text Analysis. URL: https://www.researchgate.net/publication/359324324_Understanding_the_Use_Strengths_and_Limitations_of_Automated_Text_Analysis
2. Что такое Томита-парсер, как Яндекс с его помощью понимает естественный язык, и как вы с его помощью сможете извлекать факты из текстов. URL: https://habr.com/ru/companies/yandex/articles/219311/
3. Real AI семантический анализатор. URL: https://habr.com/ru/articles/712140/
4. Elasticsearch platform. URL: https://www.elastic.co/platform
5. Blazegraph Database. URL: https://blazegraph.com/
6. Redis data structured store. URL: https://redis.io/

УДК 004.415.2.031.43

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

СЕРВЕРНАЯ ЧАСТЬ АВТОМАТИЗАЦИИ РАБОТЫ ТЕПЛИЦЫ

Современные технологии автоматизации и удаленного управления в сельском хозяйстве становятся все более популярными, особенно для частных фермеров с небольшими теплицами. Фермерам хотелось бы иметь систему, которая позволяет эффективно контролировать параметры и управлять ими в теплице с использованием датчиков измерения, и данный проект можно реализовать с помощью исполнительных устройств и протокола MQTT [1].

В реализации данной работы будет проведено проектирование и разработка серверной части веб-приложения. Приложение предназначено для обслуживания небольших теплиц с

несколькими грядками и обеспечивает возможность удаленного мониторинга и управления параметрами. Таким образом, частные фермеры могут эффективно контролировать условия выращивания растений в своих теплицах посредством удаленного доступа.

Цель работы включает в себя создание серверной части функционального веб-приложения, способного получать данные от датчиков через MQTT протокол и отправлять команды исполнительным устройствам для контроля условий в теплице.

Для разработки данной программы были поставлены следующие задачи:

– Подробное изучение реализации серверной части в веб-приложениях, использующих клиент-серверную архитектуру

– Реализация UML-диаграммы [2] и базы данных, благодаря которым программу можно легко масштабировать

– Настройка подключения к датчикам и устройствам управления

– Разработка сервиса, который собирает данные, принимает решения на основе этих данных и передает команды управляющим устройствам

Система автоматизации и удаленного управления в теплице может быть настроена для автоматического управления параметрами в теплице на основе данных, собранных датчиками. Например, если температура в теплице становится слишком высокой, система может автоматически открыть или приоткрыть окошко, чтобы снизить температуру. Если влажность на конкретной грядке становится ниже установленных показателей, система может автоматически включить систему капельного полива, чтобы достичь требуемых показателей влажности. Также реализован контроль освещенности, благодаря которому в темное время суток или при облачной погоде будут включаться ультрафиолетовые лампы для улучшения качества роста растений. Контроль за освещенностью также осуществлен с помощью датчиков. Пользователи могут отслеживать текущие значения параметров, устанавливать желаемые значения и получать уведомления о состоянии теплицы в течение дня. Также пользователи могут включить ручной режим, чтобы самостоятельно управлять показателями теплицы.

Для надежности работы системы теплица использует несколько датчиков каждого вида - освещенности, увлажненности и температуры. Это решение было принято в связи с тем, что датчики имеют тенденцию со временем выходить из строя и передавать неверные данные. Также в программе предусмотрена система валидации, чтобы в принятии решений системы учитывались только данные с корректно работающих датчиков.

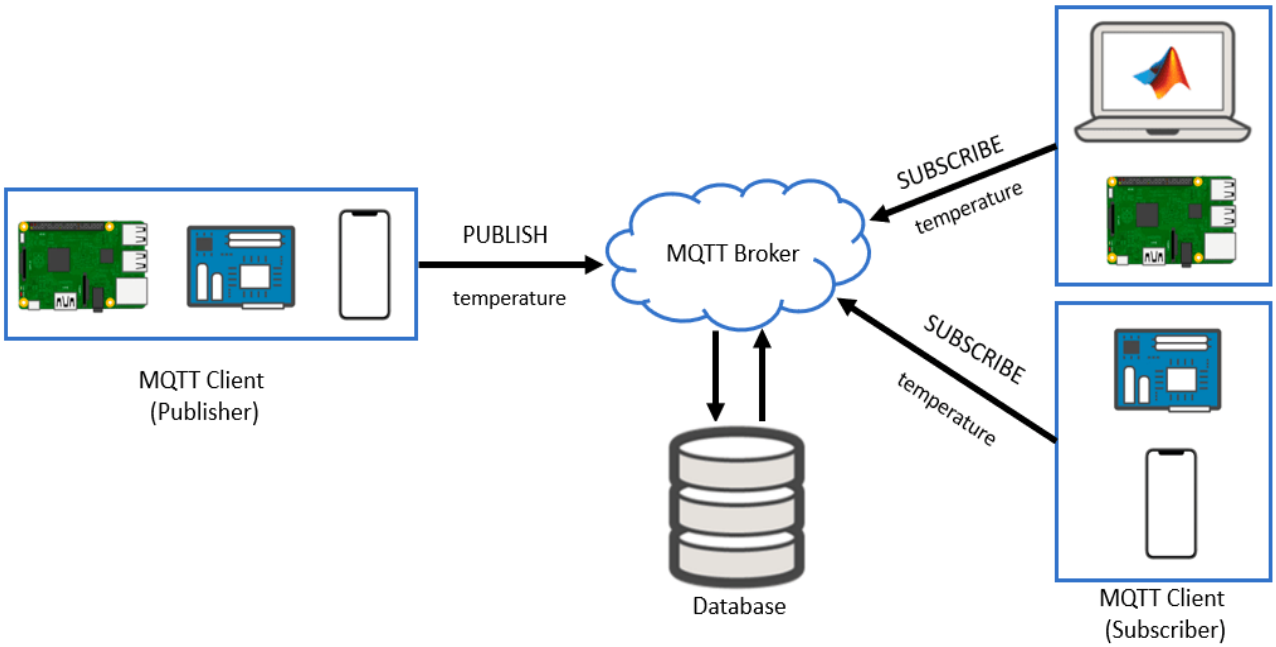


Рисунок 1 – Принцип работы MQTT протокола

Подход был реализован в программном средстве на языке Java [3] с использованием сборщика проектов Maven [4], фреймворка Spring Boot [5], базы данных PostgreSQL [6] и протокола MQTT. Реализация серверной части приложения позволяет конечному пользователю следить за состоянием теплицы и грядок в реальном времени.

ЛИТЕРАТУРА

1. MQTT - концептуальное погружение. [Электронный ресурс] Режим доступа: https://habr.com/ru/articles/463669/
2. UML (Undefined modeling language) class diagram. [Электронный ресурс] Режим доступа: https://www.geeksforgeeks.org/unified-modeling-language-uml-class-diagrams/
3. Java. [Электронный ресурс] Режим доступа: https://dev.java/
4. Maven. [Электронный ресурс] Режим доступа: https://maven.apache.org/
5. Spring boot. [Электронный ресурс] Режим доступа: https://spring.io/projects/spring-boot
6. PostgreSQL. [Электронный ресурс] Режим доступа: https://www.postgresql.org/

УДК 004.85

В. А. Каврин (2 курс магистратуры),
Г. Ф. Малыхина, д.т.н., профессор

ПОВЫШЕНИЕ КАЧЕСТВА УПРАВЛЕНИЯ ТРАНСПОРТНЫМ СРЕДСТВОМ С ИСПОЛЬЗОВАНИЕМ MPC

По статистике, которую составило ГИБДД [1] (Государственная Инспекция Безопасности Дорожного Движения), с наиболее частыми нарушениями, которые заканчиваются ДТП (Дорожно-Транспортное Происшествие) с пострадавшими или погибшими – первое место занимает несоблюдение дистанции. Действительно, человеку сложно предсказать поведение впереди идущей машины и даже, если это возможно, зачастую не хватает, банально, реакции, чтобы вовремя нажать на педаль тормоза, даже если водитель будет соблюдать указанные в ПДД (Правила Дорожного Движения) величины дистанции между двумя автомобилями.

В современном мире активно развивается область науки, которая занимается увеличением безопасности вождения транспортных средств. Одно из направлений – это предсказание дорожной обстановки с использованием MPC (Model Predictive Control)

MPC – это алгоритм управления с обратной связью, который использует модель для прогнозирования будущих результатов процесса [2]. Данный алгоритм, является одним из самых передовых технологий в области беспилотного управления транспортным средством.

Одним из ключевых преимуществ использования MPC является возможность адаптации автомобиля к изменяющимся условиям на дороге. Например, если автомобиль движется по скользкой дороге, MPC может автоматически уменьшить скорость и применить тормоза, чтобы предотвратить возможное скольжение.

Также использование MPC помогает улучшить общую безопасность вождения, поскольку она помогает автомобилю избегать столкновений и других опасных ситуаций на дороге.

Существуют и ограничения использования MPC в автомобильной индустрии. Одним из них является необходимость большого количества данных для обучения модели. Это может быть проблемой для новых моделей автомобилей, которые еще не имеют большого объема данных о поведении на дороге.

Кроме того, MPC требует высокой вычислительной мощности, что может быть проблемой для некоторых автомобилей с ограниченными ресурсами.

Таким образом, целью данной работы является оптимизация процессов управления транспортным средством с применением методов MPC, а именно процесс плавного торможения и ускорения на динамической дороге.

Для достижения этой цели необходимо решить следующие задачи:

1. Обзор существующих систем для оптимизации процессов управления транспортным средством.
2. Проведение сравнительного анализа найденных подходов.
3. Программирование контроллера с использованием Matlab.
4. Симуляция поведения транспортного средства с использованием Simulink.
5. Анализ полученных результатов.
6. Демонстрация системы оптимизации процесса управления.

Модель прогнозирующего поведения использует модель устройства, в нашем случае автомобиль (рисунок 1) для составления прогнозов о будущем его поведении. Она также использует оптимизатор, который гарантирует, что прогнозируемое поведение соответствует ожиданиям[3].

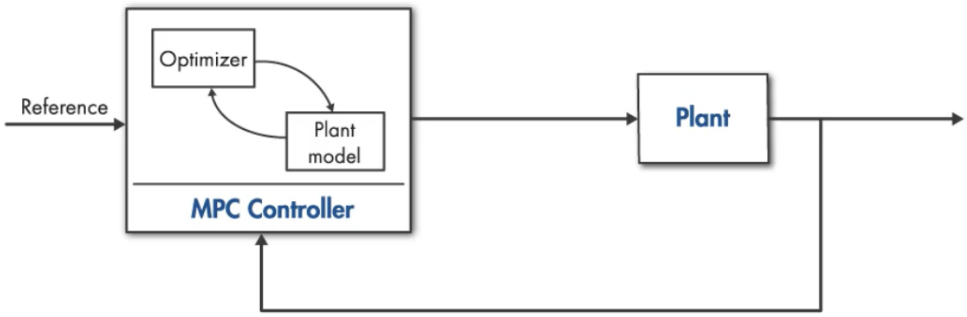


Рисунок 1 – Упрощенная схема работы прогнозирующей модели

При проектировании контроллера MPC важно правильно подобрать входные параметры, так как они влияют не только на производительность контроллера, но и на вычислительную сложность алгоритма MPC. Основные параметры включают в себя: время выборки, горизонты прогнозирования и управления, ограничения, веса [4].

После завершения этапа проектировании контроллера MPC, следует построение автономной системы с использованием инструмента Simulink.

Основными входными/выходными данными для объекта (plant) будут являться расстояние, скорость.

Выходные данные объекта подаются на вход контроллера. Далее с помощью Automated Driving System Toolbox создается пользовательское эталонное поведение автомобиля, после экспорта которого создается специальный блок, который подключен к контроллеру (reference).

Завершающим этапом является моделирование, после чего можно увидеть результаты и провести анализ на удовлетворение их – требованиям системы.

Используя результаты распознавания дорожных знаков с помощью сверточной нейронной сети YOLOv5 (You Only Look Once version 5) [5] система в дальнейшем рассматривается как предсказание поведения автомобиля на дороге, то есть медленное снижение или повышение скорости, исходя из вида распознанного дорожного знака или светофора.

Для решения такой задачи целесообразно применять LSTM (Long short-term memory) нейронные сети с предсказанием.

ЛИТЕРАТУРА

1. Рейтинг наиболее частых нарушений, которые заканчиваются ДТП [Электронный ресурс] Режим доступа: https://rg.ru/2022/04/11/reg-cfo/gibdd-sostavila-rejting-naibolee-chastyh-narushenij-kotorye-zakanchivaiutsia-dtp.html

2. Understanding Model Predictive Control, part 1 [Электронный ресурс] https://www.mathworks.com/support/search.html/videos/understanding-model-predictive-control-part-1-why-use-mpc--1526484715269.html?fq%5B%5D=asset_type_name:video&fq%5B%5D=category:mpc/index&page=1
3. Transportation Research Part C: Emerging Technologies. Congestion-mitigating MPC design for adaptive cruise control based on Newell's car following model: History outperforms prediction. Hao Zhou, Annie Zhou, Tienan Li, Danjue Chen, Srinivas Peeta, Jorge Laval, 9/2022 (сентябрь 2022), том 142 https://www.sciencedirect.com/science/article/abs/pii/S0968090X22002285
4. Understanding Model Predictive Control, part 2 [Электронный ресурс] https://www.mathworks.com/videos/understanding-model-predictive-control-part-2-what-is-mpc--1528106359076.html
5. Поиск и распознавание дорожных знаков с помощью нейронных сетей [электронный ресурс] https://elib.spbstu.ru/dl/3/2022/vr/vr22-2700.pdf/info

УДК 004.41

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

РАЗРАБОТКА ANDROID-ПРИЛОЖЕНИЯ ДЛЯ ТЕХНИЧЕСКОЙ ПОМОЩИ НА ДОРОГЕ

В современном мире автомобиль стал неотъемлемой частью жизни большинства людей. С ростом числа автомобилей увеличивается и вероятность возникновения дорожно-транспортных происшествий, технических неисправностей и других ситуаций, требующих экстренной помощи на дороге. Это создает потребность в надежных и эффективных способах получения помощи для водителей, что подчеркивает актуальность разработки приложения.

Целью проекта является создание мобильного приложения для операционной системы Android [1,2], которое позволит водителям быстро получать техническую помощь на дороге. Задачи включают в себя разработку удобного пользовательского интерфейса, интеграцию с современными технологиями для определения местоположения, а также обеспечение надежной работы приложения.

На основе анализа аналогичных решений были сформулированы преимущества разрабатываемого приложения:

- интуитивно понятный интерфейс приложения позволяет водителям с минимальными усилиями получить необходимую помощь [3];

- использование GPS для точного определения местоположения ускоряет процесс оказания помощи;

- высокая производительность приложения;

- приложение не только соединяет водителей с профессиональными поставщиками услуг, но и позволяет формировать сообщество взаимопомощи между водителями;

- современные технологии разработки.

Для реализации приложения планируется использовать язык программирования Kotlin [4]. Это современный язык, который обеспечивает простоту и безопасность кода; полностью совместим с Java, что позволяет легко интегрировать Kotlin в существующие проекты Android; обладает более совершенными функциями для обработки ошибок и улучшенным синтаксисом по сравнению с Java.

Для бэкенда и базы данных будет использоваться Firebase [5], предоставляющий широкий спектр сервисов, включая облачное хранение данных, аутентификацию пользователей, аналитику, работу в режиме реального времени, что идеально подходит для приложений, требующих мгновенного обновления данных (например, отображения запросов на помощь на карте). Firebase предлагает интуитивно понятные инструменты и хорошо документированное API, что упрощает процесс разработки и сокращает время выхода

продукта на рынок.

Интеграция карты реализуется Google Maps API [6] - надежным сервисом картографии, что критически важно для приложений, требующих точного отображения местоположения; API предоставляет множество функций, таких как маршрутизация, геолокация и настройка внешнего вида карты, что позволяет создать богатый и интерактивный пользовательский интерфейс.

Архитектура приложения будет основана на паттерне MVVM (Model-View-ViewModel), который заключается в отделении бизнес-логики от пользовательского интерфейса и позволяет разрабатывать более структурированный и управляемый код, что упрощает тестирование и поддержку приложения [7].

Среди других применяемых технологий можно выделить Room для управления базой данных, Retrofit для сетевых запросов и Coroutines для асинхронной обработки.

Выбранные технологии предоставляют надежные решения для создания интуитивно понятного и функционального пользовательского интерфейса, эффективного управления данными и обеспечения высокой производительности приложения.

ЛИТЕРАТУРА

1. Логинов, А. В. Разработка веб-приложения сопровождения процесса отчетности сотрудников / А. В. Логинов, А. И. Тышкевич // 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. 42-44. – EDN DFYLXR.
2. Шушарин, А. В. Клиентское приложение на базе Android для литературного портала / А. В. Шушарин, А. И. Тышкевич // Современные технологии в теории и практике программирования: Сборник материалов научно-практической конференции студентов, аспирантов и молодых ученых, Санкт-Петербург, 26–27 апреля 2023 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2023. – С. 187-188. – EDN MEGIAA.
3. Зеленова, Д. Д. Автоматизация тестирования пользовательского интерфейса web-приложения / Д. Д. Зеленова, Н. В. Воинов, Т. В. Леонтьева // Современные технологии в теории и практике программирования: сборник материалов конференции, Санкт-Петербург, 19 апреля 2019 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2019. – С. 104-106. – EDN BVYPKM.
4. Официальная документация языка программирования Kotlin. [Электронный ресурс] Режим доступа: https://kotlinlang.ru/
5. Руководства и материалы Firebase. [Электронный ресурс] Режим доступа: https://firebase.google.com/docs?hl=ru
6. Документация Google Maps API. [Электронный ресурс] Режим доступа: https://developers.google.com/maps/documentation/android-sdk?hl=ru
7. Томилин, И. С. Развертывание микросервисной архитектуры с использованием consul, vault, nomad / И. С. Томилин, С. А. Федоров, В. Э. Шмаков // Современные технологии в теории и практике программирования: Сборник материалов научно-практической конференции студентов, аспирантов и молодых ученых, Санкт-Петербург, 26–27 апреля 2023 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2023. – С. 255-256. – EDN FUYZVW.

УДК 004.42

И. С. Князев (4 курс бакалавриата),
Е. С. Орлов, ст. преподаватель

РАЗРАБОТКА WEB 3.0 ОБРАЗОВАТЕЛЬНОЙ ПЛАТФОРМЫ С ИСПОЛЬЗОВАНИЕМ ТЕХНОЛОГИИ БЛОКЧЕЙН И IPFS

В настоящее время существует множество различных образовательных платформ: Skillbox, Stepik и другие. Каждая из них имеет свои плюсы и минусы как для авторов курса, так и для учащихся. Мне бы хотелось выделить общие проблемы.

Во-первых, централизация. Вы в любой момент можете потерять доступ к материалам курсов как из-за технического сбоя, так и из-за блокировки аккаунта. В то время как децентрализованные платформы предоставляют пользователям полный контроль над своими данными и хранение их на своих устройствах. Децентрализованные платформы также позволяют пользователям получать доступ к своим данным с любого устройства и в любое время, что делает их более доступными и удобными для использования.

Во-вторых, хранение дипломов и сертификатов. Сейчас информация об академических достижениях хранится в ФИС ФРДО. Эта система служит для обеспечения сбора и обработки сведений об академических достижениях[1].

В ФИС ФРДО вся информация о документах заполняется по шаблону, который предоставляется системой, а хранится в централизованной базе данных. Такой подход для хранения данных может повлечь за собой потерю информации при сбое оборудования, а также централизованные базы данных подвержены риску взлома с дальнейшей фальсификацией информации.

Для борьбы с этими проблемами как нельзя лучше подходит блокчейн в связке с IPFS. Использование этих технологий обеспечит прозрачность и надежность проверки подлинности документов, а также сохранность данных.

В рамках данного проекта была поставлена цель разработать WEB 3.0 образовательную платформу, в которой информация об академических достижениях будет храниться в IPFS, а связь сертификата/диплома с его владельцем будет закреплена в блокчейне при помощи смарт-контракта. Помимо этого, все учебные материалы в свою очередь также будут храниться в IPFS.

Для достижения поставленной цели необходимо решение следующих задач:

– реализация смарт-контрактов для сохранения информации об академических достижениях, а также информации о курсах;

– реализация API для взаимодействия с блокчейном и IPFS;

– реализация пользовательского интерфейса.

Архитектуру приложения можно представить в виде следующей диаграммы компонентов:

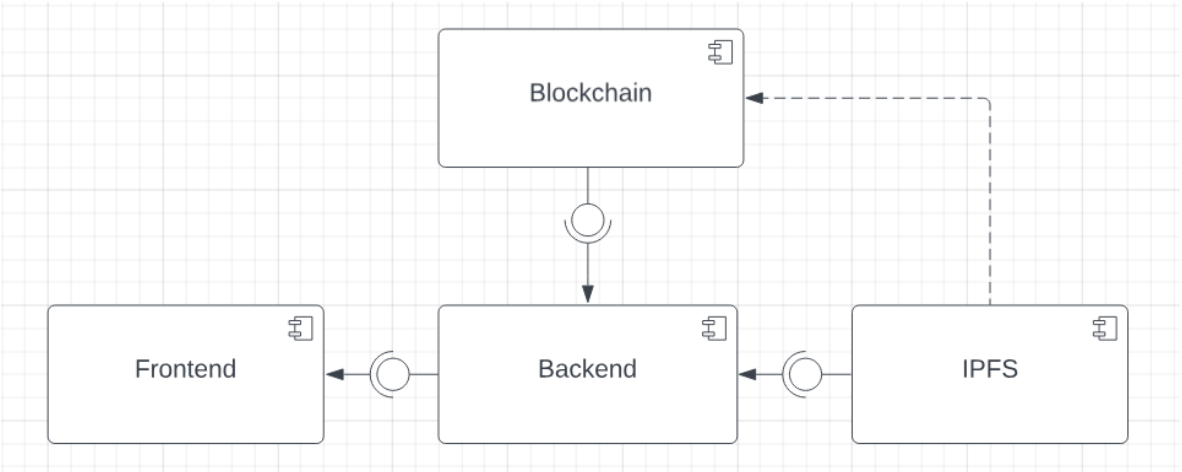


Рисунок 1 – Диаграмма компонентов.

Для реализации был выбран следующий стек технологий:

– TypeScript [2] (среда разработки: WebStorm 2023.2 [3]). Используется для написания веб-приложения;

– Solidity [4] (среда разработки: Remix [5]). Используется для написания смарт-контрактов;

– Frontend: React [6];

– Backend: NestJS [7];

– Блокчейн Arbitrum [8];

– IPFS [9].

В результате работы получилось создать веб-приложение со следующей основной функциональностью:

– возможность создания курса;

– возможность записаться на курс (как на платный, так и на бесплатный);

– получение сертификата после успешного прохождения курса;

ЛИТЕРАТУРА

1. Формирование и ведение Федерального реестра сведений о документах об образовании и (или) о квалификации, документах об обучении. [Электронный ресурс]. – URL: https://obrnadzor.gov.ru/gosudarstvennye-uslugi-i-funkczii/7701537808-gosfunction/formirovanie-i-vedenie-federalnogo-reestra-svedenij-o-dokumentah-ob-obrazovanii-i-ili-o-kvalifikaczii-dokumentah-ob-obuchenii/ (дата обращения: 12.03.2024).
2. TypeScript Documentation. [Электронный ресурс]. – URL: https://www.typescriptlang.org/docs/ (дата обращения: 12.03.2024).
3. WebStorm: IDE JetBrains для JavaScript и TypeScript. [Электронный ресурс]. – URL: https://www.jetbrains.com/webstorm/ (дата обращения: 12.03.2024).
4. Solidity - Solidity 0.8.23 documentation. [Электронный ресурс]. – URL: https://docs.soliditylang.org/ (дата обращения: 12.03.2024).
5. Remix - Ethereum IDE. [Электронный ресурс]. – URL: https://remix.ethereum.org/ (дата обращения: 12.03.2024).
6. React. [Электронный ресурс]. – URL: https://legacy.reactjs.org/ (дата обращения: 12.03.2024).
7. Documentation | NestJS - A node.js framework built on top of TypeScript. [Электронный ресурс]. – URL: https://docs.nestjs.com (дата обращения: 12.03.2024).
8. Arbitrum Docs. [Электронный ресурс]. – URL: https://docs.arbitrum.io/ (дата обращения: 12.03.2024).
9. IPFS: Docs. [Электронный ресурс]. – URL: https://docs.ipfs.tech (дата обращения: 04.03.2024).

УДК 004.422.833:004.652:004.514

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

ПЕРЕРАБОТКА И МОДЕРНИЗАЦИЯ СИСТЕМЫ КОММЕНТИРОВАНИЯ ЗАПИСЕЙ В СОЦИАЛЬНОЙ СЕТИ «ОДНОКЛАССНИКИ»

Одноклассники [1] – одна из ведущих социальных и контентных платформ РФ и ближнего зарубежья. Российская аудитория ОК – 43 000 000 чел. В сети представлены все поколения людей: дети, подростки, ядро аудитории 25-44 года и люди старшего возраста. ОК связывают разные возрастные и социальные группы и помогают выстраивать уникальные для российского интернета вертикальные связи между людьми.

В сутки пользователи сети оставляют больше 100 млн “классов” и комментариев к постам, из чего следует вывод, что этот функционал можно назвать ключевым [2]. Тем самым модификация и упрощение процесса общения и навигации по комментариям является одним из ключевых направлений развития социальной сети.

В работе была произведена переработка и модернизация раздела комментариев на всех платформах, устаревшая плоская структура была преобразована в древовидную, соответствующую современным стандартам во многих сетях, таких как YouTube, VK, Пикабу, Reddit, внешний вид изменений представлен на рисунке 1. Главным преимуществом системы можно выделить упрощение ориентирования в списке комментариев и поддержания диалога. Действительно, включение функционала привело к повышению количества написанных ответов на комментарии, а также количества комментариев в целом.

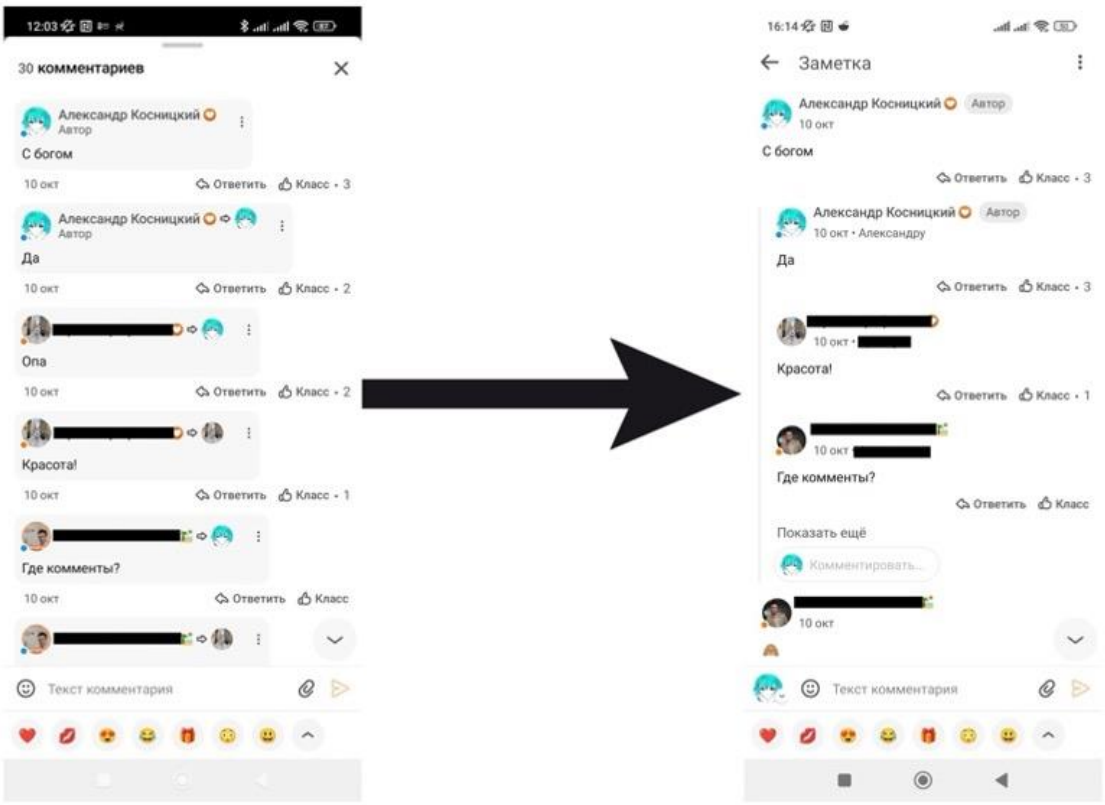


Рисунок 1 – Сравнение старой и новой структуры комментариев

Основной проблемой, возникшей во время выполнения данной задачи, был вопрос сохранения старых комментариев дискуссиях и их перевод в новый формат. В то же время поэтапное включение нового функционала на пользователей [3] означало необходимость поддержки отображения одной и той же дискуссии в обоих форматах одновременно.

Изначально была произведена попытка решить задачу через создание новой базы данных под древовидную структуру. Предполагалось, что новые данные будут записываться в обе б.д. одновременно, а также будет осуществляться копирование данных из старой б.д. в новую. После окончания копирования старая база данных будет плавно отключена, и оставлена только новая. К сожалению, реализация данного способа была затруднена рядом причин:

- Б.Д. комментариев одноклассников представляет собой NoSQL распределенное хранилище, реализованное на основе сильно модифицированной Cassandra [4] и расположенное в трех разных дата центрах по 32 сервера в каждом. Поддержка двух баз такого размера и их синхронизация, является довольно сложным процессом.
- Количество дискуссий на одном сервере в среднем превышает несколько сотен миллионов, причем в каждой дискуссии может быть более миллиона комментариев [5]. Перенос такого количества данных может занять очень долгое время.
- Данный подход подразумевает полный отказ от старой системы показа комментариев, как только старую базу данных отключат. К сожалению, все не так просто – есть пользователи, чьи мобильные устройства слишком устарели, чтобы поставить новую

версию клиента одноклассников, или же они просто не обновляют версию долгое время. Отключение старой базы данных может привести к тому, что все пользователи с устаревшими версиями не смогут просматривать комментарии вообще, что потенциально может стать большим ударом по порталу.

С учетом всего вышесказанного, был предложен новый подход, предполагающий разработку древовидного индекса, по которому сервер сможет получать указатели на нужные комментарии из базы данных с неизмененной структурой, тем самым представляя их в новом веточном виде. Система работы данного индекса проиллюстрирована на рисунке 2. Именно этот способ был выбран и успешно реализован в рамках поставленной задачи.

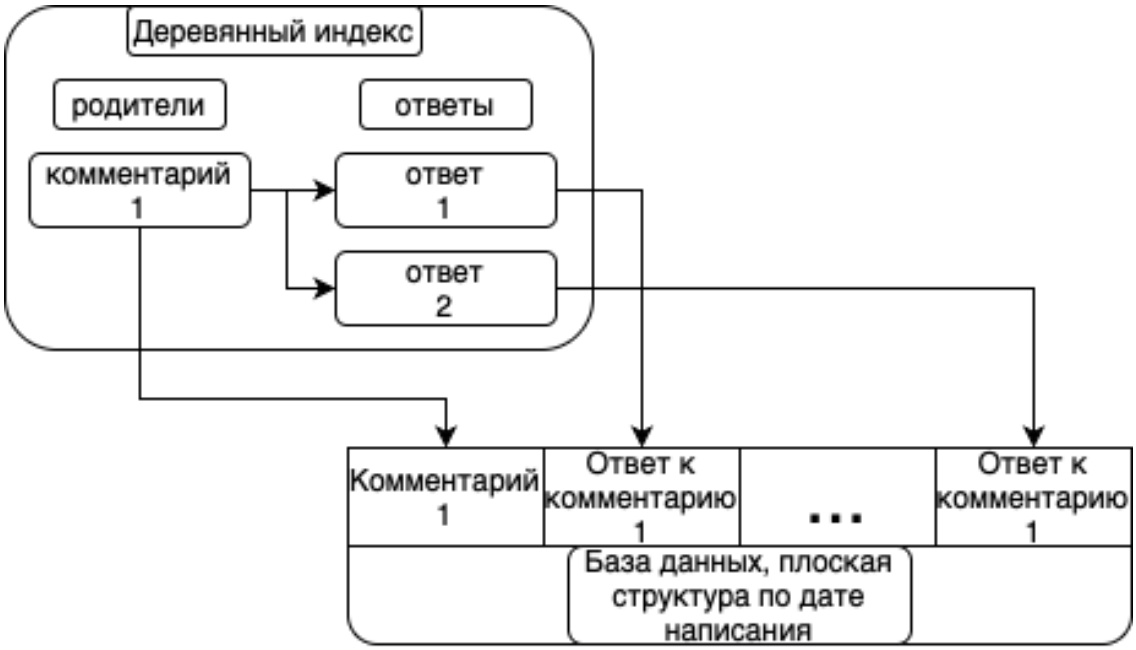


Рисунок 2 – Деревянный индекс комментариев

Дискуссии, созданные до запуска индекса, были успешно проиндексированы рабочим, создание которого привело к разработке библиотеки для упрощения проведения подобных работ над базой данных в дальнейшем. Для увеличения скорости получившейся системы был использован off-heap кеш, уже успешно применявшийся в социальной сети на практике [6]

На данный момент новая древовидная система комментариев успешно включена на всех пользователей портала «Одноклассники». Запуск прошел без серьезных происшествий, а фидбек от аудитории социальной сети был положительным.

ЛИТЕРАТУРА

1. История и этапы развития соцсети Одноклассники [Электронный ресурс], URL: https://insideok.ru/info/
2. Новый медиакит: Что нужно знать об Одноклассниках в 2022 году [Электронный ресурс], URL: https://insideok.ru/blog/novyj-mediakit-chto-nuzhno-znat-ob-odnoklassnikah-v-2022-godu/
3. Работа с A/B-тестами в крупной соцсети: подробно об A/B платформе Одноклассников [Электронный ресурс], URL: https://habr.com/ru/companies/odnoklassniki/articles/772958/
4. Блог компании Одноклассники: NewSQL = NoSQL+ACID [Электронный ресурс], URL: https://habr.com/ru/companies/odnoklassniki/articles/417593/
5. Мини-интервью Олега Анастасьева: отказоустойчивость в Apache Cassandra [Электронный ресурс], URL: https://habr.com/ru/companies/odnoklassniki/articles/465475/
6. Как мы избавились от пауз GC с помощью собственного java off-heap storage решения [Электронный ресурс], URL: https://habr.com/ru/companies/odnoklassniki/articles/139185/

УДК 004.4

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

СЕРВИС МНОГОУРОВНЕВОГО ОПОВЕЩЕНИЯ ПОЛЬЗОВАТЕЛЕЙ В СИСТЕМАХ МАССОВОГО ИНФОРМИРОВАНИЯ

Современные компании сталкиваются с задачей эффективного оповещения большого числа клиентов, гарантируя, что каждый получит уведомление. Решить эту задачу предлагается с помощью разработки сервиса, обеспечивающего следующие функции:

- Доставку уведомлений до конечного пользователя.
- Максимизацию вероятности взаимодействия пользователя с сообщением.
- Соответствие правовым нормам (контроль количества и частоты отправляемых уведомлений).

В информационных системах важно обеспечить не только доставку уведомлений до пользователя, но и сделать этот процесс максимально удобным и не навязчивым. Разработанный сервис предлагает комплексное решение, сочетающее в себе следующие возможности:

1. Автоматическая отправка уведомлений через разнообразные каналы связи, включая автоматические звонки, SMS и email-сообщения. Это позволяет охватить широкий спектр предпочтений пользователей.
2. Прямые звонки клиентам через платформу, обеспечивающие личный подход в общении с клиентами.
3. Многоуровневая система оповещений, классифицирующая клиентов по разным уровням в зависимости от их предпочтений и истории взаимодействий. Это позволяет настроить индивидуальные каналы связи и частоту уведомлений для каждой группы пользователей.
4. Настройка лимитов уведомлений в сервисе для обеспечения прав клиентов на приватность и минимизации информационного шума.
5. Контроль за доставкой сообщений и соблюдение установленных ограничений, гарантирующий, что каждое уведомление достигнет своего адресата.
6. Гибкость настройки параметров оповещений, адаптируемых под меняющиеся требования бизнеса и предпочтения клиентов.

Сервис способствует достижению баланса между необходимостью информирования клиентов и соблюдением их прав на конфиденциальность, повышая вероятность положительного взаимодействия с уведомлениями.

Система оповещения клиентов строится на четырех основных сервисах, обеспечивающих ее функционирование:

Segmentation Service - обрабатывает данные организации, классифицируя их для взаимодействия с клиентами. Важная роль сервиса заключается в публикации в Kafka топиках сведений о действиях, требующихся от клиента, и обновлениях данных.

Activity Planner Service - осуществляет планирование действий для клиентов, основываясь на поступивших сообщениях. Сервис управляет базой данных с информацией о запланированных активностях и их сроках, инициируя отправку уведомлений при наступлении времени действия.

Client Service - предоставляет доступ к контактным данным клиентов, таким как телефон или email, хранящимся в базе данных. Этот сервис обрабатывает запросы от Activity Planner Service для сбора необходимых данных для уведомлений.

Notification Service - отвечает за отправку уведомлений клиентам через внешние каналы. В случае неудачной попытки сервис повторяет запрос с определенным таймаутом до успешной доставки сообщения.

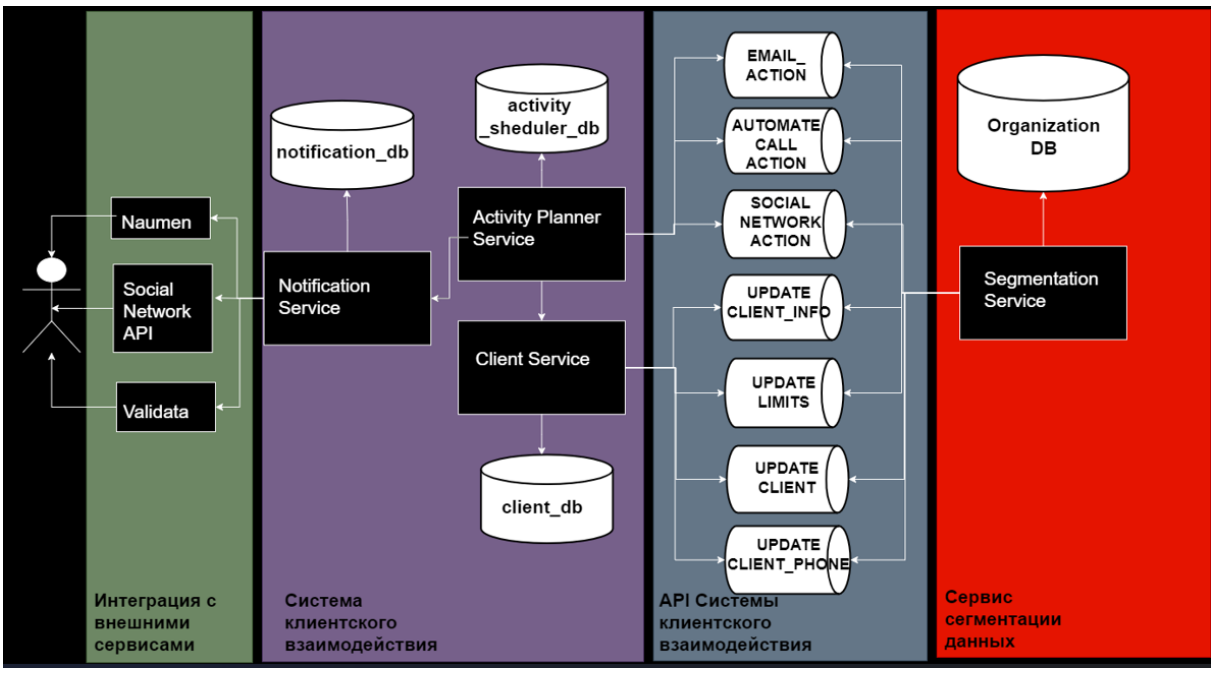


Рисунок 1 – Архитектура сервиса

Сервис реализован с использованием микросервисной архитектуры и паттерна "Database per service" для обеспечения масштабируемости, надежности и гибкости системы. Микросервисы упрощают добавление и обновление компонентов системы, повышая ее доступность за счет изоляции отказов [1]. Индивидуальные базы данных для каждого сервиса позволяют оптимизировать производительность [2]. Apache Kafka используется для взаимодействия между микросервисами, поддерживая высокую пропускную способность и декуплирование компонентов, что критически важно для систем оповещения и обеспечивает их масштабируемость и гибкость [3]. Для реализации использовался JDK 21, обеспечивающий поддержку новейших функций Java [4]. В качестве системы управления базами данных выбрана PostgreSQL 14.11 [5].

ЛИТЕРАТУРА

1. Martin Fowler. Microservices [Электронный ресурс]. – Доступно по URL: https://martinfowler.com/articles/microservices.html
2. Chris Richardson. Pattern: Database per service [Электронный ресурс]. – Доступно по URL: https://microservices.io/patterns/data/database-per-service.html
3. Официальный сайт Apache Kafka [Электронный ресурс]. – Доступно по URL: https://kafka.apache.org/ (дата обращения: 01.03.2024).
4. Jdk 21 Documentation [Электронный ресурс]. – Доступно по URL: https://docs.oracle.com/en/java/javase/21/ (дата обращения: 01.03.2024).
5. PostgreSQL 14.11 Documentation [Электронный ресурс]. – Доступно по URL: https://www.postgresql.org/docs/14/static/index.html (дата обращения: 01.03.2024).

УДК 004.422.81

Р. В. Кораблев (4 курс бакалавриата),
А. П. Маслаков, ст. преподаватель

РАЗРАБОТКА ПЛАТФОРМЫ ДЛЯ ОТДАЧИ ПОЛЬЗОВАТЕЛЬСКИХ КОММУНИКАЦИЙ

Целью работы является создание платформы коммуникаций, которая обеспечит своевременную и эффективную доставку уведомлений пользователям для повышения их вовлечённости.

Существует множество коммерческих и открытых систем управления промоблоками, таких как AdFox [1], Yandex.Direct [2], Google Ads [3] и другие. Однако анализ этих систем выявил существенные недостатки. Основные функциональные проблемы заключаются в

невозможности построить четкую маркетинговую стратегию с показом уведомлений в зависимости от предыдущих взаимодействий с коммуникациями. Также существующие платформы не позволяют использовать кроссплатформенный и кросс-сервисный идентификатор пользователя, что не позволяет таргетироваться на взаимодействия пользователя с коммуникациями из другого сервиса.

Разработка нового сервиса для управления пользовательскими нотификациями может решить эту проблему и предоставить сервисам возможность более точно настраивать таргетирование уведомлений на пользователя. Например, маркетолог может настраивать уведомления только для пользователей, которые просмотрели определённое количество серий конкретного сериала, или только для тех, кто имеет истекающую подписку.

Такой подход позволит пользователям платформы значительно повысить эффективность своих маркетинговых кампаний и улучшить пользовательский опыт. Также платформа позволит не надоедать пользователю с однотипными промокомпаниями в разных приложениях.

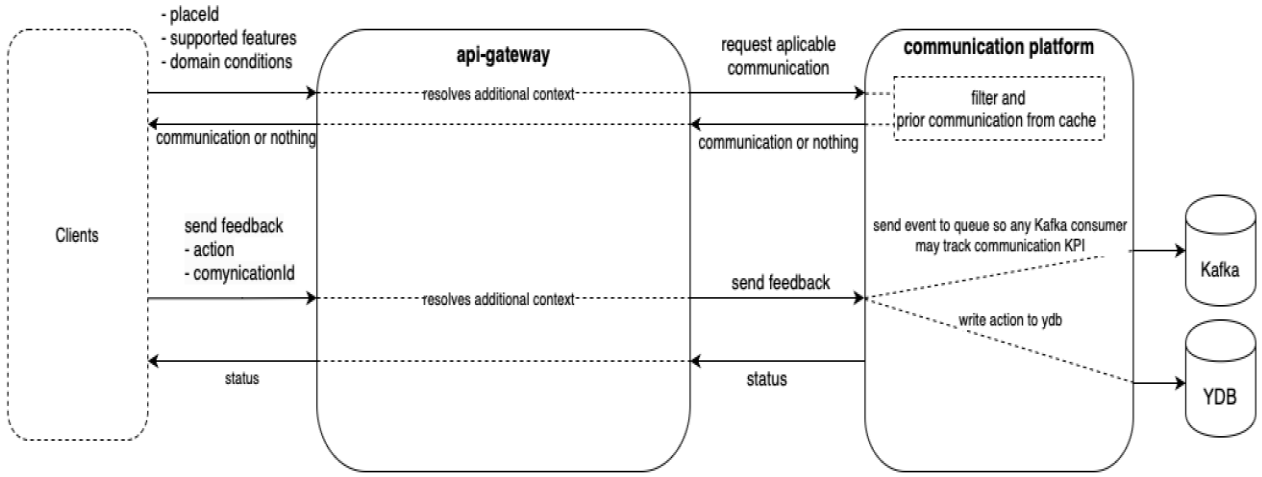


Рисунок 1 – Архитектура платформы коммуникаций

Архитектура платформы представлена на рисунке 1. Клиент отправляет запрос в абстрактный API Gateway-сервис [4] с указанием места на сайте или в приложении запрошенной коммуникации, кроме того, отправляет список поддерживаемых функциональных возможностей клиента и абстрактные доменные условия.

API Gateway валидирует авторизацию ползователя и обогащает контекст пользователя. В случае успеха он отправляет дальнейший запрос в платформу.

Платформа на своей стороне промеряет применимость каждой из коммуникаций на применимость для данного запроса и отдаёт самую приоритетную из них.

Далее клиент показывает полученную коммуникацию и отправляет информацию о просмотре в платформу. Эта информация будет записана в БД и отправлена в брокер сообщений для дальнейшей обработки аналитиками.

В качестве БД для платформы будет выступать реляционная БД YDB [5], которая имеет механизмы автоматического масштабирования и шардирования без потери производительности и отказоустойчивости.

В качестве брокера сообщений будет использоваться Kafka [6], так как она обеспечивает высокую пропускную способность, отказоустойчивость и гарантирует однократную доставку сообщений. Благодаря своей распределённой архитектуре Kafka позволяет эффективно обрабатывать большие объёмы данных и поддерживать высокую доступность системы. Кроме того, она обладает гибкой конфигурацией и масштабируемостью, что делает её отличным выбором для передачи сообщений в реальном времени.

Разработка платформы для отдачи пользовательских коммуникаций с одной стороны позволит улучшить пользовательский опыт взаимодействия с коммуникациями, а с другой — повысит эффективность маркетинговых компаний, нацеленных на вовлечение пользователей в использование продукта.

ЛИТЕРАТУРА

1. AdFox [Электронный ресурс] URL: https://yandex.ru/support/adfox-sites
2. Yandex Direct [Электронный ресурс] URL: https://direct.yandex.ru
3. Google ads [Электронный ресурс] URL: https://ads.google.com/intl/ru_ALL/home
4. Amazon API Gateway [Электронный ресурс] URL: https://aws.amazon.com/ru/api-gateway
5. Распределённая отказоустойчивая реляционная система управления базами данных компании Яндекс с открытым исходным кодом YDB [Электронный ресурс] URL: https://ydb.tech/ru
6. Распределённый программный брокер сообщений Kafka [Электронный ресурс] URL: https://kafka.apache.org

УДК 004.02

Г. В. Куксов (2 курс магистратуры),
А. Г. Сиднев, к.т.н., доцент

ГЕНЕРАТОР ИМИТАЦИОННЫХ МОДЕЛЕЙ БИЗНЕС-ПРОЦЕССОВ В СИСТЕМЕ ANYLOGIC

Бизнес-процесс – это структурированный, измеряемый набор действий, направленных на преобразование ресурсов организации (используемых на входе в процесс) в конечный продукт (получаемый на выходе из процесса), и имеющий ценность для клиента или рынка. Таким образом, бизнес-процесс представляет собой набор последовательных действий, которые приводят к решению определенной предпринимательской задачи.[1] На данный момент существует несколько общепринятых определений, но общими элементами всех дефиниций бизнес-процесса является наличие некоторых ресурсов на входе, преобразования (непосредственно сам процесс) и некоторого результата на выходе.

Оптимизация производственных процессов является одной из ключевых задач любой производственной компании – если процесс достаточно прост, то он может быть представлен точной моделью, которая может быть эффективно оптимизирована. Адекватной математической моделью рассматриваемого класса объектов является сеть систем массового обслуживания (СМО):

- Порядок выполняемых действий задается соответствующим образом, например, с помощью матрицы передач или в форме специального графа
- Каждой работе ставится в соответствие ее исполнитель. Время занятости исполнителя работы является случайной величиной с заданным законом распределения.
- Допускается распараллеливание процесса выполнения работ с последующим объединением параллельных ветвей
- Допускается назначение одного и того же исполнителя на несколько работ

В случае моделирования функционирования сложных систем используется имитационное моделирование, примерами которых могут служить промышленные предприятия, транспортные сети, больница, человек и машина, которой он управляет – модели такой степени сложности, что предсказание их поведения принципиально возможно только в ходе компьютерного модельного эксперимента. Исторически первым сложился аналитический подход к исследованию систем, однако такое исследование, содержащее сотни или тысячи элементов, сталкивается со значительными трудностями, преодоление которых невозможно.[2]

На данный момент одним из лидирующих инструментов имитационного моделирования для бизнеса является ПО, разработанное российской компанией “The AnyLogic Company” –

AnyLogic. Он помогает аналитикам, инженерам и руководителям из разных отраслей получать детальное представление о бизнес-системах и процессах и оптимизировать их.

Однако для создания исполняемой имитационной модели в выбранной среде необходимо проделать трудоемкую и многоэтапную работу – изучить систему моделирования и понять принципы построения моделей, а также научиться адекватно их представлять; понимать принципы проведения оптимизационных экспериментов с последующим формированием файла отчета с необходимой информацией.

Ставится задача создания программного продукта, обеспечивающего задание формального описания бизнес-процесса, и формирования исполняемого файла имитационной модели вне системы моделирования AnyLogic для последующих экспериментов в ней. Подобная задача автоматизации построения имитационных моделей решалась на примере Словенской мебельной компании[3] – ввиду невозможности ручного построения модели, состоящей из 30000 подпроцессов, Тадей Кандук и Блаз Родик разработали метод автоматизированного построения модели. Анализировались открытые заказы и создавалась соответствующая модель, путем редактирования исполняемого проектного файла с расширением .alp. Данная возможность обусловлена тем, что вся информация о блоках, агентах и связях между ними задается тэгами в древовидной структуре XML. Также в файле можно задавать параметры оптимизационных экспериментов на модели.[4]

Для решения поставленной задачи предлагается создание расширения для графического редактора с открытым исходным кодом Violet UML Editor[5] - таким образом пользователь сможет задавать формальное описание путем перетаскивания различных блоков с указанием связей и необходимых параметров. Помимо имеющихся опций экспорта в форматах .pdf и .png реализована возможность экспортировать созданную модель в формате проектного файла .alp, что позволяет сразу запустить ее в среде моделирования и проводить оптимизационные эксперименты.

ЛИТЕРАТУРА

1. Бизнес-процесс. Большая российская энциклопедия. https://bigenc.ru/c/biznes-protsess-e6db80 (дата обращения: 26.01.2024)
2. А.Л. Королев. Компьютерное моделирование: Учебное пособие. – Челябинск, 2019.
3. Kanduč, T.; Rodič, B. (2014). Automated simulation model building for a complex furniture manufacturing process, 23rd International Electrotechnical and Computer Science Conference ERK 2014, Slovenia section IEEE.
4. Оптимизационный эксперимент. AnyLogic. https://anylogic.help/ru/anylogic/experiments/optimization-experiment.html (дата обращения: 26.01.2024)
5. Developer guide // Violet UML Editor URL: http://alexdp.free.fr/violetumleditor/page.php?id=en:developerguide (дата обращения: 18.03.2024).

УДК 004.416

М. Ю. Куликов (1 курс магистратуры),
В. В. Амосов, к.т.н., доцент

РАЗРАБОТКА ВЕБ-СЕРВИСА ИЗМЕНЕНИЯ ЭМОЦИЙ В РЕЧИ

Работа с компьютерными устройствами происходит в три этапа: ввод данных, обработка данных и их вывод. Синтез речи - одна из возможных форм вывода информации. В широком смысле синтез речи - это формирование голосового сигнала по каким–либо передаваемым параметрам [1]. В основном, синтез речи состоит из перевода печатного текста в звук - данный процесс известен как Text-To-Speech (TTS). Цель TTS заключается в синтезе сигналов, подобных человеческой речи, чтобы лингвистическая информация могла быть передана чётко и без наличия нескольких интерпретаций [2]. Одна из вариаций данного метода - обучение

машины закономерностям эмоций и последовательное наложение этих закономерностей на нейтральную речь.

Синтез речи имеет широкий спектр применений - от помощи в изучении языков и использования в индустрии развлечений до подмоги инвалидам и интеграции в разработках искусственного интеллекта. Так, например, синтез речи используется в качестве инструмента при работе с детьми, страдающими дислексией, а применение в call-центрах в качестве системы интерактивного голосового ответа давно стало привычной практикой. На данный момент разговорный стиль синтезированного голосового сигнала нейтрален, и избавление от этой нейтральности приведёт к ускоренному увеличению сфер применения.

Целью работы является разработка информационной системы, основанной на методах распознавания эмоций в речи и синтеза эмоционального окрашивания речи с использованием различных методов, а также последующее создание веб-сервиса с применением разработанной системы.

Функциональные требования:

- Регистрация новых пользователей.
- Хранение данных о пользователях.
- Возможность работы с аудиофайлами, включая загрузку пользователями аудиофайлов, изменение параметров загруженных аудиофайлов, воспроизведение измененных аудиофайлов и скачивание измененных аудиофайлов.

Нефункциональные требования:

- Приложение должно соответствовать стандарту безопасности ГОСТ Р 59494-2021.
- Приложение должно быть масштабируемым - база данных должна обеспечивать стабильное хранение до 100 тысяч документов.
- Время отклика системы для основных операций не должно превышать порог в 3 секунды.
- Система должна быть доступной 99% времени, чтобы минимизировать перерывы в обслуживании.
- Интерфейс должен быть легким в использовании и интуитивно понятным.

Один из методов синтеза речи - метод нейронных сетей. Этот метод использует глубокие нейронные сети для синтеза речи с определенными эмоциональными характеристиками [3]. Нейронные сети обучаются на большом объеме аудиоданных, представляющих различные эмоции, такие как радость, грусть, злость и др. Затем модель анализирует эти данные и на основе полученных паттернов может преобразовывать обычную речь в эмоционально окрашенную [4]. Этот метод позволяет достичь высокого уровня реалистичности и выразительности в синтезе речи с разными эмоциональными оттенками, но для этого необходимо большое количество подготовленных данных [5].

В качестве языка программирования был выбран Python [6]. Python был создан как универсальный язык программирования с акцентом на простоту и читаемость кода. Он широко используется в различных областях, таких как веб-разработка, научные вычисления, искусственный интеллект, анализ данных и многое другое. Python отличается своей кроссплатформенностью, что позволяет запускать программы на различных операционных системах без изменений в коде. Это делает его популярным выбором для создания мультиплатформенных приложений.

В качестве среды разработки был выбран Google Colaboratory. Google Colaboratory - это бесплатный облачный сервис от Google, предназначенный для совместной работы с кодом и машинным обучением. Он основан на Jupyter Notebook, популярной веб-среде, которая позволяет создавать и делиться документами, содержащими живой код, уравнения, визуализации и текстовые описания.

В качестве системы управления базами данных (СУБД) была выбрана MongoDB [7]. MongoDB - это популярная СУБД, которая относится к категории NoSQL. В базах данных этой категории не реализована реляционная модель, что приводит к хорошей горизонтальной масштабируемости.

Для моделирования эмоций в речи используется нейронная сеть, обученная на большом количестве данных, и библиотека librosa. Librosa - это библиотека Python для анализа музыки и аудио, набор инструментов для извлечения информации из аудиоданных. Данная библиотека используется для изменения параметров аудиофайла, таких как, тональность голоса, скорость речи, громкость, благодаря чему будет меняться восприятие эмоций.

Для анализа звуковых волн используется библиотека Aniemore для языка программирования Python. Aniemore является мощной и гибкой библиотекой, специально разработанной для анализа эмоций, и предлагает ряд передовых моделей и алгоритмов, которые обеспечивают точное и надежное определение эмоций в аудиоматериалах. В основе Aniemore лежит использование глубоких нейронных сетей, обученных на большом объеме данных, содержащих различные эмоциональные выражения.

ЛИТЕРАТУРА

1. Katsuki Inoue, Sunao Hara, Masanobu Abe, Nobukatsu Hojo, Yusuke Ijima. "Model architectures to extrapolate emotional expressions in DNN– based text– to– speech" [Электронный ресурс] Режим доступа: https://arxiv.org/abs/2102.10345
2. Rich Caruana. "Multitask learning" [Электронный ресурс] Режим доступа: https://link.springer.com/article/10.1023/a:1007379606734
3. Generative Emotional AI for Speech Emotion Recognition: The Case for Synthetic Emotional Speech Augmentation [Электронный ресурс] Режим доступа: https://arxiv.org/pdf/2301.03751.pdf
4. Transfer Learning from Speaker Verification to Multispeaker Text-To-Speech Synthesis [Электронный ресурс] Режим доступа: https://www.semanticscholar.org/reader/8fc09dfcff78ac9057ff0834a83d23eb38ca198a
5. Speech Synthesis with Mixed Emotions [Электронный ресурс] Режим доступа: https://arxiv.org/pdf/2208.05890.pdf
6. Python 3.12.1 Documentation / [Электронный ресурс] Режим доступа: https://docs.python.org/3/
7. MongoDB Documentation / [Электронный ресурс] Режим доступа: https://www.mongodb.com/docs/

УДК 004.453

В. П. Лукьянов (4 курс бакалавриата),
Т. В. Коликова, ст. преподаватель

ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА МОБИЛЬНОГО ПРИЛОЖЕНИЯ ДЛЯ СОЦИАЛЬНОЙ ПОМОЩИ

В современном мире люди давно не представляют свою жизнь без технологий, с каждым днем увеличивается их интеграция во всех сферах, в том числе и в тех, которые веками не претерпевали изменений и были сложены еще задолго до появления первых вычислительных машин.

Пересечение и объединение разных отраслей с информационными технологиями позволяют добиваться немыслимых результатов, выводить на новый уровень прогресс в них, а программные продукты, являющиеся результатом такого симбиоза, практически всегда становятся экономически успешными и плотно входят в нашу жизнь. Карты городов, банковские приложения, музыкальные сервисы – все это стало успешным благодаря, в первую очередь, тому, как с помощью них быстро и эффективно удовлетворяются базовые и специфические потребности людей, в сравнении с не таким далеким прошлым.

На сегодняшний день активное взаимодействие между людьми и их кооперация отходят на второй план, заменяясь индивидуализмом. К тому же подавляющее большинство людей большую часть времени проводят в гаджетах. В связи с этим, актуальной является разработка полноценного мобильного приложения, объединяющего в себе функциональность социальной сети и агрегатора волонтерской деятельности.

Разрабатываемое приложение предназначено для облегчения взаимодействия людей в повседневных ситуациях. Его ключевое отличие от других подобных продуктов заключается в

том, что оно предлагает тонкое разделение пользователей на группы интересов. Благодаря этому пользователь будут получать только те события, которые ему интересны. В зависимости от настроек функциональность создаваемых событий так же может изменяться. Приложение также имеет систему оценивания пользователей, чтобы исключить недобросовестных.

Параллельно со своим основным предназначением приложение будет способствовать улучшению коммуникации между людьми и повышению гражданской активности.

Таким образом, целью данной работы является проектирование и разработка приложения для двух крупнейших мобильных платформ: Android и iOS, а также его серверной части REST API, готовой к увеличению числа пользователей, сохраняя при этом высокую производительность, отзывчивость, отказоустойчивость и безопасность.

Для достижения поставленной цели необходимо выполнить следующие задачи:

1. Обзор существующих решений для разработки высоконагруженных приложений
2. Проведение сравнительного анализа найденных решений
3. Разработка требований к программному продукту
4. Разработка прототипа приложения
5. Проектирование архитектуры и разработка спецификации программного продукта
6. Разработка программного продукта

В ходе сравнения средств для разработки API, было принято решение использовать Java-фреймворк Spring Framework [2, 3]. Он предоставляет базу для создания приложений с высокой производительностью и масштабируемостью, способных обрабатывать большие объемы трафика. Кроме того, он включает в себя поддержку кэширования, что позволяет улучшить производительность приложений, требующих частого доступа к данным. Наконец, Spring Framework предлагает полный набор инструментов для мониторинга и управления производительностью приложений, что помогает разработчикам быстро выявлять и устранять проблемы с производительностью.

При разработке приложений разработчики стараются охватить как можно больше платформ, на которых может функционировать продукт. Это является очень важным аспектом разработки, так как фокус только на одной платформе может снизить возможную аудиторию приложения.

Однако каждая мобильная платформа имеет собственный инструментарий для разработки приложений для них, в результате чего разработчику необходимо реализовать продукт отдельно для каждой системы. Выходом из данной ситуации является использование кроссплатформенных фреймворков приложений, они обеспечивают независимость разработчика от платформы.

В результате было принято решение использовать одним из таких фреймворков – Flutter [4, 5]. Это кроссплатформенный фреймворк для создания мобильных приложений под Android и iOS, веб-приложений, а также настольных приложений под Windows, macOS и Linux с использованием языка программирования Dart [6].

ЛИТЕРАТУРА

1. Бубнов О. О. Анализ архитектур построения высоконагруженных WEB-сервисов и приложений // Прикладные исследования и технологии ART2016. - М.: Негосударственное образовательное учреждение высшего образования Московский технологический институт, 2016. - С. 58–62.
2. Филисов Д. А. Spring Framework в высоконагруженных приложениях // Инновации и инвестиции. - 2023. - №5. - С. 206–210.
3. Spring [Электронный ресурс] Режим доступа: https://spring.io/
4. Вениаминов Н. Ю, Ковалев С. В. Проектирование мобильного приложения на фреймворке Flutter // Информатика и вычислительная техника. - Чебоксары: Чувашский государственный университет имени И. Н. Ульянова, 2021. - С. 52–56.
5. Flutter. [Электронный ресурс] Режим доступа: https://flutter.dev/
6. Dart programming language. [Электронный ресурс] Режим доступа: https://dart.dev/

УДК 004.41

А. Д. Лембер (4 курс бакалавриата),
Л. П. Котлярова, ст. преподаватель

РАЗРАБОТКА БОТА ДЛЯ СОЦИАЛЬНОЙ СЕТИ TELEGRAM ДЛЯ ОТСЛЕЖИВАНИЯ ОБНОВЛЕНИЙ НА РЕСУРСЕ «РОССИЙСКАЯ ОБЩЕСТВЕННАЯ ИНИЦИАТИВА»

«Российская общественная инициатива» (РОИ) [1] — интернет-ресурс для размещения гражданами Российской Федерации своих инициатив по социально-экономическому развитию РФ, а также по муниципальному и государственному управлению. РОИ — не единственный петиционный ресурс в России [2]. Однако только данный ресурс учреждён указом Президента РФ, и инициативы только с данного ресурса, набравшие определённое минимальное число голосов, обязаны быть рассмотрены экспертной рабочей группой с вынесением мотивированного заключения. В настоящий момент на ресурсе РОИ почти каждый день появляются новые инициативы, а многие из существующих уже набрали десятки тысяч голосов в их поддержку.

Учитывая особый, привилегированный статус РОИ среди других петиционных ресурсов РФ, мне стало интересно проанализировать его функциональность и удобство его использования. В процессе данного анализа я заметил такой недостаток у ресурса РОИ, как его весьма ограниченные возможности в плане автоматической рассылки пользователям новостей и различных обновлений по инициативам: РОИ предоставляет только возможность подписки на ежедневную рассылку новых инициатив. Если же пользователь хочет узнавать не только о новых инициативах, но и о самых популярных, хочет следить за прогрессом голосования по конкретным инициативам и изменениям в их статусе, а также хочет быть в курсе новостей ресурса, то единственный выход для пользователя — это регулярно посещать сайт РОИ. Для сравнения, другой популярный ресурс, Change.org, предоставляет пользователям возможность подписки на куда более широкий и разнообразный перечень рассылок, а в статье [2] подчёркивается, что рассылка данным ресурсом информации о количестве голосов, собранных той или иной инициативой, способствует повышению осведомлённости пользователей ресурса об идеях данной инициативы. Таким образом, можно заключить, что разработка программного продукта для ресурса РОИ, автоматизирующего получение пользователем всей вышеописанной информации, является актуальной задачей. А оформление данного программного продукта в виде бота для социальной сети Telegram будет особенно удачным решением по следующим причинам:

1. Во-первых, это позволит получить доступ ко всей функциональности разработанного продукта прямо внутри приложения для данной соцсети, без необходимости устанавливать отдельное дополнительное приложение.
2. Во-вторых, это может поспособствовать повышению популярности «Российской общественной инициативы», так как в России Telegram имеет достаточно большую аудиторию и «...заслуженно считается одним из лучших мессенджеров для распространения политической агитации и активности» [3].

На момент написания данной статьи мне не удалось найти решения, аналогичные по функциональности моему боту.

Архитектура программной системы, ядром которой является разрабатываемый бот, представлена на рисунке 1. Первый компонент системы, с которым взаимодействует бот — это сервер социальной сети Telegram. Бот получает с сервера сообщения пользователей, отправленные ему, и отправляет на сервер свои ответы пользователям. Взаимодействие с сервером осуществляется с помощью Telegram Bot API. Второй компонент системы, с которым взаимодействует бот — это собственно ресурс РОИ. Бот получает данные с ресурса двумя способами: часть данных он получает с помощью API РОИ (так как json-файлы, получаемые таким образом, содержат куда меньше ненужной информации по сравнению с веб-страницами), а для получения той части данных, которую API не предоставляет, бот

загружает и анализирует соответствующие веб-страницы. Наконец, бот взаимодействует с базой данных, записывая и считывая информацию об обслуживаемых им пользователях, а также об инициативах с ресурса РОИ.

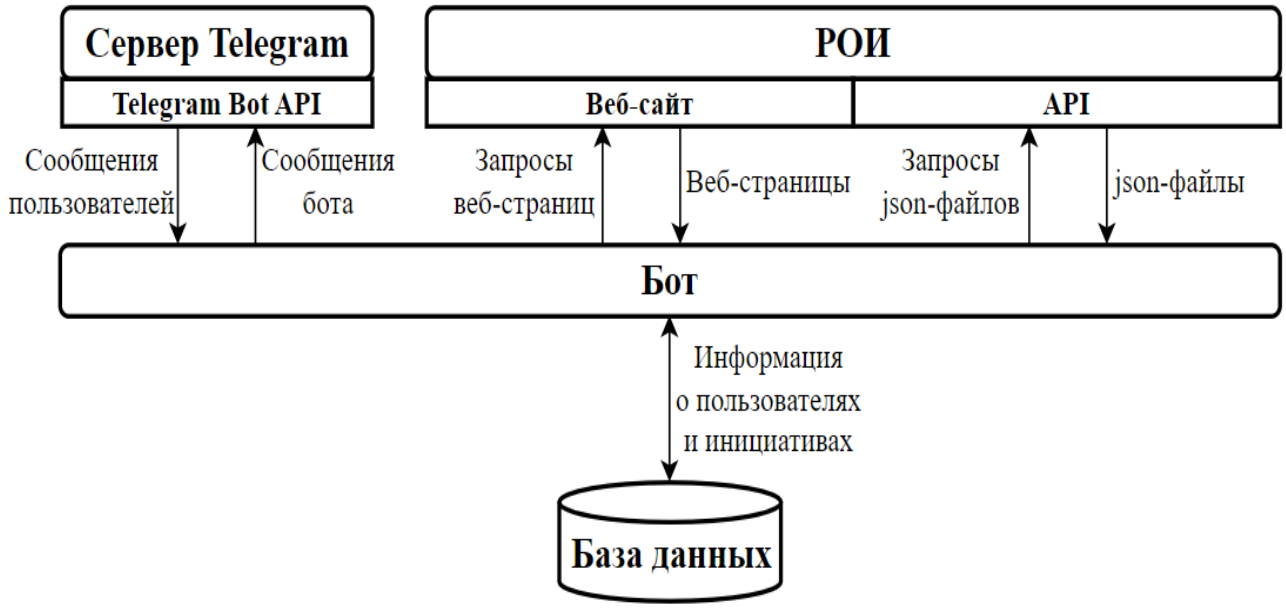


Рисунок 1 — Архитектура программной системы

Для разработки бота был подобран стек технологий. Язык программирования — Python версии 3.12. IDE — широко используемая [4] PyCharm Community Edition. Система хранения версий исходного кода — GitHub. Взаимодействие с Telegram Bot API осуществляется с помощью библиотеки python-telegram-bot [5]. СУБД — популярная [6] PostgreSQL версии 16. Для упрощения взаимодействия с БД используется ORM SQLAlchemy [7].

На данный момент бóльшая часть функциональности бота уже реализована. Пользователь может получать новости и регулярную сводку с ресурса РОИ, а также искать размещённые на нём инициативы и подписываться на них для регулярного получения по ним актуальной информации — и всё это прямо в боте, без посещения сайта ресурса.

ЛИТЕРАТУРА

1. Российская общественная инициатива [Электронный ресурс]. – URL: https://www.roi.ru/ (дата обращения: 14.03.2024).
2. Широков И. С. Онлайн-петиции и электронные обращения как форма электронной демократии в России: проблемы и перспективы // Известия Иркутского государственного университета. Серия Политология. Религиоведение. – 2022. – Т. 39. – С. 46–55. – DOI 10.26516/2073–3380.2022.39.46.
3. Литвин В. В. Пространство политических интернет-коммуникаций в современном российском обществе // Социодинамика. – 2023. – № 6. – С. 58–66. – DOI 10.25136/2409–7144.2023.6.40930.
4. Python Developers Survey 2022 Results // JetBrains [Электронный ресурс]. – URL: https://lp.jetbrains.com/python-developers-survey-2022/#DevelopmentTools (дата обращения: 14.03.2024).
5. python-telegram-bot // GitHub [Электронный ресурс]. – URL: https://github.com/python-telegram-bot/python-telegram-bot (дата обращения: 14.03.2024).
6. Most popular relational DBMS 2023 // Statista [Электронный ресурс]. – URL: https://www.statista.com/statistics/1131568/worldwide-popularity-ranking-relational-database-management-systems/ (дата обращения: 14.03.2024).
7. SQLAlchemy // GitHub [Электронный ресурс]. – URL: https://github.com/sqlalchemy/sqlalchemy (дата обращения: 14.03.2024).

УДК 004.453

В. А. Лямин (2 курс магистратуры),
И. П. Соловьев., к.ф.-м..н., доцент,
Н. Б. Ампилова, к.ф.-м..н., доцент

СРАВНЕНИЕ ИЗОБРАЖЕНИЙ КРИСТАЛЛОВ ПЛАЦЕБО И ЛЕКАРСТВЕННЫХ ПРЕПАРАТОВ МЕТОДАМИ ФРАКТАЛЬНОГО АНАЛИЗА

Во многих случаях цифровые изображения можно интерпретировать как фазовые портреты сложных процессов в разные моменты времени. Анализ этих представлений математическими методами дает возможность описать свойства наблюдаемых процессов, используя различные характеристики (особенности) изображений.

Поскольку распределение интенсивностей пикселей является основной информацией о цифровом изображении, в этих терминах часто формулируются характеристики изображения. Как правило, математические методы анализа используют нормированное распределение по ячейкам заданного размера, которое является вероятностной мерой, заданной на изображении. Нормирование (выбор меры) может осуществляться разными способами в зависимости от задачи.

Самый простой способ — это отношение суммы интенсивностей пикселей в ячейке к общей сумме интенсивностей изображения. Такой метод дает усреднение интенсивностей, а иногда и сглаживание разницы между областями изображения. Более предпочтителен выбор отношения суммы интенсивностей из данного интервала в ячейке к сумме интенсивностей из такого интервала для всего изображения.

Подготовка изображения к применению математических методов является не менее важной и весьма значимой частью исследования. Необходимо найти область интереса и преобразовать изображение к стандартному представлению, удобному для изучения задач определенного класса. Эту работу можно выполнить как с помощью графических редакторов, так и специально разработанных алгоритмов.

При изучении характеристик различных медицинских препаратов часто используется метод получения их кристаллических форм методом тезиографии. Для этого вещество помещают в соляную или масляную основу и оставляют кристаллизоваться при температуре примерно 18-20°С на протяжении суток. Добавление вещества в раствор приводит к образованию кристалла некоторой формы. Но есть определенные сложности получения стандартизированных изображений, так как результат кристаллизации зависит от множества факторов, таких как температура, соотношение массы вещества и раствора, а также само количество вещества.

Методы предварительной обработки изображений используются как для фильтрации, так и для решения задачи улучшения качества изображения, чтобы выделить и подчеркнуть те или иные свойства изображений и облегчить их визуальное восприятие экспертом в заданной предметной области. В медицинских приложениях такой подход помогает эксперту уточнить диагноз, оценить качество препарата, сравнить с некоторым стандартом и т.д.

Начальным этапом при подготовке изображений для методов фрактального анализа является поиск кристаллов, полученных на различных препаратах, примерно одинакового размера и формы. Одно из необходимых условий для корректной работы методов фрактального анализа это одинаковый размер изображения. Это позволяет сохранить структуру кристалла, так как размер всех изображений приводится к наименьшему в выборке. После этого для корректной работы методов фрактального анализа необходимо привести их к монохромному изображению.

Далее по мере необходимости следует применить различные методы фильтрации изображений, чтобы получить более выраженное отличие результатов исследования различных препаратов.

При исследовании кристаллических структур обычно применяют достаточно тонкие математические методы, а именно различные методы фрактального анализа, такие как метод обобщенной фрактальной сигнатуры, спектр обобщенных размерностей Реньи и другие.

Методы фрактального анализа описывают изображение с помощью фрактальных размерностей. Они применяется для расчета размерности Минковского для монохромных изображений [1]. Поскольку одно числовое значение не может охарактеризовать структуру изображения, необходимо применить некоторые модификации для получения векторных характеристик.

Мультифрактальные методы полезны, если анализируемые изображения можно интерпретировать как объединение нескольких фрактальных подмножеств. В этом случае мы можем получить набор фрактальных размерностей подмножеств (мультифрактальный спектр) или спектры Реньи [2], измеряющие изменения исходного распределения интенсивностей пикселей при его последовательных перенормировках. Наш опыт анализа биомедицинских изображений позволяет предложить комбинированное применение фрактальных и мультифрактальных методов для получения достоверных результатов [3].

Эксперименты проводились для 3 классов изображений медицинских препаратов. Все результаты продемонстрировали разделение рассчитанных признаков, а также, что сочетание предварительной обработки изображений и методов фрактального анализа позволяет получать разделенные графики характеристик.

ЛИТЕРАТУРА

1. N. Ampilova**,** I. Soloviev **,** J.-G. Barth**.** Application of fractal analysis methods to images obtained by crystallization modified by an additive. Journal of Measurements in Engineering, Vol. 7, Issue 2, 2019, p. 48-57. https://doi.org/10.21595/jme.2019.20436
2. Рогов А.А., Спиридогов К.Н. Применение спектра фрактальных размерностей Реньи как инварианта графического изображения // Вестник СПбГУ. -- 2008. -- С. 30-42.
3. Ампилова Н.Б., Соловьев И.П. Алгоритмы фрактального анализа изображений // Компьютерные инструменты в образовании. -- 2012.

УДК 004.052.42

А. С. Луценко, В. Е. Устинова (4 курс бакалавриата),
В. А. Ивлев, Г. В. Мироненков (3 курс аспирантуры)

РЕАЛИЗАЦИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЕ ДЛЯ ВЕРИФИКАЦИИ СООТВЕТСТВИЯ МОДЕЛЬНОГО И ВИЗУАЛЬНОГО ПРЕДСТАВЛЕНИЯ ПЛАНА ЖЕЛЕЗНОДОРОЖНОЙ СТАНЦИИ

С переходом к использованию средств автоматизации [1] и информатизации на железнодорожном транспорте появилась потребность в хранении планов железнодорожных станций в электронном формате, который был бы пригоден для использования приложениями, направленными на решение ряда задач, связанных с железнодорожной инфраструктурой.

Особенно актуальным является запрос на верификацию соответствия [2, 3] модельного и визуального представлений плана железнодорожной станции, который обычно в компании НИИАС, являющейся частью холдинга РЖД, решался путём ручного тестирования. Ручная верификация занимает от нескольких дней до нескольких недель рабочего времени в зависимости от размера станции. Поэтому существует необходимость в автоматизации этого процесса [4].

Цель работы – разработка и реализация алгоритма, позволяющего верифицировать модельное и визуальное представление железнодорожной станции.

Чтобы приступить к верификации соответствия для начала необходимо привести данные модельного и визуального представлений планов железнодорожной станции к общему виду. После анализа подходов к построению графов [5, 6] было принято решение строить графы с помощью списков смежности.

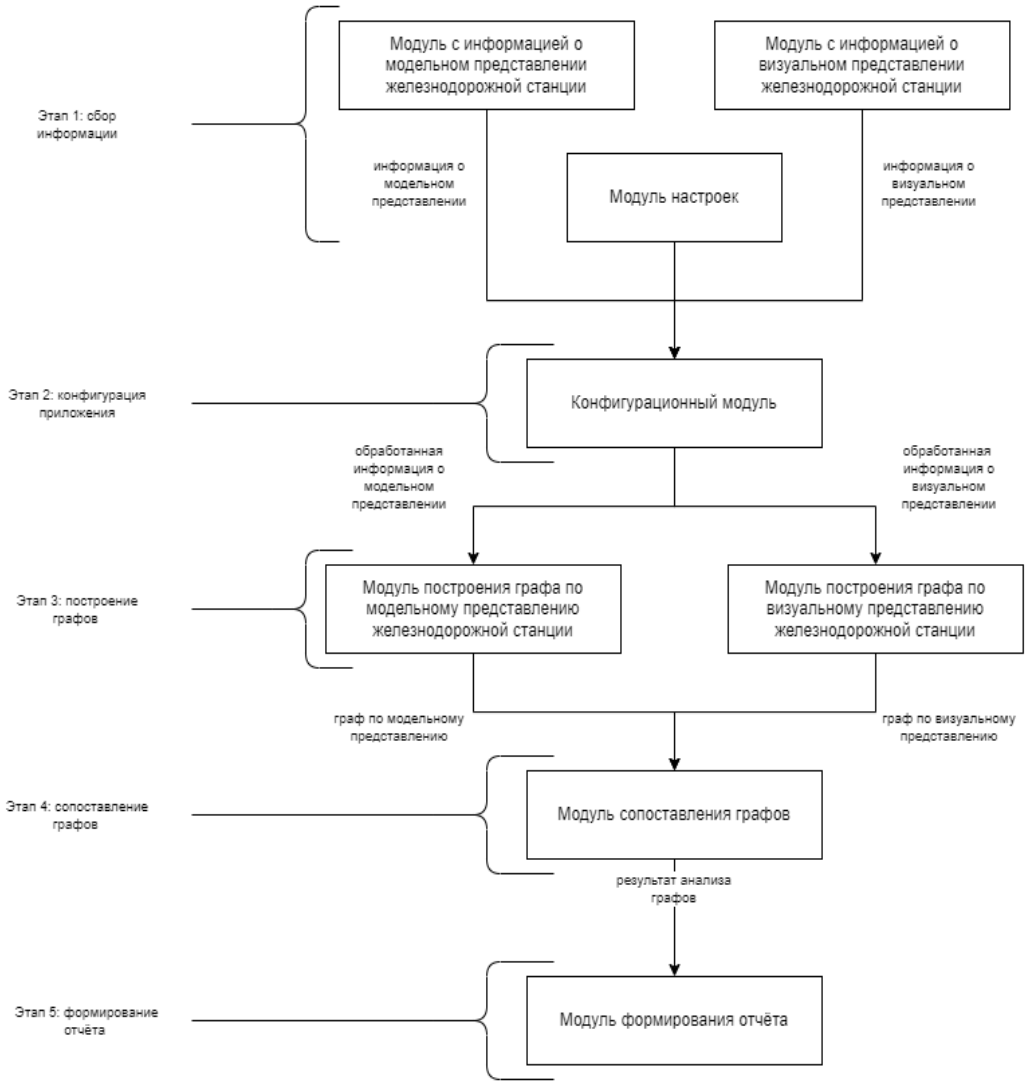


Рисунок 1 – Общая схема алгоритма.

На рисунке 1 изображена общая схема работы программного обеспечения, реализующая алгоритм разделён на пять этапов и заключается в последовательной обработке входных данных о модельном и визуальном представлениях железнодорожной станции в независимых компонентах.

Сам алгоритм верификации соответствия предполагает поэтапное сопоставление объектов железнодорожной станции друг другу. Сначала определяются “идентичные” объекты – те, которые имеют одинаковый тип (рельсовая цепь или стрелка), имя, а также соседей с одинаковыми типами. Такие объекты на визуальном представлении будут покрашены зелёным. Далее для всех “идентичных” объектов, у которых соседи не стали “идентичными”, начинается проверка и если типы соседей у проверяемого и “идентичного” для проверяемого совпали, то оба этих соседа также помечаются “идентичными”, но с отличием в имени. Эти объекты будут окрашены в жёлтый цвет, чтобы просигнализировать инженеру о том, что в модельном и визуальном представлении один и тот же объект назван по-разному. Далее повторяется обход, пока за итерацию ни одна “идентичная” пара не найдётся. Оставшиеся объекты – аномалии, которые не совпали ни с одним другим объектом, они будут окрашены в красный.

На первом этапе инженер загружает соответствующую информацию о модельном и визуальном представлении станции, а также указывает ряд настроек, необходимых для корректной подготовки входных данных [7]. На втором этапе вся собранная информация объединяется и преобразуется в необходимый формат в соответствии с настройками в

компоненте, отвечающем за конфигурацию приложения. На третьем этапе на основе подготовленной информации о представлениях железнодорожной станции происходит построение графов, с помощью которых на четвёртом этапе произойдёт анализ и верификация соответствия данных моделей. На последнем этапе полученные результаты анализа предоставляются в удобном для человека формате – с помощью текстового и визуального пояснения о расхождениях в загруженных инженером представлениях, облегчая таким образом поиск ошибок для их дальнейшего устранения.

Предложенный алгоритм лёг в основу построения системы автоматизации, использованной в компании НИИАС. Было создано программное обеспечение, реализующее алгоритм, которое позволило существенно сократить затрачиваемое время на верификацию соответствия модельного и визуального представлений железнодорожной станции. После внедрения был проведён эксперимент на тестовых примерах, содержащих упрощённые варианты станций: сначала было замерено время работы инженера, использующего метод ручной верификации, а затем с использованием разработанного программного обеспечения. В ходе данного эксперимента выяснилось, что среднее время с помощью предложенного средства автоматизации удалось сократить с 70 до 16 минут.

Таким образом, применение алгоритма помогло увеличить скорость верификации соответствия модельного и визуального представления плана железнодорожной станции в 5 раз.

ЛИТЕРАТУРА

1. Исследование систем автоматизации настройки инфраструктуры ИТ-проекта / В. А. Ивлев, Ф. Ю. Чемашкин, И. В. Никифоров, А. Д. Ковалев // Современные технологии в теории и практике программирования : Сборник материалов научно-практической конференции студентов, аспирантов и молодых ученых, Санкт-Петербург, 26–27 апреля 2023 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2023. – С. 204-205. – EDN HYVLSK.
2. Шалин, А. П. Верификация и валидация при оценке соответствия / А. П. Шалин, В. Н. Батраков // Контроль качества продукции. – 2022. – № 4. – С. 47-50. – EDN EWUOTS.
3. Сироткин, А. А. Некоторые основные компоненты цифровизации транспортно-логистической деятельности на железнодорожном транспорте / А. А. Сироткин, И. А. Тучина // Логистические системы в глобальной экономике. – 2019. – № 9. – С. 223-227. – EDN MYGRML.
4. Сысоев, И. М. Создание подхода автоматизации обновления конфигурационных файлов программного обеспечения / И. М. Сысоев, И. В. Никифоров, Е. В. Каплан // 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. – EDN YARVRM.
5. Пак, В. Г. Дискретная математика. Комбинаторный подход : учеб, пособие / В. Г. Пак. - СПб. : Изд-во Политехн. ун-та, 2010. - 235 с.
6. Свидетельство о государственной регистрации программы для ЭВМ № 2017617670 Российская Федерация. Построение и визуальное редактирование графа с сокращенными списками смежности вершин : № 2017614444 : заявл. 12.05.2017 : опубл. 11.07.2017 / А. М. Магомедов ; заявитель Федеральное государственное бюджетное учреждение науки Дагестанский научный центр Российской академии наук. – EDN GTRLZX.
7. 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. – EDN VDVCYX.

УДК 004.4

А. С. Марашов (2 курс магистратуры),
Г. Ф. Малыхина, д.т.н., профессор

ЭФФЕКТИВНОЕ ОБНАРУЖЕНИЕ МОШЕННИЧЕСКИХ ССЫЛОК С ПОМОЩЬЮ МАСКИРУЮЩЕГОСЯ ПОД ПОЛЬЗОВАТЕЛЯ ВЕБ-СКРАПЕРА

Исследователи [1] прогнозируют, что к 2025 году глобальный экономический ущерб от киберпреступлений достигнет порядка 10.5 триллионов долларов ежегодно, что эквивалентно приблизительно 1% от мирового валового внутреннего продукта. Одним из основных инструментов, используемых в целях совершения таких преступлений, являются мошеннические сайты, суть которых сводится к обману посетителей для получения выгоды.

Для решения проблемы обнаружения мошеннических ссылок существует множество подходов, в том числе оценка сайта по подозрительным критериям: молодой возраст домена, наличие известных логотипов, присутствие формы для ввода персональных данных [3].

Большинство методов исследования сайтов [2, 3] требуют доступа к содержимому страницы для принятия решения о подозрении, однако на практике в мошеннических сайтах часто используется механизм для разделения трафика, который на основе анализа информации о посетителе делает вывод, является ли тот пользователем или же это агент системы модерации: человек-модератор или бот – компьютер под управлением программного обеспечения.

Задача обнаружения сайтов с разделением трафика непрерывно усложняется: мошенники придумывают более сложные схемы маскировки контента, в то время как отделы безопасности компаний вынуждены создавать более совершенные системы обнаружения. Если в 2005 году для решения проблемы исследователи [4] акцентировали внимание на различиях между ссылками и словами, которые при загрузке сайтов были получены веб-скраперами (парсерами веб-страниц) с разными http-заголовками «user-agent» и разными IP-адресами, то в статье от 2021 года [5] авторы уделяют больше внимания проблеме разделения трафика на стороне клиента, принудительно исполняя Javascript-код страницы и сравнивая изображения получившихся веб-ресурсов с помощью алгоритмов перцептивного хеширования.

Рассмотренные методы обнаружения сайтов с разделением трафика либо имеют невысокую точность [4], либо требуют существенных затрат временных и вычислительных ресурсов [5], из-за чего обработка большого объёма ссылок за единицу времени не представляется возможной.

Таким образом, целью данной работы является построение веб-скрапера для эффективного обнаружения мошеннических ссылок, которые используют разделение трафика. Для достижения цели необходимо решить следующие задачи:

1. Провести аналитический обзор инструментов для построения веб-скрапера
2. Придумать и описать алгоритмы модерации ссылок и сравнения контента веб-страниц
3. Разработать веб-скрапер и провести тестирование его конфигураций

В данной работе используется объектно-ориентированный язык программирования Java. Для создания веб-скрапера был выбран фреймворк «Selenium» с его возможностями маскировки программного обеспечения под пользователя за счёт поддержки современных браузеров и гибкой настройки http-клиента.

Архитектура веб-скрапера представлена на рисунке 1. В качестве источника ссылок выступает HTTP-запрос, на который может быть дан синхронный ответ в случае наличия свежего результата для запрошенной ссылки или асинхронный через брокера сообщений Apache Kafka, когда сайт удастся обработать с помощью модуля извлечения данных и модуля анализа контента.

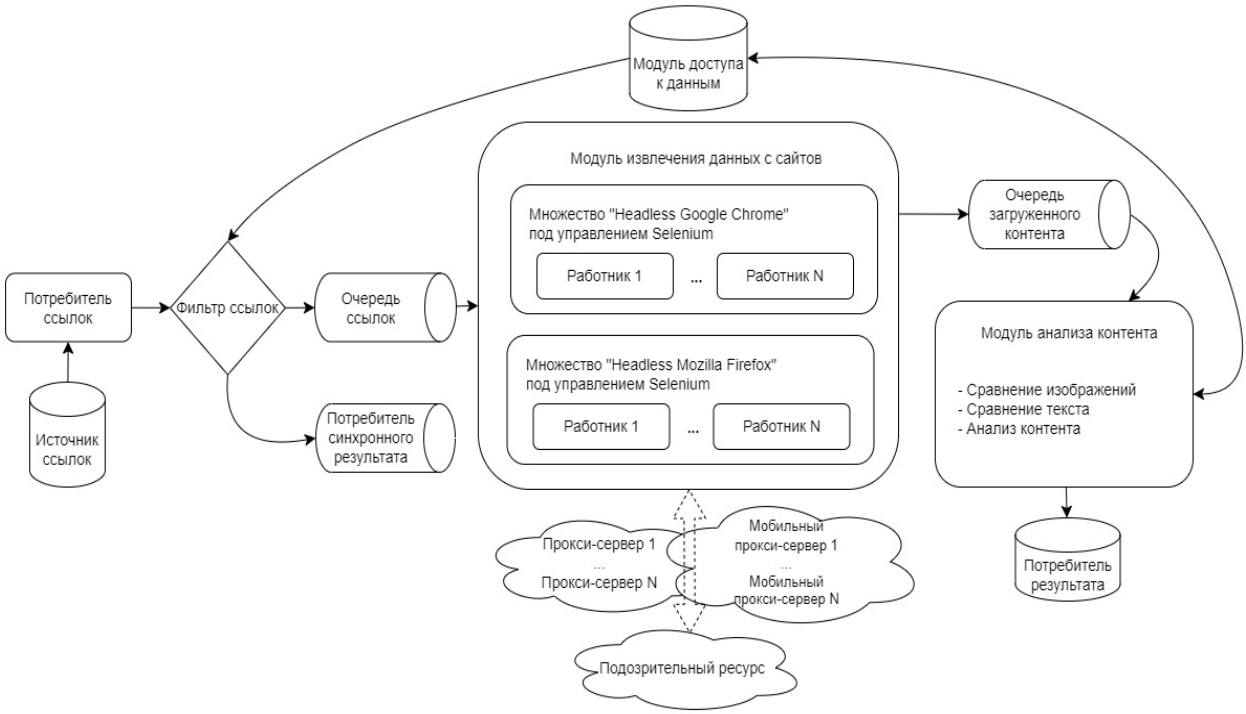


Рисунок 1 – Архитектура веб-скрапера

В сервисе используются лучшие практики обнаружения сайтов с разделением трафика, в том числе ротация IP-адресов с помощью использования прокси-серверов, сравнение полученных в разное время изображений веб-страниц и работа с несколькими конфигурациями браузеров.

По сравнению с существующими, разработанный веб-скрапер ориентирован не только на правильность решений, но и на высокую пропускную способность. Для сокращения времени обработки сайта используются «headless» браузеры, страницы загружаются параллельными потоками, а специальный алгоритм позволяет отказаться от загрузки лишних веб-ресурсов.

При этом в качестве хранилища данных выступает распределённая система управления базами данных Apache Cassandra. Использование этого инструмента обеспечивает оптимальное время чтения при высокой скорости записи, свойственной NoSQL-системам, а также закладывает фундамент для горизонтального масштабирования веб-скрапера.

ЛИТЕРАТУРА

1. A literature review of financial losses statistics for cyber security and future trend / S. Md Haris, M. Mehmood. // World Journal of Advanced Research and Reviews. – 2022. – N 15. – P. 138 – 156.
2. Maktabdar M., Zainal, A., Maarof M., Kassim M., Content based Fraudulent Website Detection Using Supervised Machine Learning Techniques, // Hybrid Intelligent Systems. HIS 2017. Advances in Intelligent Systems and Computing, vol 734, 2017
3. Park A. J., Quadari R. N., Tsang H. H., Phishing website detection framework through web scraping and data mining // 2017 8th IEEE Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON), Vancouver, BC, Canada, 2017. – P. 680 – 684
4. Wu B., Davison B., Cloaking and Redirection: A Preliminary Study // Proceedings of the First International Workshop on Adversarial Information Retrieval on the Web AIRWeb'05, 2005 – P. 1 – 10
5. Zhang P. et al., CrawlPhish: Large-scale Analysis of Client-side Cloaking Techniques in Phishing, // 2021 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 2021. – P. 1109 – 1124

УДК 004.43

А. В. Мейник (1 курс магистратуры),
С. П. Воскобойников, к.ф.-м.н., доцент,
Е.Н. Попов, к.ф.-м.н., н.с.

РАЗРАБОТКА ПРОГРАММЫ ДЛЯ МОДЕЛИРОВАНИЯ ВЗАИМОДЕЙСТВИЯ ДВУХМЕРНОГО АНСАМБЛЯ ДИПОЛЕЙ НА ПЛОСКОСТИ

Актуальность разработки программы связана с тем, что рассеяние электромагнитного излучения атомными ансамблями представляет интерес для исследований из-за возникновения коллективных эффектов при переизлучении поля, особенно в квантовом описании, что важно при проведении высокочувствительных измерений.

Задачами моделирования будут оценка изменения свойств ансамбля рассеивателей при увеличении их плотности и расчёт диаграммы направленности рассеянного света при разном количестве и расположении диполей. Решение задачи осложняется близостью рассеивателей, их взаимным перепоглощением и временными задержки требующими нелокальной формулировки задачи, что порождает сложную систему дифференциальных уравнений второго порядка.

Целью работы является разработка программы, позволяющей проводить взаимодействия ансамбля диполей на плоскости. Заметим, что работы [1-2] были посвящены моделированию взаимодействия одномерных ансамблей диполей.

Математическая модель взаимодействия описывается системой N линейных дифференциальных уравнений второго порядка с запаздывающим аргументом [3].

На плоскости около начал координат располагается N точечных рассеивателей. Координаты рассеивателей $\mathrm{r_i} = \{\mathrm{x_i}, \mathrm{y_i}\}$ принадлежат двухмерному нормальному распределению с шириной a. Каждый i-й рассеиватель характеризуется вектором дипольного момента $\mathbf{D}_\mathrm{i}$, который лежит в плоскости XOY. Каждый диполь является индуцированным и подчиняется уравнению:

$$\frac{\mathrm{d}^2\mathbf{D}_\mathrm{i}}{\mathrm{dt}^2} + \gamma\frac{\mathrm{d}\mathbf{D}_\mathrm{i}}{\mathrm{dt}} + \omega_0\mathbf{D}_\mathrm{i} = \alpha_\mathrm{n}\mathbf{E} + \alpha_\mathrm{n}\sum_{\mathrm{m}\neq\mathrm{i}}\mathbf{E}_{\mathrm{i,m}}(\mathrm{t}-\mathrm{r}_{\mathrm{i,m}}/\mathrm{c}) \tag{1}$$

$$\mathbf{E} = \mathbf{e}_x\varepsilon\cos(\omega t) + \mathbf{e}_y\varepsilon\sin(\omega t) \tag{2}$$

$$\omega_0 = \omega = \frac{2\pi}{\mathrm{T}}, \qquad r_{i,m} = \left|\mathbf{r}_m - \mathbf{r}_i\right| \tag{3}$$

где ω_0 и ω – это собственная частота рассеивателя и частота внешнего поля, γ – скорость затухания диполя рассеивателя, α – поляризуемость рассеивателя, ε – амплитуда внешнего поля, $E_{i,m}$ – поле, создаваемое m-тым рассеивателем в точке, где находится i-тый рассеиватель, c – это скорость света, $r_{i,m}$ – это расстояние между i-тым и m-тым рассеивателем.

$$\mathbf{E}_{i,m} = -\frac{\mathbf{D}_m(t)}{r_{i,m}^3} + 3\frac{U_{i,m}\mathbf{r}_{i,m}}{r_{i,m}^5} - \frac{\mathbf{v}_m(t)}{r_{i,m}^2}, \tag{4}$$

$$\mathbf{r}_{i,m} = \mathbf{r}_m - \mathbf{r}_i, \qquad \mathbf{v}(t) = \frac{1}{c}\frac{d\mathbf{D}_m}{dt}, \qquad U_{i,m} = \left(\mathbf{r}_{i,m}\cdot\mathbf{D}_m\right) \tag{5}$$

Начальные условия

$$\mathbf{D}_i(t) = \{0,0\}, \qquad -\frac{r_{i,m}}{c} \le t \le 0, \tag{6}$$

$$\frac{d\mathbf{D}_i(t)}{dt} = \{0,0\}, \qquad -\frac{x_{i,m}}{c} \le t \le 0 \tag{7}$$

Для оценки характеристик системы используется функция мощности рассеянного света в единицу телесного угла $I_n(\theta,\varphi)$, которая может быть вычислена следующим образом:

$$I_n(\theta,\varphi)=\lim_{\tau\to\infty}\int_t^{t+T}\left(\sum_{i=1}^{N}\mathbf{Y}_i\left(t-\frac{y_i'}{c}\right)\right)^2 dt \tag{8}$$

$$y_i'=\left(y_i\cos\varphi-x_i\sin\varphi\right)\sin\theta \tag{9}$$

$$\mathbf{Y}_i=\mathbf{M}_i+\mathbf{e}_z a_{s,i}\sin\theta\cos\theta \tag{10}$$

$$\mathbf{M}_i=\mathbf{a}_i\left(v_{s,i}\sin\theta-1\right)+\mathbf{s}a_{s,i}\sin^2\theta-\mathbf{v}_i a_{s,i}\sin\theta \tag{11}$$

$$\mathbf{s}=\mathbf{e}_x\cos(\varphi)+\mathbf{e}_y\sin(\varphi) \tag{12}$$

$$a_{s,i}=\left(\mathbf{s}\cdot\mathbf{a}_i\right),\qquad v_{s,i}=\left(\mathbf{s}\cdot\mathbf{v}_i\right),\qquad \mathbf{a}_i=\frac{d^2\mathbf{D}_i}{dt^2} \tag{13}$$

Для наблюдения за влиянием дискретного характера ансамбля рассеивателей необходимо построить зависимость усреднённой по ансамблю функции $\tilde{I}(n)$ от числа рассеивателей:

$$\tilde{I}(n)=\left\langle I_n(\theta,\varphi)\right\rangle=\frac{1}{n}\sum_{j=1}^{N}I_n(\theta,\varphi)\Big|_j \tag{14}$$

Так как в (8) требуется использовать установившееся решение системы (1), то в работе реализован метод нахождения квазистационарного решения, минуя непосредственное интегрирование системы дифференциальных уравнений с запаздывающим аргументом (1), обладающей свойством жёсткости [4].

Численное моделирование проводилось при следующих численных значениях параметров: $\omega_0=\omega=10^{15}c^{-1}$, $\gamma=10^7 c$, $c=3\cdot10^8 м/с$, $\varepsilon=1$, $\alpha=1$.

В работе приводятся результаты применения разработанной программы для вычисления $I_n(\theta,\varphi)$ и $\tilde{I}(n)$.

Разработка программы осуществлена на языке программирования C++ с использованием среды программирования Clion 2023.2.2.

ЛИТЕРАТУРА

1. Шалгуева С.Л., Воскобойников С.П., Попов Е.Н. Разработка программы для моделирования взаимодействия электромагнитных рассеивателей. В сборнике: Современные технологии в теории и практике программирования. Сборник материалов научно-практической конференции. Санкт-Петербург, 2021. С. 181-182.
2. Мейник А.В., Воскобойников С.П., Попов Е.Н. Разработка программы для моделирования взаимодействия цепочки диполей. В сборнике: Современные технологии в теории и практике программирования. Сборник материалов научно-практической конференции студентов, аспирантов и молодых ученых. Санкт-Петербург, 2023. С. 155-157.
3. Мышкис А.Д. Линейные дифференциальные уравнения с запаздывающим аргументом. Изд. 3, стереот. URSS. 2014. 360 с.
4. Ракитский Ю.В.,Устинов С.М., Черноруцкий И.Г. Численные методы решения жёстких систем. М., Наука. 1979. 199 с.

УДК 004.416

Л. А. Мясоедов (1 курс магистратуры),
В. В. Амосов, к.т.н., доцент

РАЗРАБОТКА ВЕБ-ПРИЛОЖЕНИЯ ДЛЯ АНАЛИЗА ЗДОРОВЬЯ СПОРТСМЕНОВ И СОСТАВЛЕНИЯ ПЛАНА ТРЕНИРОВОК

Целью разработки информационной системы для спортсменов является создание приложения, которое на основе изображений фигуры пользователя будет автоматически генерировать персонализированный план тренировок.

Функциональные требования:

1. Регистрация новых пользователей с возможностью создания персонального профиля.
2. Загрузка фотографий тела пользователем для анализа.
3. Анализ физических параметров на основе загруженных фотографий, включая оценку процента жира, массы мышц, обхватов и т.д.
4. Генерация персонализированного плана тренировок с учетом целей пользователя (например, набор массы, снижение жира, улучшение выносливости).
5. Предоставление рекомендаций по питанию и режиму отдыха на основе целей тренировок.
6. Возможность отслеживания прогресса выполнения тренировок и корректировка плана в соответствии с достигнутыми результатами.

Нефункциональные требования:

1. Безопасность данных пользователей в соответствии со стандартами GDPR и прочими законодательными требованиями.
2. Масштабируемость приложения для обработки данных большого объема пользователей.
3. Оптимизированное время отклика системы для быстрой генерации планов тренировок и анализа данных.
4. Высокая доступность системы для минимизации простоев и обеспечения непрерывного доступа пользователей.
5. Интуитивно понятный и удобный интерфейс для удобства использования приложения спортсменами всех уровней подготовки.

Для реализации данного приложения можно использовать методы машинного обучения и компьютерного зрения для анализа фотографий и генерации персонализированных тренировочных программ. Также важно провести тестирование системы на различных типах фигур и целях тренировок, чтобы обеспечить точность анализа и эффективность планов тренировок для всех пользователей.

В качестве языка программирования был выбран Java[6]. Java был создан как мультиплатформенный язык программирования с акцентом на простоту и читаемость кода.

В качестве среды разработки был выбран Intelij Idea.

В качестве системы управления базами данных (СУБД) была выбрана Postgresql [7]. Postgresql - это популярная СУБД, которая относится к категории SQL.

ЛИТЕРАТУРА

1. Horstmann C.S., Cornell G. Core Java Volume I--Fundamentals. Pearson Education, 2019.
2. Sierra K., Bates B. Head First Java. O'Reilly Media, 2020.
3. Lea D. Java Concurrency in Practice. Addison-Wesley Professional, 2006.
4. Evans M. PostgreSQL: Up and Running. O'Reilly Media, 2017.
5. Riggs R. PostgreSQL 13 Administration Cookbook: Over 150 Recipes for PostgreSQL 13 Administrators to Manage and Maintain High-Performance PostgreSQL Database. Packt Publishing, 2021.
6. Fehily C. SQL: Visual QuickStart Guide. Peachpit Press, 2008.

7. Fowler M. Patterns of Enterprise Application Architecture. Addison-Wesley Professional, 2003.
8. Johnson R., Hoeller J. Expert One-on-One J2EE Design and Development. Wrox Press, 2002.
9. Bloch J. Effective Java. Addison-Wesley Professional, 2017.
10. Kimball R., Ross M. The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling. Wiley, 2013.

УДК 004.415.2

О. Д. Меркадо (2 курс магистратуры),
И. В. Никифоров, к.т.н., доцент

ПОСТРОЕНИЕ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ ПРЕДИКТИВНОГО АНАЛИЗА ДАННЫХ ПО НЕДВИЖИМОСТИ С ИСПОЛЬЗОВАНИЕМ МАШИННОГО ОБУЧЕНИЯ

Введение. Постоянный рост стоимости жизни во всем мире привел к значительному увеличению стоимости недвижимости. Это явление было обусловлено растущим спросом и предложением на недвижимость, которая соответствует различным характеристикам и адаптирована к различным сегментам рынка. Однако эта динамика сделала прогнозирование роста стоимости недвижимости все более сложной задачей.

В этом контексте возникает необходимость в разработке автоматизированной системы прогнозного анализа данных о недвижимости. Эта система предназначена для использования инструментов анализа данных и машинного обучения для предоставления прогнозов стоимости свойств. Хотя существуют различные онлайн-платформы для публикации информации о недвижимости и предложения цен, ни одна из них не предоставляет полного анализа или прогноза стоимости недвижимости. Хотя cian.ru он имеет небольшую систему оценки недвижимости, но все же это не то, что может помочь арендодателю или продавцу недвижимости установить конкурентоспособную цену на недвижимость на основе статистических данных и проанализированных данных.

Для решения этой проблемы предлагается создать автоматизированную систему прогнозного анализа данных о недвижимости с использованием машинного обучения. В рамках этой системы были разработаны модули:

1. Сервисе на Python, с помощью которой будут собираться данные с веб-сайтов публикаций о недвижимости [6]. Для этого нам нужно будет одновременно иметь возможность синхронизировать задания каждого из подчиненных процессов. Для синхронизации будет использоваться распределенная система обмена сообщениями в реальном времени между серверными приложениями, в данном случае будут использоваться Kafka [5] и Zookeper из-за их высокой скорости, масштабируемости и отказоустойчивости. Для этого будет использоваться многопоточный и многопоточный подход [3].
2. Также был разработан модуль очистки и стандартизации данных, чтобы иметь возможность хранить данные в одном формате, что помогает лучше анализировать информацию [7]. Чтобы иметь возможность анализировать данные в режиме реального времени и, кроме того, иметь возможность часто обновляться, мы будем использовать процесс репликации данных с помощью MongoDB и Monstage.
3. Модуль, который отвечает за возможность обучения модели [2] с помощью машинное обучение. Эта модель будет учитывать различные характеристики, которые может предложить недвижимость [1].
4. Модуль визуализации данных, а также прогноз стоимости недвижимости на основе характеристик, которые пользователь может ввести.

Архитектура системы. Далее рассматривается архитектура, используемая для разработки решения. Следует отметить, что я использую подход с использованием контейнеров с помощью Docker [4, 8].

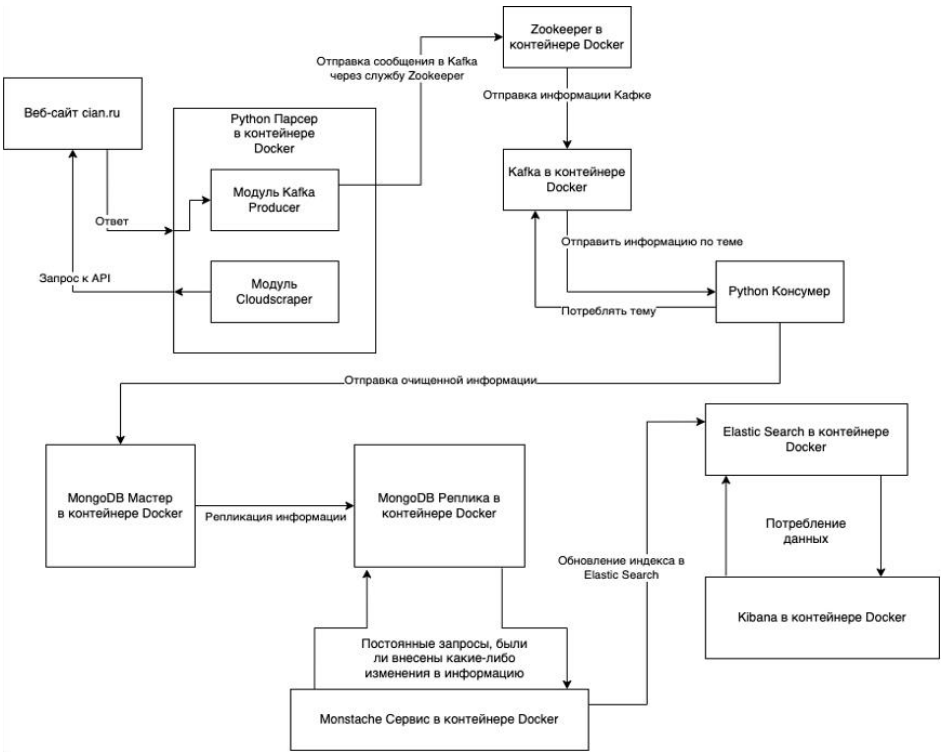


Рисунок 1 – Архитектура системы

В результате работы были разработаны вышеупомянутые модули, а также возможность обучения модели машинного обучения. Доказано, что существуют характеристики, которые имеют большее значение при определении цены недвижимости. Кроме того, мы можем оценить, что подход "ведущий-ведомый" в сервисах, отвечающих за сбор информации с сайтов размещения объявлений о недвижимости, имеет очень низкий уровень отказов.

ЛИТЕРАТУРА

1. Canaz Sevgen S., Tanrivermiş Y. Comparison of Machine Learning Algorithms for Mass Appraisal of Real Estate Data // Real Estate Management and Valuation. 2024. (32).
2. Cao P., He X. Machine Learning Solutions for Fast Real Estate Derivatives Pricing // Computational Economics. 2023. С. 1–30.
3. Leang B. [и др.]. Improvement of Kafka Streaming Using Partition and Multi-Threading in Big Data Environment // Sensors. 2019. (19). С. 134.
4. Özgüven Y., Gönener U., Eken S. A Dockerized big data architecture for sports analytics // Computer Science and Information Systems. 2022. (19). С. 10–10.
5. Park S., Huh J.-H. A Study on Big Data Collecting and Utilizing Smart Factory Based Grid Networking Big Data Using Apache Kafka // IEEE Access. 2023. (PP). С. 1–1.
6. Eyzenakh, D. S. High performance distributed web-scraper / D. S. Eyzenakh, A. S. Rameykov, I. V. Nikiforov // 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. – EDN SIPWXY.
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. – EDN XPRQXB.
8. Сафронов, Д. Автоматическая балансировка нагрузки между потоковой обработкой данных и внутренними задачами кластера с использованием Kubernetes / Д. Сафронов, К. М. Стоноженко, И. В. Никифоров // Современные технологии в теории и практике программирования : сборник материалов конференции, Санкт-Петербург, 23 апреля 2020 года / Санкт-Петербургский политехнический университет Петра Великого; Dell Technologies; EPAM Systems. – Санкт-Петербург: ПОЛИТЕХ-ПРЕСС, 2020. – С. 165-167. – EDN VOXGIM.

УДК 004.45

С. А. Мищенко (4 курс бакалавриата),
Ю. В. Литвинов, к.т.н., ст. преподаватель,
А. В. Корнилова, инженер-исследователь,
Д. С. Ярош, ведущий программист-разработчик

ОПТИМИЗАЦИЯ РАБОТЫ СЦЕНЫ В ПРИЛОЖЕНИИ SOLVE

На сегодняшний день все большее распространение получают автономные робототехнические системы, применяемые для выполнения разнообразных действий на местности. Наиболее важной задачей для такого рода устройств является задача определения положения робота в пространстве (локализация). При разработке подобных систем часто прибегают к использованию метода SLAM (Simultaneous Localization and Mapping) [1] — одновременная локализация и построение карты. Решения, использующие данный подход, как правило, состоят из двух компонентов: бэкенда и фронтенда. В зону ответственности бэкенда входит выделение ориентиров на полученных кадрах (детекция) и их сопоставление (ассоциация). Компонент фронтенда на основании полученных на предыдущем этапе данных занимается построением карты местности и локализацией на ней робота.

Качество работы SLAM-систем во многом связано с работой используемых алгоритмов детекции и ассоциации. По этой причине, часто возникает потребность в отладке отдельных компонентов решения. Для отладки подобных систем в большинстве случаев используется визуализация ориентиров и ассоциаций в реальном времени.

Существующие решения предоставляют такую возможность [2, 3], однако зачастую визуализация в них является неотделимой от остальной реализации составляющей, что создаёт проблемы при отладке отдельных компонентов фронтенда.

Для решения данной проблемы в Лаборатории мобильной робототехники разрабатывается приложение SOLVE [4], предназначенное для визуализации ориентиров в SLAM-алгоритмах. Основным компонентом данного инструмента является сцена, используемая для отображения кадров с размеченными на них ориентирами трёх типов: точек, линий и плоскостей. В разработке используется язык программирования Kotlin [5], фреймворк TornadoFX [6] и библиотека JavaFX [7]. Датасеты, используемые в разработке и тестировании SLAM-алгоритмов, могут состоять из сотен и даже тысяч кадров, каждый из которых может содержать несколько слоёв с ориентирами разных типов. По этой причине к производительности компонента сцены приложения предъявляются повышенные требования. Рассматриваемый инструмент активно используется сотрудниками Лаборатории мобильной робототехники при работе на проектах, связанных с SLAM-системами. Производительность первой версии приложения была рассчитана на работу с небольшими датасетами, содержащими несколько десятков кадров. По этой причине, на реальных наборах данных возникали проблемы с производительностью компонента сцены, что отрицательно сказывалось на работе всего инструмента, а также приводило к увеличению объема потребляемой оперативной памяти. Описанные проблемы, согласно результатам проведённой апробации, оказались критическими и отрицательно сказывались на опыте пользователей.

Таким образом, *целью* данной работы является оптимизация работы сцены приложения SOLVE.

Для достижения этой цели были поставлены следующие задачи:

1. Выполнить обзор разрабатываемого приложения и используемых в нём технологий.
2. Выполнить обзор решений для рендеринга в рамках JavaFX-приложения.
3. Разработать систему для отрисовки кадров, ориентиров и ассоциаций с применением выбранного решения.
4. Выполнить обзор методов измерения производительности сцены и сравнить реализации.
5. Провести апробацию приложения с оптимизированной сценой.

В ходе проведения обзора приложения было выяснено, что невозможно существенно улучшить производительность существующей реализации компонента сцены, поскольку имеющиеся проблемы связаны с неоптимальным выбором инструментов для реализации системы рендеринга кадров и ориентиров, а также с особенностями выбранной архитектуры.

В связи с этим, была решено начать разработку нового компонента сцены с нуля с учётом ошибок предыдущей реализации. В результате проведённого обзора решений для рендеринга, выбор был остановлен на библиотеках, позволяющих использовать мощности графических ускорителей для рендеринга содержимого сцены. В качестве библиотеки для реализации новой системы рендеринга была выбрана библиотека OpenGLFX [8]. Данное решение позволяет внедрить в приложение программный интерфейс OpenGL, предоставляющий доступ к графической аппаратуре.

Большая часть логики, связанной с рендерингом кадров, была перенесена в шейдерные программы, исполняемые на GPU, что положительно сказалось на скорости отрисовки содержимого сцены. Система отрисовки сцены была разделена на независимые компоненты, отвечающие за рендеринг кадров и ориентиров трёх типов: точек, линий и плоскостей. Благодаря такому разделению стало возможно использование дополнительных оптимизаций при отрисовке содержимого сцены.

На момент написания работы, *была реализована* новая версия компонента сцены приложения, позволяющая визуализировать кадры с размеченными на них ориентирами точек и линий. С целью сравнения производительности двух реализаций был сформирован тестовый стенд, в котором проводились замеры времени отрисовки кадров на проекте, содержащем 100 кадров с ориентирами. Усреднённые результаты 2000 замеров времени отрисовки кадров представлены в таблице 1. Как видно из результатов сравнения, новая реализация компонента сцены выигрывает в скорости отрисовки содержимого сцены и обладает стабильной частотой кадров, чего нельзя сказать о предыдущей реализации.

Таблица 1 – Результаты сравнения компонентов сцен

	Среднее время отрисовки кадра, мс
Предыдущая реализация компонента сцены	72 ± 45
Новая реализация компонента сцены	17.4 ± 2.5

ЛИТЕРАТУРА

1. Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age / Cesar Cadena, Luca Carlone, Henry Carrillo et al. // IEEE Transactions on Robotics. —2016. — Vol. 32, no. 6. — P. 1309–1332.
2. Cartographer repository. — URL: https://github.com/cartographer-project/cartographer (дата обращения: 2024-03-17).
3. ORB-SLAM2 repository. — URL: https://github.com/raulmur/ ORB_SLAM2 (дата обращения: 2024-03-17).
4. SOLVE repository. — URL: https://github.com/prime-slam/SOLVE/ (дата обращения: 2024-03-17).
5. Kotlin programming language official site. — URL: https://kotlinlang.org/ (дата обращения: 2024-03-17).
6. TorandoFX framework official site. — URL: https://tornadofx.io/ (дата обращения: 2024-03-17).
7. JavaFX library official site. — URL: https://openjfx.io/ (дата обращения: 2024-03-17).
8. OpenGLFX repository. — URL: https://github.com/husker-dev/openglfx/ (дата обращения: 2024-03-17).

УДК 004.4

Э. В. Мухамадиева (4 курс бакалавриата),
А. В. Петров, ст. преподаватель

РАЗРАБОТКА СЕРВЕРНОЙ ЧАСТИ ВЕБ-ПРИЛОЖЕНИЯ ДЛЯ МОНИТОРИНГА ЦЕН НА МАРКЕТПЛЕЙСАХ

За последние несколько лет маркетплейсы стали неотъемлемой частью жизни каждого человека. Людям больше нет необходимости тратить время на поездки до торговых центров с целью найти нужный товар, достаточно открыть сайт или приложение маркетплейса, ввести в поисковике запрос, и вот – предложения от различных продавцов перед глазами. Маркетплейсы обладают не только преимуществами, но и имеют некоторые проблемы. Зачастую продавцы на маркетплейсах завышают цены на товары перед началом распродаж и акций, либо же когда имеется высокий спрос на позицию. Бывают периоды, когда продавцы снижают цены, например, чтобы увеличить объем продаж или превзойти предложения конкурентов. Для того, чтобы быть в курсе падения цены или же появления товара в наличии, необходимо иметь инструмент, позволяющий оперативно собирать актуальную информацию с маркетплейсов и уведомлять об изменениях. Благодаря данному инструменту люди смогут разумно распределять личные финансы и делать более осознанные покупки.

На сегодняшний день существует несколько телеграм-ботов и веб-приложений, которые позволяют пользователям подписываться на определенные товары и получать уведомления об изменении их цен. Однако у них имеются ограничения в функционале, они зачастую разрешают бесплатно отслеживать только один товар, что очень мало для пользователя, также бот или приложение ограничиваются лишь одним маркетплейсом. Веб-приложение для мониторинга цен будет иметь несколько преимуществ по сравнению с существующими решениями. Во-первых, оно предоставит пользователю более широкий и гибкий функционал для отслеживания цен, возможность настройки уведомлений по различным параметрам (например, определенный процент снижения цены). Во-вторых, в приложении будет возможность добавлять товары сразу с нескольких маркетплейсов, таких, как Wildberries, Ozon и AliExpress.

Целью данной работы является разработка серверной части веб-приложения для мониторинга цен на маркетплейсах, пользователи которого смогут отслеживать изменения цен на интересующие товары и их наличие, определять оптимальное время для совершения покупок.

В качестве архитектурного стиля, который определяет правила обмена данными между клиентом и сервером, был выбран REST API [6], он использует HTTP в качестве протокола передачи данных для запросов и ответов. Серверная часть приложения реализована на языке программирования Python с использованием Django Rest Framework [1]. DRF – это библиотека, которая работает со стандартными моделями Django [3] для создания гибкого и мощного API для проекта. Django предоставляет мощный и удобный в использовании ORM (Object-Relational Mapping), что делает работу с базой данных удобной и эффективной. В качестве СУБД была выбрана PostgreSQL [5].

Архитектура Django REST API приложения представлена на рисунке 1. Принцип работы таков: сначала приходит запрос от клиента, который обрабатывается маршрутизатором фреймворка Django; далее, с поступившим запросом связано одно из представлений (views), которое реализует API-сайта. Задачей представления является обработка запроса и отправка результата пользователю. Так как для обработки необходимо сформировать некоторые данные, как правило, в формате JSON, то для этого управление передается специальному объекту – сериализатору. Именно они формируют данные для ответа на API-запросы, а также выполняют парсинг входной информации.

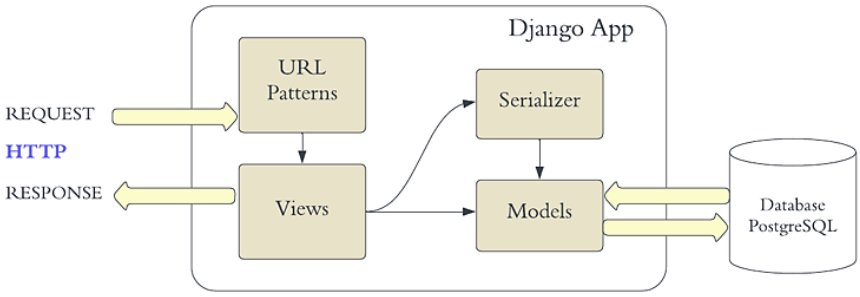


Рисунок 1 – Архитектура Django REST API приложения

Приложение для мониторинга цен на маркетплейсах подразумевает оповещение пользователей по электронной почте и планирование выполнения повторяющихся задач для отслеживания. Это затратные по времени задачи, которые являются частью рабочего процесса приложения. Для выполнения задач в фоновом режиме и гибкой настройки параметров для каждой задачи используется инструмент Celery [2]. Celery – это распределенная очередь задач, которая может собирать, записывать, планировать и выполнять задачи вне основной программы. Celery отслеживает задачи, которые необходимо выполнить, и в которой есть набор обработчиков (workers), которые будут выполнять эти задачи. Для хранения задач есть отдельный сервис, называемый брокером сообщений (message broker) Redis [4], который по сути своей является очередью. Схема работы Celery представлена на рисунке 2.

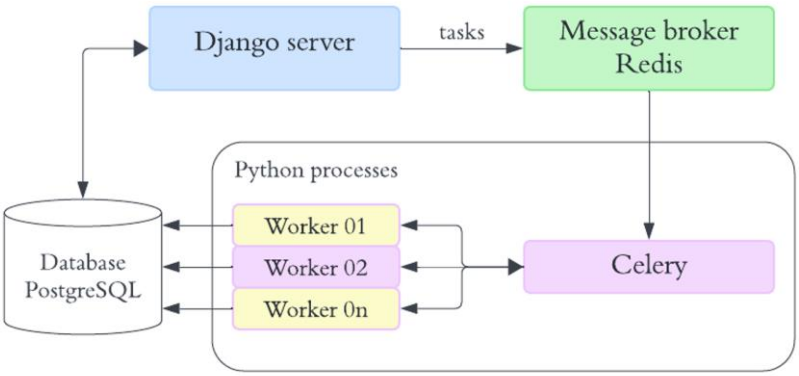


Рисунок 2 – Схема работы Celery с Django

Таким образом, описанная выше архитектура приложения позволит эффективно собирать данные с маркетплейсов, анализировать изменения цен и информировать пользователей об актуальных предложениях.

ЛИТЕРАТУРА

1. Django Rest Framework documentation. [Электронный ресурс] Режим доступа: https://www.django-rest-framework.org
2. Celery documentation. [Электронный ресурс] Режим доступа: https://docs.celeryq.dev/
3. Django documentation. [Электронный ресурс] Режим доступа: https://docs.djangoproject.com/en/5.0/
4. Redis documentation [Электронный ресурс] Режим доступа: https://redis.io/docs/
5. PostgreSQL documentation [Электронный ресурс] Режим доступа: https://www.postgresql.org/docs/
6. Meshram S. U. Evolution of Modern Web Services–REST API with its Architecture and Design //International Journal of Research in Engineering, Science and Management. – 2021. – Т. 4. – №. 7. – С. 83-86.

УДК 004.045

Мэн Цзянин (1 курс магистратуры),
О. А. Юсупова, ассистент

АНАЛИЗ ПОЛЬЗОВАТЕЛЕЙ ВИДЕО ПЛАТФОРМЫ BILIBILI С ИСПОЛЬЗОВАНИЕМ РАСПРЕДЕЛЕННЫХ ИНСТРУМЕНТОВ

Платформа bilibili — это видеохостинг, основанный в Китае в 2009 году, специализирующийся на контенте, связанном с ACG (аниме, комиксы, игры). Со временем он превратился в платформу, объединяющую обмен видео и взаимодействие пользователей и является одной из крупнейших видео платформ в Китае [1].

Учитывая уникальность пользователей bilibili и широкое влияние платформы, данное исследование направлено на глубокий анализ характеристик пользователей с помощью распределенных инструментов [2]. Анализ будет основан на следующих полях: уровень длительности использования, желание платить и других, чтобы изучить отношения между ними.

Структура проекта состоит из 5 модулей: сбор данных, предварительная обработка данных (очистка данных), распределенное хранение и обработка данных, анализ данных и визуализация данных [3]. Для этого на трех машинах установлены и настроены среды Python 3.9, MySQL 5.7, Hadoop 3.1.3, Hive 3.1.2 и Superset 2.1.2. Используемая архитектура программного средства и процесс передачи информации представлены на рисунке 1.

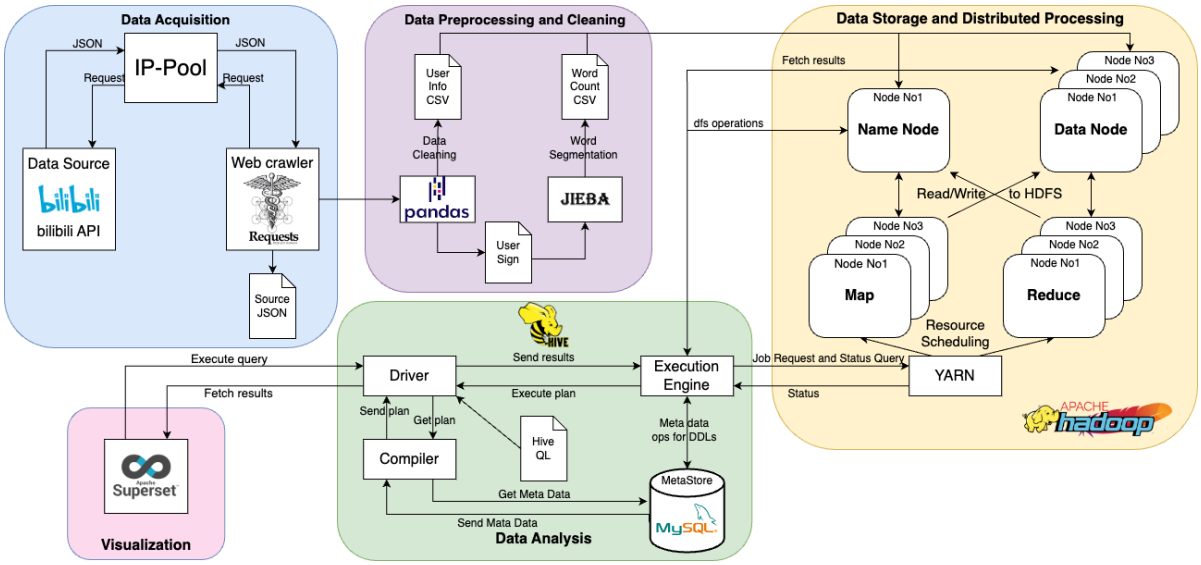


Рисунок 1 – Архитектура и поток обработки

1. Сначала мы используем Python-скрапер для отправки запросов к API bilibili через пул IP-адресов, и сервер возвращает данные JSON с общедоступной информацией пользователей.

2. С помощью инструмента pandas каждый JSON-ответ очищается и сохраняется в файл CSV. Учитывая отсутствие пробелов между китайскими символами, как в индоевропейских языках, личное описание пользователя обрабатывается с помощью китайского сегментатора Jieba и сохраняется в отдельный файл CSV для последующего анализа [4].

3. С использованием Hive CSV-файлы с информацией о пользователях и сегментированными личными описаниями импортируются в кластер с установленной системой HDFS на Hadoop, исходные данные хранятся в базе данных MySQL [5].

4. Через Hive мы используем HiveQL для прямого вызова параллельных вычислений Hadoop YARN и MapReduce для анализа данных, хранящихся в системе HDFS, и результаты анализа сохраняются обратно в систему HDFS в виде таблиц данных.

5. С помощью Apache Superset результаты анализа визуализируются

На этапе сбора данных использовалось 4 процесса, каждый из которых отправлял серверу 15–20 запросов в секунду через пул IP-адресов. За месяц сбора данных было собрано 100 миллионов пользовательских данных (ID пользователей от 1 до 100,000,000). После удаления данных об удаленных пользователях осталось 99,4 миллиона действительных пользовательских записей, которые занимали 13 ГБ места в формате CSV. При импорте данных в распределенную систему и выполнении запросов с помощью Hive время выполнения каждого запроса составляло от 5 до 25 секунд.

После анализа данных 99,4 миллионов пользователей мы обнаружили, что большинство пользователей сосредоточены на уровне Lv.0, который составляет 67.46%. С повышением уровня пользователя доля пользователей постепенно уменьшается, конкретное распределение следующее: Lv.1 - 2.24%, Lv.2 - 7.63%, Lv.3 -4.68%, Lv.4 -5.43%, Lv.5 -9.71%, Lv.6 - 2.85%.

В соответствии с рисунком 2 мы наблюдаем, что 10.02% пользователей когда-либо оформляли подписку на один месяц, а 5.99% пользователей — на один год и более. Кроме того, желание пользователей платить имеет явную положительную корреляцию с уровнем пользователя: на Lv.0 98.69% пользователей никогда не оформляли подписку, в то время как на максимальном Lv.6 только 4.99% пользователей никогда не оформляли подписку, 39.2% пользователей оформляли подписку на один месяц, а 55.81% пользователей - на более чем один год. Это также отражает, что как платформа, сосредоточенная на взаимодействии пользователей, приверженность пользователей bilibili напрямую влияет на их желание платить [6].

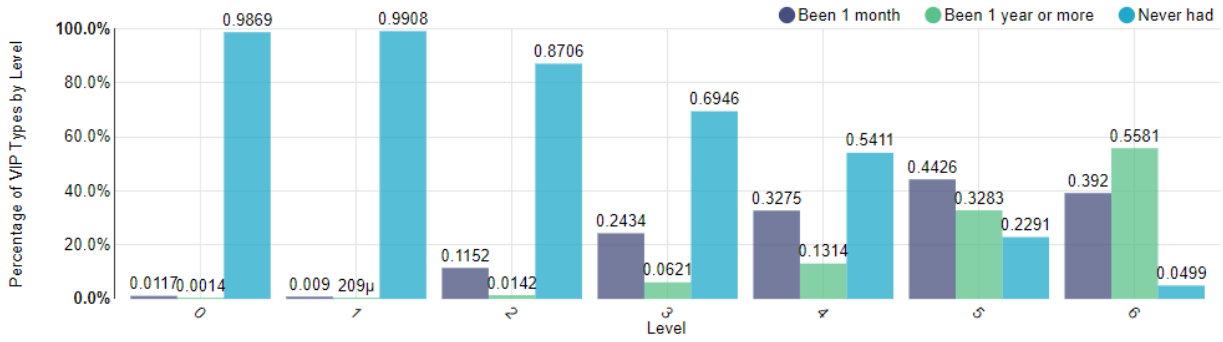


Рисунок 2 – Отношение между уровнем пользователя и желанием платить

В соответствии с рисунком 3 мы наблюдаем, что с повышением уровня пользователя доля мужчин среди пользователей с известным полом постепенно увеличивается. Особенно на уровне Lv.6 доля мужчин достигает 42.86%, что в 3.3 раза больше доли женщин. Это может быть связано с тем, что первоначальный контент bilibili в основном сосредоточен вокруг ACG, который более привлекателен для мужчин, поэтому ранние пользователи bilibili, вероятно, в основном были мужчины. Несмотря на то, что платформа продолжала привлекать больше женщин, доля мужчин с высоким уровнем пользователя сохраняется на определенном уровне.

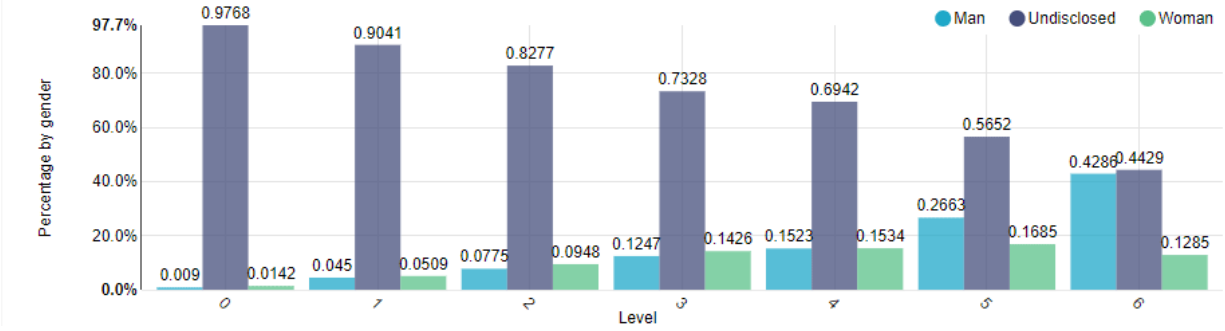


Рисунок 3 – Отношения между полом и уровнем пользователя

ЛИТЕРАТУРА

1. Bilibili - wikipedia. [Электронный ресурс] Режим доступа: https://en.wikipedia.org/wiki/Bilibili.
2. Zeng Xiaoqin. Implementation of Chinese Jieba Segmentation Technology Based on Python // Information and Computer, 2019, Pages 38-39.

3. Анализ данных трендов YouTube / В. К. Сычев, Ю. В. Ленькова, Е. А. Цветкова, И. В. Никифоров // Современные технологии в теории и практике программирования : сборник материалов конференции, Санкт-Петербург, 23 апреля 2020 года / Санкт-Петербургский политехнический университет Петра Великого; Dell Technologies; EPAM Systems. – Санкт-Петербург: ПОЛИТЕХ-ПРЕСС, 2020. – С. 131-132. – EDN GGZRXB.
4. 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. – EDN XPRQXB.
5. Peter Mccaffrey. Overview of big data tools: Hadoop, Spark, and Kafka // An Introduction to Healthcare Informatics, 2020, Pages 291-305 - https://doi.org/10.1016/B978-0-12-814915-7.00020-X.
6. Ke Rong, Fei Xiao. Platform strategies and user stickiness in the online video industry // Technological Forecasting and Social Change, Vol. 143, 2019, P. 249-259 - https://doi.org/10.1016/j.techfore.2019.01.023.

УДК 004.94

Г. В. Мироненков, В. А. Ивлев (3 курс аспирантуры),
Н. В. Воинов, к.т.н., доцент

АЛГОРИТМ ПОСТРОЕНИЯ НАВИГАЦИИ В N – МЕРНЫХ ЗАМКНУТЫХ ПРОСТРАНСТВАХ ДИНАМИЧЕСКИХ АДАПТИВНЫХ СРЕД

Многие динамические адаптивные среды (ДАС) [1], предоставляющие агентам возможность навигации, требуют эффективной реализации метода поиска кратчайшего пути. Данная задача требует решения двух подзадач: поиска оптимального пути и преобразование найденного пути в исполняемую агентами последовательность действий. Для простых ДАС, передвижение по которым реализовано через N-мерную сеть, поиск оптимального пути не составляет большой сложности, однако для геометрически сложных ДАС потребуются методы оптимизации пространства [2].

Особый интерес представляет навигация агентов в рамках ДАС, имитирующих ограниченные замкнутые пространства, такие как завалы [3]. Однако зачастую подобные ДАС представляют собой геометрически сложные пространства, требующие упрощения для построения графа навигации. В данной работе будут рассмотрены 2 существующих алгоритма оптимизации пространства: метод решётчатого деления и метод нормалей. Методы будут рассмотрены в двухмерном пространстве, ДАС будет представлена в виде неправильного многоугольника, агент будет изображён в виде круга.

Метод решётчатого деления [4] делит навигационное пространство N-мерной сеткой на квадраты, по которым происходит поиск и построение оптимального пути. Все квадраты, не содержащие в себе граней многоугольника, считаются доступными для перемещения. Данный алгоритм отличает высокая скорость работы и простота реализации, однако в случае некорректного подбора шага алгоритм может не допустить перемещение агента по узким местам (промежуток GC на рисунке 1).

Метод нормалей [5] заключается в построении N-мерных плоскостей параллельных границам геометрического пространства ДАС на расстоянии радиуса агента, образуя зону навигации. В отличие от метода решётчатого деления, скорость работы алгоритма напрямую зависит от сложности геометрического пространства ДАС, однако он позволяет построить навигацию через проблемные для аналога места (промежуток GC на рисунке 1).

Для построения навигационного пространства, покрывающего наибольшую площадь, необходимо улучшить метод построения нормалей путём добавления N-мерных на каждую вершину N-мерного многогранника. Таким образом, N-мерные плоскости, построенные в рамках метода нормалей, будут являться касательными к сферам, построенным от соседних вершин, что позволяет данному методу построить навигацию по местам, недоступным

предыдущим алгоритмам. К сожалению, скорость ниже скорости работы метода нормалей и напрямую зависит от количества вершин.

В данных тезисах был произведён краткий обзор методов навигации в пространстве и предложен собственный метод для решения задачи построения навигации в N-мерных замкнутых пространствах.

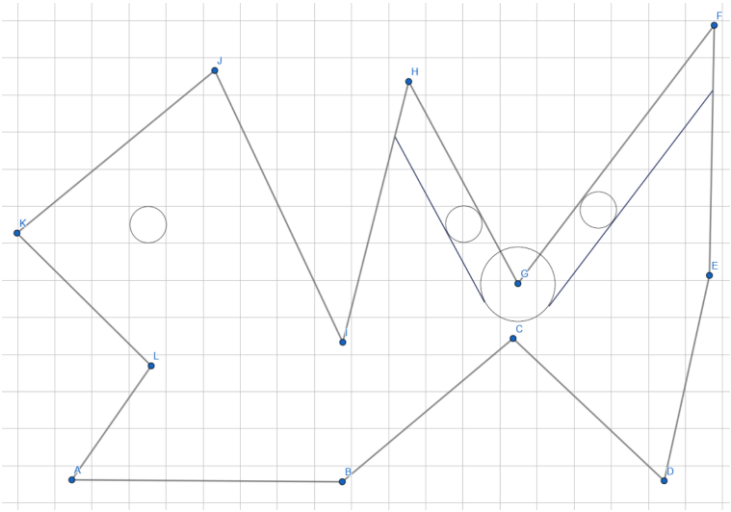


Рисунок 1 – Архитектура разрабатываемой системы

ЛИТЕРАТУРА

1. Ivlev, V. A. Automation method for configuring IT infrastructure for IT projects / V. A. Ivlev, I. V. Nikiforov, O. A. Yusupova // International Conference on Digital Transformation: Informatics, Economics,and Education (DTIEE2023) : Proceedings of SPIE - The International Society for Optical Engineering, Fergana, Uzbekistan, 20–22 марта 2023 года. – Washington: SPIE-SOC PHOTO-OPTICAL INSTRUMENTATION ENGINEERS, 2023. – P. 126370D. – DOI 10.1117/12.2680779. – EDN BYZHIK.
2. Y. Zhao, Z. Yang, C. Song and D. Xiong, "Vehicle dynamic model-based integrated navigation system for land vehicles," *2018 25th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS)*, St. Petersburg, Russia, 2018, pp. 1-4, doi: 10.23919/ICINS.2018.8405846.
3. J. Ren and X. Huang, "Dynamic Programming Inspired Global Optimal Path Planning for Mobile Robots," *2021 IEEE 4th International Conference on Information Systems and Computer Aided Education (ICISCAE)*, Dalian, China, 2021, pp. 461-465, doi: 10.1109/ICISCAE52414.2021.9590740.
4. M. Imran and F. Kunwar, "A hybrid path planning technique developed by integrating global and local path planner," *2016 International Conference on Intelligent Systems Engineering (ICISE)*, Islamabad, Pakistan, 2016, pp. 118-122, doi: 10.1109/INTELSE.2016.7475172.
5. H. Yuan, G. Zhang, Y. Li, K. Liu and J. Yu, "Research and implementation of intelligent vehicle path planning based on four-layer neural network," *2019 Chinese Automation Congress (CAC)*, Hangzhou, China, 2019, pp. 578-582, doi: 10.1109/CAC48633.2019.8997265.

УДК 62-50

Л. И. Нафикова (4 курс бакалавриата),
О. Н. Граничин, д.ф.-м.н., профессор

ПРОТОТИП СИСТЕМЫ ОПРЕДЕЛЕНИЯ ТРЕНДА ВО ВРЕМЕННЫХ РЯДАХ

Практически каждая современная кампания рано или поздно сталкивается с необходимостью анализировать данные, чтобы извлекать из них ценную информацию для увеличения прибыли и принятия ключевых решений, составления прогнозов. Однако иногда данных для анализа может не хватать. Например, в сфере медицины недостаточное количество данных о пациентах может затруднить разработку методов лечения. В сфере бизнеса отсутствие разнообразия данных о потребительском спросе или конкурентной среде может привести к неудачным маркетинговым стратегиям.

Недостаток данных может быть связан с несколькими причинами. Примерами являются дорогостоящий мониторинг, необходимость быстро принимать решения или невозможность проводить эксперименты из-за трудности моделирования ситуаций[1].

Для решения проблемы недостатка данных можно использовать различные стратегии. Например, компании могут сотрудничать с другими организациями для обмена данными или приобретения информации из внешних источников. Но это также чаще всего является дорогостоящим решением. Другой способ основывается на использовании методов анализа данных, которые позволяют работать с небольшим объемом информации. Например, методы машинного обучения позволяют создавать модели на основе ограниченных данных и делать прогнозы. Также можно использовать методы статистического анализа для извлечения ценной информации.

Важно помнить, что данные – это ключевой ресурс для принятия обоснованных решений. Таким образом, *целью* этой работы является разработка прототипа системы, которая будет способна предоставлять предсказания на основе имеющихся данных и определять в них тренд.

Для достижения этой цели были поставлены следующие *задачи*:

1. Обзор существующих решений.
2. Разработка архитектуры системы.
3. Реализация прототипа.
4. Апробация системы.

Важным шагом к разработке системы является составление *требований*. Система должна позволять гибко настраивать параметры алгоритмов и иметь возможность задавать различные модели шума. Также немаловажной частью является возможность получать данные из разных источников. Пользователь должен иметь возможность обрабатывать разные типы входных данных, представлять результаты в виде графиков, сохранять предсказания. Для достижения этого необходим удобный пользовательский интерфейс. Главная его цель – тестирование алгоритмов без погружения в детали реализации. Поэтому было решено создать веб-приложение, в котором пользователям будет удобно получать прогнозы и анализировать данные.

Языком реализации серверной части был выбран Python[4], потому что для него написано большое количество библиотек с реализациями основных алгоритмов для прогнозирования. Например, такими как фильтр Кальмана[2], метод знаковозмущённых сумм[3]. В качестве фреймворка для создания сервера был выбран Flask[6], потому что он более легковесный, прост в изучении. Клиентская часть реализуется на языке JavaScript[5] с помощью библиотеки для создания пользовательских интерфейсов React[7].

ЛИТЕРАТУРА

1. Марина Волкова. Рандомизированные алгоритмы оценивания параметров инкубационных процессов в условиях неопределённостей и конечного числа наблюдений: Диссертация/Волкова Марина ; СПбГУ. – 2018
2. Kalman Rudolph Emil. A New Approach to Linear Filtering and Prediction Problems // Transactions of the ASME – Journal of Basic Engineering. – 1960. – Vol. 82, no. Series D. – P. 35–45
3. Csáji Balázs, Campi Marco, Weyer Erik. Sign-Perturbed Sums: A New System Identification Approach for Constructing Exact Non-Asymptotic Confidence Regions in Linear Regression Model s // Signal Processing, IEEE Transactions on. – 2015. – 01. – Vol. 63. – P. 169–181
4. Python programming language. [Электронный ресурс] Режим доступа: https://www.python.org/doc/.
5. JavaScript programming language. [Электронный ресурс] Режим доступа: https://www.javascript.com/.
6. Flask web framework. [Электронный ресурс] Режим доступа: https://flask.palletsprojects.com/en/3.0.x/.
7. React library. [Электронный ресурс] Режим доступа: https://react.dev/.

УДК 004.453

А. М. Никитина (4 курс, бакалавриата),
А. Н. Терехов., д.ф.-м.н., профессор

РАЗРАБОТКА КОМПЛЕКСА ПРОГРАММ ДЛЯ ГРАФИЧЕСКОГО ПРОГРАММИРОВАНИЯ ОБОРУДОВАНИЯ 4G

Сети 4G существуют с 2009 года, то есть уже около пятнадцати лет [1], однако до недавнего времени настройкой сетей занимались только специально обученные люди, работающие у операторов сотовой связи. В последние несколько лет наметилась тенденция внедрения технологий 4G в самые разнообразные области, туда, где они раньше не использовались. Например, на промышленные предприятия или в транспортную отрасль [2].

Компании ищут способы внедрить новые технологии в работу, затратив при этом минимальное количество ресурсов, но далеко не на всех предприятиях имеются сотрудники, квалификация которых позволяет управлять сетями. Существует два решения данной проблемы.

Первое из них заключается в обучении сотрудников новым технологиям, однако это требует довольно больших затрат, как по финансам, так и по времени.

Вторым вариантом решения является упрощение самого процесса настройки сетевых систем. Если сделать управление сетями более понятным и доступным, им могут заняться даже люди, квалификация которых изначально не связана с сетями. Это поможет сэкономить ресурсы, которые могли быть затрачены на переквалификацию имеющихся кадров или поиск новых.

Таким образом появилась потребность в продукте, который позволит настраивать параметры сетей 4G людям без глубоких знаний сетевых технологий. Выбор был сделан именно в сторону визуального программирования [3] по причине того, что графические средства разработки являются наиболее понятными и доступными для людей, которые имеют минимальные знания о программировании. В данном случае графическая настройка сетей позволит не вникать в устройство самих конфигурационных файлов и оборудования.

Итак, *целью* данной работы является разработка продукта, который позволяется настраивать параметры оборудования сетей 4G посредством графического программирования.

Для достижения этой цели необходимо решение следующих *задач*:

1. Обзор предметной области.
2. Обзор возможных подходов к разработке комплекса программ для графической разработки сети.
3. Проведение анализа найденных подходов.
4. Обзор графических редакторов, которые могут быть взяты за основу в реализации инструмента.
5. Реализация инструмента на основе выбранного редактора.

В качестве способа разработки инструмента для графического программирования оборудования 4G был выбран подход, при котором предполагается использование стороннего графического редактора в качестве основы для конструирования диаграмм сети. После проведенного анализа и разработки требований для инструментов редактирования диаграмм, был выбран редактор Draw.io [4]. На рисунке 1 представлен пример диаграммы сети LTE, составленной в данном редакторе.

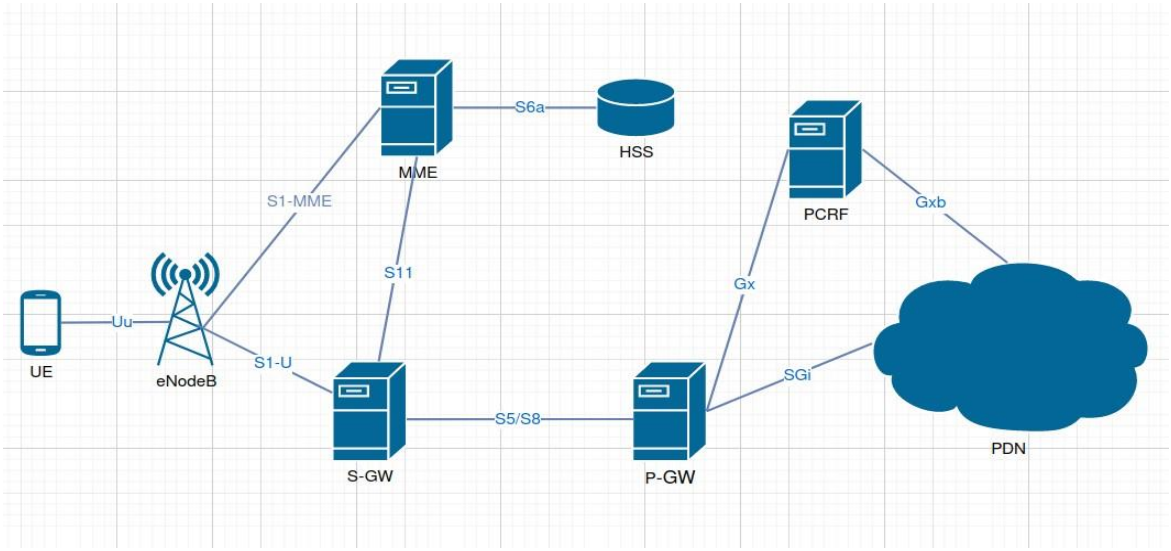


Рисунок 1 – Пример диаграммы сети в графическом редакторе Draw.io

Изначальное взаимодействие с пользователем происходит через десктопное приложение, которое позволяет выбрать директорию, в которой хранятся конфигурационные файлы, полученные с оборудования. Далее конфигурационные файлы автоматически конвертируются и открываются для пользователя в виде диаграмм в Draw.io. При внесении изменений в диаграмму происходит обновление параметров конфигурационных файлов, после чего они отправляются обратно на оборудование.

Если же изменения в конфигурационных файлах произошли напрямую, без вмешательства пользователя через графический редактор, все измененные параметры также подгружаются в приложение, таким образом реализуется обратная связь с оборудованием.

Инструмент *был реализован* с помощью языка программирования Python [5], в котором представлено большое количество библиотек для обработки разных форматов конфигурационных файлов, а также диаграмм в виде XML. Для разработки графического интерфейса использовалась библиотека PyQT [6].

ЛИТЕРАТУРА

1. M. Hicham, N. Abghour, M. Ouzzif. 4G system: Network Architecture and Performance // International Journal of Innovatice Research in Advanced Engineering – 2015 – Vol. 2 – No. 4. – P. 215–220
2. Krishnamurthy Raghunandan. Introduction to Wireless Communications and Networks – 2017
3. Д.В. Кознов. Основы визуального моделирования. — М: ИнтернетУниверситет Информационных Технологий; БИНОМ. Лаборатория знаний, 2007.
4. Draw.io редактор диаграмм. [Электронный ресурс] Режим доступа: https://www.drawio.com/
5. Python programming language. [Электронный ресурс] Режим доступа: https://www.python.org/doc/.
6. PyQT. [Электронный ресурс] Режим доступа: https://doc.qt.io/qtforpython-6/

УДК 004.42

А. А. Осипов (4 курс бакалавриата),
Т. В. Коликова, ст. преподаватель

РАЗРАБОТКА ВЕБ ПРИЛОЖЕНИЯ ДЛЯ АВТОМАТИЗАЦИИ УПРАВЛЕНИЯ ОТЕЛЕМ: ФУНКЦИОНАЛЬНЫЕ ВОЗМОЖНОСТИ, ТЕХНОЛОГИИ И МЕТОДЫ РЕАЛИЗАЦИИ

В настоящее время автоматизация управления отелями становится все более важной задачей в гостиничной индустрии. Развитие информационных технологий и интернета приводит к необходимости создания эффективных веб-приложений, способных обеспечить удобное и оперативное управление всеми аспектами работы отеля. В данной научной работе рассматриваются функциональные возможности, технологии и методы реализации веб-

приложения для автоматизации управления отелем. Анализируются основные требования к такому приложению, рассматриваются современные технологии разработки веб-приложений и методы их реализации. Представляются практические примеры использования веб-приложения для управления отелем и оцениваются потенциальные выгоды от его внедрения.

Актуальность темы автоматизации управления отелями с помощью веб-приложений обусловлена рядом факторов:

1. Рост конкуренции: В гостиничной индустрии наблюдается увеличение конкуренции, и отели стремятся повысить эффективность своей деятельности, включая улучшение процессов управления.

2. Требования гостей: Современные гости отелей ожидают быстрого и удобного сервиса, который можно обеспечить с помощью автоматизации и веб-приложений.

3. Эффективность и экономия ресурсов: Автоматизация управления отелем позволяет оптимизировать рабочие процессы, сократить временные затраты и уменьшить вероятность ошибок.

4. Возможность персонализации: Веб-приложения позволяют отелям предлагать персонализированные услуги для каждого гостя, что способствует улучшению качества обслуживания.

5. Развитие технологий: С появлением новых технологий, таких как облачные вычисления, интернет вещей и искусственный интеллект, становится возможным создание более инновационных и эффективных систем управления отелями.

Для разработки веб-приложения для автоматизации управления отелем были выбраны следующие технологии:

1. Java: Java является отличным выбором для создания веб-приложений благодаря своей платформенной независимости, высокой производительности и широкому набору инструментов и библиотек. В данном случае, Java может быть использована для написания бэкенд-части приложения, где будет обрабатываться бизнес-логика, взаимодействие с базой данных и другие задачи.

2. Spring Framework: Spring Framework предоставляет мощный инструментарий для создания веб-приложений. Он содержит множество модулей, которые упрощают разработку, такие как Spring Boot для быстрого создания приложений, Spring MVC для обработки HTTP-запросов, Spring Data JPA для работы с базой данных и другие. Spring Framework также поддерживает принципы инверсии управления (IoC) и внедрения зависимостей (DI), что способствует созданию гибкого и расширяемого кода.

3. PostgreSQL: Для хранения данных об отелях, номерах, бронированиях и других сущностях можно использовать PostgreSQL - мощную реляционную базу данных с отличной производительностью и надежностью. PostgreSQL поддерживает широкий спектр функциональности, включая транзакции, хранимые процедуры, триггеры и многое другое, что делает его отличным выбором для приложений с высокими требованиями к данным.

4. Hibernate: Hibernate - это ORM (Object-Relational Mapping) фреймворк, который позволяет работать с данными в базе данных с использованием объектно-ориентированного подхода. Hibernate упрощает взаимодействие с базой данных, позволяя разработчикам оперировать объектами Java вместо SQL-запросов. Это повышает производительность разработки и обеспечивает более чистый и понятный код.

5. HTTP: Протокол HTTP играет ключевую роль в веб-приложениях, поскольку он используется для передачи запросов и ответов между клиентом (браузером пользователя) и сервером. В контексте управления отелем, HTTP будет использоваться для отправки запросов на сервер для получения информации о бронированиях, изменении статусов номеров и других операций.

Использование указанных технологий позволит создать эффективное и надежное веб-приложение для автоматизации управления отелем, которое будет обладать высокой производительностью, легкостью расширения и удобством использования для конечных пользователей.

Архитектура веб-приложения для автоматизации управления отелем на Java с использованием Spring Framework может быть построена на основе трехслойной архитектуры: представления (View), бизнес-логики (Service) и доступа к данным (Data Access). Давайте рассмотрим каждый слой подробнее:

Представление (View): - В этом слое находится пользовательский интерфейс, через который пользователи будут взаимодействовать с приложением. Для разработки пользовательского интерфейса можно использовать шаблонизаторы, такие как Thymeleaf или FreeMarker, которые позволяют создавать динамические страницы, взаимодействующие с бэкендом. Контроллеры (Controllers) Spring MVC используются для обработки HTTP-запросов, вызова соответствующих сервисов и передачи данных между представлением и бизнес-логикой.

Бизнес-логика (Service): этом слое содержится вся бизнес-логика приложения, такая как обработка бронирований, управление номерами отеля, расчет стоимости проживания и другие операции.

Доступ к данным (Data Access): В этом слое происходит взаимодействие с базой данных для сохранения и получения данных

ЛИТЕРАТУРА

1. Шапиро Д. Spring в действии // ДМК Пресс, 2017.
2. Каргин А. Java. Эффективное программирование // Питер, 2019.
3. Вершигора В. Hibernate для начинающих // БХВ-Петербург, 2021.
4. Фаулер М. Проектирование и рефакторинг: паттерны объектно-ориентированного проектирования // Питер, 2018.
5. Калленберг Д. PostgreSQL: вводный курс // ДМК Пресс, 2020.
6. Фримен Э., Сиера К. Паттерны проектирования // Питер, 2016.
7. Филдинг Р., Тейлор Р. Протокол передачи гипертекста HTTP/1.1 // Издательство Символ-Плюс, 2018.

УДК 004.4

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

СОЗДАНИЕ РАСПРЕДЕЛЕННОЙ SIEM СИСТЕМЫ СБОРА И АНАЛИЗА ЛОГОВ ДЛЯ СЕТИ КОМПЬЮТЕРОВ ОРГАНИЗАЦИИ

В современную цифровую эпоху организации все больше полагаются на компьютерные сети для выполнения своих повседневных операций. В результате объем данных, генерируемый компаниями, растет экспоненциальными темпами. Для решения возникшей проблемы многие компании внедряют SIEM системы для управления журналами. Такие комплексы программ обеспечивают централизованную платформу для сбора, анализа и корреляции данных из различных источников по всей сети организации. Результатом является улучшенная видимость сетевой активности, максимально точное обнаружение угроз и более быстрое реагирование на инциденты.

Существует три основных типа систем безопасности [1]:

- Системы обнаружения и предотвращения вторжений (IDPS)
- Управление информацией и событиями безопасности (SIEM)
- Аналитика больших данных [2]

На практике три типа систем безопасности часто объединяются в единую SIEM-систему для обеспечения защиты от угроз и анализа событий. Современные SIEM-системы модульные, позволяя выбирать компоненты. Готовые решения "из коробки" популярны благодаря простоте настройки и скорости внедрения, но их недостатком является высокая стоимость. Поэтому будем рассматривать в первую очередь продукты с открытым исходным кодом [3].

Цель работы – найти и построить распределенную систему безопасности, выполняющую функции системы сбора и анализа логов с возможностью быстрого внедрения и улучшения функциональности системы в дальнейшем. Необходимые шаги для достижения цели: 1. Исследовать и сравнить доступные open-source SIEM системы. 2. Внедрить и настроить выбранную SIEM систему. 3. Разработать и внедрить Android-приложение для мониторинга данных мобильного устройства.

В ходе исследования были рассмотрены шесть SIEM систем: Grafana Loki, GrayLog, Suricata, Wazuh, Prelude OSS, AlienVault OSSIM.

Grafana Loki, GrayLog часто ошибочно относят к SIEM-системам, хотя они предоставляют только частичный функционал SIEM. Они удобны в качестве вспомогательного инструмента для предварительной обработки информации перед отправкой в SIEM-систему.

Suricata - это система обнаружения вторжений в сеть (NIDS), анализирующая трафик и генерирующая предупреждения о угрозах. Она обладает способностью к анализу протоколов, основана на правилах и способна обнаруживать аномалии в трафике, обеспечивая высокую производительность и поддержку различных форматов вывода.

Prelude OSS, AlienVault OSSIM имеют немалое количество недостатков: Prelude OSS ограничен в функциональности и рекомендован для тестирования в малых средах, а AlienVault OSSIM не обновлялся с 2017 года и не имеет официальной документации.

Остановимся на детальном рассмотрении именно SIEM системы Wazuh, потому что она обладает наибольшей функциональностью, полной документацией, а также возможностью подключения Suricata – получения, обработки и визуализации логов с данной NIDS системы.

Wazuh — это HIDS-система с открытым исходным кодом, которая включает в себя следующие возможности для обнаружения вторжений и обеспечения безопасности: аналитика безопасности, система обнаружения вторжений, анализатор журналов, файловый мониторинг, обнаружение уязвимостей, реакция на инциденты безопасности.

Архитектура Wazuh включает в себя базу данных OpenSearch, Wazuh agent для сбора логов, реагирования на инциденты и Wazuh server для анализа данных, поступивших от агентов, а также оповещения об угрозах [4].

Распределенная тестовая система, которую мы реализуем, состоит из трех виртуальных машин, отвечающих за обработку информации и двух подконтрольных систем, на которые установлены агенты для сбора информации. Помимо стандартной функциональности была добавлена аутентификация отправителя по паролю с соответствующем распределением логов по различным индексам в зависимости от отправителя.

SIEM-система состоит из мониторинговой части и контролируемых компонентов. Контролируемые системы включают мобильное приложение WazuhAndroid и агенты Wazuh для сбора данных с использованием Filebeat для передачи логов на центральный узел VM-2 на ПК Windows и Linux.

К сожалению, в настоящее время нет приложения Wazuh для ОС Android, которое может передавать логи на сервер, что связано с ограниченными логами для непривилегированных пользователей [5]. Однако сбор информации из Logcat требуется, так как среди используемых устройств организации смартфоны Android с пользовательским приложением для работы с гео-данными.

Поэтому было разработано приложение, способное не только передавать логи на сервер, но и обеспечивать дополнительную функционал защиты устройства. Таким образом, приложение включает в себя два модуля: сбор логов от других приложений и встроенную антивирусную защиту.

Таким образом, в рамках данной работы был проведен всесторонний анализ различных открытых SIEM-технологий и реализована собственная SIEM-архитектура с использованием оптимальных компонентов, включая Wazuh в качестве основного SIEM-движка благодаря его широким возможностям в области безопасности. Также была выполнена кастомизация Logstash для обеспечения безопасного подключения агентов и успешно реализована

архитектура с мониторинговыми системами, центральным узлом и управляющим узлом. Помимо этого, был разработан собственный Android-антивирус для сбора и пересылки логов, обеспечивающий дополнительную защиту устройств.

ЛИТЕРАТУРА

1. Sekharan S. S., Kandasamy K. Profiling SIEM tools and correlation engines for security analytics //2017 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET). – IEEE, 2017. – С. 717-721.
2. Voinov N. et al. Big data processing system for analysis of GitHub events //2019 XXII International Conference on Soft Computing and Measurements (SCM)). – IEEE, 2019. – С. 187-190.
3. González-Granadillo G., González-Zarzosa S., Diaz R. Security information and event management (SIEM): analysis, trends, and usage in critical infrastructures //Sensors. – 2021. – Т. 21. – №. 14. – С. 4759.
4. Sheeraz M. et al. Effective security monitoring using efficient SIEM architecture //Hum.-Centric Comput. Inf. Sci. – 2023. – Т. 13. – С. 1-18.
5. Cinque M., Cotroneo D., Pecchia A. Challenges and directions in security information and event management (SIEM) //2018 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW). – IEEE, 2018. – С. 95-99.

УДК 004.415.2

А. М. Пантюхин (2 курс магистратуры),
С. А. Молодяков, д.т.н., профессор

РЕАЛИЗАЦИЯ ПРОГРАММНОЙ СИСТЕМЫ АНАЛИЗА ДАННЫХ НА ОСНОВЕ ELK-СТЕКА

В эпоху, когда объемы данных растут с невероятной скоростью, их сбор и анализ становятся важными для достижения успеха в различных сферах бизнеса. В данной работе рассматривается разработка компонента программной системы анализа данных, задача которого — обеспечить эффективную обработку, хранение и визуализацию информации с интернет-ресурсов.

Применение ELK-стека, объединяющего Elasticsearch для анализа и поиска данных, Logstash для их сбора и агрегации, а также Kibana для визуализации, делает возможной обработку больших объемов данных в реальном времени. Подобные системы предоставляют компаниям возможность принимать обоснованные решения, оперативно реагировать на меняющиеся условия рынка и повышать общую эффективность деятельности.

При оценке существующего рынка систем анализа данных выделяются несколько ведущих платформ бизнес-аналитики. Прежде всего, стоит упомянуть Tableau, мощную систему визуализации данных, предлагающую широкие возможности для анализа и представления данных. Она привлекает пользователей интуитивным интерфейсом и глубокими аналитическими возможностями. Ещё одним конкурентом является Microsoft Power BI, платформа, интегрированная с множеством сервисов Microsoft, что делает её популярной среди компаний, использующих продукты Microsoft Office.

Каждая из этих платформ обладает своими уникальными особенностями и сильными сторонами, но рассматриваемая в данной работе система выделяется гибкостью конфигурации и способностью к масштабированию. В отличие от многих BI-решений, которые часто требуют значительных начальных инвестиций, наша система позволит пользователям собирать, анализировать и визуализировать данные, используя открытые источники и инструменты. Кроме того, наш подход к интеграции модулей машинного обучения и NLP-алгоритмов предоставляет дополнительную ценность, позволяя более тонко анализировать и интерпретировать данные.

Для тестирования разрабатываемой системы был выбран ресурс с объявлениями о продаже автомобилей Авито Авто. Сбор и анализ таких данных, как объявления о продаже

автомобилей может быть полезен для компаний, анализирующих вторичный рынок, например занимающихся подбором б/у автомобилей.

Стоит учитывать, что подобные системы имеют гораздо более широкую область применения. Необходимо сделать систему в достаточной степени конфигурируемой, гибкой и расширяемой, чтобы иметь возможность проводить анализ большого объема данных об объявлениях определенного типа на разных ресурсах, с использованием заданной пользователем модели данных об объявлениях. Нужно реализовать возможность конфигурирования системы для анализа не только автомобильного рынка, но и объявлений о продаже других товаров, которые содержат в том или ином виде текстовое описание.

Таким образом, можно выделить следующие задачи:

- Рассмотреть существующие BI-платформы и системы анализа данных

- Предложить новую архитектуру программной системы, описать ее отличия от сухого применения ELK-стека

- Реализовать платформу в программном средстве, внутри которой будет одним из модулей - модуль коннектора

- Продемонстрировать работоспособность системы на примере анализа данных с Авито

Компоненты разрабатываемой системы представлены на рисунке 1. Ключевой особенностью является её многоуровневая структура, позволяющая разделять процессы сбора, обработки, хранения и визуализации данных. Это не только улучшает управление данными, но и повышает отказоустойчивость системы, так как каждый компонент может быть оптимизирован и масштабирован независимо от других.

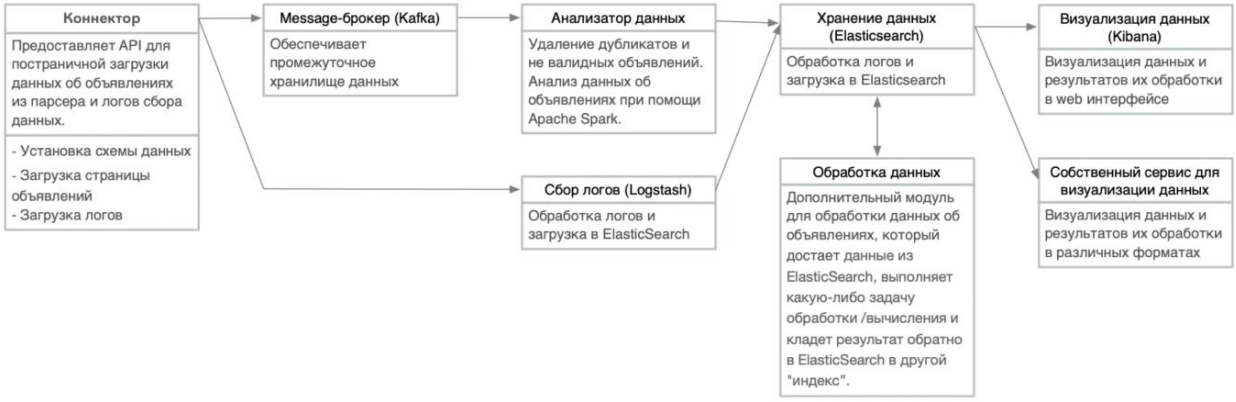


Рисунок 1 – Компоненты системы

В архитектуре системы предусмотрен специализированный модуль обработки данных, который извлекает информацию из ElasticSearch, проводит необходимую операцию обработки или вычисления — будь то применение методов машинного обучения или NLP-алгоритмов — и затем сохраняет результаты в новый 'индекс' в ElasticSearch. Этот подход обеспечивает высокую гибкость системы, позволяя ей адаптироваться к разнообразным задачам обработки данных.

Также один из главных вопросов, возникающих при работе с ELK-стеком, касается возможности интеграции собственных модулей или плагинов в его состав. Интерес к расширяемости ELK-стека через добавление пользовательских модулей, например, для реализации уникальных методов визуализации в Kibana, подчеркивает необходимость в глубоком понимании его модульной системы и возможностей для настройки и адаптации под специфические потребности проекта.

В целом, спроектированная система является отражением лучших практик в области анализа данных и представляет собой устойчивое решение, которое может быть адаптировано под различные бизнес-задачи и масштабировано в соответствии с растущими потребностями предприятия.

ЛИТЕРАТУРА

1. Elastic. Logstash Reference. [Электронный ресурс] Режим доступа: https://www.elastic.co/guide/en/logstash/current/index.html
2. The Apache Software Foundation. Apache Kafka. [Электронный ресурс] Режим доступа: https://kafka.apache.org/24/documentation.html
3. Docker docs. [Электронный ресурс] Режим доступа: https://docs.docker.com
4. Docker Compose overview. [Электронный ресурс] Режим доступа: https://docs.docker.com/compose/
5. GitHub Actions documentation. [Электронный ресурс] Режим доступа: https://docs.github.com/en/actions/

УДК 004.942

М. И. Попов (4 курс бакалавриата),
Ю. Б. Сениченков, д.т.н., профессор

ГЕНЕРАЦИЯ ЛИНИЙ ОТКЛИКА ПО ПЭТ-ИЗОБРАЖЕНИЮ ДЛЯ МОДЕЛИРОВАНИЯ ЛУЧЕВОЙ ТЕРАПИИ ПОД БИОЛОГИЧЕСКИМ КОНТРОЛЕМ

Лучевая терапия представляет собой высокоэффективный метод лечения онкологических заболеваний, базирующийся на точной доставке дозы облучения в определенную область человеческого тела. Однако, несмотря на значительные успехи в этой области, существует фундаментальная проблема – методики лучевой терапии, принятые в настоящее время, основаны на предположении о статичности опухоли. Это предположение играет критическую роль в точности процедуры, и любое движение пациента, будь то дыхание или случайные телесные движения, может привести к существенным погрешностям в доставке облучения.

В основе современных разработок в области лучевой терапии лежит стремление к максимальной точности и эффективности в доставке облучения опухолевой ткани. В этом контексте, ключевую роль играет подход лучевой терапии под биологическим контролем, предполагающий использование аппаратуры с обратной связью [1]. Этот подход предоставляет возможность активного реагирования на динамические изменения внутренней структуры пациента в режиме реального времени, что повышает точность и предсказуемость процедуры.

Подобный аппарат лучевой терапии активно мониторит и анализирует биологические параметры организма пациента в процессе облучения. Одним из ключевых элементов этого подхода является использование позитронно-эмиссионной томографии (ПЭТ) в качестве "биологического ориентира" [2]. ПЭТ предоставляет детальную информацию о метаболизме тканей, что делает его ценным инструментом для наблюдения за изменениями внутри организма пациента в реальном времени [3].

Однако, несмотря на потенциальные преимущества данного метода, возникает проблема, связанная с продолжительным временем, требующимся для получения высококачественных ПЭТ-изображений [4]. Решение этой проблемы находится на поверхности – необходимо использовать быстро получаемые "частичные изображения" [2]. Это позволяет учесть динамику изменений в организме пациента и, таким образом, активно корректировать направление луча в режиме реального времени.

Разработка и внедрение лучевой терапии под биологическим контролем представляет собой важный шаг в улучшении результатов лечения онкологических заболеваний. Этот подход не только повышает точность доставки облучения, но и открывает новые перспективы для персонализированного и эффективного лечения пациентов.

Процесс тестирования аппарата лучевой терапии, изображенный на рисунке 1, требует наличия модели пациента. В настоящее время для этих целей активно применяются фантомы – контейнеры, имитирующие форму конкретной части человеческого тела, оснащенные

цилиндрами внутри. Заполненный различными жидкостями, фантом подвергается сканированию.

Однако, несмотря на эффективность использования фантомов, существует проблема – тестирование приборов становится сложным процессом из-за зависимости лучей от зашумленных данных ПЭТ. Точность результатов тестирования напрямую зависит от конкретной реализации вероятностного распределения линий отклика ПЭТ, что создает дополнительные сложности при анализе и интерпретации данных.

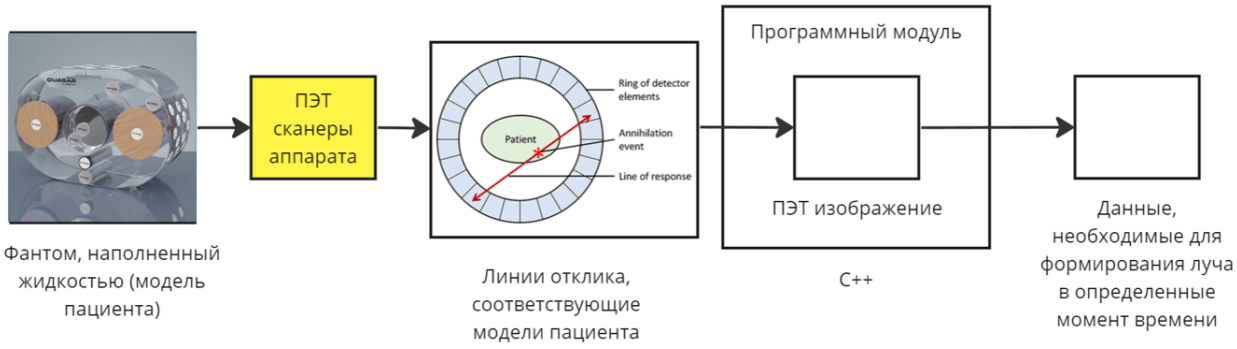


Рисунок 1 – Настоящий процесс тестирования аппарата

Для преодоления сложностей, связанных с натурными моделями, предлагается использовать компьютерные модели пациента и подход, рассмотренный на рисунке 2. Зная внутреннюю структуру фантома, можно ручным образом составить "идеальное" ПЭТ-изображение. Это изображение становится основой для создания компьютерной модели пациента.

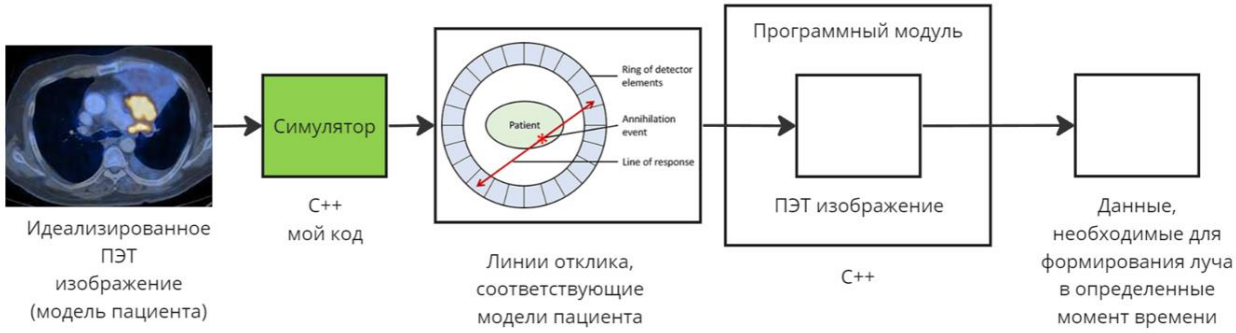


Рисунок 2 – Планируемый процесс тестирования аппарата

Результатом реализации такого подхода будет являться автоматизация процесса тестирования. Генерация линий отклика происходит на основе идеализированного ПЭТ-изображения. Сгенерированные данные подаются на вход программного модуля, который работает с ними как с данными, полученными естественным путем.

Такой компьютерный подход к моделированию позволит более эффективно проводить тестирование приборов лучевой терапии, минимизируя влияние шумов и предоставляя более стабильные результаты. Это открывает новые возможности для повышения эффективности методов лучевой терапии под биологическим контролем.

Симулятор ПЭТ сканера, получающий на вход ПЭТ изображение, составленное специалистом, и возвращающий соответствующий набор линий отклика, состоит из следующих модулей [5]:

– преобразование Радона – переход от осей (X, Y, Z) к осям (θ, S, Z);
– scatter-коррекция – избавление от искажений, связанных с геометрией пациента;
– извлечение слайсов синограммы – установка соответствия между частями синограммы и положением кушетки;
– генерация случайных данных – генерация нормально распределенных данных и масштабирование;

– попиксельная сумма синограмм;
– ACF-коррекция – избавление от искажений, связанных с плотностью такней;
– генерация событий из вероятностного распределения.

Симулятор ПЭТ сканера реализуется на языках MATLAB и C++, использует ранее реализованные классы на языке MATLAB и взаимодействует с программным модулем аппарата лучевой терапии на языке C++. Совместимость кода MATLAB и кода C++ поддерживается при помощи MATLAB Engine API и технологии MEX-файлов.

В начале работы симулятора происходит соединение с базой данных (класс PlanningSession), после чего файл формата json (класс ImageSeries), ссылающийся на серию ПЭТ изображений, извлекается в оперативную память или на диск. Симулятор работает в режиме реального времени, генерируя события (фиксацию линий отклика), при этом часть из преобразований, выполняемых модулями, можно вычислить заранее. Последний модуль получает на вход синограмму, каждое значение которой является параметром λ для распределения Пуассона $p(k) = \frac{\lambda^k}{k!} e^{-\lambda}$. Эта синограмма дает представление о вероятностном распределении для каждого пикселя исходного ПЭТ изображения. В результате работы программы сгенерированные данные передаются дальше программному модулю аппарата лучевой терапии, которые возвращает данные, необходимые для формирования луча в определенные моменты времени.

ЛИТЕРАТУРА

1. Daniel Pham, BS Eric Simiele, PhD Dylan Breitkreutz, PhD Dante Capaldi, PhD Bin Han, PhD Murat Surucu, PhD Seyi Oderinde, PhD Lucas Vitzthum, MD Michael Gensheimer, MD Hilary Bagshaw, MD Alex Chin, MD Lei Xing, PhD DT Chang, MD Natalyia Kovalchuk, PhD IMRT and SBRT Treatment Planning Study for the First Clinical Biology-Guided Radiotherapy System // Technology in Cancer Research & Treatment. - 2022. - №21.
2. M. Unterrainer, C. Eze, H. Ilhan, S. Marschner, O. Roengvoraphoj, N. S. Schmidt-Hegemann, F. Walter, W. G. Kunz, P. Munck af Rosenschöld, R. Jeraj, N. L. Albert, A. L. Grosu, M. Niyazi, P. Bartenstein & C. Belka Recent advances of PET imaging in clinical radiation oncology // Radiation Oncology. - 2020. - №88
3. Matthias Mäurer, Lukas Käsmann, Daniel F. Fleischmann, Michael Oertel, Danny Jazmati, Daniel Medenwald & Young DEGRO Trial Group PET/CT-based adaptive radiotherapy of locally advanced non-small cell lung cancer in multicenter yDEGRO ARO 2017-01 cohort study // Radiation Oncology. - 2022. - №29
4. DBarbara Pawlak, MSc. Richard Gordon, Ph.D.* Density Estimation for Positron Emission Tomography // Technology in Cancer Research & Treatment. - 2005. - №4. - С. 131 - 141.
5. Andreas Maier, Stefan Steidl, Vincent Christlein, Joachim Hornegger Medical Imaging Systems: An Introductory Guide (Image Processing, Computer Vision, Pattern Recognition, and Graphics). - 1st ed. - Springer, 2018. - 269 pp

УДК 004.8

А. Д. Плетнева (4 курс бакалавриата),
О. Г. Малеев, к.т.н., доцент

РАЗРАБОТКА ВЕБ-ПРИЛОЖЕНИЯ ДЛЯ РАСПОЗНАВАНИЯ ПЕСЕН ПО НАПЕВАНИЮ С ИСПОЛЬЗОВАНИЕМ МАШИННОГО ОБУЧЕНИЯ

В современном мире существует постоянно растущий объем музыкального контента. Примером этого может служить количество песен, которые индексируются в цифровых базах данных, таких как Spotify, где каждый день добавляется 40 000 новых композиций. Из-за этого возникает необходимость в методах поиска в больших объемах музыкальных данных. Например, один из способов поиска песни - по ее названию или исполнителю. Однако это бесполезно, если пользователь не знает название песни или какую-либо идентифицирующую

информацию, отличную от мелодии песни. Однако уже существую приложения, которые решают эту проблему. Наиболее популярное из них – Shazam. Shazam [1] - это приложение, которое может идентифицировать музыку на основе короткого сэмпла, воспроизводимого с помощью микрофона на устройстве. Но Shazam эффективен только в том, случае, когда оригинал интересующей нас композиции мы можем записать прямо сейчас. Также существуют приложения, в которых поддерживается распознавание песен по напеванию, однако они имеют ряд недостатков, таких как устаревший интерфейс, возможность распознавания трека только по загруженному файлу, отсутствие поддержки русского языка, большое количество рекламы, высокая стоимость, отсутствие истории распознанных композиций.

Таким образом, целью данной работы является создание веб-приложения, которое позволит пользователям распознавать песни по их напевам, обрабатывая и анализируя предоставленные ими аудиофайлы с помощью алгоритмов машинного обучения и предоставляя соответствующие результаты. Приложение должно быть бесплатным, иметь современный простой интерфейс на русском языке, а также авторизованные пользователи будут иметь возможность посмотреть историю ранее распознанных треков.

В данном проекте будет использоваться комбинация различных технологий для реализации веб-приложения для распознавания песни по напеванию. Архитектура приложения представлена на рисунке 1.

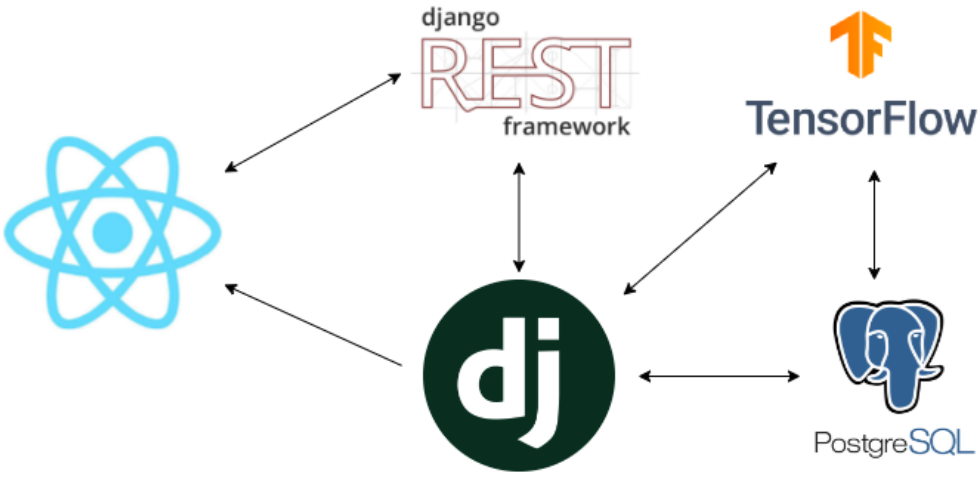


Рисунок 1 – Архитектура приложения

Frontend часть будет разработана с использованием React.js. React [2] — это библиотека JavaScript, разработанная Facebook в 2013 году, которая отлично подходит для создания современных приложений любого размера и масштаба. Она очень быстрая, благодаря реализации React Virtual DOM и различным оптимизациям рендеринга.

Чтобы реализовать нейросеть, будет использован фреймворк Tensorflow [3]. Для ее реализации понадобится собрать и подготовить набор данных, содержащий записи различных песен и соответствующие им напетые мелодии, разработать алгоритмы для предобработки аудиоданных для оптимизации распознавания песни. Для реализации распознавая песен по напеванию уже разработаны такие алгоритмы, как Query-by-Humming [4], музыкальная дактилоскопия на основе расстояния Бхаттачарии [5], алгоритм, основанный на облачных вычислениях [6] и т. д. Предстоит выбрать и усовершенствовать наиболее эффективный алгоритм для данной задачи.

Backend будет написан на django. Django [7] — свободный фреймворк для веб-приложений на языке Python, использующий шаблон проектирования MVC, имеет огромное количество расширений, библиотек и пакетов, созданных сообществом разработчиков. Также Django поставляется с встроенной административной панелью, которая предоставляет удобный интерфейс для управления данными и моделями приложения.

Хранение данных будет осуществляться с помощью СУБД PostgreSQL [8]. Данная система является одной из лидирующих среди направления систем управления базами данных. Она предоставляет поддержку JSON, а также возможность удобного хранения различных типов данных.

React может получать и публиковать данные через REST API, Django REST Framework сериализует данные из Django ORM и разрешает доступ/обновления через RESTful API, Django ORM создает модели базы данных и запросы и управляет ими, с помощью Tensorflow происходит распознавание мелодии, напетой пользователем.

ЛИТЕРАТУРА

1. Wang A. An Industrial Strength Audio Search Algorithm. // Proceedings of the 4th International Society for Music Information [Конференция], Baltimore, Maryland (USA) - 2003 г..
2. React Reference Overview - React. [Электронный ресурс] Режим доступа: https://react.dev/reference/react.
3. Tensorflow. [Электронный ресурс] Режим доступа: https://www.tensorflow.org/?hl=ru.
4. A.K.Tripathy N.Chhatre, N.Surendranath, M.Kalsi Query by humming system // INTERNATIONAL JOURNAL OF RECENT TRENDS IN ENGINEERING & RESEARCH. - 2009 г..
5. Riyanarto Sarno Dedy Rahman Wijaya, Muhammad Nezar Mahardika Music fingerprinting based on bhattacharyya distance for song and cover song recognition // International Journal of Electrical and Computer Engineering (IJECE). - 2019 г..
6. Lei Du IOP Song recognition in music library based on cloud computing. // Conference Series Materials Science and Engineering - 2020 г..
7. Django: Documentation. [Электронный ресурс] Режим доступа: https://docs.djangoproject.com/en/5.0/.
8. PostgreSQL: Documentation. [Электронный ресурс] Режим доступа: https://www.postgresql.org/docs/.

УДК 004.05

А. И. Подоба (4 курс бакалавриата),
Ю. Б. Сениченков, д.т.н., профессор

МОДЕЛИРОВАНИЕ МОРСКОГО ПОРТА КАК СИСТЕМЫ МАССОВОГО ОБСЛУЖИВАНИЯ В ANYDYNAMICS

В современном мире моделирование систем массового обслуживания становится все более востребованным инструментом для анализа и оптимизации различных процессов в бизнесе, индустрии, транспорте и других сферах человеческой деятельности. Точное и эффективное моделирование систем массового обслуживания позволяет прогнозировать и управлять потоками, оптимизировать распределение ресурсов и повышать качество обслуживания. В настоящее время разработано множество программных продуктов, предоставляющих возможность создавать и анализировать модели систем массового обслуживания. В данной статье мы рассмотрим программный продукт AnyDynamics, а также расскажем о преимуществах использования данного программного продукта.

AnyDynamics – это система имитационного моделирования, позволяющая создавать модели сложных динамических систем и проводить вычислительные эксперименты с ними.

Целью работы является выяснить удобно ли разрабатывать модели СМО в AnyDynamics. Необходимые шаги для достижения цели: 1. Рассмотреть и изучить сложную систему СМО. 2. Разработать библиотеку компонентов СМО в AnyDynamics. 3. Разработать изученную модель в AnyDynamics.

Модели систем массового обслуживания в AnyDynamics рассматривались в работе [1-2]. Но данные модели учебные, они не имеют конкретной цели и не такие сложные как реальные системы массового обслуживания.

Для задания была выбрана модель морского порта как системы массового обслуживания [3-5].

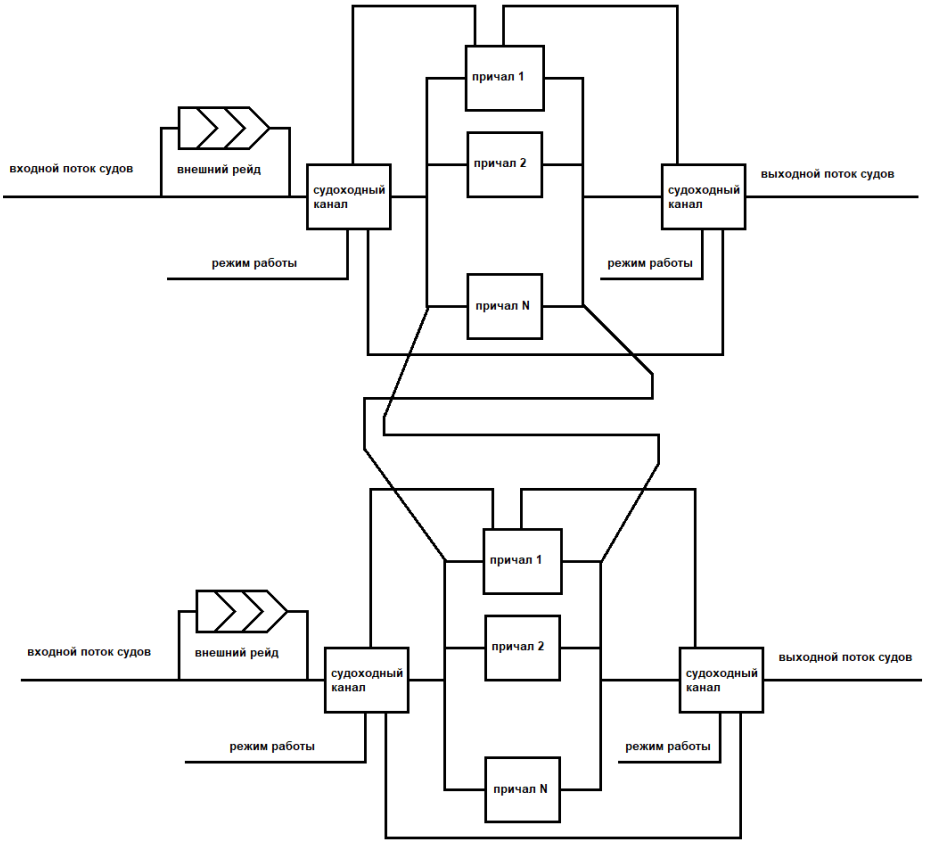


Рисунок 1 – Морской порт с двумя терминалами как система массового обслуживания.

Описание работы модели. Для каждого терминала имеется входящий поток судов, представленный расписанием прибытия судов в систему. Суда, заходящие в порт, могут зайти только на верхний терминал, только на нижний терминал, сначала на терминал верхний и потом на нижний терминал, сначала на нижний терминал и потом на верхний терминал. Суда, пришедшие в терминал отправляются в свободные причалы через судоходный канал, ограниченный по интенсивности пропуска судов или характеризуемый режимом его работы, если свободных причалов нет, то они встают в очередь. Суда, обработанные у терминала и следующие ко второму терминалу порта, дожидаются освобождения причалов последнего, продолжая занимать причал первого терминала. При освобождении причала терминала назначения они имеют приоритет перед судами, ожидающими входа в канал на рейде. После последнего обслуживания суда выходят из терминалов через судоходные каналы. Каждый канал обслуживания характеризуется временем обработки заявок, которым в рассматриваемом случае служит время выполнения погрузо-разгрузочных T_{op}. В свою очередь, это время определяется объемом выгружаемой и погружаемой грузовых партий судна (V_{imp} и V_{exp}), и производительностью причала P_{berth}, то есть $T_{op} = \frac{V_{imp} + V_{exp}}{P_{berth}}$

Условия задачи – разработать модель в AnyDynamics, которая работает по описанию модели. Модель должна вычислять: среднее время погрузочно-разгрузочных операций (ПРР), среднее время простоя у причала, среднее время обслуживания у причала, среднее время ожидания на рейде.

Из схемы модели можно выделить следующие компоненты:

- Заявка (Судно)
- Входной поток заявок (Входной поток судов)
- Очередь (Внешний рейд)
- Точка обслуживания с пропускной способностью (Судоходный канал)

– Ресурс за счёт которой обслуживается сервис (Причал)
– Выходной поток заявок (Выходной поток судов)
– Расписание (Режим работы для судоходных каналов)

Также можно добавить следующий компонент:

– Сервис – он будет распределять приходящие судна в причалы, а также из отработавших причалов направлять судна дальше.

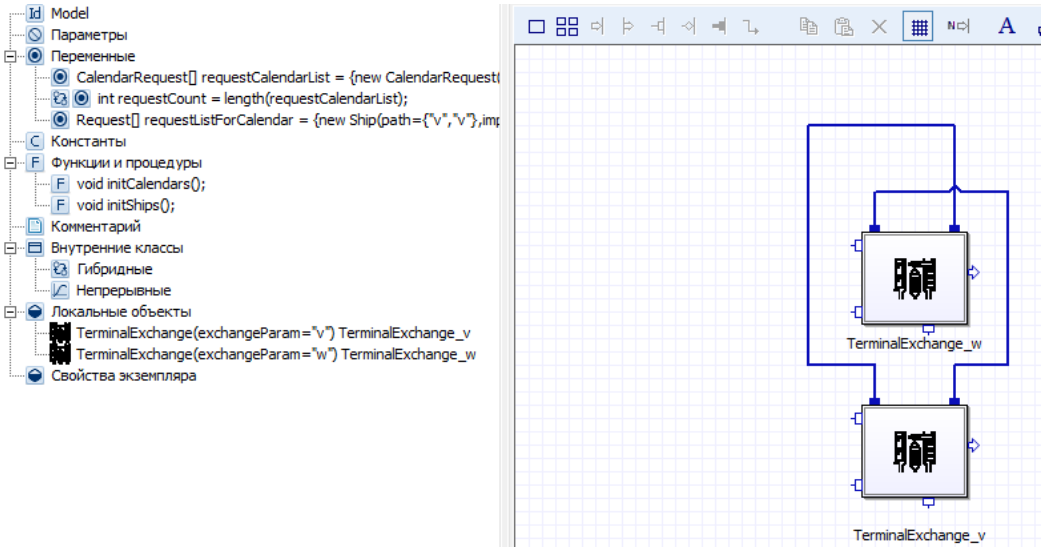


Рисунок 2 – Реализация порта с двумя терминалами как СМО в AnyDynamics.

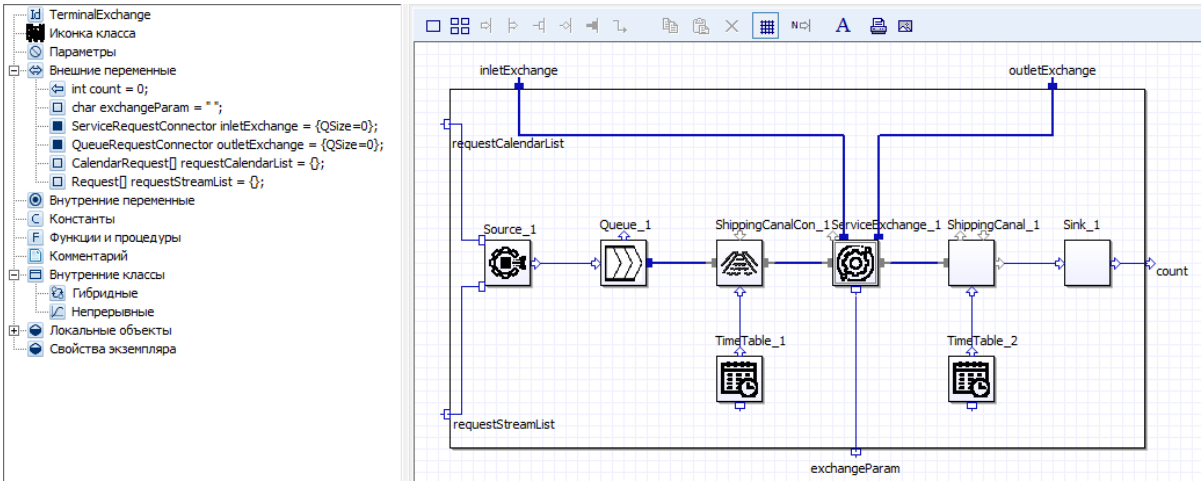


Рисунок 3 – Реализация терминала в AnyDynamics.

При работе данной модели получили такие результаты:

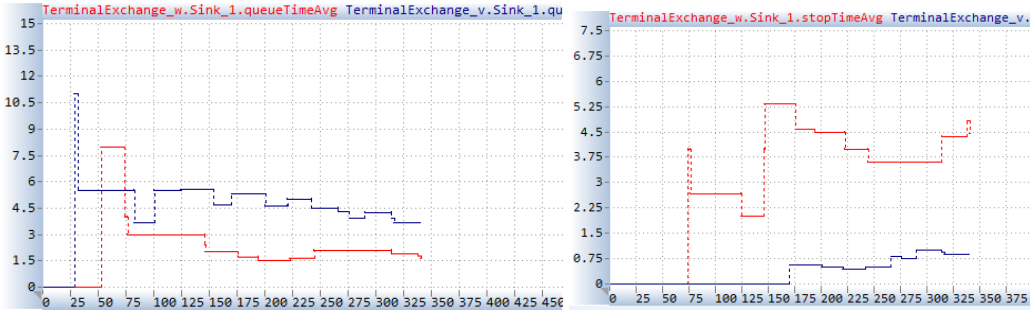


Рисунок 4 – Среднее время ожидания в очереди, среднее время простоя в ресурсе.

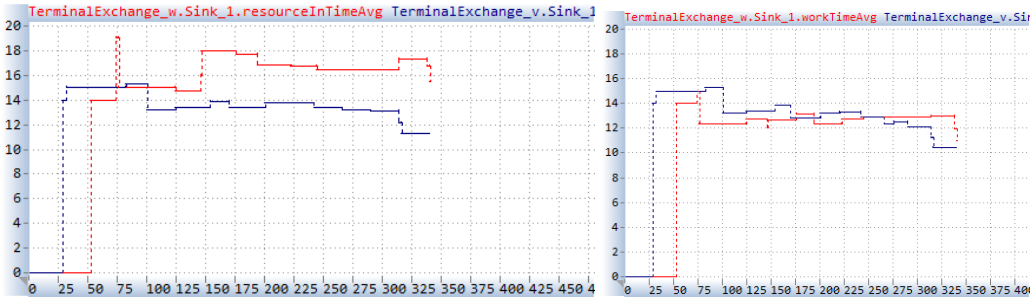


Рисунок 5 – Среднее время нахождения в причале, среднее время погрузочно-разгрузочных операций.

Получив результаты работы модели AnyDynamics, можно сделать выводы, что модель отработала корректно, схема модели разработанная в AnyDynamics схожа с оригинальной схемой. Сделаем вывод: AnyDynamics удобен в моделировании систем массового обслуживания.

ЛИТЕРАТУРА

1. Колесов Ю.Б. Объектно-ориентированное моделирование систем массового обслуживания с помощью Rand Model Designer 7. - М.: 2014. - 30 с.
2. Колесов Ю.Б., Сениченков Ю.Б. Математическое моделирования гибридных динамических систем. - СПб.: Издательство Политехнического университета, 2014. - 236 с.
3. Кузнецов А.Л., Семенов А.Д. Расчет параметров контейнерного порта методами теории массового обслуживания и имитационного моделирования // ИКМ МТМТС. - 2023. - С. 110-116.
4. Кузнецов А.Л., Кириченко А.В., Зайкин Д.А. Моделирование работы морского грузового фронта // Вестник государственного университета морского и речного флота имени адмирала С.О. Макарова. - 2019. - С. 33-42.
5. А. Л. Кузнецов, А. В. Кириченко, В. А. Погодин, В. Н. Щербакова-Слюсаренко Роль имитационного моделирования в технологическом проектировании и оценке параметров грузовых терминалов // Вестник АГТУ. - 2017. - №2. - С. 93-102.

УДК 004.42

В. С. Почернин (4 курс бакалавриата),
А. П. Маслаков, ст. преподаватель

РАЗРАБОТКА СЕРВЕРНОЙ ЧАСТИ ПРИЛОЖЕНИЯ - АГРЕГАТОРА ЦИФРОВЫХ ФИНАНСОВЫХ АКТИВОВ

Цифровые финансовые активы или же ЦФА – это новое явление на финансовом рынке Российской Федерации. Они создаются на базе технологии распределенных реестров, что позволяет автоматизировать исполнение сделок с ними за счет применения смарт-контрактов. Для того, чтобы в стране была обеспечена стабильность обращения цифровых финансовых активов, их выпуском и учетом занимаются операторы информационных систем, для которых существует отдельный реестр ЦБ РФ [1]. Компании-эмитенты, желающие выпустить свой цифровой финансовый актив должны обращаться за этим к представленным операторам. На данный момент на рынке отсутствует сервис, который бы агрегировал в себе все ЦФА, выпущенные различными операторами информационных систем, что затрудняет потенциальному инвестору поиск подходящего актива.

Целью данной работы является создание серверной части приложения, которое было бы способно агрегировать в себе цифровые финансовые активы, выпущенные различными операторами информационных систем, предоставлять пользователю информацию по ним в удобном виде, добавлять активы в избранное, а также предлагать возможные инвестиционные портфели.

Серверная часть приложения написана с использованием REST API, что является более эффективным подходом (по сравнению с WebSocket) при использовании одноразовых HTTP-

запросов и ответов [2]. Использование данного подхода также делает наше приложение универсальным – используя отдельный сервер, мы можем реализовать множество клиентов на различных платформах, которые будут использовать единый API, работая с консистентным набором данных. Архитектура серверной части приложения представлена на рисунке 1.

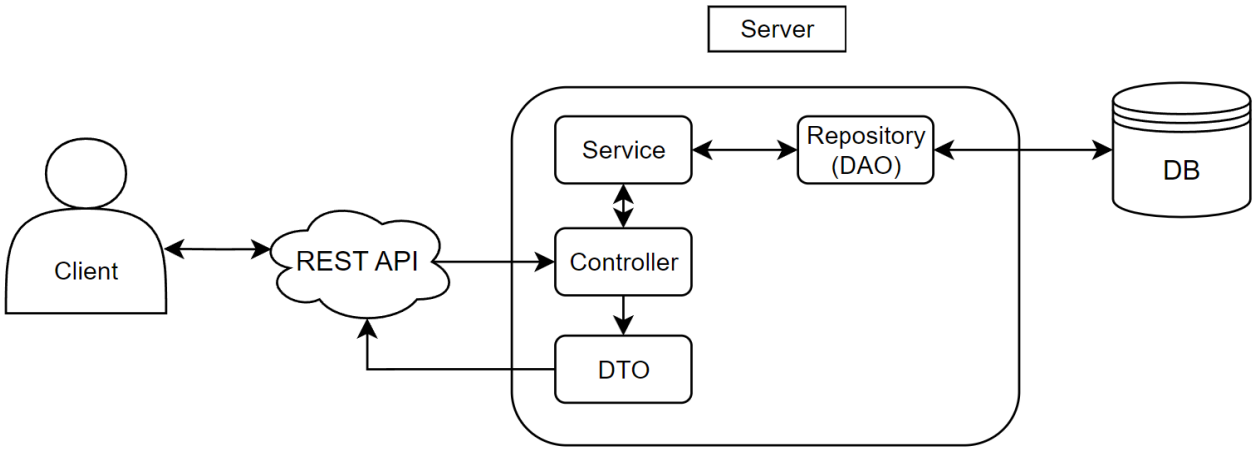


Рисунок 1 – Архитектура серверной части приложения

Хранилищем данных выступает СУБД PostgreSQL, которая является одним из самых популярных решений в России для построения архитектуры хранения данных, обладающей быстротой и надежностью [3]. В базе хранится вся информация приложения: данные о цифровых финансовых активах, компаниях-эмитентах, операторах информационных систем, зарегистрированных пользователях, их портфелях и избранных активах. В отдельной таблице хранятся данные о стоимостях ЦФА в различные моменты времени, что дает возможность для анализа изменения стоимости актива с течением времени.

Сам сервер написан на языке программирование Java с использованием фреймворка Spring Boot, который де-факто является стандартом для современных веб-приложений, написанных на Java. Данный фреймворк используется множеством IT-компаний, поскольку в нем постоянно актуализируется функциональность под нужды рынка веб-приложений [4]. С помощью Spring Data фреймворк позволяет настроить удобное взаимодействия с сущностями базы данных, отображая их на внутренние объекты Java кода. Стандартный сценарий работы сервера таков: на вход REST-контроллера приходит HTTP запрос, содержащий токен аутентификации и тело запроса. Обрабатывая запрос, контроллер обращается к сервису, в котором реализуется основная бизнес-логика работы с данными. Сами же данные берутся из репозитория – специального класса, реализующего доступ к базе данных. Обработанные данные отображаются в DTO (Data Transfer Object) и возвращаются в виде HTTP-ответа, содержащего, кроме прочего, JSON строку. Клиент, получая ответ, анализирует код ответа, а также полученные данные и выводит на экран пользователя информацию в удобном виде.

Система аутентификации в приложении основана на использовании Spring Security и JWT [5]. Открытыми для использования без аутентификации остаются точки входа или регистрации. При успешном входе или регистрации на сервере клиенту возвращается Json Web Token, в котором содержится заголовок с общей информацией, данные о пользователе, а также подпись, не позволяющая его подделать. При обращении к любой другой точке клиент (который после аутентификации запоминает полученный токен) прикладывает его к запросу в заголовке Authorization. Spring Security, обрабатывая запрос, первым делом проверяет корректность токена и только после этого отправляет запрос на выполнение.

Также, в репозитории, хранящем код приложения, была настроена система непрерывной интеграции, которая автоматически запускает написанные модульные, интеграционные и системны тесты при запросе на извлечение изменений в мастер-ветку. Написанные тесты

покрывают 96% кода серверной части приложения и проверяют все основные сценарии использования существующей функциональности.

В результате, был разработан веб-сервер, специализирующийся на хранении, обработке и выдаче цифровых финансовых активов. Вместе с клиентским приложением – это дает возможность потенциальным инвесторам ознакомиться со всевозможными цифровыми финансовыми активами, представленными на рынке, независимо от того, каким оператором информационной системы они выпущены, что способствует популяризации данного вида инвестирования и, как следствие, развитию экономики РФ.

ЛИТЕРАТУРА

1. Туфетулов А.М., Абдуллин А.А. Цифровые финансовые активы: сущность и перспективы на финансовом рынке // Экономические науки, № 217, 2022. С. 58-62.
2. Шишигина Д.В., Шемякин А.А. Сравнение двух наиболее популярных вариантов взаимодействия клиент-сервер в сети интернет // Актуальные научные исследования в современном мире, № 7-1 (51), 2019. С. 70-71.
3. Богатов И.В. Эффективная оптимизация запросов в СУБД POSTGRES // Академическая публицистика, № 5-2, 2022. С. 59-64.
4. Шилкина М.Л., Чернобай А.П. Вэб-приложение виртуальной библиотеки с использованием фреймворка Java Spring Boot // информационные системы и технологии: теория и практика, № 15, 2023. С. 155-167.
5. Абдурайимов Л.Н. Аутентификация в REST-приложениях с использованием Spring Security и JWT // Информационно-компьютерные технологии в экономике, образовании и социальной сфере, № 4 (34), 2021. С. 5-13.

УДК 004.4

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

СИСТЕМА МОНИТОРИНГА ПЕРЕМЕЩЕНИЙ

Мониторинг перемещений, в частности транспорта, является важным инструментом для обеспечения безопасности, контроля и оптимизации процессов в различных сферах деятельности.

Целью данной работы является разработка инструмента мониторинга перемещений, обеспечивающего настраиваемые параметры мониторинга в соответствии с индивидуальными требованиями.

Разрабатываемый продукт должен отвечать определённым требованиям, таким как модульность конструкции трекеров, возможность продолжительной автономной работы и развёртывание системы на собственных серверах. Несмотря на то, что на рынке представлено множество решений для мониторинга транспорта, нет такого, который бы полностью соответствовал всем требованиям.

В ходе проекта было разработано ПО для трекеров и создан графический интерфейс для удобного взаимодействия пользователей с системой.

Осуществлять работу с системой возможно как на мобильных устройствах через мобильное приложение на Android, так и на компьютерах через веб-приложение (или desktop-приложение). Также реализована ограниченная настройка и получение информации используя СМС.

Через пользовательский интерфейс можно получить информацию о перемещениях трекера, включая текущее местоположение, историю перемещений за определенный период времени, а также настройки и оставшийся уровень заряда батареи. Возможно получить и другие параметры, в зависимости от уровня доступа, настроек и комплектации конкретного трекера.

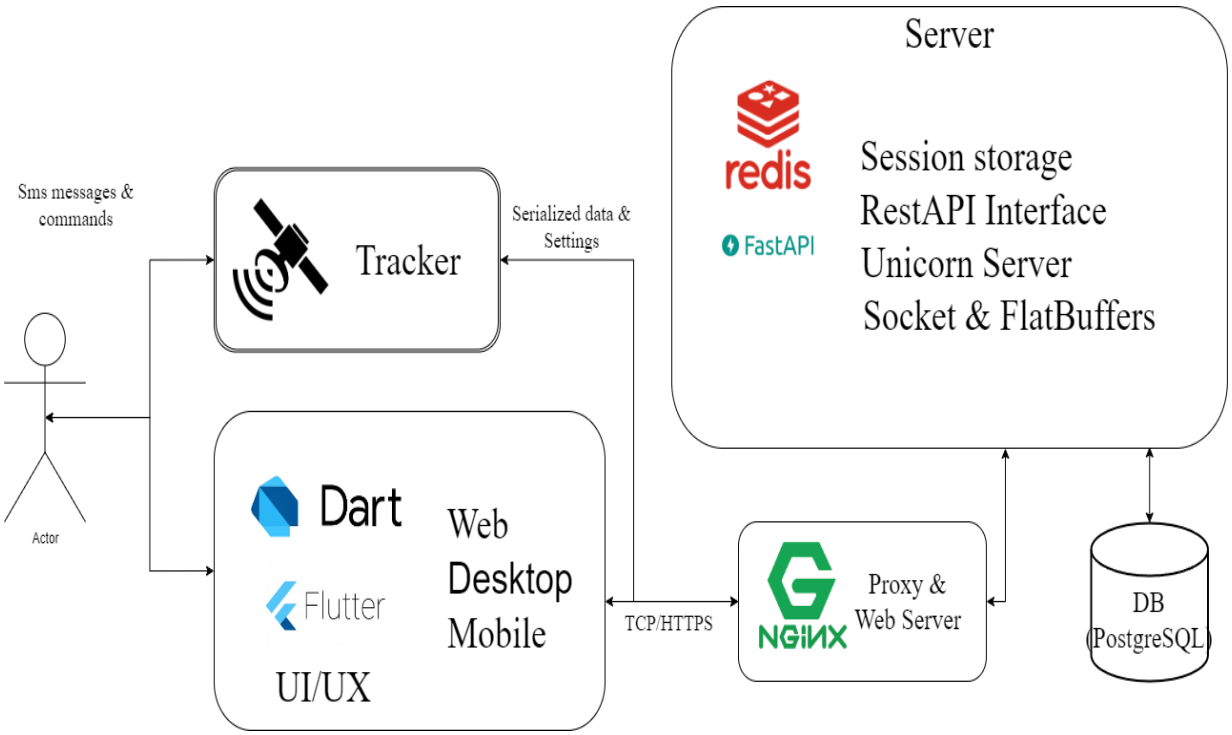


Рисунок 1 – Архитектура системы

Как видно из рисунка 1, при разработке использовались следующие технологии:
– FlatBuffers. Технология используется для сериализации данных, отправляемых с трекера на сервер, используя TCP сокеты;
– Nginx. Веб-сервер и почтовый прокси-сервер. В проекте используется для обеспечения доступа к API серверу, а также он используется для доступа к веб-приложению;
– FastAPI. Фреймворк для создания api. Совместно с веб-сервером Unicorn обеспечивает функционирование бэкенда;
– PostgreSQL. Реляционная база данных, необходимая для хранения пользовательских данных, логов с трекеров и их настроек, а также для разграничения прав доступа;
– Redis. NoSQL база данных. В проекте используется для кэширования сессий;
– Flutter. Open source фреймворк для разработки кроссплатформенных приложений. Flutter был выбран из-за обширной библиотеки плагинов и высокой производительности в сравнении с другими кроссплатформенными фреймворками, такими как React-Native и Angular;
– ПО для трекера писано на языке Си и предоставляет возможности: отправки данных с трекера на заданный сервер, работы в соответствии с полученными по сети или через СМС настройками и взаимодействия с дополнительными датчиками по настраиваемому скрипту.

ЛИТЕРАТУРА

1. Саперштейн Е. Б., Сергеева И. А., Ярославцева С. И. Декодирование данных формата NMEA. – 2017.GPSD.
2. Langley R. B. NMEA 0183: A GPS receiver interface standard //GPS world. – 1995. – Т. 6. – №. 7. – С. 54-57.
3. Lathkar M. High-Performance Web Apps with FastAPI: The Asynchronous Web Framework Based on Modern Python. – Apress, 2023.
4. AL-atraqchi O. M. A. A proposed model for build a secure restful api to connect between server side and mobile application using laravel framework with flutter toolkits //Cihan University-Erbil Scientific Journal. – 2022. – Т. 6. – №. 2. – С. 28-35.
5. Kernighan B. W., Ritchie D. M. The C programming language. – 2002.

УДК 004.4'2

Р. В. Проффен (2 курс магистратуры),
А. В. Самочадин, к.т.н, доцент

РАЗРАБОТКА ПРОГРАММНОГО КОМПЛЕКСА ДЛЯ ОБСЛУЖИВАНИЯ БАЗ ДАННЫХ В МЕДИЦИНСКОЙ ИНФОРМАЦИОННОЙ СИСТЕМЕ

В работе рассматривается медицинская информационная система "WEB.Поликлиника" [1], разработанная компанией "ВСД" и предназначенной для автоматизации деятельности медицинского учреждения при решении задач хранения, поиска и отображения данных о прикрепленном населении, электронных амбулаторных карт пациентов, публикации сведений о свободных ресурсах учреждения и так далее.

В информационной системе осуществляется различные действия с данными, хранение которых осуществляется в СУБД SQLite [2], в связи с чем необходимо разработать программный комплекс, осуществляющий обслуживание таких баз данных с учётом специфики работы медицинских учреждений.

Обычно в СУБД функции обслуживания являются встроенными. Например, к таким системам относятся PostgreSQL [3] и Microsoft SQL Server [4]. SQLite является однофайловой встраиваемой СУБД и не имеет подобных функций.

Таким образом, целью работы является разработка программного комплекса, который будет осуществлять обслуживание баз данных в медицинской информационной системе "WEB.Поликлиника".

Информационная система "WEB.Поликлиника", для которой будет осуществляться разработка ПО, является комплексом веб-приложений, разворачиваемым на серверах отдельных медицинских учреждений. Всего таких учреждений с развёрнутыми на их серверах информационной системой порядка 50 в Санкт-Петербурге.

Система является модульной, её общая схема с перечислением основных модулей и баз данных представлена на рисунке 1. Разрабатываемое средство обслуживания баз данных относится к одному из модулей системы.

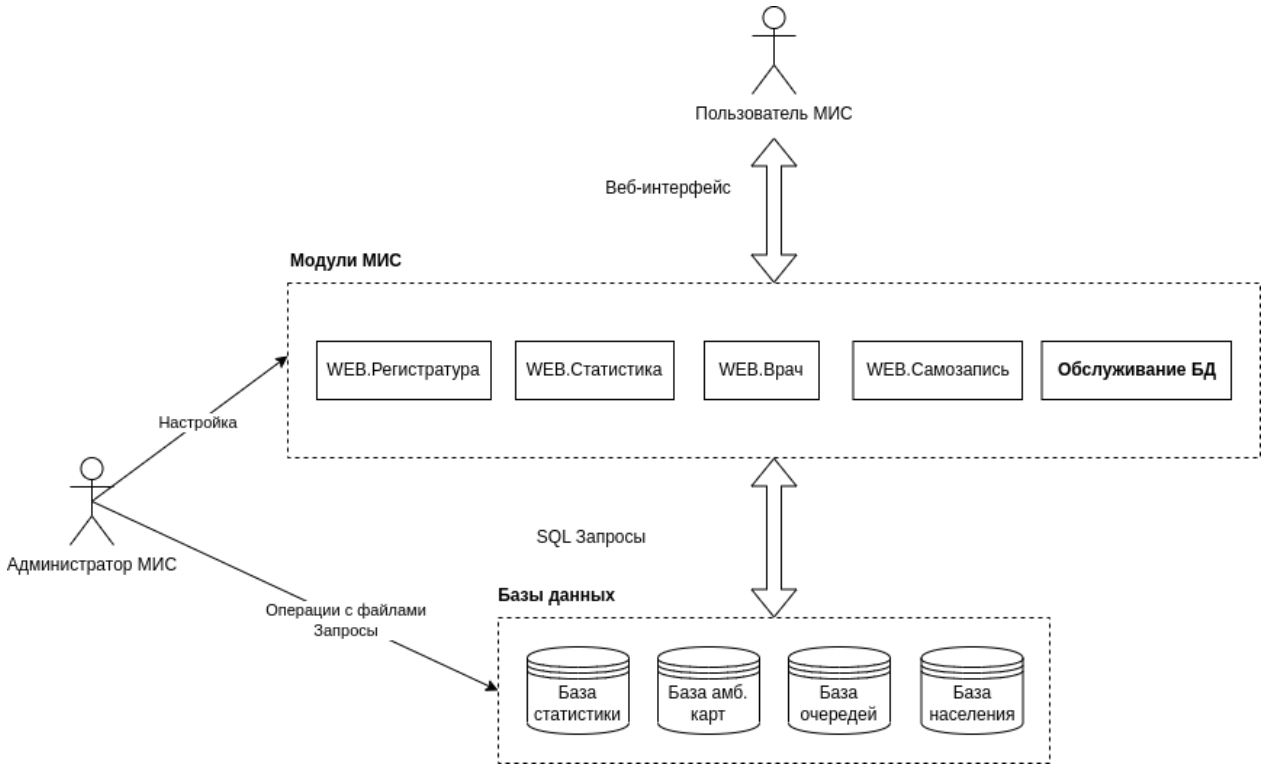


Рисунок 1 – Общая схема информационной системы

При разработке ПО необходимо учитывать, что система функционирует в государственных учреждениях и должна удовлетворять соответствующим нормативно-правовым актам. Также необходимо учитывать особенности СУБД SQLite, вследствие которых, например, конфликты при конкурентных подключениях могут приводить к ошибкам.

Следовательно:

– Итоговое решение должно быть исполняемым .exe модулем с закрытым исходным кодом.

– Для проведения действий по обслуживанию БД необходимо разработать соответствующие алгоритмы и средства.

– Для предотвращения и выявления ошибок с БД все действия с ними необходимо логировать.

Предлагаемое решение состоит из клиентской и серверной частей (рисунок 2).

Клиент является осуществляет обслуживание баз данных, а также посылает запросы в брокер сообщений (приложение написано на языке Python [5]). В тексте запросов содержатся логи действий с базами данных.

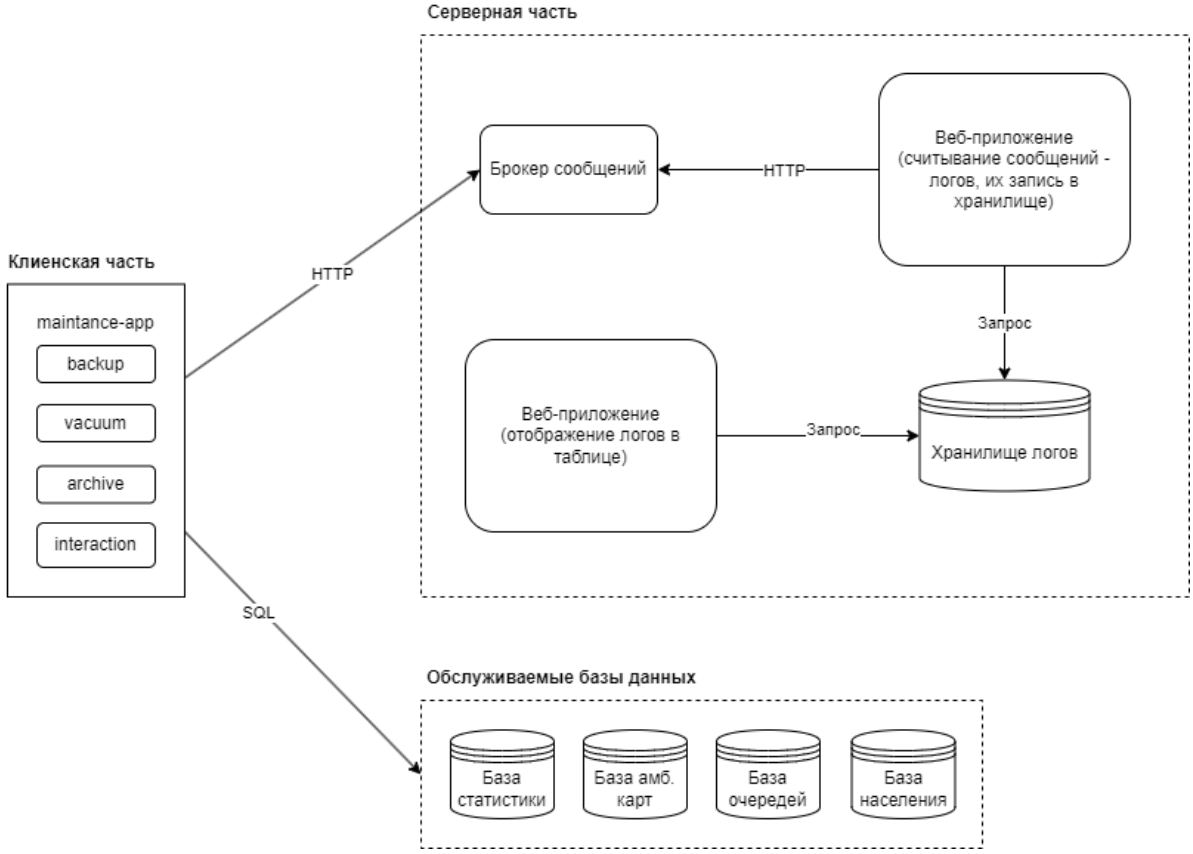


Рисунок 2 – Общая схема предлагаемого решения

Серверная часть состоит из:

– брокера сообщения RabbitMQ, развёрнутом на сервере организации;

– веб приложения, "слушающего" RabbitMQ и выгружающего сообщения в хранилище логов (Java Spring [6]);

– хранилища, где находятся сообщения (логи), отправляемые приложением, находящимся в поликлинике (PostgreSQL);

– приложения, отображающего в таблице логи, считываемые из хранилища (Java Spring).

ЛИТЕРАТУРА

1. Автоматизированная информационная система “WEB.Поликлиника” [Электронный ресурс] Режим доступа: https://vsd.spb.ru/mis/
2. Система управления базами данных “SQLite” [Электронный ресурс] Режим доступа: https://www.sqlite.org/index.html

3. Система управления базами данных “PostgreSQL” [Электронный ресурс] Режим доступа: https://www.postgresql.org/
4. Система управления базами данных “Microsoft SQL Server” [Электронный ресурс] Режим доступа: https://www.microsoft.com/en-us/sql-server/
5. Брокер сообщений RabbitMQ [Электронный ресурс] Режим доступа: https://www.rabbitmq.com/
6. Язык Python [Электронный ресурс] Режим доступа: https://www.python.org/
7. Фреймворк для создания Java веб-приложений Spring [Электронный ресурс] Режим доступа: https://spring.io/

УДК 004.9

А. О. Переятенцев, Д. Г. Чертков, В. О. Ячменев (2 курс бакалавриата),
К. В. Пшеничная, к.т.н., доцент

СИСТЕМА АНАЛИЗА WEB-САЙТОВ ОБРАЗОВАТЕЛЬНЫХ УЧРЕЖДЕНИЙ

В настоящее время актуально повышение эффективности взаимодействия участников учебного процесса с Web-ресурсами образовательных учреждений (ОУ) [1]. Для реализации этой цели предложена система, использующая модульно-модифицируемую архитектуру, базирующуюся на языке Python.

Предложенный подход позволяет асинхронно обрабатывать данные с Web-сайтов ОУ и трансформировать их в форматы, пригодные для анализа и последующего использования. В основе системы лежат принципы модульности и расширяемости, что предоставляет возможности для адаптации под изменяющиеся потребности ОУ.

Рассмотрен пример работы системы на базе расписания СПбГМТУ [2]. Структура модульной программы представлена на рисунке 1.

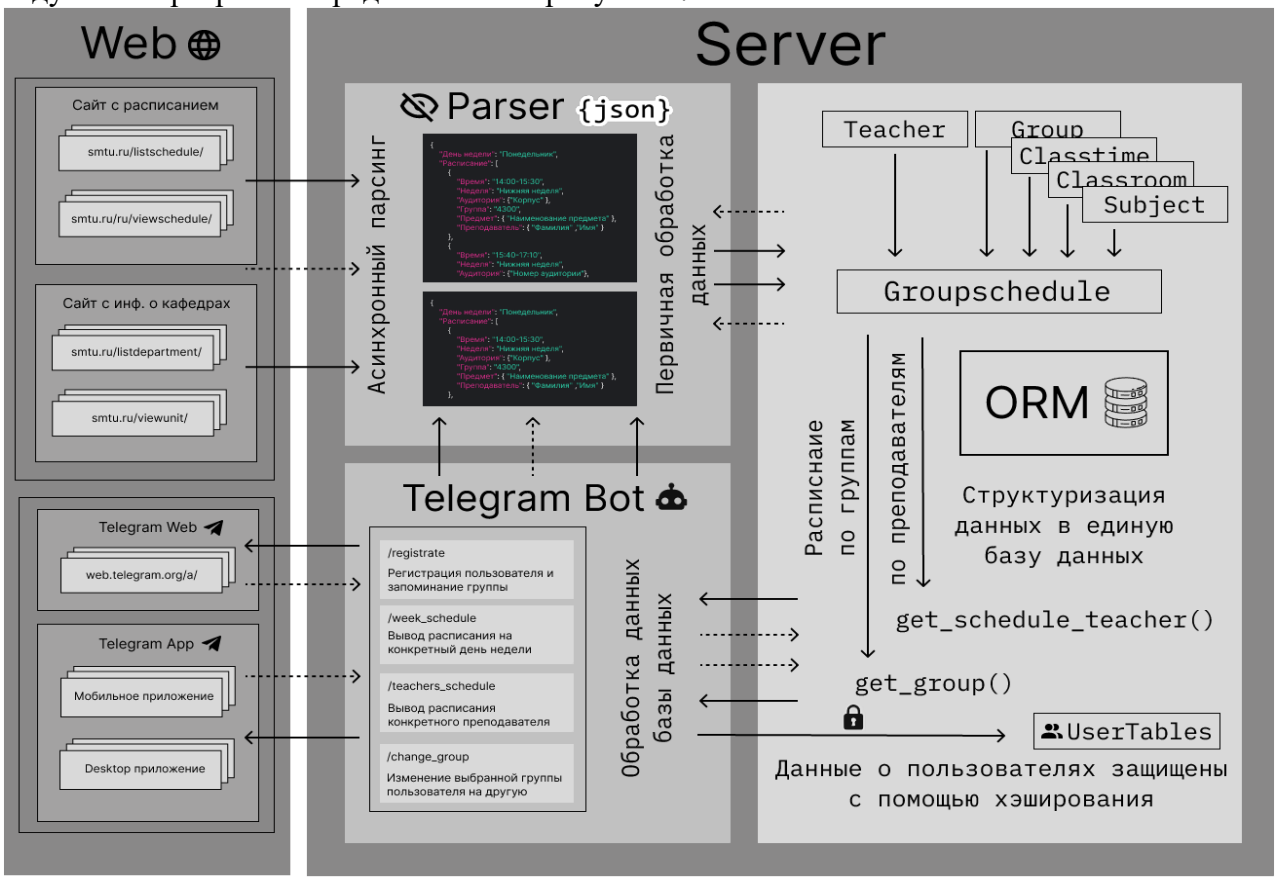


Рисунок 1 – Структурная модульной программы

Интерфейсы Web-сайтов ОУ служат источником исходных данных. Парсер, воплощающий асинхронные механизмы, выполняет сбор и предварительную обработку этих данных, конвертируя их в формат JSON. Данный процесс повышает адаптивность и скорость

обработки информации, а также обеспечивает универсальность системы: замена источника данных на другое ОУ не влияет на последующие этапы обработки.

Технология ORM [3], заложенная в фундамент системы, отвечает за структурирование данных в базе данных (БД), созданной с использованием модели объектно-реляционного отображения. С помощью ORM осуществляется доступ к информации и управление данными.

Telegram Bot, разработанный на базе библиотеки aiogram [4], служит пользовательским интерфейсом для доступа к данным БД. Он включает в себя функции для регистрации и управления профилем пользователя, получения расписания, а также подачи заявок и предложений. Тем не менее, система предусматривает гибкость в выборе средств коммуникации: вместо Telegram Bot в качестве транслятора данных может быть использовано специализированное приложение или другая платформа. Это подчёркивает модульность и расширяемость системы, позволяя ей адаптироваться к широкому спектру информационных каналов и пользовательских сценариев.

Функции Парсера:

1. Считывание сырых данных. Выполняются HTTP-запросы к веб-ресурсам ОУ и извлекаются сырые HTML-данные. Выполняется анализ DOM-структуры страниц и выявление необходимых элементов данных (время занятий, дисциплины, преподаватели, аудитории).
2. Создание структурированных файлов. Из сырых данных формируются структурированные JSON-файлы. HTML-контент преобразуется в машиночитаемый формат, что облегчает последующую интеграцию с БД.
3. Асинхронная обработка. Осуществляется асинхронная обработка, что позволяет эффективно масштабировать процесс сбора данных без перегрузки серверов ОУ.

Функции ORM:

1. Интеграция данных. ORM трансформирует структурированные файлы в формат, пригодный для работы с БД, путём отображения JSON-структур в таблицы БД. Это обеспечивает согласованность данных и их легкую интеграцию в систему.
2. Обеспечение целостности. СУБД с помощью ORM поддерживает отношения между различными таблицами, гарантируя целостность и согласованность данных.
3. Упрощение запросов. ORM реализует объектно-ориентированный подход, что упрощает выполнение сложных запросов.

Функции Telegram Bot (tg BOT):

1. Обеспечение интерактивности. Бот обеспечивает регистрацию пользователей, управление группами и предоставляет интерактивный доступ к расписаниям и информации о преподавателях через привычный интерфейс мессенджера.
2. Обработка запросов. Команды бота позволяют запросить расписание на конкретный день, получить различную информацию или отправить пожелание.

Предложенная система предоставляет ряд возможностей для разных пользователей. Некоторые из них перечислены ниже:

• Администрация факультета получает инструменты для анализа загруженности учебных аудиторий и управления аудиторным фондом.

• Администрация кафедры получает инструменты для автоматического формирования расписания кафедры.

• Студенты получают прямой доступ к актуальной информации о расписании на любой конкретный день.

Система обеспечивает интеграцию информационных ресурсов ОУ с повседневными академическими и административными действиями всех заинтересованных сторон. Результатом является повышение эффективности учебного процесса, а также расширение возможностей для управления образовательным процессом.

ЛИТЕРАТУРА

1. Т.Н. Носкова, Т.Б. Павлова. Взаимодействие с цифровыми ресурсами: продуктивность образовательной деятельности. // Человек и образование – 2019, № 3 (60), с.44-50.

2. Сайт СПбГМТУ. Режим доступа https://www.smtu.ru (дата обращения 12.03.2024).
3. SQLAlchemy ORM. [Электронный ресурс]. Режим доступа: https://docs.sqlalchemy.org/en/20/orm/index.html.
4. aiogram Documentation Release 3.4.1. (2024). Режим доступа: https://aiogram.readthedocs.io/_/downloads/en/latest/pdf/.

УДК 004.42

А. А. Рыжова (4 курс бакалавриата),
Т. В. Коликова, ст. преподаватель

РАЗРАБОТКА ANDROID ПРИЛОЖЕНИЯ ДЛЯ ПОИСКА И ОЦЕНКИ КНИГ

В современном мире наблюдается значительный интерес к чтению книг, что приводит к появлению огромного количества литературных произведений в различных жанрах от разнообразных авторов. Из-за этого изобилия материала часто возникают сложности при выборе подходящей книги для чтения.

Анализируя текущее состояние рынка приложений, связанных с оценкой и выбором книг, можно отметить, что большинство из них предлагают обширные библиотеки литературных произведений. Это создает дополнительную сложность для пользователей, поскольку процесс поиска интересного материала может быть длительным и требовать значительных временных затрат.

Для оптимизации процесса выбора литературы и улучшения пользовательского опыта было предложено разработать приложение, которое позволяет пользователям оценивать книги и просматривать оценки, оставленные другими пользователями, которых они добавили в друзья. Такой подход направлен на упрощение выбора книги путем предоставления информации о рейтинге и рецензиях, а также способствует формированию сообщества читателей, обменивающихся литературными рекомендациями.

Для реализации приложения был выбран язык Kotlin, который является официальным языком программирования для разработки Android приложений.

Для работы с HTTP-запросами была выбрана связка Retrofit и OkHttp3, так как они поддерживают основные методы HTTP-протокола, предоставляют гибкую систему для работы с параметрами запроса и, конечно же, возможность выполнять запросы асинхронно, что поможет не нагружать главный UI поток, и, как итог, поможет избежать ошибки ANR (Application Not Responding). Эти факторы обеспечивают надежность, масштабируемость и эффективность при взаимодействии с сетью.

Для эффективной реализации DI (Dependency Injection) в рамках разработки приложения было принято использовать фреймворк Dagger2. Данный инструмент представляет собой мощный инструмент для управления зависимостями в приложении, обладающий рядом преимуществ, которые способствуют улучшению архитектуры и общей структуры кода. Одним из ключевых преимуществ Dagger2 является его гибкая конфигурация зависимостей. Фреймворк позволяет определять зависимости и их жизненный цикл с помощью аннотаций, что обеспечивает прозрачность и удобство в управлении зависимостями. Кроме того, использование Dagger2 повышает читаемость и помогает разделить код на независимые компоненты, что обеспечивает более легкую поддержку и возможности для расширения функционала.

Чтобы локально сохранять историю поиска пользователя, используется фреймворк Room. Это самый современный фреймворк для локального хранения данных, использующий SQLite.

Архитектура включает в себя несколько слоев – слой данных, слой бизнес-логики и слой представления. В слое данных реализована работа с сетью, обработка сетевых ошибок, взаимодействие с локальной базой данных. Слой бизнес-логики отвечает за основную логику приложения. В этом слое сформированы главные алгоритмы, которые определяют поведение

приложения. Слой представления отвечает за UI логику, в нем будет использован паттерн MVVM.

ЛИТЕРАТУРА

1. Раджабов И. Н., Рысин М. Л. Чистая архитектура и паттерн MVVM в практике разработки Android-приложения // Сборник материалов XXII Международной научно-практической конференции. - Москва: Печатный цех, 2023
2. Казаков И. А., Якимчук А. В. Применение инструмента "Retrofit" при разработке программного обеспечения под операционную систему Android // Экономика и социум. - 2019. - №2 (57). - С. 460-462.
3. Шумилова Е. А., Свищёв А. В. Внедрение зависимостей в Android-разработке. Особенности и сравнение DI-фреймворков платформы Android // Моя профессиональная карьера. - 2023. - №54. - С. 245-257.
4. Тимофеев А. Г. Использование корутин в приложениях на платформах Android // Труды ростовского государственного университета путей сообщения. - 2021. - №3 (56). - С. 44-46.
5. Сарманин А. М. Современный подход к реализации архитектурного паттерна в android-приложениях // Электронные библиотеки. - 2020. - №5. - С. 1058-1075.
6. Жемеров Д. Б., Искакова С С. Kotlin в действии. - 1-е изд. - М.: ДМК-Пресс, 2018. - 402 с.

УДК 004.032.26

С. Ю. Рышкова (4 курс бакалавриата),
О. В. Прокофьев, ст. преподаватель

РАЗРАБОТКА ПРИЛОЖЕНИЯ ДЛЯ РАСПОЗНАВАНИЯ БОЛЕЗНЕЙ РАСТЕНИЙ

Точная диагностика болезней растений играет важную роль в сельском хозяйстве, поскольку фермерам часто приходится решать, достаточно ли хорош урожай, который они собирают. Раннее выявление и профилактика заболеваний необходимы для предотвращения потенциальных разрушительных последствий для человечества, например, глобальной нехватки продовольствия, избавляют от ненужных финансовых затрат и повышают качество жизни населения. К сожалению, полагаться исключительно на невооруженный глаз профессионала для выявления всех видов болезней растений сложно и трудоемко. К тому же ручная идентификация является менее точной и может выполняться только на небольших участках за один раз [1].

Внедряя автоматизированную систему, основанную на методах глубокого обучения и компьютерного зрения, мы способны решить вышеупомянутые проблемы. С её помощью болезни растений могут быть выявлены непосредственно на начальной стадии, а инструменты борьбы с вредителями и инфекциями возможно использовать для успешного решения проблем при минимизации рисков для людей и окружающей среды [2].

Таким образом, целью данной работы является решение проблемы автоматизации классификации болезней растений за счет создания веб-приложения для конечных пользователей, что обеспечит надежное и эффективное решение для точной диагностики в сельскохозяйственном секторе.

Для достижения этой цели необходимо выполнить следующие задачи:

1. Изучение предметной области и анализ существующих подходов к распознаванию болезней растений.
2. Сбор и подготовка тестового датасета.
3. Выбор и адаптация нейронной сети.
4. Обучение нейронной сети на тренировочных данных и настройка ее параметров для достижения оптимальной производительности.
5. Разработка приложения, способного загружать изображения растений и использовать обученную модель для классификации болезней на основе входных данных.

В данной работе нам потребуется специально обученная нейронная сеть, способная выявлять возможные заболевания растений на основе внешних признаков [3]. Для этой цели подойдут сверточные нейронные сети класс архитектур искусственных нейронных сетей (ИНС), позволяющих эффективно распознавать и анализировать визуальные данные. Их основная идея состоит в том, чтобы автоматически извлекать иерархические признаки из входных изображений [4].

Этапы создания и представления конечным пользователям модели ИНС сверточной архитектуры представлены на рисунке 1. В работе был использован общедоступный набор данных, взятый с сервиса Kaggle [5], состоящий примерно из 87 тыс. rgb-изображений здоровых и больных листьев сельскохозяйственных культур, которые разделены на 38 различных классов. Общий набор данных делится на обучающий и проверочный наборы в соотношении 80/20 с сохранением структуры каталогов.

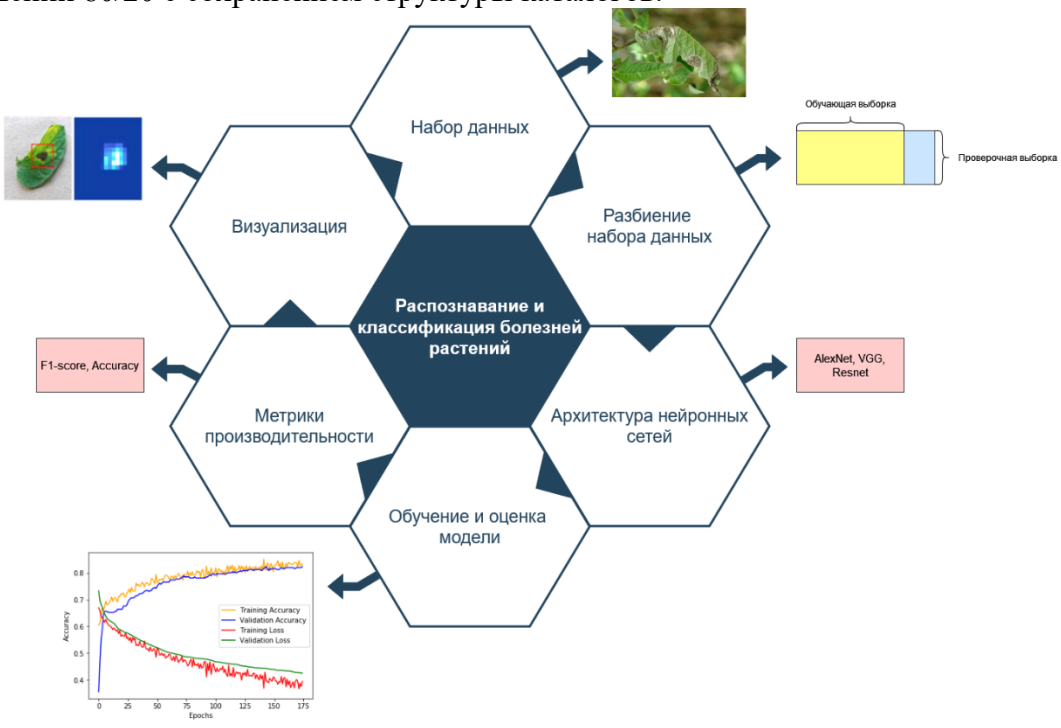


Рисунок 1 – Архитектура системы

В качестве основе нашей модели машинного обучения была применена архитектура VGG19, состоящая из 19 слоев и способная извлекать высокоуровневые признаки из изображений благодаря глубокой структуре сети [6]. С ее использованием мы достигли точности нашей модели, равной 85%.

Обученная модель была интегрирована в веб-сайт, разработанный с использованием HTML и CSS, а связь между серверной и пользовательской частями установлена при помощи Flask [7] фреймворка для создания веб-приложений на языке программирования Python. Основываясь на предоставленном пользователем изображении, модель успешно определяет, каким заболеванием страдает то или иное растение. Данное знание позволяет нам принимать обоснованные решения относительно дальнейшего лечения.

ЛИТЕРАТУРА

1. Khalid M., Sarfraz M. S., Iqbal U., Aftab M. U., Niedbala G., Rauf H. T. Real-Time Plant Health Detection Using Deep Convolutional Neural Networks // Agriculture (Switzerland). 2023. Vol. 13, no. 2.
2. Аббасов И. Б., Дешмух Р. Р. Распознавание изображений сельскохозяйственных культур, растений и лесных массивов // Известия ЮФУ. Технические науки. 2020. №3 (213). DOI 10.18522/2311-3103-2020-3-202-212.

3. Шереужева М. А., Шереужев М. А., Альбекова З. М. Использование сверточных нейронных сетей для задач автоматического обнаружения заболеваний // Известия Кабардино-Балкарского научного центра РАН. 2023. №5(115). DOI 10.35330/1991-6639-2023-5-115-41-51.
4. Convolutional Neural Networks for Visual Recognition. [Электронный ресурс] Режим доступа https://cs231n.github.io/convolutional-networks/
5. Kaggle: Your Machine Learning and Data Science Community. [Электронный ресурс] Режим доступа https://www.kaggle.com/
6. Neural Style Transfer VGG19. [Электронный ресурс] Режим доступа https://medium.com/software-dev-explore/neural-style-transfer-vgg19-dab643ec6160
7. Flask Documentation (3.0.x). [Электронный ресурс] Режим доступа https://flask.palletsprojects.com

УДК 004.62

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

РАЗРАБОТКА СРЕДСТВА ФОРМИРОВАНИЯ РАСПИСАНИЯ ПРЕПОДАВАТЕЛЕЙ В КОНТЕКСТЕ ЛОКАЛЬНОЙ ОБРАЗОВАТЕЛЬНОЙ СРЕДЫ

Образовательные учреждения часто сталкиваются с задачей эффективного управления ресурсами, особенно при формировании расписания. Важность этого проявляется в оптимизации времени менеджеров онлайн-школ и сокращении пропусков уроков студентами из-за непредвиденных обстоятельств у преподавателей. Расписание должно соответствовать целям учебного заведения, включать все необходимые курсы, учитывать предпочтения преподавателей и учебные потребности студентов. [1]

Одним из способов улучшения составления расписания является разработка системы для распределения учебных групп по свободным временным слотам, выбранными самими преподавателями и их руководителями. В данной работе разработка ведется в контексте существующей образовательной платформы, поскольку она уже содержит данные для распределения групп и обеспечивает заказчику гибкость в корректировках и добавлении функциональности. Однако нужно понимать, что ни готовые решения, ни разработанное мной средство не составляют окончательную версию расписания, ее может сформировать только человек, а средства лишь уменьшают трудоемкость предложенной задачи. [2]

Таким образом, целью данной работы является создание средств формирования расписания в онлайн школе программирования для детей, которая будет легко адаптироваться к возможным изменениям, позволять внедрять новую функциональность, успешно интегрироваться с существующей образовательной платформой заказчика. Для достижения этой цели необходимо решение следующих задач:

1. Проектирование моделей (сущности Django) с данными о расписании;
2. Создание логики обработки всевозможных HTTP-запросов [3] для взаимодействия с расписанием пользователями разных ролей (разработка представлений Django).
3. Разработка элементов интерфейса, их локализация на 8 действующих языков платформы и настройка доступов к ним для разных ролей.

Текущая образовательная платформа разработана на языке программирования Python с использованием фреймворка Django, поэтому разработка средства данной работы велась с этим же стеком.

Архитектура спроектирована следующим образом: пользователь взаимодействует с сервером через HTTP – запросы, каждый из которых имеет свой URL. Они зарегистрированы в проекте, благодаря чему их обрабатывают представления (view), и в процессе обработки обращаются к базе данных посредством предварительной сериализации данных. Затем представление возвращает серверу ответ на запрос, который отображается пользователю в виде определенных разработанных компонент интерфейса.

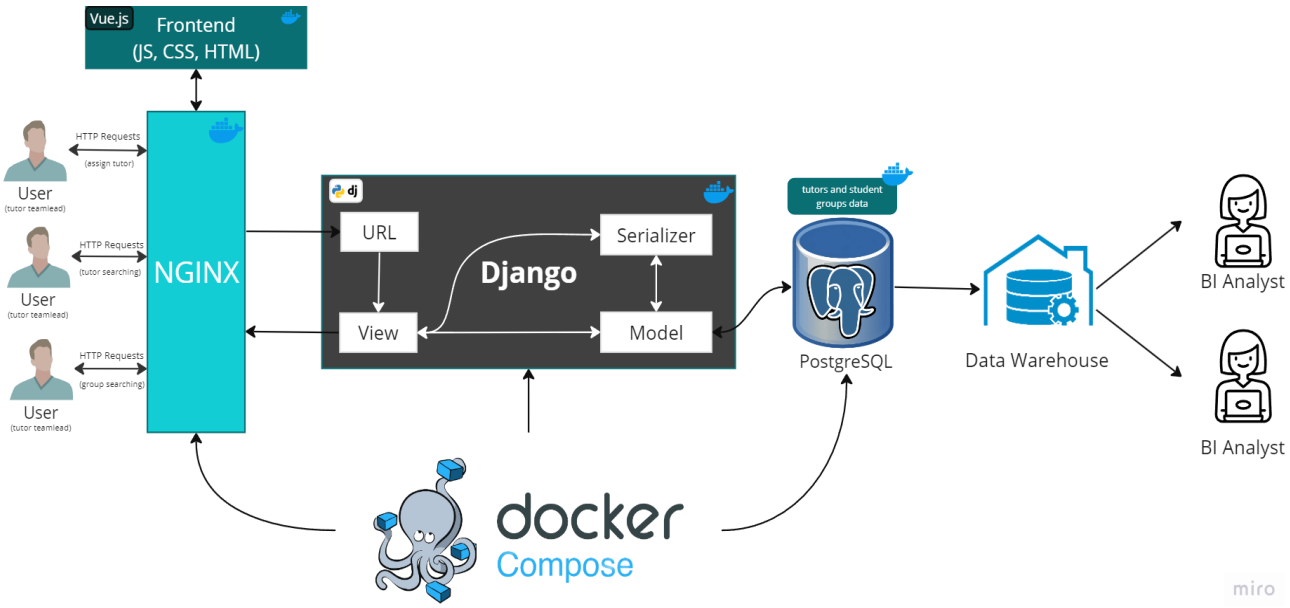


Рисунок 1 – Архитектура системы формирования расписания

Данные из базы данных (PostgreSQL) [5] собираются в DWH (Amazon Redshift) для дальнейшей аналитики. С этой же целью было разработано несколько моделей, содержащих все данные об изменениях объектов моделей. Для упрощения развертывания, управления всей инфраструктурой и изолирования, и обеспечения изолированной среды выполнения используется Docker Compose, а все элементы архитектуры упаковываются в контейнеры.

Логика работы с расписанием выглядит следующим образом: менеджер заходит на вкладку со всеми преподавателями, выставляет необходимые ему фильтры (курс, тип группы, часовой пояс, руководителя, от 1 до 7 временных слотов занятий, тип замены). Доступны разовое назначение преподавателя на группу, назначение с даты старта группы, либо с выбранной даты и до конца обучения группы.

Представление обрабатывает данные, и выдает доступных под выбранные фильтры преподавателей, около которых доступна кнопка назначения группы. Если преподаватель по фильтрам найден, а групп, которые учатся в выбранные даты нет, кнопка является неактивной. Также назначить на временной слот преподавателя можно на странице каждый группы, где отображается актуальное расписание обучения. Пользователь может назначить на урок преподавателя как разово, так и выбрать дату, после которой он хочет поставить его на все оставшиеся занятия.

ЛИТЕРАТУРА

1. Annette Perez. 8 Steps to Building an Elementary School Schedule // Edutopia. 2022. URL: https://www.edutopia.org/article/8-steps-building-elementary-school-schedule/. (Дата обращения: 25.02.2024).
2. Gordana S. How to create school schedule // Unify High School. 2020. URL:
1. https://unifyhighschool.org/create-school-schedule. (Дата обращения: 25.02.2024).
3. Gourley, D., Totty, B., Sayer, M., Aggarwal, A., Reddy, S. (2009). HTTP: The Definitive Guide. USA: O'Reilly Media.
4. Dauzon, S., Bendoraitis, A., Ravindran, A. (2016). Django: Web Development with Python. UK: Packt Publishing.
5. PostgreSQL: Документация: [Электронный ресурс]. URL: https://postgrespro.ru/docs/postgresql. (Дата обращения: 19.01.2024).

УДК 004.4

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

РАЗРАБОТКА ПРИЛОЖЕНИЯ С ИСПОЛЬЗОВАНИЕМ ФРЕЙМВОРКА NEXT.JS ДЛЯ АВТОМАТИЧЕСКОЙ ДИАГНОСТИКИ БОЛЕЗНЕЙ РАСТЕНИЙ

В современном мире сельское хозяйство играет ключевую роль в обеспечении продовольственной безопасности. Одной из важных частей этой сферы является забота о здоровье растений.

Болезни растений представляют серьезную угрозу для сельского хозяйства. Помимо прямых экономических потерь, болезни растений также приводят к другим проблемам в сельском хозяйстве, таким как снижение урожайности, повышение затрат на защитные мероприятия и использование химических препаратов [5; 6]. Следовательно, эффективное управление здоровьем растений представляет собой важную задачу для обеспечения продовольственной безопасности и устойчивого развития сельского хозяйства.

Ранняя диагностика заболеваний позволяет своевременно принимать меры по их контролю и предотвращению распространения, а также способствует эффективному использованию агрохимических средств и сокращению негативного воздействия на окружающую среду.

Симптомы многих болезней растений проявляются в видимом спектре [1], что открывает возможности для ручной диагностики человеком. Однако, этот подход имеет свои ограничения, включая сложность и множественность симптомов, а также требования к высокой квалификации специалистов, что сдерживает оперативность реакции на возможные угрозы. Кроме того, в условиях больших площадей сельскохозяйственных угодий, необходимость в более эффективных методах диагностики становится крайне актуальной. В связи с этим возникает необходимость в разработке автоматических систем диагностики болезней растений, способных оперативно и точно определять заболевания на основе анализа фотографий в видимом спектре.

Таким образом, *целью* данной работы является разработка веб-приложения для автоматической диагностики болезней растений по фотографиям, полученным в видимом спектре. Для достижения поставленной цели предлагается решить следующие *задачи*:

1. Обзор существующих программных средств для автоматической диагностики болезней растений с использованием методов машинного обучения.
2. Сбор данных для обучения, их разметка и предобработка.
3. Определение модели машинного обучения для решения задачи диагностики болезней растений и её обучение на подготовленных данных.
4. Оценка качества и эффективности обученной модели.
5. Разработка веб-приложения, интегрирующего обученную модель и позволяющего пользователям загружать фотографии листьев растений для автоматической диагностики болезней.

Разрабатываемое приложение должно уметь решать две задачи: сегментация пользовательского изображения и его классификация. Сегментация входного изображения предполагает выделение области заднего фона и области листа растения. Так как невозможно делать предположения о характере пользовательского изображения: цвете фона, освещённости и так далее, то для сегментации входного изображения должна быть использована отдельная модель [2]. Для решения этой задачи была взята предобученную модель для сегментации YOLO [3] и дообучена на подготовленных изображениях.

Классификация входного изображения предполагает определение, каким заболеванием поражен лист, представленный на изображении. Для обучения классификатора были использованы не сами изображения, а их компактное цифровое описание — текстурные признаки. Сначала из исходного изображения выделяется серия цветовых компонент,

соответствующих характерным цветам листьев, пораженных болезнью: оттенки желтого, коричневого, серого, черного цветов (рисунок 1). Затем для каждой из компонент вычисляются текстурные признаки Харалика [4].

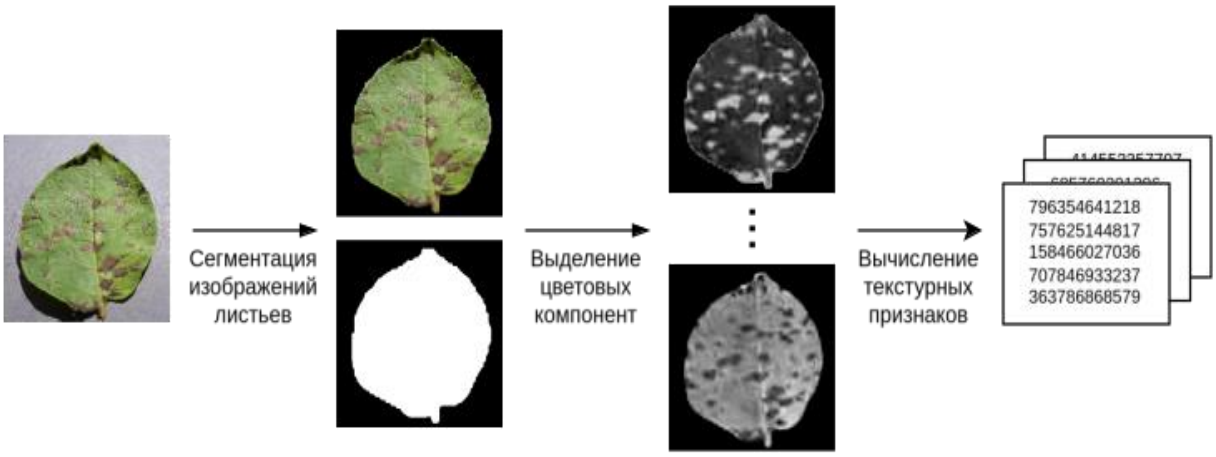


Рисунок 1 – сбор текстурных признаков из изображений листьев

Такой подход позволяет значительно уменьшить размер данных для обучения [7], а также позволяет достаточно просто производить аугментацию данных путем добавления случайного шума [7], что становится необходимым, когда количество данных, доступных для обучения ограничено.

Архитектура разрабатываемого приложения клиент-серверного типа, где клиентская часть разработана с использованием фреймворка Next.js, а серверная часть - на базе Express.js. Для обеспечения масштабируемости и автоматического развертывания применяется контейнерная инфраструктура с использованием Docker и Docker Compose. Модели машинного обучения разрабатываются на языке Python с применением фреймворка PyTorch, а эксперименты отслеживаются и управляются с помощью MLflow.

ЛИТЕРАТУРА

1. Arnal Barbedo J. G. Digital image processing techniques for detecting, quantifying and classifying plant diseases // SpringerPlus. – 2013. – Дек. – Т. 2, № 1.
2. Gajjar, R. Real-time detection and identification of plant leaf diseases using convolutional neural networks on an embedded platform / R. Gajjar, N. Gajjar, V. J. Thakor, N. P. Patel, S. Ruparelia // The Visual Computer. – 2021. – Июнь. – Т. 38, № 8. – С. 2923–2938.
3. Redmon, J. You Only Look Once: Unified, Real-Time Object Detection / J. Redmon, S. Divvala, R. Girshick, A. Farhadi // 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). – Los Alamitos, CA, USA : IEEE Computer Society, 06.2016. – С. 779–788.
4. Haralick R. M., Shanmugam, K., Dinstein, I. H. Textural features for image classification // IEEE Transactions on systems, man, and cybernetics. – 1973. – № 6. – С. 610–621.
5. Pathogens, precipitation and produce prices // Nature Climate Change. – 2021. – Авг. – Т. 11, № 8.
6. Говоров Д., Живых А., Ипатова Н., Защита растений в Российской Федерации: сколько стоит, что дает? // Защита и карантин растений. – 2015. – Окт. – № 12. – С. 7–8.
7. Тутыгин В. С., Прокофьев О. В. Новый метод диагностики болезней растений на основе цифрового описания изображений листьев и нейронной сети прямого распространения // Современная наука: актуальные проблемы теории и практики. Серия: Естественные и Технические Науки. – М., 2022. – Окт. – № 10. – С. 135–143.

УДК 004.94

К. А. Семенова (4 курс бакалавриата),
В. Н. Волкова, д.э.н., профессор

ВЫБОР CASE-СРЕДСТВА ДЛЯ МОДЕЛИРОВАНИЯ ПРОЦЕССОВ ВЗАИМОДЕЙСТВИЯ С КЛИЕНТАМИ В ОРГАНИЗАЦИИ

Управление по целям на средних и крупных предприятиях предполагает применение процессного подхода, в основе которого лежит бизнес-процесс. Бизнес-процессом называется логически завершенный набор этапов работ, поддерживающий деятельность предприятия и направленный на достижение поставленных целей [1, 2].

В повышении эффективности бизнес-процессов взаимодействия с клиентами важную роль играет автоматизация. На различных этапах автоматизации целесообразно построение моделей процессов и систем. Для этого широко используются методологии и нотации моделирования, для работы с которыми применяются различные CASE-средства.

1. Обзор методологий и нотаций моделирования процессов и систем.

Нотация IDEF0 (Integration Definition for Function Modeling) предназначена для описания процессов на верхнем уровне и для рассмотрения логических отношений между элементами процесса [3]. Нотация применяется на этапе анализа бизнес-процессов.

Нотация BPMN (Business Process Model and Notation) используется для описания процессов нижнего уровня. Диаграмма в нотации BPMN представляет собой алгоритм выполнения процесса, включающий описание взаимодействия объектов данных [3, 4].

Диаграмма вариантов использования (Use Case diagram) обеспечивает описание функциональных возможностей системы [5]. Она применяется на этапе определения основных сценариев, участников процесса и их отношений с другими участниками.

Диаграмма последовательности (Sequence diagram) демонстрирует, каким образом объекты взаимодействуют друг с другом. Построение диаграммы последовательности целесообразно при реализации интеграций компонентов системы или нескольких систем.

Диаграмма классов (Class diagram) описывает статическую структуру системы [6].

Нотация DFD (Data Flow Diagram) позволяет описывать процессы передачи данных и используется при распределении потоков данных в системе.

ER-диаграмма (Entity-Relationship Diagram) используется при создании базы данных для описания сущностей, атрибутов и отношений.

2. Выбор CASE-средства для моделирования процессов и систем.

CASE (Computer-Aided Software Engineering) — это набор инструментов и методов, предназначенных для автоматизированного проектирования систем. Наиболее известные CASE средства: «BPWin», «ERWin», «AnyLogic», «Bizagi», «Microsoft Visio».

Для сравнения CASE-средств мной были выбраны следующие критерии сравнения:

1. Возможность разработки моделей в различных нотациях: IDEF0, BPMN, use case, sequence diagram, class diagram, DFD, ER-diagram;
2. Удобство пользования (понятный интерфейс, наличие шаблонов);
3. Доступность и удобство процесса загрузки и установки, наличие документации.

Характеристики CASE-средств могут быть представлены совокупностью множеств: $S = < W, F, R >$, где $W = \{w_i\}, i = \overline{1,n}$ – множество CASE-средств ($n = 5$); $F = \{f_j\}, j = \overline{1,m}$ – множество критериев сравнения ($m = 3$); $R = \{r_{ij}\}$, $i = \overline{1,n}$; $j = \overline{1,m}$ – множество, характеризующее связи между W и F.

Для более точной оценки программ необходимо учитывать значимость критериев, которые были выбраны для сравнения. Для этого я применила методику ПАТТЕРН [7].

Шаг 1. Введение обозначений:

$\{\beta_i\}, i = \overline{1,m}$ – относительные веса (значимость) критериев. Причем $\sum_{i=1}^{m} \beta_i = 1$

$\{\gamma_i\}, i = \overline{1,n}$ – относительные веса (значимость) рассматриваемых программ.

$\{q_{ij}\}$ – соответствие i программы j критерию. Причем $\sum_{i=1}^{n} q_{ij} = 1,\ j = \overline{1,m}$

Шаг 2. Оценка соответствия i программы j критерию (таблица 1).

Таблица 1 – Соответствие CASE-средств критериям сравнения (оценки q_{ij})

Критерии / Программы	f_1	f_2	f_3
w_1	0,22	0,19	0,19
w_2	0,18	0,19	0,21
w_3	0,21	0,20	0,18
w_4	0,16	0,20	0,21
w_5	0,23	0,22	0,21

Шаг 3. Оценка значимости критериев сравнения (таблица 2).

Таблица 2 – Значимость критериев сравнения CASE-средств (оценки β_i)

Критерии / Значимость	f_1	f_2	f_3
β_i	0,5	0,3	0,2

Шаг 4. Вычисление весов CASE-средств: $\gamma_i = \sum_{j=1}^{m} q_{ij} \times \beta_j, \sum_{i=1}^{n} \gamma_i = 1$ (таблица 3).

Таблица 3 – Весовые коэффициенты CASE-средств

Программы / Весовые коэффициенты	w_1	w_2	w_3	w_4	w_5
γ_i	0,205	0,189	0,201	0,182	0,223

Шаг 5. Выбор CASE-средств: $\gamma_{max} = \max_i \gamma_i$

Таким образом, наиболее высокая оценка принадлежит CASE-средству «Microsoft Visio». Данная программа может применяться в качестве инструмента для моделирования процессов и систем на разных этапах автоматизации: от анализа процессов и функций системы, до проектирования базы данных.

ЛИТЕРАТУРА

1. Волкова В.Н., Юрьев В.Н. и др. Прикладная информатика: справочник: учеб. пособие – М.: Финансы и статистика, 2021. – 768 с.
2. Волкова В.Н., Юрьев В.Н. Информационные системы в экономике: учебник. – СПб.: Изд-во Политехи, ун-та, 2006. – 538 с.
3. Димитриади Г.Д. Основные методологии и подходы к моделированию бизнес-процессов компании //Международный науч. ж-л «Символ науки», №.11-2-1,2023,с. 53-56.
4. Сапожкова Т.Е. Сравнительный анализ подходов к моделированию бизнес-процессов // Сборник «Прикладная информатика», (37) 2012, стр. 14-19.
5. Линник Е.А., Крикунов Д.О., Трифонов Г.И., Митрофанов Д.В. Построение методического аппарата обоснования требований к разработке информационных систем // Журнал: Теория и практика, №23, 2022, стр. 34-42.
6. Лапшова А.А.. Разработка графического описания программного обеспечения с помощью языка UML // Ж-л: Теория и практика современной науки, №6 (36), 2018, с. 894.
7. Волкова В.Н., Денисов А.А. Основы теории систем и системного анализа: Учебник для студентов вузов, обучающихся по направлению «Системный анализ и управление». – СПб.: Издательство СПбГПУ, 2004. – 520 с.

УДК 004.02

А. А. Селецкая (2 курс магистратуры),
А. Г. Сиднев, к.т.н., доцент

АНАЛИЗ И ОПТИМИЗАЦИЯ ПРОЦЕССОВ МЕЛКОСЕРИЙНОГО ПРОИЗВОДСТВА С ИСПОЛЬЗОВАНИЕМ АНАЛИТИЧЕСКОГО И ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ

Теория массового обслуживания широко применяется во многих сферах жизнедеятельности человека. Применение аппарата сетей систем массового обслуживания позволяет решать задачи оценки эффективности и оптимизации сложных процессов на производстве, в медицине, транспортной отрасли и других областях.

Интерес к моделированию производственных процессов непрерывно растет в связи с постоянным увеличением требований к ресурсоемкости и экономичности производства. В данной работе рассматриваются возможности теории массового обслуживания для представления моделей процессов мелкосерийного производства или так называемых job-shop систем.

Под job-shop системами понимается производство, выпускающее некоторый спектр продукции в небольших объемах. Для обеспечения конкурентоспособности производства и планирования необходимо решать задачи, связанные с изучением функционирования системы и оценкой ее характеристик. В случае с мелкосерийным производством такими характеристиками может быть, например, время обработки одной заявки.

Применение теории массового обслуживания для моделирования процессов производства позволяет на основе модели проводить оценку рассматриваемой системы и решать задачи оптимизации. Оценка характеристик модели может быть применена, например, для определения стратегии по увеличению дохода с учетом затрат и имеющегося оборудования.

Рассматриваемые job-shop системы могут быть представлены при помощи замкнутых сетей систем массового обслуживания. Так как подразумевается, что система производит товары нескольких классов, рассматриваемая сеть будет неоднородной. Также можно считать ее немарковской исходя из того, что закон поступления заявок в сеть может быть произвольным. Например, в работе «Determining inventory levels in a CONWIP controlled job shop» [1] рассматривается модель замкнутой сети для решения задачи оптимизации уровня запасов в цехе.

Перемещения заявок по сети описываются вероятностной матрицей переходов. В замкнутых сетях число заявок, постоянно находящихся в сети, определено заранее. Распределения заявок в узлах сети описываются начальными моментами – математическим ожиданием и дисперсией или квадратами коэффициентов вариации.

Для расчета характеристик построенных моделей применяются аналитические и имитационные методы. Стоит отметить, что для некоторых сетей точные методы расчета могут быть слишком трудоемкими или вовсе неприменимыми. В таких случаях используются приближенные алгоритмы.

Один из таких алгоритмов, основанный на балансе заявок, описан в статье «Solving general multi-class closed queuing networks using parametric decomposition» [2]. Этот метод подразумевает замену замкнутой сети разомкнутой с условием, что среднее суммарное количество заявок в разомкнутой сети будет равняться заранее заданному числу заявок в замкнутой сети. Это равенство достигается итерационно. Метод баланса заявок также лежит в основе алгоритмов, предложенных в ряде работ Ю. И. Рыжикова [3, 4].

Для реализации алгоритмов расчета и оптимизации неоднородных немарковских сетей систем массового обслуживания выбрана среда MATLAB, которая предоставляет множество функций для работы со сложными математическими вычислениями. Чтобы оценить погрешности приближенных алгоритмов, эталонными будут считаться результаты, полученные при имитационном моделировании. Имитационная модель позволяет оценить

параметры исследуемой системы, проверить реализуемость составленного производственного плана, а также выполнить необходимые расчеты и провести корректировки в системе [5].

В качестве инструмента для имитационного моделирования выбран язык GPSS и среда GPSS World. Код на языке GPSS состоит из блоков, описывающих компоненты модели и взаимосвязи между ними. Построенная модель имитирует реальную систему или процесс, и по результатам моделирования среда формирует отчет с рассчитанными характеристиками.

Целью работы является исследование методов аналитического и имитационного моделирования для анализа и оптимизации процессов мелкосерийного производства. В рамках данного исследования в среде MATLAB реализованы алгоритмы расчета для замкнутых сетей систем массового обслуживания. Рассмотрены однородные и неоднородные, одноканальные и многоканальные сети, сети Джексона и немарковские сети. Предложенные методы расчета основаны на поиске среднего числа заявок в сети путем последовательного изменения потока заявок во всех узлах сети. В алгоритмах расчета для замкнутых сетей был использован корректирующий фактор, учитывающий тот факт, что узел в составе замкнутой сети может в отличие от того же узла в составе разомкнутой сети содержать ограниченное число заявок. Поэтому при той же интенсивности на входе ожидание обслуживания в узле замкнутой сети будет несколько меньшим, чем при прочих равных условиях в таком же узле разомкнутой сети. Для улучшения сходимости итерационных алгоритмов использовался метод Вегстейна. Приближенные алгоритмы расчета показали хорошую точность в сравнении с результатами имитационного моделирования.

Также в среде MATLAB реализован алгоритм для решения задачи оптимального назначения интенсивностей обслуживания заявок в узлах одноканальной однородной замкнутой немарковской сети. В ходе алгоритма вычисляются производные функции, зависящей от интенсивностей обслуживания заявок, и на основе полученных значений производится корректировка интенсивностей для поиска оптимального решения.

ЛИТЕРАТУРА

1. Ryan, Sarah & Baynat, Bruno & Choobineh, Fred. (2000). Determining inventory levels in a CONWIP controlled job shop. IIE Transactions. 32. 105-114. 10.1023/A:1007602028992.
2. Satyam, Kumar & Krishnamurthy, Ananth & Kamath, Manjunath. (2013). Solving general multi-class closed queuing networks using parametric decomposition. Computers & Operations Research. 40. 1777–1789. 10.1016/j.cor.2013.01.014.
3. Рыжиков Ю.И. Теория очередей и управление запасами – 2001 г – 376 с
4. Рыжиков Ю.И. Численные методы теории очередей – 2019 г – 512 с
5. Русских П.А., Капулин Д.В. МУЛЬТИАГЕНТНАЯ МОДЕЛЬ МНОГОНОМЕНКЛАТУРНОГО МЕЛКОСЕРИЙНОГО ПРОИЗВОДСТВА // Вестник ЮУрГУ. Серия: Компьютерные технологии, управление, радиоэлектроника. 2021. №4.

УДК 004.588

И. О. Сачук (2 курс магистратуры), В. К. Фролов (1 курс),
П. Д. Дробинцев., к.т.н., доцент

РАЗРАБОТКА ОБУЧАЮЩЕГО ANDROID ПРИЛОЖЕНИЯ ДЛЯ СТУДЕНТОВ БИОЛОГОВ С ВОЗМОЖНОСТЬЮ ПРОСМОТРА УЧЕБНЫХ МАТЕРИАЛОВ И АВТОМАТИЧКСКОЙ ПРОВЕРКОЙ ЗАДАНИЙ

В современном мире цифровые технологии являются важнейшей частью жизни общества. Поэтому одна из важнейших вещей – введение цифровых технологий в образовательный процесс, что дает удобный и эффективный способ обучения студентов. Благодаря цифровым технология у студентов повышается мобильность выполнения заданий и изучения материала, увеличивается наглядность материала, а также появляется индивидуализация процесса обучения [1].

Разработка приложений для студентов медицинских вузов по изучаемым предметам, имеет большое значение для улучшения усвоения материала, так как позволяет студентам всегда держать под рукой средство, с помощью которого они могут изучать различную учебную информацию [2].

Существует множество учебных пособий, которые помогают студентам углубиться в изучаемую тему, но при этом существует недостаток приложений, который фокусируются на узконаправленные темы, и так же приложений, которые могут проверять изучаемый материал студентами [3].

Таким образом, целью данной работы является создание приложение, которые может помочь студентам медицинских вузов изучать различные темы, а также проверять полученные знания.

Для достижения этой цели необходимо решение следующих задач:

1. Обзор существующих приложений, которые нацелены на обучение и проверку знаний в медицинской сфере.
2. Создание походов автоматизации проверки знаний студентов.
3. Реализация приложения.
4. Демонстрация приложения и способов проверки заданий.

В качестве платформы для разработки приложения был выбран Android, который является самой популярной мобильной ОС [4].

Для написания приложения был выбран язык программирования Kotlin – официальный язык для написания Android приложений [5].

Для написания UI приложения был использован новый подход в Android – Jetpack Compose [6], который представляет современный набор средств от компании Google для создания приложений под ОС Android на языке программирования Kotlin. Jetpack Compose упрощает написание и обновление визуального интерфейса приложения, предоставляя декларативный подход [7].

Приложение написано, используя архитектурный паттерн MVVM (Model-View-ViewModel), который состоит из трех частей: Model - логика, которая связаны с данными приложения, View - это UI приложения, ViewModel - объект, в котором описывается логика поведения View в зависимости от результата работы Model.

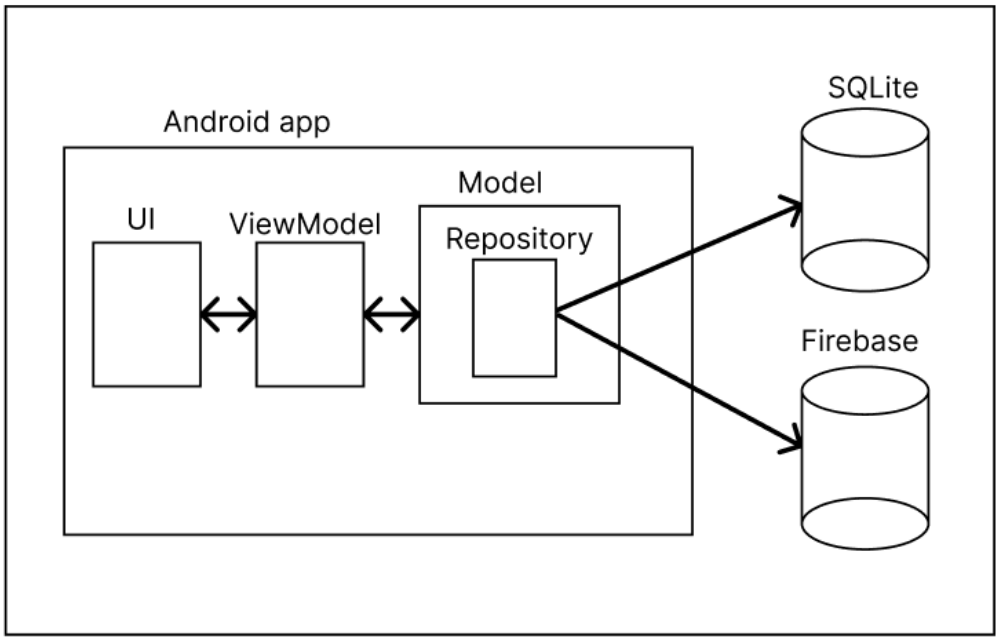


Рисунок 1 – Архитектура приложения

Для проверки заданий сделаны тесты, а также возможность давать пользователям проверять задания, загружая их в приложение, после чего получая ответ. Для этого

используется нейронная сеть, которая получает текст из фотографии, а после чего идет проверка написанного задания пользователем. Для работы с нейронной сетью используется библиотека Huawei ML Kit [8], которая предоставляет API для работы с нейронной сетью. Данная нейронная сеть позволяет распознавать 10 языков, включая русский. Также данная библиотека имеет и другие функции: распознавание звуков, преобразование текста в голос, перевод и др.

ЛИТЕРАТУРА

1. Ваганова О. И., Гладков А. В., Коновалова Е. Ю., Воронина И. Р. Цифровые технологии в образовательном пространстве // БГЖ. 2020. №2 (31). С.53-56
2. Aldekhyyel R, Almulhem J, Binkheder S, et al. User Perceptions and Use of Decision Support Medical Apps Among Medical Students: Cross-Sectional Study. Stud Health Technol Inform. 2024;310:1216-1220. doi:10.3233/SHTI231158
3. Абдукаева Н.С., Косенкова Н.С., Грачева Т.И., Васильева Н.В. Клетка - миниатюрная биосистема. Санкт-Петербург, 2018.
4. Android документация. [Электронный ресурс] Режим доступа: https://developer.android.com/develop.
5. Kotlin документация. [Электронный ресурс] Режим доступа: https://kotlinlang.org/docs/home.html.
6. Android Jetpack Compose. [Электронный ресурс] Режим доступа: https://developer.android.com/jetpack/compose.
7. Milla, E., Radonjić, M. (2023). ANALYSIS OF DEVELOPING NATIVE ANDROID APPLICATIONS USING XML AND JETPACK COMPOSE. Balkan Journal of Applied Mathematics and Informatics, 6(2), 167-178.
8. Huawei ML Kit. [Электронный ресурс] Режим доступа: https://developer.huawei.com/consumer/en/hms/huawei-mlkit/

УДК 004.457

В. В. Сечко (4 курс бакалавриата),
Б. М. Медведев, к.т.н., доцент

РАЗРАБОТКА СЕРВЕРНОЙ ЧАСТИ ПРОГРАММНЫХ СРЕДСТВ УПРАВЛЕНИЯ ОБНОВЛЕНИЕМ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ НА КОСМИЧЕСКИХ АППАРАТАХ

В настоящее время космическая индустрия находится в стадии интенсивного развития. Многие космические аппараты сталкиваются с необходимостью частого обновления программного обеспечения. Это вызвано не только стремительным прогрессом в технологиях, но и появлением новых задач и требований, которые требуют изменения программного обеспечения. Космические миссии требуют высокого уровня безопасности и надежности программного обеспечения. Регулярные обновления позволяют внедрять последние технологии и корректировать выявленные уязвимости, что существенно повышает безопасность космических аппаратов.

В качестве основы для разработки системы обновления программных средств космических аппаратов целесообразно использовать стандарты O-RAN [2], которые разработаны альянсом «O-RAN ALLIANCE» [3]. Этот альянс предприятий стал всемирным сообществом операторов мобильных сетей, поставщиков, а также исследовательских и академических учреждений, работающих в отрасли сетей радиодоступа (RAN). Стандарт O-RAN.WG4.MP.0-R003 определяет структуру пакетов, содержащих загружаемые программные средства, а также схему обновления. Однако, в связи с примененным подходом открытости в стандартах недостаточно уделено внимание вопросам защиты информации в процессе передачи и обновления программного обеспечения.

Таким образом, целью данной работы является разработка серверной части программных средств управления обновлением программного обеспечения на космических аппаратах, основанных на стандартах O-RAN.WG4.MP.0-R003 и содержащих дополнительные средства защиты: проверку целостности и цифровых подписей пакетов.

Для достижения цели необходимо решить следующие задачи:

1) Разработка программных средств центра обновления программного обеспечения космического аппарата на базе стандарта O-RAN.

2) Разработка программных средств формирования пакетов, содержащих загружаемое ПО.

3) Разработка программных средств космического аппарата для проверки целостности и подписей пакетов.

Разработаны следующие требования к программным средствам обновления ПО для космического аппарата: проверка на наличие ошибок и целостности файлов с использованием циклического кода (CRC32) и цифровых подписей пакета.

Требования к программным средствам центра управления:

1. Расчет проверочных символов для каждого файла пакета и формирование его цифровых подписей.

2. Реализация стандарта O-RAN.WG4.MP.0-R003 для обновления ПО.

3. Требуются средства контроля версий ПО и обеспечение обратной совместимости.

На рисунке 1 представлена разработанная архитектура ПО: ПМУ – программный модуль управления, СУСС – системы управления сетью и сервисами, Lib O1 – разрабатываемая библиотека, которая обеспечивает взаимодействие через IPC (межпроцессорное взаимодействие), а также получает управляющие сообщения от СУСС через REST/HTTPS.

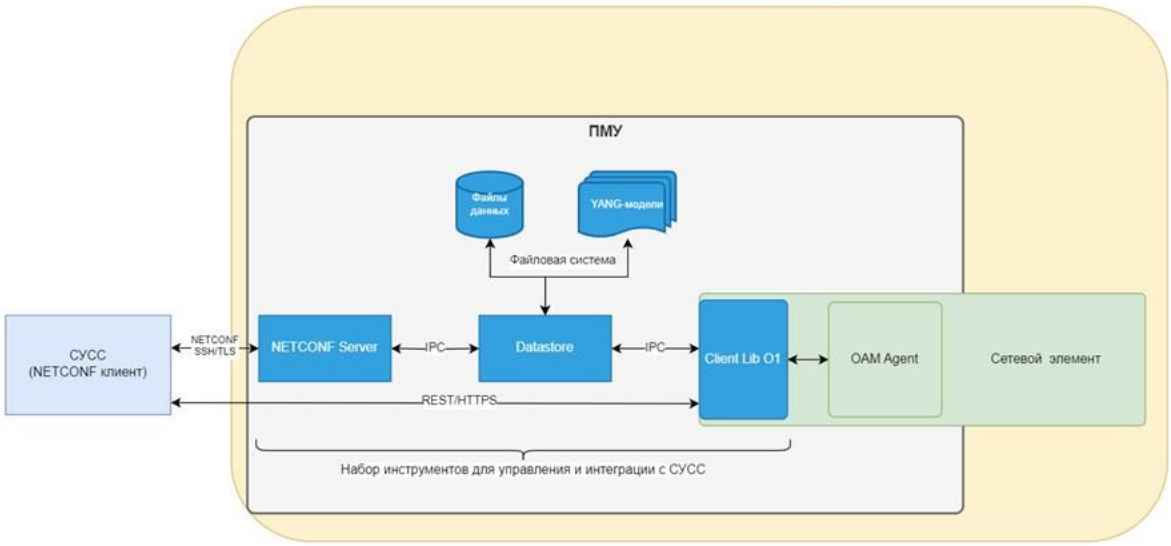


Рисунок 1 – Архитектура ПО

Библиотека состоит из следующих модулей:

1. Heartbeat – отслеживает состояние библиотеки.

2. Performance Metrics – измеряет производительность.

3. Alarm – отправляет оповещения о сбоях.

4. Software Package Management – управляет пакетами ПО.

5. Reset tracker – отслеживает причины перезагрузки.

6. Configuration change notifications – уведомляет об изменениях конфигурации (для хранения конфигурации использована Sysrepo [4]).

7. Event logger – регистрирует события.

Разработано приложение для создания пакетов программного обеспечения, которое формирует zip архив, содержащий файлы ПО и манифест в формате xml с описанием поставщика, номера версии, включенных файлов.

Вся разработка проведена с использованием языка программирования C++ 20 [5] для операционной системы Linux.

Разработанные программные средства будут использованы в составе системы обновления программного обеспечения космических аппаратов, разрабатываемых в компании «Бюро 1440».

ЛИТЕРАТУРА

1. ООО «Бюро 1440». [Электронный ресурс] Режим доступа: https://1440.space/.
2. O-RAN. [Электронный ресурс] Режим доступа: https://open-ran.org/
3. O-RAN ALLIANCE. [Электронный ресурс] Режим доступа: https://www.o-ran.org/.
4. Sysrepo. [Электронный ресурс] Режим доступа: https://www.sysrepo.org/
5. C++. [Электронный ресурс] Режим доступа: https://isocpp.org/

УДК 004.9

Р. А. Симонов (1 курс магистратуры),
П. А. Шагалова, к.т.н., доцент

РАЗРАБОТКА ПРИЛОЖЕНИЯ ДЛЯ ОБРАБОТКИ И УПРАВЛЕНИЯ ДАННЫМИ В ОБРАЗОВАТЕЛЬНЫХ СИСТЕМАХ

Развитие информационных технологий в образовании значительно улучшает качество и удобство обучения. Однако существующие IT-решения имеют недостатки, такие как ограниченная функциональность, сложность использования и высокая стоимость. Создание приложения для управления данными в образовательных системах решает эти проблемы. Такое приложение должно обладать расширенными возможностями, простым интерфейсом и доступностью для всех участников образовательного процесса. Это актуальное направление развития, которое упрощает и ускоряет работу преподавателей и студентов, повышает качество образования и оптимизирует учебный процесс.

Разработанное приложение представляет собой чат-бот на платформе Телеграмм. Данное приложение было разработано исходя из функциональных требований, которые включат в себя:

1. Стабильность работы системы - чат-бот реализован на языке программирования Python с использованием библиотеки iogram, которая является асинхронной и позволяет обрабатывать большой поток пользователей.
2. Масштабируемость - все компоненты платформы должны работать в контейнерах, что позволяет масштабировать платформу горизонтально и вертикально [4].
3. Автоматическое тестирование - позволяет выявить программные ошибки.

Разработанный чат-бот представляет собой модульную систему:

1. Рекомендательный модуль - основанный на машинном обучении, который позволяет получить ответы по вопросам поступления в ВУЗ, а также студентов – по стандартным вопросам учебного процесса, включая расписание и поиск преподавателя, а именно в какой аудитории он находиться [5, 7].
2. Модуль документооборота помогает в оформлении и заполнении документов. Данный модуль основывается на шаблонах и сравнении конечного результата с эталоном, что позволяет исключить некорректное заполнение документов, тем самым исключив фактор ошибок от ручного заполнения документов.

Разработанный бот входит в поддерживаемую платформу, которая включает в себя средства автоматизации и обработки данных, представлена архитектурно на рисунке 1. Данная архитектура включает не только разрабатываемое приложение(логику), но, а также вспомогательные компоненты, которые автоматизируют процесс развертывания приложения, мониторинга и хранения данных. Вспомогательные компоненты включают в себя:

1. ELK который позволяет централизировано вести журнал для выявления проблем каждого модуля. Данный инструмент поможет в дальнейшем интегрировать новые модули системы и централизировано вести процесс их логирования [2].
2. Zabbix – средство мониторинга, которое позволяет мониторить инфраструктурные компоненты системы, такие как сетевой трафик, нагрузку на систему в целом, что позволяет предпринять необходимые меры на этапе анализа метрик.
3. База данных, представлена в лице PostgreSQL, которая позволяет хранить данные чат-бота, а также метрики средства мониторинга Zabbix.
4. Ansible – средство управления конфигурациями, которое позволяет развернуть компоненты платформы или обновление уже имеющихся.

Так же разработаны автотесты имитирующие конечного пользователя, данный этап позволяет на этапе разработки, выявить все критические и блокирующие проблемы системы. Что позволяет на ранних этапах исправить все недостатки системы, тем самым сэкономив ресурсы команды проекта [1, 6].

Интеграция дополнительных компонентов (которые развернуты в контейнерах Docker) позволяют легко масштабировать систему в дальнейшем [3]. С системой взаимодействуют конечные пользователи при обращении к функционалу. А также инженер, который осуществляет развертывание нового кода чат-бота, в том числе и мониторинг всей системы.

Непосредственное взаимодействие с функционалом системы происходит на устройстве конечного пользователя: персональный компьютер, смартфон. Взаимодействие с дополнительными компонентами и логикой (кодом) осуществляется инженером посредством запуска сценариев Ansible.

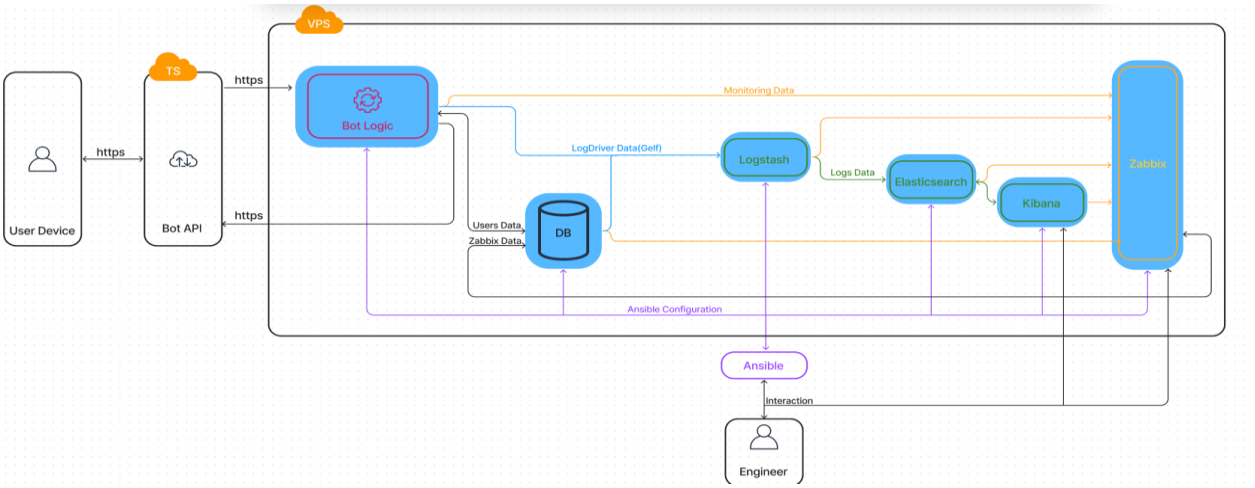


Рисунок 1 – Архитектура системы

В рамках проделанной работы была осуществлена разработка приложения, цель которого заключалась в обработке и управлении данными в образовательных системах. Главная цель данной разработки заключалась в создании функционального, простого в использовании и эффективного приложения, которое способствует улучшению эффективности и качества обучения через оптимизацию процесса работы с данными. Разработка подобных приложений для образовательных систем – является важной и перспективной задачей, нацеленной на модернизацию и улучшение образовательного процесса. Благодаря применению новейших технологий и инструментов разработанное приложение представляет собой универсальную, надежную и удобную платформу, которая будет полезна как для образовательных учреждений, так и для преподавателей и студентов.

ЛИТЕРАТУРА

1. Аниче М. Эффективное тестирование программного обеспечения / М. Аниче. – М.: ДМК Пресс, 2023. – 370 с.

2. Гвоздева, В. А. Информатика, автоматизированные информационные технологии и системы / В.А. Гвоздева. - М.: Форум, Инфра-М, 2017. – 544 с.
3. Криницкий, Н.А. Автоматизированные информационные системы / Н.А. Криницкий, Г.А. Миронов, Г.Д. Фролов. - М.: Наука, 2017. – 382 с.
4. Любарский, Ю.Я. Интеллектуальные информационные системы / Ю.Я. Любарский. - М.: Наука, 2016. – 232 с.
5. Машинное обучение: что такое обработка естественного языка? [Электронный ресурс]. Режим доступа: https://blog.lingoda.com/ru/obrabotka-yestestvennogoyazyka/
6. Савин Р. Тестирование Дот Ком, или Пособие по жесткому обращению с багами в интернет-стартапах / Р. Савин. – М.: Ридеро, 2017. – 312 с.
7. Что такое NLP в машинном обучении [Электронный ресурс]. Режим доступа: https://mchost.ru/articles/chto-takoe-nlp-v-mashinnom-obuchenii/

УДК 004.453

И. О. Смирнов, А. А. Литвинов (4 курс бакалавриата),
Н. В. Воинов, к.т.н., доцент

РАЗРАБОТКА ВЕБ-САЙТА ДЛЯ ОНЛАЙН-КУРСОВ ПО ПРОГРАММИРОВАНИЮ

В условиях стремительного развития технологий и всеобщего доступа к интернету онлайн-образование становится все более популярным среди пользователей [1]. Особое внимание привлекает область онлайн-курсов, которая предоставляет уникальную возможность обучаться и повышать квалификацию в удобном формате, доступном каждому.

Развитие IT-индустрии и программирования вносит значительный вклад в рост спроса на обучение в соответствующих сферах. Этот рост активности стимулирует развитие онлайн-платформ для обучения. Создание веб-приложений, предлагающих курсы по программированию и включающих функционал тестирования знаний, играет важную роль в совершенствовании образовательных технологий [1,2]. Такие приложения обеспечивают удобный и доступный способ обучения, а также дают возможность студентам оценивать свой прогресс и уровень владения материалом. Этот шаг в развитии онлайн-образования отвечает на растущий спрос на обучение в IT-сфере и обогащает образовательную среду новыми возможностями и инструментами для эффективного обучения.

Не менее важно отметить тот факт, что существующие решения на рынке услуг не могут в полной мере удовлетворить потребности всей потенциальной аудитории. Анализ аналогов показал, что в предлагаемых системах отсутствует качественное предварительное тестирование знаний. Такая функциональность позволит не перегружать информацией потенциальных пользователей и предоставит возможность выбирать курсы самостоятельно, исходя из полученных результатов. Именно поэтому основной особенностью разрабатываемого проекта будет реализация проведения тестовых опросов, которые помогут определить уровень знания в данной области потенциального обучающегося в рамках конкретных модулей.

Целью данной работы является разработка веб-сайта для онлайн-курсов по программированию, который предоставит студентам доступ к качественным образовательным материалам и практическим заданиям, необходимым только им. Задача заключается в создании удобной и эффективной образовательной платформы, способной удовлетворить потребности аудитории. Проект представляет собой разработку веб-сайта, предназначенного для обучения программированию в онлайн-формате. Веб-сайт будет являться образовательной платформой, предоставляющей студентам доступ к различным курсам и обучающим материалам по программированию.

В рамках работы необходимо решить следующие задачи:

– Обзор существующих решений.

– Разработка удобного интерфейса.

– Обеспечение доступа к качественным образовательным материалам.

– Гарантия конфиденциальности и безопасности данных.

– Реализация клиентской и серверной части.

– Демонстрация работы проекта.

Для реализации поставленных задач будут применяться следующие технологии:

– Язык программирования Java для реализации серверной части приложения [3].

– Фреймворк Spring Boot для упрощения процесса за счет интеграции с другими компонентами Spring, встроенными средствами безопасности и т.д. [4]

– СУБД PostgreSQL для управления базами данных [5].

– JavaScript для реализации клиентской части приложения [6].

– React для построения пользовательского интерфейса [7].

Выбранный стек технологий обеспечивает все необходимые инструменты для успешной реализации поставленных задач.

ЛИТЕРАТУРА

1 Analysis of student activity on the e-learning course based on "OpenEdu" platform logs / N. D. Barsukov, I. M. Sysoev, A. A. Pereskokova [et al.] // Proceedings of the Institute for System Programming of the RAS. – 2020. – Vol. 32, No. 3. – P. 91-100. – DOI 10.15514/ISPRAS-2020-32(3)-8. – EDN QYVJBQ.

2. Ивлев, В. А. Обработка данных в геоинформационных системах для выбора местоположения рекламы / В. А. Ивлев, И. В. Никифоров, Т. В. Леонтьева // Современные технологии в теории и практике программирования: сборник материалов конференции, Санкт-Петербург, 19 апреля 2019 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2019. – С. 27-30. – EDN DNQPMC.

3. Бойков, К. М. Разработка клиент-серверной архитекутры для сервиса по управлению интерактивными подписками / К. М. Бойков, О. С. Командина, Н. В. Воинов // Современные технологии в теории и практике программирования: сборник материалов конференции, Санкт-Петербург, 23 апреля 2020 года / Санкт-Петербургский политехнический университет Петра Великого; Dell Technologies; EPAM Systems. – Санкт-Петербург: ПОЛИТЕХ-ПРЕСС, 2020. – С. 113-114. – EDN DTYFAF.

4. 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. – EDN CFRNSJ.

5. Smirnov, P. A. A Web Application to Promote Blood Donation in Russia / P. A. Smirnov, V. V. Malinovskaya, N. V. Voinov // Proceedings of the Institute for System Programming of the RAS. – 2022. – Vol. 34, No. 2. – P. 179-190. – DOI 10.15514/ISPRAS-2022-34(2)-14. – EDN SGENXV.

6. Чавес, К. Разработка javascript-фреймворка для генерации серверных веб-приложений / К. Чавес, Н. В. Воинов, Е. В. Каплан // Современные технологии в теории и практике программирования: Сборник материалов конференции, Санкт-Петербург, 26 апреля 2022 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2022. – С. 108-110. – EDN SHFUQC.

7. Применение нейронной сети для оценки пользовательского опыта работы с приложением / В. А. Юркин, С. Э. Сараджишвили, Н. В. Воинов, С. А. Молодяков // IV Международная конференция по нейронным сетям и нейротехнологиям (NeuroNt'2023): Сборник докладов конференции, Санкт-Петербург, 16 июня 2023 года. – Санкт-Петербург: Санкт-Петербургский государственный электротехнический университет "ЛЭТИ" им. В.И. Ульянова (Ленина), 2023. – С. 72-75. – EDN YNANGK.

УДК 004.418

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

ОЦЕНКА СОЛНЕЧНОЙ ГЕНЕРАЦИИ ДЛЯ РАСЧЕТА МОЩНОСТИ СЭС

В современном мире качество и стабильность электроснабжения играют важную роль. Перебои в электроснабжении даже небольших населённых пунктов чреваты финансовыми потерями. Например, в отопительный сезон прекращение электроснабжения может привести к замерзанию воды в линиях теплоснабжения и выходу их из строя. В последние годы ведутся работы по модернизации автономных дизельных электростанций. Энергия, вырабатываемая на таких станциях, имеет большую цену ввиду дороговизны дизельного топлива и сложности его доставки в отдалённые районы страны, где зачастую и располагаются такие электростанции. Идея модернизации сводится к добавлению возобновляемых источников энергии, в частности солнечных электростанций, для экономии дизельного топлива. Таким образом, дизельная электростанция превращается в гибридную. Однако солнечная генерация, в отличие от дизельной, является нестабильной. Помимо сезонности проблемы вносит и внезапное снижение генерации из-за уменьшения солнечной инсоляции, например, вследствие перекрытия солнца облаками. Для автономных систем такое не прогнозируемое и быстрое снижение генерации критично, поскольку возможно нарушение баланса мощностей. Это может приводить либо к работе дизельных генераторов под критически высокой нагрузкой, либо к аварийной остановке оборудования из-за возникшего дефицита энергии.

Для борьбы с данной проблемой необходимо вводить в работу дополнительные дизельные генераторы, которые будут работать вхолостую под небольшой нагрузкой, но иметь достаточный запас мощности, чтобы в случае снижения солнечной генерации компенсировать возникший дефицит. Такое решение приводит к значительному снижению экономии дизельного топлива.

Другим решением является установка системы накопления энергии, которая сможет быстро компенсировать дефицит мощности. Однако такие системы достаточно дорогие, а также требуют поддержания приемлемого уровня заряда для возможности своевременной и полной компенсации дефицита.

Наиболее интересным представляется внедрение системы, которая способна прогнозировать снижение солнечной генерации. Если такая система прогнозирования сможет предсказать снижение солнечной генерации за одну-пять минут до непосредственного снижения, это позволит своевременно ввести в работу дополнительные дизельные генераторы, тем самым обеспечив необходимый запас мощности в энергосистеме на случай снижения солнечной генерации.

Проект нацелен на повышение точности прогнозирования вырабатываемой мощности солнечной электростанции путем разработки программной системы, которая реализует подход к прогнозированию, основанный на новом методе сегментации облачности на изображениях полного неба.

Основным параметром, влияющим на изменение располагаемой мощности солнечной электростанции (СЭС), является уровень солнечного излучения [1]. Он складывается из двух составляющих: прямой нормальной освещенности DNI (Direct Normal Irradiation) и рассеянного горизонтального излучения DHI (Diffuse Horizontal Irradiance). DNI - это излучение, которое исходит непосредственно от солнечного пятна, DHI - это рассеянное излучение поступающее со всех направлений, кроме солнечного пятна. Их суммой с учетом направления DNI является величина глобальной горизонтальной освещенности GHI (Global Horizontal Irradiance). Значение GHI показывает, сколько Ватт энергии излучения поступает на один квадратный метр горизонтальной поверхности.

Существуют программные имплементации соответствующих математических моделей, позволяющих рассчитать мощность СЭС по известным характеристикам солнечного излучения. К таким программным решениям относится библиотека PVLib [2] или

программный пакет Pvsyst. Библиотека PVLib активно применяется в работах, посвящённых прогнозированию мощности СЭС [3,4]. Эта библиотека наравне с аналогичным ПО оперирует описанием схемы СЭС, включающей параметры и положение солнечных панелей, характеристики солнечных инверторов. Наличие таких инструментов позволяет свести задачу прогнозирования мощности СЭС к задаче прогнозирования солнечного излучения.

В проекте для прогнозирования солнечного излучения будет использоваться подход на основе явного прогнозирования облачной сцены. При таком подходе на точность прогноза оказывает наибольшее влияние точность сегментации облаков при анализе изображений полного неба. Существуют различные методы сегментации облаков, наиболее распространенные их которых - RBR (Read Blue Ration) [5], BRBG (Red-Blue Red-Green Ratio) [6], CSL (Clear Sky Library) [7], HYTA (Hybrid Thresholding Algorithm) [8], SACI (Smart adaptive cloud identification method) [9]. Результаты сравнения точности работы данных методов на размеченных вручную изображениях показали, что методы склонны ошибаться в околосолнечной области, за исключением метода CSL. Однако и он имеет ряд недостатков: необходимость формирования базы данных с изображениями чистого неба, снижение точности при несовпадении положения солнца на исходном и референсном изображениях, а также других параметров, например, мутности атмосферы. Для устранения недостатков предлагается собственный метод сегментации в рамках подхода к прогнозированию солнечного излучения.

ЛИТЕРАТУРА

1. Smets A. et al. Solar Energy: The Physics and Engineering of Photovoltaic Conversion //Technologies and Systems. – 2016.
2. Stein J. S. et al. PVLIB: Open source photovoltaic performance modeling functions for Matlab and Python //2016 ieee 43rd photovoltaic specialists conference (pvsc). – IEEE, 2016. – С. 3425-3430.
3. Holmgren W. F., Lorenzo A. T., Hansen C. A comparison of pv power forecasts using pvlib-python //2017 IEEE 44th Photovoltaic Specialist Conference (PVSC). – IEEE, 2017. – С. 1127-1131.
4. Holmgren W. F., Groenendyk D. G. An open source solar power forecasting tool using PVLIB-Python //2016 ieee 43rd photovoltaic specialists conference (pvsc). – IEEE, 2016. – С. 972-975.
5. Kassianov E., Long C. N., Ovtchinnikov M. Cloud sky cover versus cloud fraction: Whole-sky simulations and observations //Journal of Applied Meteorology and Climatology. – 2005. – Т. 44. – №. 1. – С. 86-98.
6. Tang J. et al. An improved cloud recognition and classification method for photovoltaic power prediction based on total-sky-images //The Journal of Engineering. – 2019. – Т. 2019. – №. 18. – С. 4922-4926.
7. Ghonima M. S. et al. A method for cloud detection and opacity classification based on ground based sky imagery //Atmospheric Measurement Techniques. – 2012. – Т. 5. – №. 11. – С. 2881-2892
8. Li Q., Lu W., Yang J. A hybrid thresholding algorithm for cloud detection on ground-based color images //Journal of atmospheric and oceanic technology. – 2011. – Т. 28. – №. 10. – С. 1286-1296.
9. Chu Y. et al. A smart image-based cloud detection system for intrahour solar irradiance forecasts //Journal of Atmospheric and Oceanic Technology. – 2014. – Т. 31. – №. 9. – С. 1995-2007.

УДК 004.42

Г. В. Смородников, Р. А. Золотарев (1 курс магистратуры),
Р. Р. Попов, К. А. Каширин (1 курс магистратуры)

РАЗРАБОТКА СИСТЕМЫ АНАЛИЗА ДАННЫХ СЕРВИСА КИНОПОИСК

С ростом цифровизации в современном обществе объем данных, создаваемых в киноиндустрии, стремительно увеличивается, создавая потребность в эффективных методах анализа для выявления трендов и понимания предпочтений зрителей. Это обуславливает актуальность и перспективность разработки системы анализа данных, основанной на информации от ведущих ресурсов о кинематографе, таких как Кинопоиск.

Киноиндустрия остается одной из наиболее прибыльных развлекательных отраслей в мире, несмотря на вызванный COVID-19 спад, и продолжает ежегодно выпускать тысячи фильмов и сериалов [2]. Этот стремительный поток информации представляет сложности в его анализе и интерпретации, требуя эффективных методов обработки и аналитики больших данных.

Кинопоиск, как один из крупнейших онлайн-сервисов о кино в мире, содержит огромное количество данных о фильмах, их оценках, рецензиях, а также информацию об актерах и других участниках индустрии [1]. Обработка такого объема информации представляет собой сложную задачу из-за необходимости эффективной обработки больших данных и выделения значимых закономерностей.

Целью данной работы является разработка комплексной системы для сбора, обработки и хранения больших массивов информации с ведущего ресурса о кинематографе - Кинопоиска. Эта система будет предназначена для последующего анализа и визуализации данных, что позволит исследователям, аналитикам и другим заинтересованным сторонам извлекать ценные инсайты из обширной базы данных о фильмах, сериалах, актерах и других участниках кинопроизводства.

Для достижения этой цели необходимо решение следующих *задач*:

1. Создание архитектуры системы выгрузки, обработки, хранения и визуализации данных.
2. Реализация выгрузки большого количества данных в систему.
3. Статистическая обработка выгруженных данных.
4. Визуализация результатов обработки.

Архитектура проекта представлена на рисунке 1, она включает четыре основных модуля: модуль источника данных (Data source), который использует сторонний API Кинопоиска [3], модуль обработки данных (ETL), содержащий сервисы по выгрузке, обработке и хранению данных, модуль хранения (Storage layer) и модуль визуализации (Presentation layer). Модуль ETL состоит из трех основных сервисов: сервис API Data Extractor, отвечающий за выгрузку данных с API с помощью планировщика задач Celery для контроля объема данных и избежания превышения лимитов использования [4]; сервис API Data Receiver, который получает данные и передает их в брокер RabbitMQ для эффективного распределения нагрузки на обработку [5]; и сервис API Data Normalizer, использующий Apache Spark [6] для обработки данных и передачи их в систему хранения ElasticSearch через Logstash для дальнейшего анализа и поиска [7].

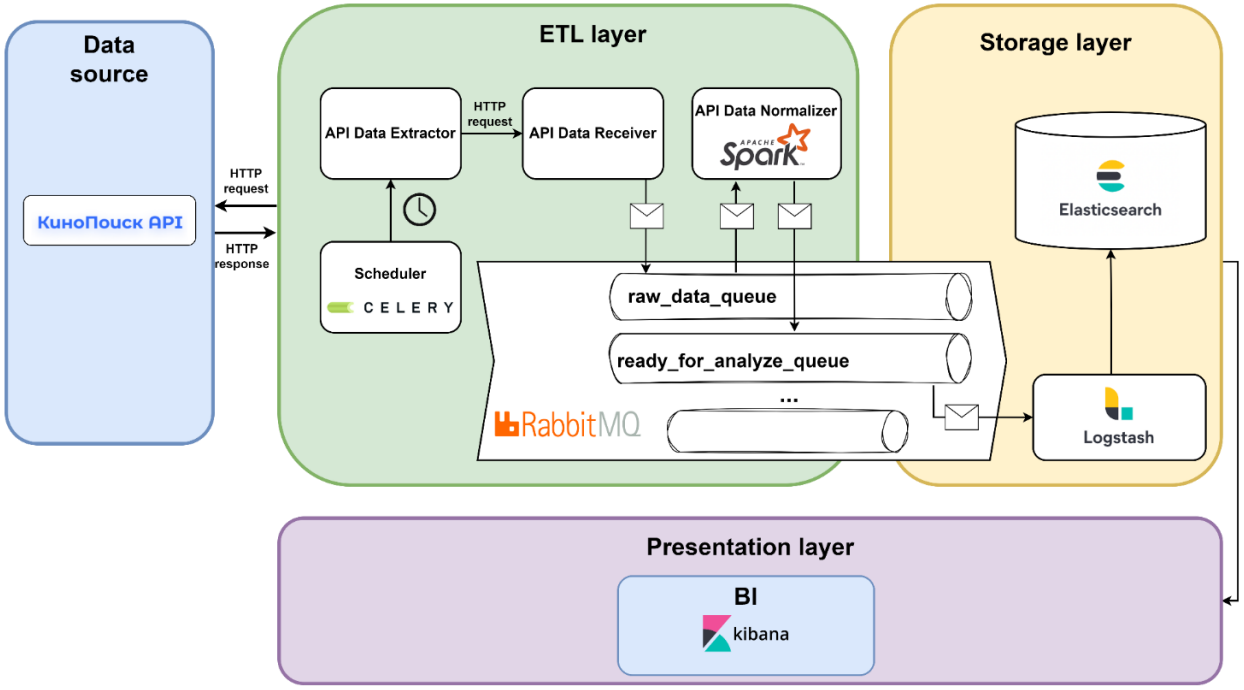


Рисунок 1 – Архитектура системы анализа данных сервиса Кинопоиск

В результате работы была успешно спроектирована и реализована система анализа данных о фильмах сервиса Кинопоиск. Процесс сбора данных был осложнен ограничениями стороннего API, такими как ограничение по количеству запросов и записей в запросе. Однако, благодаря стратегии использования нескольких API ключей и управлению квотами, удалось обеспечить регулярное и непрерывное обновление данных.

ЛИТЕРАТУРА

1. Кинопоиск [Электронный ресурс] Режим доступа: https://www.kinopoisk.ru/
2. Rahmouni, L. The Impact of COVID-19 on the Cinema Industry // ELWAHAT Journal for Research and Studies (2023) - 16. 1084-1099
3. Документация API Кинопоиск. [Электронный ресурс] Режим доступа: https://api.kinopoisk.dev/documentation
4. Планировщик задач Celery. [Электронный ресурс] Режим доступа: https://docs.celeryproject.org/en/stable/
5. RabbitMQ – брокер сообщений. [Электронный ресурс] Режим доступа: https://www.rabbitmq.com/
6. Apache Spark. [Электронный ресурс] Режим доступа: https://spark.apache.org/
7. Elastic stack. [Электронный ресурс] Режим доступа: https://www.elastic.co/what-is/elk-stack.

УДК 004.738

Д. И. Стахеев (4 курс бакалавриата),
О. В. Прокофьев, ст. преподаватель

ОРГАНИЗАЦИЯ УДАЛЁННОГО ДОСТУПА ДЛЯ ПРОЕКТА УМНОЙ ТЕПЛИЦЫ

Современное сельское хозяйство стоит перед множеством вызовов, среди которых изменение климата, уменьшение площади плодородных земель и растущее население планеты. Эти проблемы требуют новых подходов в агрокультуре, направленных на повышение эффективности и производительности. С развитием технологий Интернета вещей (Internet of Things, IoT) и повышением доступности микрокомпьютеров появляются новые возможности для автоматизации сельскохозяйственных процессов. Особенно актуальным становится создание умных теплиц, способных автоматически контролировать внутренние условия с целью оптимизации роста растений.

Основная цель данной работы - организация системы удалённого доступа для проекта умной теплицы. Это позволит не только автоматизировать управление климатическими условиями в теплице, но и предоставит возможность удалённого мониторинга и управления.

Основой для организации удалённого доступа является Internet-протокол, но здесь возникает серьёзная проблема - исчерпание доступных IP-адресов в пространстве IPv4 [1]. С ростом числа устройств, подключенных к сети Интернет, спрос на IP-адреса значительно превысил предложение, что привело к необходимости использования механизмов экономии адресного пространства. Переход на IPv6 является одним из основных решений данной проблемы, поскольку IPv6 обеспечивает практически неограниченное количество адресов (340 ундециллионов). Однако этот процесс затягивается из-за больших финансовых затрат и сложностей в обновлении существующих сетевых инфраструктур. В качестве промежуточного решения используется другой механизм - преобразование сетевых адресов, также известный как Network Address Translation (NAT) [2].

Но при этом возникает проблема двойного NAT [3]. Она проявляется, когда устройство находится за несколькими шлюзами, обеспечивающими NAT, что может создать сложности при установке соединений. Для решения данной проблемы используются VPN-технологии, технологии туннелирования и P2P-сетей, а также сервисы прямого доступа. В таблице 1 приведены результаты сравнения наиболее популярных представителей каждой категории.

Таблица 1 – Сравнительная характеристика решений

	Название решения					
Критерий	OpenVPN	WireGuard	ZeroTier	Tailscale	Ngrok	LocalTunnel
Сложность установки	Высокая	Средняя	Низкая	Низкая	Низкая	Низкая
Гибкость настройки	Высокая	Высокая	Средняя	Средняя	Низкая	Низкая
Безопасность	Высокая	Высокая	Высокая	Высокая	Средняя	Средняя
Производительность	Средняя	Высокая	Низкая	Низкая	Высокая	Высокая
Масштабируемость	Высокая	Высокая	Средняя	Средняя	Средняя	Высокая
Зависимость от внешних сервисов	Низкая	Низкая	Средняя	Высокая	Высокая	Низкая

В качестве решения для организации удалённого доступа был выбран LocalTunnel [4] из-за открытости исходного кода, отсутствия ограничений в бесплатной версии, высокой производительности и возможности размещения узлов маршрутизации на собственных серверах. Также одним из наиболее важных критериев выбора стало отсутствие необходимости в использовании специального программного обеспечения на конечных устройствах. Наконец, код LocalTunnel может быть улучшен как с точки зрения безопасности, так и с точки зрения соответствия современным требованиям функциональности.

В качестве альтернативного и в то же время резервного способа управления может быть использован Telegram-бот, размещённый внутри контура Wi-Fi сети умной теплицы.

Таким образом, общая схема сетевой инфраструктуры умной теплицы примет вид, изображённый на рисунке 1.

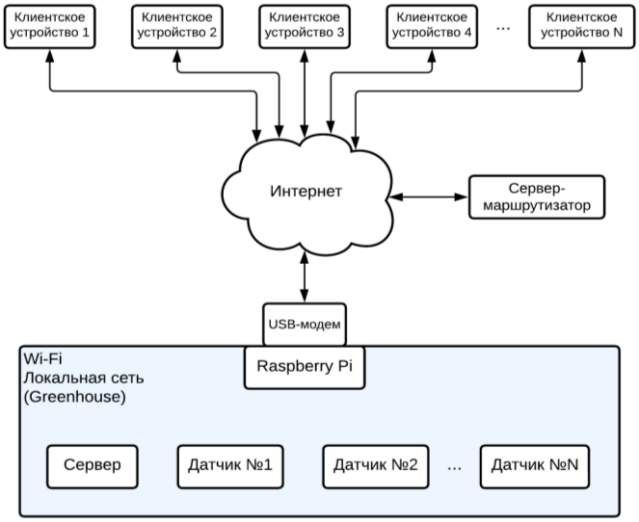


Рисунок 1 – Общая схема сетевой инфраструктуры

Диаграмма последовательности взаимодействия компонентов сетевой инфраструктуры будет иметь вид, представленный на рисунке 2.

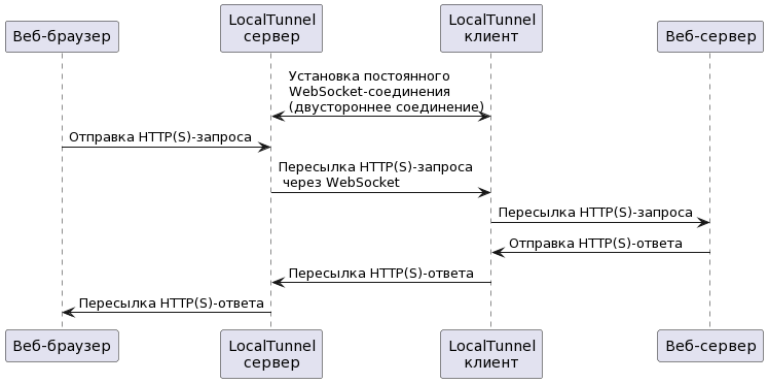


Рисунок 2 – Взаимодействие компонентов сетевой инфраструктуры

При этом немаловажным вопросом остаётся обеспечение информационной безопасности, поскольку каждый компонент может быть атакован [5] - начиная от самой Wi-Fi сети, заканчивая конечным устройством клиента.

ЛИТЕРАТУРА

1. Безопасность сети Интернет: модели, инструменты, методики: учебное пособие / Д. П. Зегжда, А. В. Козачок, Е. Ю. Павленко [и др.]; Санкт-Петербургский политехнический университет Петра Великого, Институт кибербезопасности и защиты информации. — Санкт-Петербург: ПОЛИТЕХ-ПРЕСС, 2022. – 115 с.
2. Huawei Technologies Co., Ltd.. (2023). Network Address Translation Technologies. In: Data Communications and Network Technologies. Springer, Singapore. https://doi.org/10.1007/978-981-19-3029-4_10
3. M. Karimzadeh, L. Valtulina, A. Pras et al., “Double-NAT Based Mobility Management for Future LTE Networks,” in Proceedings of the 2017 IEEE Wireless Communications and Networking Conference (WCNC), pp. 3–6, San Francisco, Calif, USA, March 2017
4. Localtunnel - Expose yourself to the world. [Электронный ресурс] Режим доступа: https://theboroer.github.io/localtunnel-www/
5. C. Kohlios and T. Hayajneh, ‘‘A comprehensive attack flow model and security analysis for Wi-Fi and WPA3,’’ Electronics, vol. 7, no. 11, p. 284, October 2018

УДК 004.9

В. О. Струк (2 курс магистратуры),
Г. Ф. Малыхина, д.т.н., профессор

БИБЛИОТЕКА ПРОГРАММ ДЛЯ ЗАЩИТЫ ИЗМЕРИТЕЛЬНОЙ ИНФОРМАЦИИ НА ТЭЦ

Тепловые электростанции (ТЭЦ) играют важную роль в энергетике России. Они являются основными источниками генерации электроэнергии и обеспечивают энергоснабжение во многих регионах страны [1].

Актуальность работы заключается в том, что на тепловой электроцентрали имеется система управления котлами, которая обеспечивает контроль работы котлов в соответствии с параметрами. Эти параметры хранятся в озере данных, откуда в виде измерительной информации передаются внутри предприятия на расстояние до 15 км. Изменение или удаление измерительных данных может привести к неправильной работе котлов и к катастрофе на предприятии.

На тепловой электроцентрали действуют меры защиты информации от злоумышленников, однако они не могут полностью предотвратить изменение данных, поскольку злоумышленник сможет обойти эти меры защиты.

Используются следующие инструменты для защиты информации.

Wallix AdminBastion. Программа для управления привилегированным доступом. Позволяет организациям развёртывать решение в соответствии с потребностями макроуровня, запуская функции по одной за раз, чтобы облегчить долгосрочный и устойчивый подход к обеспечению безопасности доступа к привилегиям [2]. Однако злоумышленник путём подбора пароля может запросто получить доступ к информации.

Cisco Firepower. Первый в отрасли полностью интегрированный межсетевой экран нового поколения с ориентацией на защиту от угроз и с унифицированным управлением [3]. Он обеспечивает расширенную защиту от угроз до, во время и после атак, но злоумышленник, поменяв свой реальный IP – адрес, может запросто обойти данный механизм защиты и изменить информацию.

Таким образом, вышеперечисленные инструменты не могут обеспечить надёжную защиту информации. Нужно разработать такой продукт, чтобы, перехватив информацию, злоумышленник не смог её прочитать или изменить.

Объектом исследования является защита измерительной информации на ТЭЦ.

Предметом исследования является создание надёжного метода защиты измерительной информации.

Целью работы является разработка библиотек программ для защиты измерительной информации на ТЭЦ, благодаря которым злоумышленник, получив информацию, не сможет её прочитать или изменить.

Задачами доклада являются:

– Провести аналитический обзор методов шифрования информации;
– Разработать методологию целостности измерительных данных ТЭЦ;
– Разработать методологию криптографической защиты данных ТЭЦ;
– Разработать библиотеку программ для защиты измерительной информации на ТЭЦ.

Основные этапы работы:

– Изучение методов криптографической защиты информации;
– Изучение методов шифрования информации;
– Разработка алгоритма создания дайджеста пакета измерительных данных
– Разработка алгоритма создания кода аутентичности сообщения пакета измерительных данных;
– Разработка алгоритма создания цифровой подписи пакета измерительных данных;
– Разработка алгоритма шифрования пакета измерительных данных;
– Реализация данных алгоритмов на языке Python;
– Тестирование программного обеспечения.

Научная новизна работы заключается в следующем:

– Разработка системы защиты измерительной информации на тепловой электроцентрали с использованием методов криптографии;
– Разработка программного обеспечения, благодаря которому измерительная информация будет защищена и злоумышленник не сможет прочитать её и изменить.

При разработке библиотеки программ измерительных данных были использованы следующие стандарты:

– ГОСТ Р 8.654-2015 Требования к программному обеспечению средств измерений [4];
– ГОСТ Р 8.883 – 2015 Государственная система обеспечения единства измерений. Программное обеспечение средств измерений. Алгоритмы обработки, хранения, защиты и передачи информации. Методы испытаний [5];
– ГОСТ Р 34.11 -2012 Криптографическая защита информации. Функция хэширования [6];
– ГОСТ Р 50.1.113 -2016 Криптографическая защита информации. Криптографические алгоритмы, сопутствующие применению алгоритмов электронной подписи и функции хэширования [7].

В результате выполнения работы было создано программное обеспечение, с помощью которого можно обеспечить криптографическую защиту информации любым из предложенных способов.

Были изучены методы шифрования информации и разработаны следующие алгоритмы методологии обеспечения целостности измерительной информации:

– Дайджест;
– Код аутентификации;
– Цифровая подпись;
– Шифрование.

После этого были выбраны инструменты для разработки программного обеспечения. В конце был написан код программы на языке программирования Python.

ЛИТЕРАТУРА

1. ТЭС в России: энергетика и электростанции [Электронный ресурс] Режим доступа: https://uproger.com/szhatie-tekstovyh-dannyh-metodom-arifmeticheskogo-kodirovaniya
2. Wallix vs balabit. Сравнение ПО по контролю админов [rdp] [Электронный ресурс] Режим доступа: https://habr.com/ru/articles/262083/

3. Cisco FirePOWER система сетевой безопасности [Электронный ресурс] Режим доступа: https://habr.com/ru/articles/262083/
4. ГОСТ Р 8.654 – 2015. Государственная система обеспечения единства измерений. Требования к программному обеспечению средств измерений. Введ. 2015-04-28. – Москва, Стандартинформ, 2019. – 9 с.
5. ГОСТ Р 8.883 – 2015. Государственная система обеспечения единства измерений.Программное обеспечение средств измерений. Алгоритмы обработки, хранения, защиты и передачи измерительной информации. Методы испытаний. Введ. 2015-04-28. – Москва, Стандартинформ, 2019. – 9 с.
6. ГОСТ Р 34.11 -2012. Криптографическая защита информации. Функция хэширования. Введ. 2012–08-07. – Москва, Стандартинформ, 2013. – 18 с.
7. ГОСТ Р 50.1.113 - 2016 Криптографическая защита информации. Криптографические алгоритмы, сопутствующие применению алгоритмов электронной подписи и функции хэширования. Введ. 2017-06-01. – Москва, Стандартинформ, 2016. – 16 с.

УДК 519.23, 519.254

Б. С. Мандрикова (аспирант),
А. Р. Лисс, д.т.н., профессор,

КОМПЛЕКСНЫЙ АЛГОРИТМ ОБНАРУЖЕНИЯ РАЗНОМАСШТАБНЫХ АНОМАЛИЙ В ПРИРОДНЫХ СИГНАЛАХ

Проблемой извлечения информации из данных является отсутствие полных априорных знаний об информационной составляющей и помехе. В ряде критических смежных областей (физика и техника, биология и медицина др.) эта проблема приводит к недостаточной эффективности существующих методов анализа данных. Особую актуальность имеют задачи обработки и анализа геофизических данных и обнаружения аномалий. Создание эффективных методов, алгоритмов и программных средств для решения таких задач является *целью* данной работы. *Объектом* исследования являются измерения нейтронных мониторов [1], регистрирующих интенсивность вариаций потоков космических лучей. Обнаружение аномалий по данным измерений нейтронных мониторов является важным прикладным аспектом исследований и направлено на предотвращение их негативного воздействия на здоровье людей и современную инфраструктуру [2]. Применяемые методы обработки и анализа данных измерений нейтронных мониторов не удовлетворяют требованиям оперативности (метод глобальной сьемки [3]) и точности (пороговые оценки [4]) и требуют разработки новых подходов в этой области.

В работе предложен *комплексный алгоритм обнаружения разномасштабных аномалий в природных сигналах*, основанный на применении нелинейных аппроксимирующих схем в вейвлет-базисах [5-7]. Нелинейные аппроксимирующие схемы используются для аппроксимации данных измерений интенсивности космических лучей (сигналов космических лучей) и обнаружения аномалий. Применяется непараметрический подход, и сигналы космических лучей рассматриваются как генерируемые изменяющимся во времени случайным процессом. Используются пороговые функций, параметры которых определяются адаптивно, по мере поступления данных в систему обработки. Использование вейвлет-базисов позволяет обнаружить аномалии разной формы и временной протяженности и оценить их параметры. Компактный носитель вейвлетов обеспечивает высокую степень локализации. Разработанный комплексный алгоритм позволяет по мере поступления данных в систему обработки обнаружить аномалии. Комплексный алгоритм включает в себя алгоритм определения информационных компонент сигнала и алгоритм адаптивной вейвлет-фильтрации. На рисунке 1 показан результат алгоритма определения информационных компонент сигнала, основанный на разработанном критерии выбора узлов дерева вейвлет-пакетов, содержащих информационные составляющие сигнала. Результат работы алгоритма адаптивной вейвлет-фильтрации представлен на рисунке 2 а. Алгоритм позволил обнаружить

аномалию в сигнале космических лучей за 3 ч до ее регистрации Международным центром прогнозов космической погоды (Space Weather Prediction Center), что подтверждает его высокую эффективность. Блок-схема *комплексного алгоритма обнаружения разномасштабных аномалий в природных сигналах и оценки их параметров* представлена на рисунке 2 б.

Работа выполнена за счет Гос. задания ИКИР ДВО РАН (рег. № темы 124012300245-2).

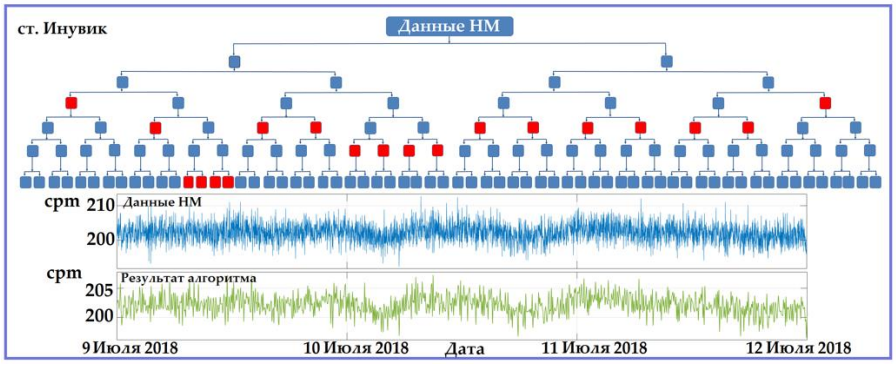
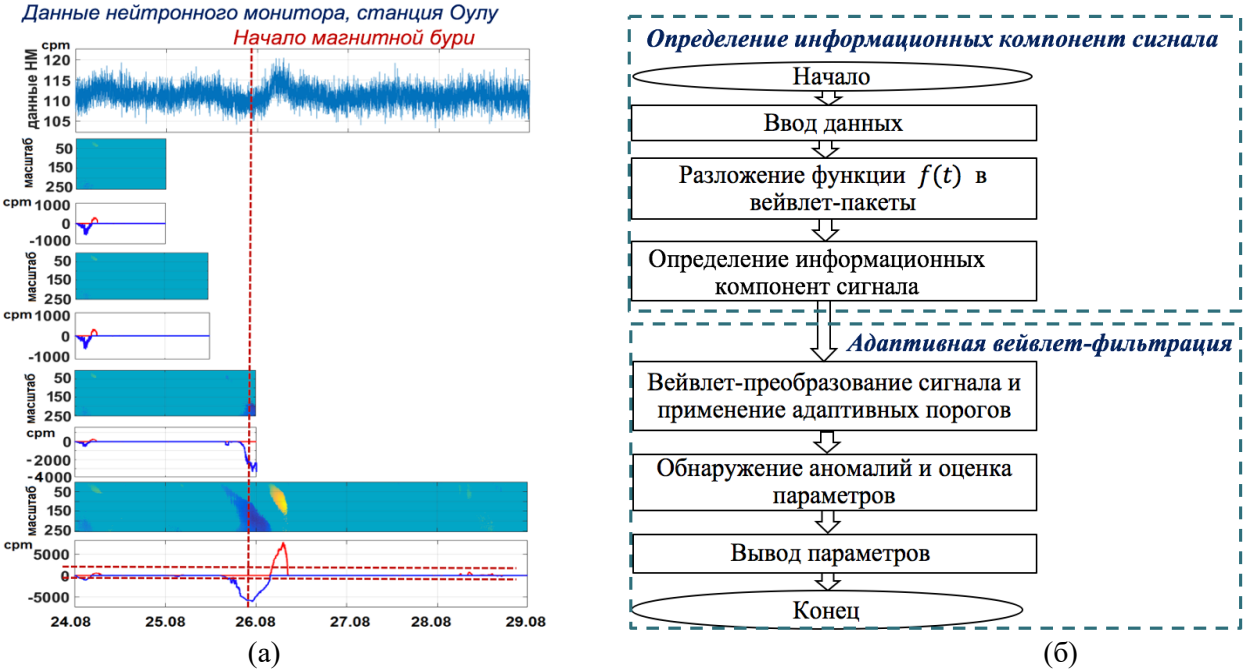


Рисунок 1 – Результат алгоритма определения информационных компонент сигнала



(а) (б)

Рисунок 2 а) Результат работы алгоритма адаптивной вейвлет-фильтрации; б) Блок-схема предложенного комплексного алгоритма

ЛИТЕРАТУРА

1. Neutron Monitor Data Base. [Электронный ресурс] Режим доступа: https://www.nmdb.eu.
2. Кузнецов В.Д. Космическая погода и риски космической деятельности // Космическая техника и технологии, 2014. – Т. 3 (6). – С. 3-13.
3. Белов А.В., Ерошенко Е.А., Янке В.Г., Оленева В.А., Абунина М.А., Абунин А.А. Метод глобальной съемки для мировой сети нейтронных мониторов // Геомагнетизм и аэрономия, 2018. – Т. 58 (3). – С. 374-389.
4. Mavromichalaki H., Paschalis P., Gerontidou M., et al. The updated GLE alert system by ANEMOS // Proceedings of the hybrid symposium on cosmic ray studies with neutron detectors, September 26-30, 2022. – P. 97-103.
5. Лисс А. Р., Мандрикова Б. С. Подавление шума и выделение когерентных структур сигнала на основе алгоритма нелинейной адаптации // Изв. СПбГЭТУ «ЛЭТИ», 2022. – Т. 15 (10). – С. 58-66. – DOI 10.32603/2071-8985-2022-15-10-58-66.
6. Мандрикова Б.С. Метод анализа сложных природных данных и обнаружения аномалий //

Информационные Технологии 2022. – Т. 28 (5). – С. 263–269. – DOI: 10.17587/it.28.263-269.

7. Геппенер В.В., Мандрикова Б.С. Обнаружение и идентификация аномалий в сложном сигнале // Автоматика и Телемеханика, 2021. – Н. 10. – С. 46-59. – DOI: 10.31857/S0005231021100056.

УДК 004.139

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

РАЗРАБОТКА IOS-ПРИЛОЖЕНИЯ ПО ГЕНЕРАЦИИ ИЗОБРАЖЕНИЙ

В настоящее время нейронные сети ворвались во все сферы нашей жизни и вывели их на новый уровень. Искусство, ставшее неотъемлемой частью нашей культуры, не могло остаться в стороне от этого технологического прорыва. Одним из самых популярных способов применения нейронных сетей является генерация изображений. Это положило начало новой эпохе в искусстве. Ранее люди вкладывали огромные усилия в создание оригинальных композиций. Теперь же с появлением нейронных сетей у них есть возможность автоматически создавать новые и уникальные изображения, которые поражают своей красотой. Это открывает неограниченные возможности для художников, дизайнеров и других творческих людей.

Все больше и больше людей отдают предпочтение мобильным приложениям. Основные преимущества заключаются в удобстве, быстром доступе, интуитивно понятном интерфейсе возможности настройки под себя. Однако, существует не так много инструментов генерации на мобильных устройствах. В процессе изучения рынка были выявлены 3 решения: Шедеврум [2], Кандинский [3] и Midjourney [4]. Однако только Шедеврум является приложением, остальные представляют из себя ботов в Telegram и Discord соответственно. В Шедевруме нет возможности настраивать seed и prompt. Так же там нельзя скачать изображение, не опубликовав его на своей странице.

Главной целью этой работы является разработка мобильного приложения для ОС IOS, которое позволит быстро и удобно генерировать изображения, а также иметь возможность настройки различных параметров процесса.

Макет приложения изображен на рисунке 1.

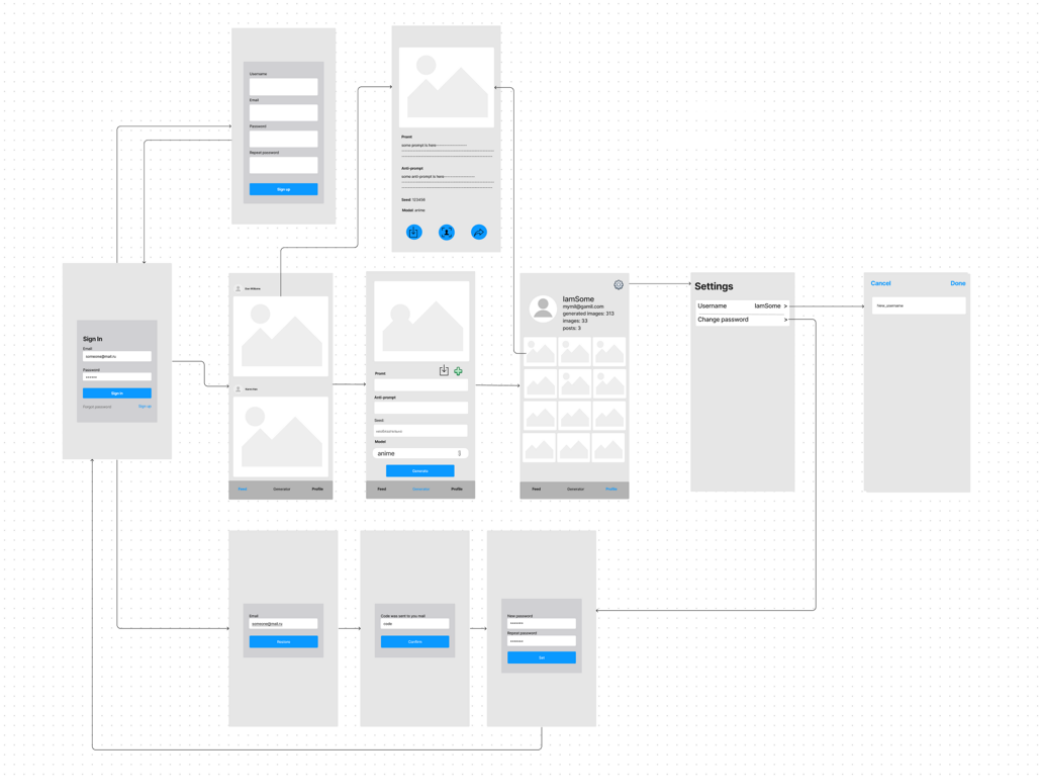


Рисунок 1 - Макет приложения

Описание схемы взаимодействия пользователя с приложением:

1. Пользователь входит в свой аккаунт и попадает на страницу с общей лентой постов
2. Если нет аккаунта, то регистрируется, после переходит на страницу входа
3. Если забыл пароль, то восстанавливает, проходя три этапа (ввод почты, код подтверждения, смену пароля), затем переходит на страницу входа
4. По таббару можно переходить между экранами: лента с постами, генератор, профиль
5. При нажатии на картинку в ленте или галерее открывается экран с подробной информацией об изображении и параметрах его генерации
6. При нажатии на кнопку в правом верхнем углу профиля осуществляется переход в настройки
7. При нажатии на смену имени в настройках откроется BottomSheet для смены имени
8. При нажатии на смену пароля в настройках откроется экран смены пароля

Вся данные пользователя хранятся на сервере. Бэкенд уже был реализован группой разработчиков. За основу взята нейронная сеть Stable-Diffusion [5]. База данных – PostgreSQL.

Основным языком программирования является Swift [7]. Для верстки интерфейса будет использоваться SwiftUI [6]. Он предлагает декларативный способ описания интерфейса. Его главной особенностью является функция "Live Preview", которая позволяет в реальном времени видеть изменения в интерфейсе при редактировании кода. К тому же он интегрирован с фреймворком Combine, который обеспечивает реактивное программирование в iOS-приложениях.

Шаблон проектирования Model-View-ViewModel [1]. MVVM является популярным паттерном, который разделяет логику и отображение данных, упрощая разработку и тестирование приложения. За пользовательский интерфейс отвечает View, за данные – Model, а ViewModel является диспетчером ответственным за извлечение данных и преобразование их в необходимый формат.

Для обеспечения хорошей производительности и оптимизации работы с изображениями, в приложении также используются различные техники, такие как кэширование изображений, асинхронная загрузка и обработка данных. Для управления многопоточностью используется фрэймворк Grand Central Dispatch (GCD). Он автоматически управляет распределением задач по доступным ядрам процессора. GCD выполняет задачи асинхронно, что позволяет избежать блокировки главного потока и обеспечить отзывчивость пользовательского интерфейса.

Другим важным аспектом разработки является поддержка различных устройств и ориентаций экрана. Для этого используются Auto Layout. Он позволяет создавать адаптивный пользовательский интерфейс, который корректно отображается на различных устройствах и в различных ориентациях. Также в приложении будут различные анимированные элементы, которые добавят интерактивности и привлекательности пользовательскому интерфейсу. Для управления анимацией используется фрэймворк Core Animation, позволяющий создавать плавные и эффектные переходы между различными состояниями приложения.

Главным результатом работы должно стать приложение, которое будет удобным в использовании даже простым пользователям. Оно предоставит больший контроль над процессом генерации, чем другие программные продукты.

ЛИТЕРАТУРА

1. Эрих Гамма, Джон Влисидис, Ричард Хелм, Ральф Джонсон Приёмы объектно-ориентированного проектирования. Па́ттерны проектирования. - СПб: "Издательский дом ""Питер""", 2020. - 448 с.
2. Шедеврум [Электронный ресурс] – URL: https://shedevrum.ai - (дата обращения: 15.02.2024)
3. Kandinsky [Электронный ресурс] – URL: https://t.me/kandinsky21_bot - (дата обращения: 15.02.2024)
4. Midjourney [Электронный ресурс] – URL: https://www.midjourney.com/home?callbackUrl=%2Fexplore - (дата обращения: 15.02.2024)
5. Stable Diffusion [Электронный ресурс] – URL: https://platform.stability.ai/docs/getting-started - (дата обращения: 14.02.2024)

6. SwiftUI [Электронный ресурс] – URL: https://developer.apple.com/xcode/swiftui/ - (дата обращения: 13.02.2024)
7. Swift [Электронный ресурс] – URL: https://developer.apple.com/swift/ - (дата обращения: 16.02.2024)

УДК 004.42

Д. И. Субботин (2 курс магистратуры),
А. В. Самочадин, к.т.н., доцент

МОДЕЛИРОВАНИЕ АЛГОРИТМОВ ПЛАНИРОВАНИЯ ВЫПОЛНЕНИЯ ЗАДАЧ В РАСПРЕДЕЛЕННОЙ ОБЛАЧНОЙ СРЕДЕ

С увеличением числа предприятий и организаций, переходящих на облачные технологии, вопросы эффективного планирования выполнения задач в распределенной облачной среде становятся критически важными для оптимизации использования ресурсов и снижения затрат [1]. Данная работа затрагивает проблематику операционного менеджмента в сфере облачных вычислений, фокусируясь на моделировании алгоритмов планирования выполнения задач в распределенной облачной среде, способных обеспечить высокую производительность и быстрый доступ к информации.

Целью работы является создание имитационной модели исследования различных алгоритмов планирования выполнения задач в распределенной облачной среде и проведение сравнительного анализа этих алгоритмов.

Для создания модели используется фреймворк с открытым исходным кодом CloudSim. CloudSim – это обобщенная и масштабируемая платформа для симуляции облачных вычислений, позволяющая оценивать гипотезы перед фактической разработкой программного обеспечения [2].

Для достижения поставленной цели необходимо решить следующие задачи:

1. Обзор алгоритмов планирования задач в распределенной облачной среде.
2. Выбор алгоритмов для моделирования и проведения исследований.
3. Формирование требований к модели.
4. Проектирование модели.
5. Реализация модели с использованием CloudSim.
6. Проведение имитационных экспериментов для выбранных алгоритмов.
7. Проведение сравнительного анализа алгоритмов

Архитектура модели представлена на рисунке 1.

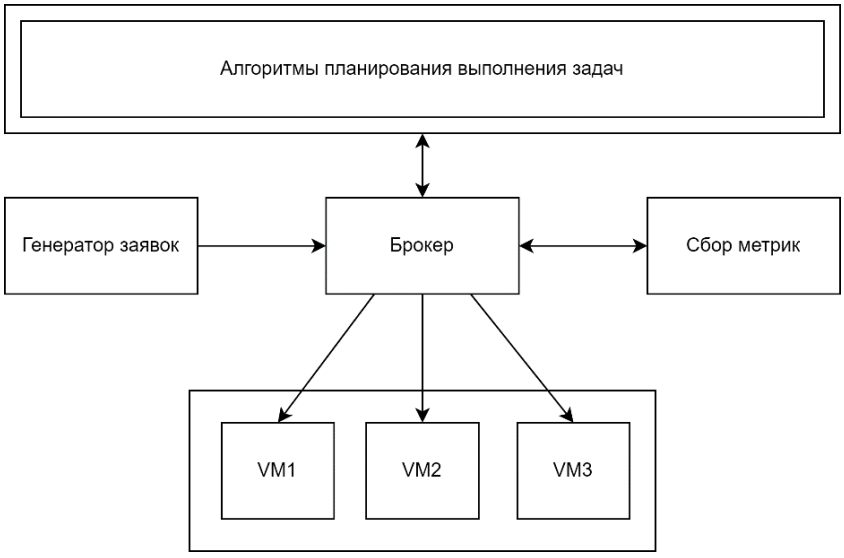


Рисунок 1 – Архитектура модели

Для моделирования предлагается использовать следующие алгоритмы планирования выполнения задач в облачной среде:

- Статические алгоритмы. Эти алгоритмы основаны на предопределенных правилах и не учитывают текущую загруженность системы или узлов, они просты в реализации и требуют минимального управления. Примерами данных алгоритмов являются: Round Robin, Random Selection, Least Loaded.
- Динамические алгоритмы. В отличии от статических, они адаптируются к изменяющейся нагрузке и условиям системы, могут использовать информацию о текущей загруженности узлов для принятия решений о распределении задач. Примерами данных алгоритмов являются: Threshold-based, Load Balancer [3].
- Основанные на машинном обучении. Эти алгоритмы адаптируются к изменяющейся нагрузке и условиям системы, они могут использовать информацию о текущей загруженности узлов для принятия решений о распределении задач. Примерами данных алгоритмов являются: Threshold-based, Load Balancer [4].

Система должна определять эффективность реализованных алгоритмов, используя различные метрики. В модели реализованы экономические метрики и метрики производительности [5, 6]. Данные для формирования метрик собираются в процессе проведения имитационных экспериментов. Совокупность формируемых метрик определяется критериями эффективности, сформированными разработчиками облачной среды.

ЛИТЕРАТУРА

1. Cloud Computing Value Chains: Research from the Operations Management Perspective [Электронный ресурс] / S. Chen [и др.] // Manufacturing & Service Operations Management. — 2023. — Т. 25, вып. 4. — С. 1338-1356. — Режим доступа: https://doi.org/10.1287/msom.2022.1178.
2. CloudSim. [Электронный ресурс] Режим доступа: http://www.cloudbus.org/cloudsim/.
3. Aslam, S. Load balancing algorithms in cloud computing: A survey of modern techniques [Электронный ресурс] / S. Aslam, M. A. Shah // 2015 National Software Engineering Conference (NSEC). — 2015. — Режим доступа: https://doi.org/10.1109/nsec.2015.7396341.
4. Machine Learning-Based Load Balancing Algorithms in Future Heterogeneous Networks: A Survey [Электронный ресурс] / E. Gures [и др.] // IEEE Access. — 2022. — Т. 10. — С. 37689-37717. — Режим доступа: https://doi.org/10.1109/access.2022.3161511.
5. Various job scheduling algorithms in cloud computing: A survey [Электронный ресурс] / Y. P. Dave [и др.] // International Conference on Information Communication and Embedded Systems (ICICES2014). — 2014. — Режим доступа: https://doi.org/10.1109/icices.2014.7033909.
6. Halawani, T. Types of Task Scheduling Algorithms in Cloud Computing Environment [Электронный ресурс] / T. Aladwani // Scheduling Problems - New Applications and Trends. — 2020. — Режим доступа: https://doi.org/10.5772/intechopen.86873.

УДК 004.42

Г. Н. Толстиков (2 курс магистратуры),
В. В. Амосов, к.т.н., доцент

РАЗРАБОТКА ANDROID ПРИЛОЖЕНИЯ СИСТЕМЫ ПОДДЕРЖКИ ПРИНЯТИЯ РЕШЕНИЙ

В рамках данной работы стояла проблема создания мобильного приложения для системы поддержки принятия решений. С необходимостью принятия решений сталкиваются и при разработке искусственного интеллекта, и при машинном обучении, и при разработке систем реального времени, и в профессиональной деятельности, и в реальной жизни. Принимать решение можно и без систем, однако, когда вариантов выбора и различных предпочтений много, то без формального подхода не обойтись.

Разработанная система поддержки основывается на качественном подходе и реализует механизмы (турнирный, доминирования, блокирующий, k-max и k-max оптимальный), по

которым производится оценка введенных пользователем вариантов по заданным предпочтениям [1]. Приложение должно давать возможность пользователю воспользоваться системой поддержки в любом месте и в любое время удобным способом. Существующие решения ограничиваются десктоп или веб-версиями, часто платные, либо узкоспециализированные и имеют несовременный интерфейс. Например, приложение «Decision Mentor: Decide Better» [2] реализует метод принятия решений на основе искусственного интеллекта, однако требует постоянного подключение к Интернету, и большая часть функционала предоставляется платно, что не позволяет нормально решить заданную проблему.

Для разработки мобильного приложения была выбрана операционная система Android как наиболее распространенная и универсальная. При этом выбранная минимальная версия для работы приложения позволит запускать его более чем на 95% устройств. Разработка проводилась при помощи современного фреймворка создания пользовательского интерфейса Jetpack Compose [3]. Jetpack Compose в отличие от view-подхода позволяет строить приложение новым декларативным способом формирования UI при помощи composable-функций стандартной библиотеки и собственных composable-функций. Это делает приложение легко расширяемым, требует меньше кода, а также обеспечивает высокую производительность – изменения в приложении контролируются изменением состояний переменных комбинацией remember и mutableStateOf. При их изменении соответствующая composable-функция, хранящая это состояние, перестраивается (а не вся страница или фрагмент), и тем самым ускоряется работа приложения. Из различных элементов были созданы отсутствующие в библиотеке таблицы, в том числе редактируемые, различные меню и страницы, позволяющие пользователю взаимодействовать с приложением. При этом разметка приложения, созданная при помощи Jetpack Compose построена таким образом, что все элементы, в том числе таблицы, корректно отображаются как на маленьких, так и на больших экранах смартфонов [3].

Система представляет собой клиент-серверное приложений с возможностью офлайн работы. Реализован backend-сервер на языке Python [4] при помощи быстрого и высокопроизводительного фреймворка FastAPI в связке с uvicorn [5], при помощи которых происходит работа приложения в режиме онлайн. Выбор Python обусловлен тем, что кроме описанного ранее качественного подхода онлайн-режим приложение подразумевает подход на основе машинного обучения и искусственного интеллекта, который лучше и проще всего реализовывается на Python. FastAPI предоставляет много возможностей для валидации и сериализации запросов при помощи библиотеки pydantic, сам код фреймворка написан на языке Rust, поэтому выполнение различных операций происходит быстро. Данный сервер предоставляет различные методы API для вычисления того или иного механизма системы поддержки. Взаимодействие с сервером происходит на стороне Android при помощи библиотеки Retrofit в API-шлюзе системы. Взаимодействие происходит асинхронно, то есть приложение не блокируется во время выполнения и обработки запросов.

Также существует сценарий работы приложения без интернета, поэтому те же вычисления были реализованы на языке Kotlin [6] и интегрированы в альтернативные вызовы API-шлюза. Таким образом при отсутствии интернет-соединения или если пользователь не хочет отправлять свои вычисления на сервер система может выполнить основную функциональность и выдать результат своими силами. FastAPI-приложение работает в Kubernetes на удаленном сервере VK Cloud, что позволяет легко масштабировать приложение, увеличивая число реплик и ресурсов, и работать более эффективно при увеличенной нагрузке. Само приложение поставляется на устройство пользователя установкой apk-файла.

ЛИТЕРАТУРА

1. Юдин Д. Б. Вычислительные методы теории принятия решений // М.: Наука. Гл. ред. Физ.-мат. лит., 1989, 320 с
2. Приложение Decision Mentor [Электронный ресурс]. Режим доступа https://decisionmentor.app/

3. Jetpack Compose Documentation [Электронный ресурс]. Режим доступа https://developer.android.com/jetpack/compose
4. Python Documentation [Электронный ресурс]. Режим доступа https://docs.python.org/3.11/
5. FastAPI Documentation [Электронный ресурс]. Режим доступа https://fastapi.tiangolo.com/
6. Kotlin Documentation [Электронный ресурс]. Режим доступа https://kotlinlang.org/

УДК 004.94

В. Е. Устинова, А. В. Шпак (4 курс бакалавриата),
В. А. Ивлев, Г. В. Мироненков (3 курс аспирантуры)

СИСТЕМА ПРЕОБРАЗОВАНИЯ КООРДИНАТ ОБЪЕКТОВ В SVG-ФАЙЛЕ ДЛЯ ПОИСКА СООТВЕТСТВИЯ МОДЕЛИ ЖЕЛЕЗНОДОРОЖНОЙ СТАНЦИИ ЕЕ ВИЗУАЛЬНОМУ ПРЕДСТАВЛЕНИЮ

На сегодняшний день в компании "РЖД" существует большое количество моделей ж/д станций в формате svg [1]. Между структурами железнодорожных станций и соответствующих им svg-моделей наблюдаются расхождения (несоответствие топологии отдельных участков, наименований объектов, направлений прямого проезда и съезда по стрелке, направлений регулировки светофора, а также отсутствие или наличие неучтенных светофоров в заданной точке), а наличие расхождений влечет невозможность проектирования новых станций до момента устранения всех несоответствий для существующих станций. Ручное обновление моделей и поиск несоответствий — это трудоемкая работа, сопровождаемая большим количеством человеческих ошибок, и именно поэтому актуальной является задача автоматизации [2] поиска соответствий модели железнодорожной станции визуальному представлению.

Алгоритм, решающий вышеназванную задачу, был поделен на части. Данная работа посвящена одной из частей реализованного алгоритма. Целью работы является снижение трудоемкости поиска соответствия модели железнодорожной станции визуальному представлению за счет алгоритма преобразования координат объектов svg-файла, который реализован в программном средстве поиска соответствия модели железнодорожной станции ее визуальному представлению.

Необходимость преобразовывать координаты объектов svg-файла обусловлена тем, что структура каждой станции представлена в двух формах: база данных и svg-файл. Чтобы сравнить представления и найти все несоответствия, необходимо построить по объектам каждого представления граф с последующим сравнением полученных графов. Тем не менее, если в базе данных представлены реальные координаты объектов, в svg-файле координаты объектов зашифрованы в виде относительных координат с описанием производимых над ними аффинных преобразований. Однако нужно иметь преставление о том, как объекты в реальности соотносятся друг с другом, из-за чего возникает необходимость реализации преобразования координат объектов [3].

Для реализации алгоритма были рассмотрены существующие решения для получения аффинных преобразований, однако они имеют недостатки, не позволяющие использовать их в данной задаче:

1. Pillow.Image [4] — библиотека Python, которая преобразует не сами численные координаты, а изображения.

2. Cv2 [5] — аналогична в этом отношении предыдущей библиотеке.

3. Shapely.affinity [6] — преобразует координаты, однако на больших данных показывает низкую скорость работы — при размере модели в миллион элементов скорость работы равна примерно 7 минутам.

4. Разработанное решение — подходит поставленным требованиям. Скорость работы выше приблизительно в 3 раза.

Алгоритм преобразования заключается в реализации трех операций — translate, rotate и scale [7]. Пример на рисунке 1 — объект до и после применения операции:

Рисунок 1 — Аффинные преобразования

Важен тот факт, что после поворота или отзеркаливания осей дальнейшие операции проводятся относительно измененных осей, что создает необходимость создания алгоритма, учитывающего все варианты порядка применения операций.

Данный алгоритм — часть программного средства, сопоставляющего представления станций. Была разработана конфигурация всей системы [8]. На Рисунке 2 представлена вся схема проекта — жирным выделена часть, касающаяся описанного алгоритма:

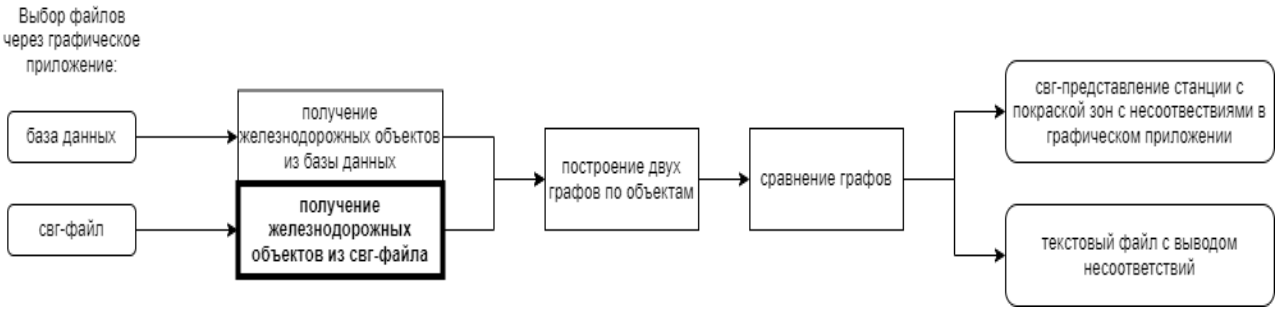


Рисунок 2 — Схема проекта

В качестве эксперимента работники компании провели сравнение представлений станций вручную и с помощью программы. Были вычислены средние значения потребовавшегося времени, из чего был сделан вывод — вместо 71 минуты, требовавшейся одному сотруднику для проверки несоответствий между двумя представлениями станциями, стало уходить 14 минут (из которых подавляющая часть времени требуется не на работу программы, а на корректировку человеком файлов на основании полученных данных). Таким образом, было достигнуто снижение трудоемкости в 5 раз.

ЛИТЕРАТУРА

1. SVG [Электронный ресурс] // Википедия: [сайт]. — URL: https://ru.wikipedia.org/wiki/SVG (дата обращения: 05.03.2024).
2. Ивлев, В. А. Актуальность автоматизированной настройки инфраструктуры ит-проекта / В. А. Ивлев, И. В. Никифоров // Современные технологии в теории и практике программирования : Сборник материалов конференции, Санкт-Петербург, 26 апреля 2022 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2022. – С. 79-80. – EDN QQAVXT.
3. Ивлев, В. А. Обработка данных в геоинформационных системах для выбора местоположения рекламы / В. А. Ивлев, И. В. Никифоров, Т. В. Леонтьева // Современные технологии в теории и практике программирования : сборник материалов конференции, Санкт-Петербург, 19 апреля 2019 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2019. – С. 27-30. – EDN DNQPMC.

4. Image Module / [Электронный ресурс] // Pillow: [сайт]. — URL: https://pillow.readthedocs.io/en/stable/reference/Image.html (дата обращения: 25.02.2024).
5. OpenCV / [Электронный ресурс] // OpenCV : [сайт]. — URL: https://opencv.org/ (дата обращения: 25.02.2024).
6. The Shapely User Manual / [Электронный ресурс] // Shapely: [сайт]. — URL: https://shapely.readthedocs.io/en/stable/manual.html (дата обращения: 25.02.2024).
7. SVG-разметка двухмерной графики: Опыт использования SVG в создании двухмерной графики / А. И. Телегин, Д. Н. Тимофеев, Д. И. Читалов, С. Г. Пудовкина. – Миасс : Южно-Уральский государственный университет (национальный исследовательский университет), 2015. – 73 с. – EDN UVMNDP.
8. Сысоев, И. М. Создание подхода автоматизации обновления конфигурационных файлов программного обеспечения / И. М. Сысоев, И. В. Никифоров, Е. В. Каплан // 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. – EDN YARVRM.

УДК 004.453

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

РЕАЛИЗАЦИЯ ПЛАНАРНОСТИ ГРАФА АВТОМОБИЛЬНЫХ ДОРОГ С ИСПОЛЬЗОВАНИЕМ МОДЕЛИ MAPREDUCE

В современном мире картографические службы играют ключевую роль в обеспечении эффективной транспортной инфраструктуры. Эффективная реализация обработки данных для таких систем является важной задачей, поскольку позволяет экономить вычислительные ресурсы и используемое место в хранилищах картографической информации. В данной работе рассматривается реализация планарности графа автомобильных дорог из OpenStreetMap [3] с использованием модели распределенных вычислений MapReduce [1].

Планарный граф – граф, который можно изобразить на плоскости без пересечений рёбер не по вершинам.

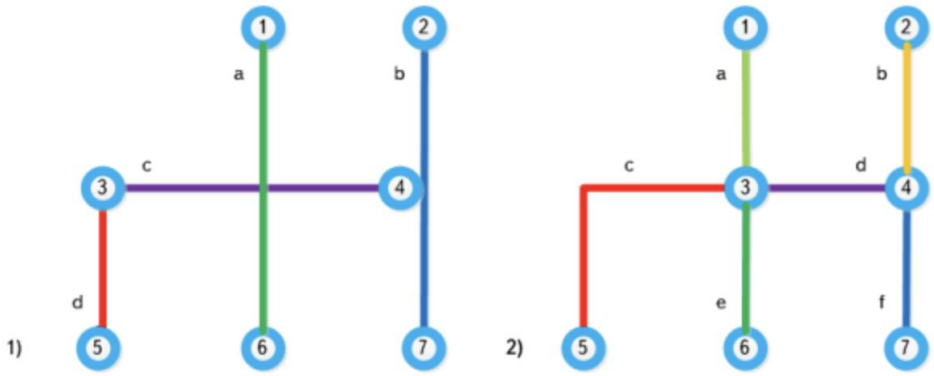


Рисунок 1 – не планарный и планарный графы.

На рисунке 1 граф под номером 2) является планарным, а граф 1) – нет.

В OpenStreetMap автодороги имеют параметр уровня относительно земли и других дорог. Например, дорога с уровнем 0 лежит на земле, а мост, проходящий над ней, лежит на уровне 1. Уровни выставляют ребрам дорожного графа.

В данной работе рассматривается подход к конвертации такого формата работы с уровнями автомобильных дорог в формат, в котором уровни назначаются узлам, соединяющим ребра графа автодорог.

Формат с выставлением уровней в узлах графа, а не в рёбрах имеет следующие преимущества:

1. Одно ребро может являться переходным этапом между уровнями 0 и 1, например, съезд с трассы или моста.
2. Такой формат облегчает добавление новых картографических данных пользователям таких систем, поскольку точнее отражает ситуацию в реальности.
3. При добавлении данных в картографическую систему не нужно создавать промежуточные ребра автомобильного графа для плавного перехода между дорогами на разных уровнях.

Таким образом *целью* данной работы является эффективная реализация обработки данных об автомобильных дорогах с конвертацией хранения информации о них с использованием модели распределенных вычислений MapReduce.

Первым и одним из основных этапов является построение индекса по геометрическим данным дорожного графа, поскольку необходимо параллельно обрабатывать и находить пересечения рёбер дорожного графа.

Индексация геометрических данных всей планеты является задачей, требующей комплексного подхода с реализацией эффективных алгоритмов индексации, однако в данной работе был использован упрощенных подход с применением системы геокодирования Geohash [2]. Это иерархическая структура пространственных данных, которая подразделяет пространство на сегменты в виде квадратов.

Рисунок 2 – схематичное изображение работы алгоритма Geohash.

Каждому из сегментов дороги назначается своя клетка из иерархической сетки Geohash с помощью Map-операции. С помощью Reduce-операции дорожные элементы объединяются по значению Geohash и по данных рёбрам графа строится геометрический индекс R-tree для эффективного нахождения пересечений между рёбрами.

В частности, R-tree индекс помогает сравнивать между собой на предмет пересечения только рёбра дорожного графа, граничные области которых пересекаются. Граничная область – это прямоугольник, в который можно вписать геометрию объекта.

Пересечения между ребрами дорожного графа записываются в выходную таблицу в следующем формате: «ребро графа» – «координата пересечения». Здесь «ребро графа» – ребро дорожного графа, которое должно быть разделено на два ребра в точке «координата пересечения». Необходимо учесть при построении новых рёбер путём разделения имеющихся, что одно ребро графа может быть разделено в нескольких местах.

Также необходимо разделять дороги в точках перекрёстков, поскольку формат хранения в OpenStreetMap не ограничивает дорожные элементы там, где реальные дороги начинаются

или пересекаются. Соответственно необходимо разделить дороги в узлах, которые принадлежат к узлам перекрестков. Определить это можно двумя способами: в тегах, описывающих узел, содержится информация о том, что он является перекрёстком или же один и тот же узел значится как начало/конец нескольких дорожных элементов (таких дорожных элементов больше одного).

После проделанных операций имеем две таблицы: таблица пересечений и таблица точек перекрёстков. Дополнительными MapReduce [4] операциями геометрия рёбер графа разделяется в этих точках. Новым точкам в местах пересечения назначаются новые сгенерированные уникальные идентификаторы.

Стоит отметить, что точка разделения в месте пересечения рёбер является общей у ребра и другого пересекающего его ребра. Это нужно для уменьшения хранимых данных об узлах графа.

В данной работе был представлен способ реализации планарности графа автомобильных дорог с использованием метода распределённых вычислений MapReduce. В качестве источника данных был использован некоммерческий веб-картографический проект OpenStreetMap. Также были сравнены два подхода к использованию относительных уровней дорожных элементов. Важным этапом работы стоит отметить индексацию геометрических данных, поскольку эта часть работы играет основную роль в быстродействии и эффективности обработки таких объемов информации.

ЛИТЕРАТУРА

1. Sakr, S., Zomaya, A.Y., “MapReduce”, Encyclopedia of Big Data Technologies. Springer, Cham – 2019.
2. Van, L.H., Takasu, A. An Efficient Distributed Index for Geospatial Databases, (eds) Database and Expert Systems Applications. Globe DEXA 2015 2015. Lecture Notes in Computer Science, vol 9261. Springer, Cham. – 2015
3. OpenStreetMap. [Электронный ресурс] Режим доступа: https://www.openstreetmap.org/about
4. Jeffrey Dean, Sanjay Ghemawat, MapReduce: Simplified Data Processing on Large Clusters, San Francisco, CA – 2004.

УДК 004.42

М. Г. Ушакова (4 курс бакалавриата),
Л. П. Котлярова, ст. преподаватель

МНОГОПОЛЬЗОВАТЕЛЬСКАЯ АВТОМАТИЗИРОВАННАЯ СИСТЕМА «ПРОИЗВОДСТВЕННАЯ БЕЗОПАСНОСТЬ»

Автоматизация учета происшествий играет ключевую роль в обеспечении производственной безопасности, особенно в организациях, где несчастные случаи и другие происшествия могут иметь серьезные последствия. Эффективное управление информацией о происшествиях позволяет оперативно реагировать на них, предотвращать повторение и минимизировать риски для сотрудников и имущества компании [1].

Автоматизированная система позволит отслеживать статус отправки извещений и актов расследования в вышестоящие органы, что поможет предотвратить возможные финансовые потери, связанные с невыполнением регуляторных требований.

Кроме того, автоматизация учета происшествий позволит своевременно обрабатывать информацию. Благодаря этому у сотрудников будет меньше времени между получением извещения о происшествии и его внесением в систему. Это позволит быстрее реагировать на происшествия и принимать необходимые меры для их устранения или предотвращения.

Таким образом, целью данной работы является решение проблемы автоматизации ведения учета происшествий за счет создания веб-приложения для конечных пользователей, что позволит повысить эффективность работы, сократить рутинные задачи и обеспечить более надежную и актуальную информацию для принятия управленческих решений [2].

Для достижения поставленной цели необходимо решение следующих задач:

1. Проанализировать текущие процессы учета происшествий и выявить проблемные аспекты.
2. Разработать структуру базы данных для хранения информации о происшествиях.
3. Настроить сервер для стабильной и безопасной работы системы учета происшествий.
4. Разработать механизм загрузки данных из извещений в формате XLSX в систему.
5. Разработать и внедрить в работу веб-приложение.

Веб-приложение будет реализовано на языке PHP [3], используя объектно-ориентированный компонентный фреймворк Yii [4], реализующий парадигму MVC (Model-View-Controller) [5].

Система разделяется на два интерфейса: для всех пользователей и для администраторов системы. Пользователь взаимодействует с веб-приложением, отправляя HTTP запросы через браузер. Запрос может содержать различные параметры и данные. Архитектура системы ведения учета происшествий представлена на рисунке 1.

Приложение использует механизм маршрутизации для определения, какой контроллер и действие должны обрабатывать полученный запрос. Маршруты могут быть настроены для различных URL, и они могут указывать на разные контроллеры и действия в зависимости от запроса.

Контроллер получает запрос от маршрутизатора и обрабатывает его. Он может выполнить различные действия, такие как чтение или запись данных в базу данных, обработка входных данных, вызов методов модели и формирование данных для отображения. Контроллер может взаимодействовать с моделью, чтобы получить доступ к данным приложения.

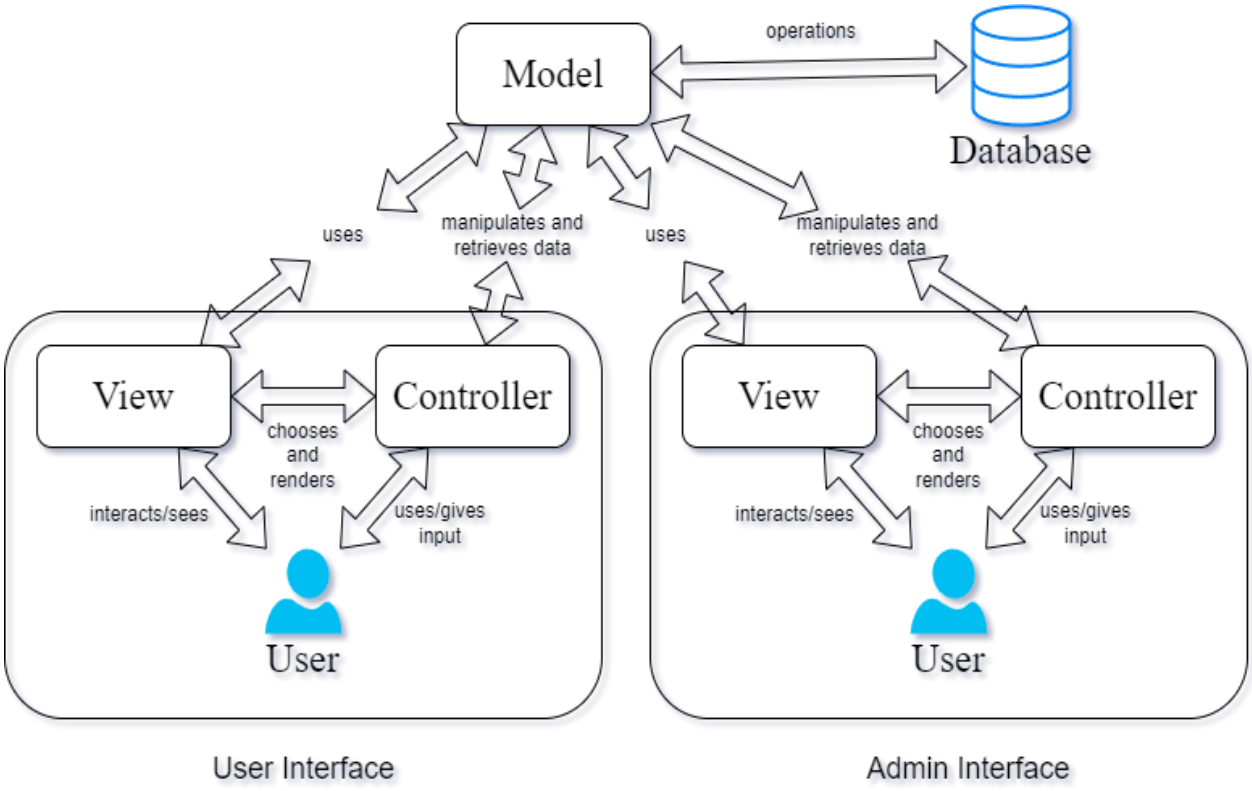


Рисунок 1 — Архитектурный шаблон системы

Модель предоставляет интерфейс для работы с базой данных или другими источниками данных, а также выполняет бизнес-логику приложения, такую как валидация данных или выполнение сложных операций. После обработки запроса контроллер передает данные представлению для формирования ответа пользователю.

Представление отвечает за отображение данных пользователю в удобном для восприятия виде. Сформированный ответ от представления отправляется обратно пользователю через веб-сервер в виде веб-страницы или другого типа ответа, который используется в зависимости от запроса и требований приложения.

На данный момент реализован учет происшествий о несчастных случаях и дорожно-транспортных происшествиях, а также личный кабинет пользователя.

ЛИТЕРАТУРА

1. Рябикина, Т. С. Система управления происшествиями без последствий как метод повышения безопасности на газовых объектах / Т. С. Рябикина // Аллея науки. – 2018. – Т. 7, № 5(21). – С. 213-217. – EDN USYWZB.
2. Астафьев, П. Профилактика производственного травматизма на основе практики расследования и анализа происшествий без последствий / П. Астафьев // Электроэнергия. Передача и распределение. – 2020. – № S1(16). – С. 35-37. – EDN EYMWFA.
3. PHP: Hypertext Preprocessor. [Электронный ресурс]. Режим доступа: https://www.php.net/.
4. Yii PHP Framework. [Электронный ресурс]. Режим доступа: https://www.yiiframework.com/.
5. Сафин, А. М. Архитектура MVC / А. М. Сафин, К. А. Кадыров // World science: problems and innovations : Сборник статей LVI Международной научно-практической конференции, Пенза, 30 августа 2021 года. – Пенза: Наука и Просвещение, 2021. – С. 53-55. – EDN GOYFER.

УДК 004.4

Т. Э. Шахбазлы (2 курс магистратуры),
Н. В. Воинов, к.т.н., доцент

РАЗРАБОТКА КРОССПЛАТФОРМЕННОГО МОБИЛЬНОГО ПРИЛОЖЕНИЯ ДЛЯ НАСТРОЙКИ И ПОДКЛЮЧЕНИЯ К VPN

Разработка мобильного приложения для нескольких платформ (iOS [1,2] и Android [3]) традиционно требует отдельной работы над каждой из них. Это означает, что разработчики должны иметь навыки и опыт работы с разными языками программирования, фреймворками и инструментами разработки для каждой платформы. Это может привести к дублированию работы, увеличению времени разработки и сложностям в поддержке приложений в долгосрочной перспективе. Кроссплатформенная разработка призвана упростить этот процесс, позволяя разработчикам создавать приложения для нескольких платформ с помощью одного и того же кода [4]. Это позволяет существенно сократить затраты на разработку и поддержку приложений, а также ускорить время выхода на рынок, что является важным конкурентным преимуществом для бизнеса.

Инструменты кроссплатформенной разработки, такие как Flutter [5], React Native [6] и Kotlin Multiplatform [7], предоставляют разработчикам возможность создания приложений для различных платформ с использованием единого кода. Flutter от Google и React Native от Facebook долгое время были популярными выборами для кроссплатформенной разработки. Однако Kotlin Multiplatform становится все более привлекательным вариантом благодаря своей интеграции с языком программирования Kotlin [8], который широко используется для разработки приложений под Android.

В данной работе основное внимание уделяется кроссплатформенности разрабатываемого мобильного приложения, что делает необходимым решение следующих задач:

- Изучение существующих решений на рынке и анализ их особенностей.
- Постановка задачи и формирование требований к разрабатываемому приложению.
- Проектирование архитектуры приложения с учетом кроссплатформенных возможностей и требований.

- Реализация приложения с использованием Kotlin Multiplatform и других необходимых технологий.

- Тестирование приложения на различных платформах и анализ его применимости для решения задачи.

Каждая из рассмотренных технологий представляет собой уникальный подход к кроссплатформенной разработке мобильных приложений. Однако Kotlin Multiplatform Mobile (KMM) выделяется как наиболее перспективное и привлекательное решение.

KMM основан на концепции создания общего кода для бизнес-логики приложения, который затем компилируется в нативный код для каждой платформы. Этот подход позволяет разработчикам использовать нативные API и инструменты разработки для каждой платформы, сохраняя при этом общую базу кода. Благодаря этому, KMM обеспечивает высокую степень переиспользования кода, упрощает процесс разработки и поддержки приложений, а также обеспечивает высокую производительность и нативный пользовательский опыт на разных платформах.

Кроме того, KMM позволяет сохранить нативный UI для каждой платформы, что делает его идеальным выбором для команд разработчиков, стремящихся к оптимизации процесса разработки и снижению затрат на поддержку приложений. Благодаря использованию языка программирования Kotlin, широко применяемого для разработки приложений под Android, KMM также обеспечивает простоту адаптации для существующих команд разработчиков.

ЛИТЕРАТУРА

1. Потапова, А. М. Разработка IOS-приложения для записи воспоминаний / А. М. Потапова, Н. В. Воинов, Ю. В. Юсупов // Современные технологии в теории и практике программирования: Сборник материалов научно-практической конференции студентов, аспирантов и молодых ученых, Санкт-Петербург, 26–27 апреля 2023 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2023. – С. 70-72. – EDN RAMWIW.
2. Сопрачев, А. К. Разработка мобильного навигационного приложения на платформе IOS / А. К. Сопрачев, Н. В. Воинов // Современные технологии в теории и практике программирования: Сборник материалов конференции, Санкт-Петербург, 26 апреля 2022 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2022. – С. 47-48. – EDN AXDODM.
3. Усов, М. А. Разработка Android-приложений на основе декларативного подхода с применением Jetpack Compose / М. А. Усов, Н. В. Воинов, И. В. Зайцев // 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. 126-127. – EDN DUTYHC.
4. Касимова, К. М. Разработка конструктора кроссплатформенных веб-сайтов / К. М. Касимова, Н. В. Воинов // Современные технологии в теории и практике программирования: Сборник материалов научно-практической конференции студентов, аспирантов и молодых ученых, Санкт-Петербург, 26–27 апреля 2023 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2023. – С. 138-140. – EDN JATXPB.
5. Flutter. [Электронный ресурс] Режим доступа: https://flutter.dev/
6. React Native. [Электронный ресурс] Режим доступа: https://reactnative.dev/
7. Kotlin Multiplatform. [Электронный ресурс] Режим доступа: https://kotlinlang.org/docs/multiplatform.html
8. Kotlin. [Электронный ресурс] Режим доступа: https://kotlinlang.org/

УДК 004.031.2

А. Ю. Шестакова (2 курс магистратуры),
И. Г. Черноруцкий, д.т.н., профессор

TELEGRAM-БОТ ДЛЯ ПЕРЕПОСТА ДАННЫХ МЕЖДУ СОЦСЕТЯМИ TELEGRAM, ВКОНТАКТЕ И ОДНОКЛАССНИКИ С ИСПОЛЬЗОВАНИЕМ ИХ API

Чтобы достичь разные сегменты целевой аудитории, бизнес и инфлюенсеры вынуждены дублировать идентичный контент в разные социальные сети. Выполнение подобной работы вручную может занимать большое количество времени [1].

Целью работы является разработка Telegram-бота для автоматического переноса данных из Telegram в Одноклассники и ВКонтакте с использованием их API.

Основными требованиями для искомого инструмента являются следующие:

- поддержка социальных сетей Telegram [2], ВКонтакте [3] и Одноклассники [4];
- возможность добавления неограниченного количества аккаунтов;
- возможность синхронизации постов из неограниченного количества Telegram-каналов;
- отсутствие необходимости использования дополнительных сервисов для выполнения формирования и переноса контента.

В рассматриваемой области кросспостинга [5] на данный момент не существует готовых решений, удовлетворяющих перечисленным выше требованиям. Кроме того, решения, удовлетворяющие части требований, требуют оформления платной подписки для их использования.

Основными компонентами программной системы являются база данных и серверная часть. Клиентскую часть предоставляет Telegram, но для удобства получения токена для авторизации через Одноклассники был также добавлен фронтенд.

Компоненты системы, взаимодействие между ними и используемые технологии представлены на рисунке 1.

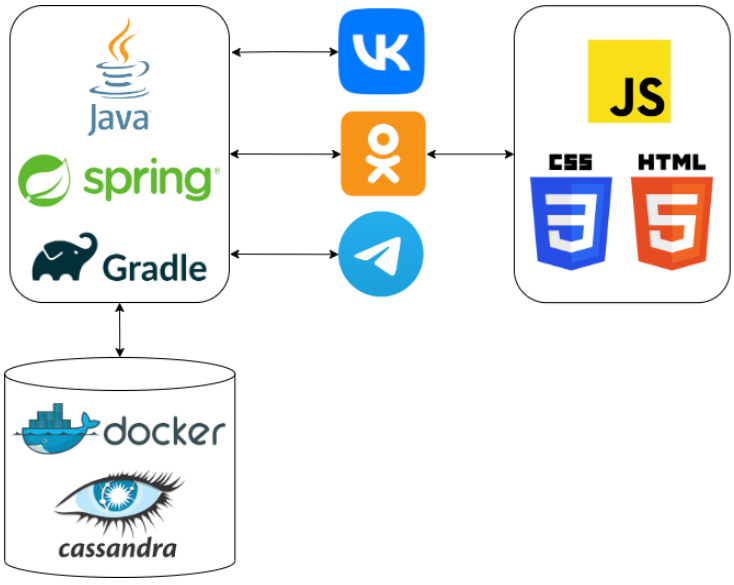


Рисунок 1 — Компоненты программной системы с используемыми технологиями

Также в разработанном продукте предусмотрены защита от спама в виде ограничения на количество публикуемых в минуту постов, защита от нарушения авторских прав (не публикуются пересылаемые из других каналов посты и сообщения других пользователей), функция уведомлений пользователя о статусе публикации постов в социальных сетях, функция включения/выключения автоматической публикации постов в канале.

Кроме того, разработанный бот развернут в VK-облаке, а благодаря использованию модульной архитектуры серверной части присутствует возможность добавления новых социальных сетей для постинга в виде отдельных модулей.

ЛИТЕРАТУРА

1. Красова К. А. Возможности PR в социальных сетях и блогах // Научные Записки ОрелГИЭТ. – 2013. – № 1(7). – С. 290а-293.
2. Шевченко В. И. Модель интерактивного взаимодействия с системами мониторинга на основе TELEGRAM API // Системы контроля окружающей среды. – 2016. – № 4(24). – С. 62-65.
3. Биков Д. И. Способы обработки запросов для чат-бота при помощи инструментов VK API // Приоритетные направления инновационной деятельности в промышленности // Сборник научных статей по итогам десятой международной научной конференции, Казань, 30–31 октября 2020 года. Том Часть 2. – Казань: Общество с ограниченной ответственностью "КОНВЕРТ", 2020. – С. 35-36.
4. Диков М. Е. API социальных сетей: сравнительный анализ функционала для разработки инструментария профориентации // Информационные технологии в науке и образовании // Материалы Международной молодежной научно-практической конференции, Новочеркасск, 14–15 августа 2021 года. – Новочеркасск: ООО "Лик", 2021. – С. 89-92.
5. Каспаринский Ф. О. Кросспостинг в авторском информационном континууме // XVII Всероссийская научная конференции, Новороссийск, 21–26 сентября 2015 года / ИПМ им. М. В. Келдыша. – Новороссийск: Институт прикладной математики им. М. В. Келдыша РАН, 2015. – С. 135-140.

УДК 004.457

А. Ю. Шестакова (2 курс магистратуры),
А. М. Сабуткевич (1 курс аспирантуры),
И. В. Никифоров, к.т.н., доцент

ПОДХОД К ВЕРИФИКАЦИИ МОДЕЛЕЙ BPMN 2.0 С ИСПОЛЬЗОВАНИЕМ АЛГОРИТМА MODEL CHECKING

Оптимизация бизнес-процессов является одним из наиболее значимых подходов к повышению эффективности деятельности предприятий, а также имеет особую значимость в процессе их цифровизации [1].

Одним из способов формального описания бизнес-процессов, используемого в дальнейшем для их оптимизации, является использование нотации Business Process Model and Notation (BPMN) [2]. Данная нотация представляет собой набор условных обозначений и их описания на языке XML, она является связующим звеном между бизнес-пользователями и разработчиками.

Одним из ключевых этапов, предшествующих дальнейшей оптимизации, является верификация построенной формальной модели процесса с использованием нотации BPMN относительно предъявляемых свойств, определяющих критерии соответствия модели и реального процесса. Верификация [3] — один из основных методов повышения качества формальных систем, гарантирующих их правильность. Осуществление ручной верификации таких моделей трудозатратно и не может гарантировать отсутствие ошибок в модели, в связи с чем модель может оказаться некачественной и далекой от реального процесса. Одним из возможных решений является использование программных средств верификации моделей, реализующих алгоритм Model Checking.

Model checking [3] — формальная проверка того, выполняется ли требуемое свойство, задаваемое посредством логической формулы на данной модели, описанной в виде структуре. Алгоритм верифкации Model Checking связан с формальной проверкой выполнения на модели реализации свойств поведения, специфицированных на языке формальной логики и сводится к исчерпывающему анализу всего пространства состояний модели системы [4].

Таким образом, актуальной является задача верификации моделей BPMN с использованием алгоритма Model Checking.

Целью работы является повышение качества моделей BPMN 2.0 за счет создания средства проверки моделей с применением алгоритма Model Checking, который поддерживается в верификаторе SPIN, с использованием языка Promela.

Для достижения поставленной цели требуется выполнить следующие задачи:

- проведение исследования существующих методов проверки моделей BPMN 2.0;

- сравнительный анализ методов проверки моделей BPMN 2.0;

- предложение метода проверки моделей BPMN 2.0, позволяющего верифицировать модель с учетом требований пользователей;

- реализация предложенного метода в программном средстве с использованием языка Promela и верификатора SPIN, отличительной особенностью которого является возможность проверки асинхронных распределенных алгоритмов с использованием свойств, выраженных формулами темпоральной логики линейного времени (LTL);

- оценка качества и эффективности применения разработанного программного средства в ходе экспериментальных исследований.

В ходе исследования предметной области были рассмотрены существующие работы, связанные с методами проверки моделей BPMN:

- «Formal approach for compliance rules checking in Business Process Models» [5, 6];

- «Model Checking Functional Integration of Human Cognition and Machine Reasoning» [7];

- «An Approach to Construct Formal Model of Business Process Model from BPMN Workflow Patterns» [8];

- «A Tool for the Automatic Verification of BPMN Choreographies» [9].

В таблице 1 представлен сравнительный анализ решений, критериями сравнения были выбраны наличие проверки базовых для верификатора SPIN свойств — свойств безопасности, живости и справедливости [2], — наличие проверки произвольных пользовательских свойств, наличие поддержки основных шлюзов BPMN и использование автоматизированной универсальной трансляции.

Таблица 1 – Существующие методы проверки моделей BPMN 2.0

Решение	Проверка свойств безопасности, живости, справедливости	Проверка произвольных свойств	Автоматизи-рованная трансляция	Поддержка основных шлюзов BPMN
1	+	-	+	3/5
2	+/-	-	-	1/5
3	-	-	+	2/5
4	-	+	+	1/5

Исходя из данных, представленных в таблице выше, можно сделать вывод, что большая часть существующих решений не позволяет провести проверку модели, удовлетворяющую пользовательским требованиям. Также рассмотренные решения поддерживают лишь значительно ограниченную часть функционала нотации BPMN, что не позволяет приведенным решениям быть универсальными и корректно работать с любым требуемым бизнес-процессом. Разрабатываемое решение спроектировано с учетом выявленных недостатков.

На рисунке 1 представлена архитектура разрабатываемого решения: основными компонентами системы являются:

- XML-парсер — принимает на вход BPMN-модель и отдает на выходе список объектов, соответствующих обнаруженным в модели сущностям;

- Генератор кода — принимает на вход объекты от парсера и пользовательскую LTL-формулу, на выходе дает транслированную на язык Promela модель;

- SPIN — получает модель на языке Promela, производит проверку базовых и пользовательских свойств, на выходе отдает результат верификации — сообщение об успехе или контрпример.

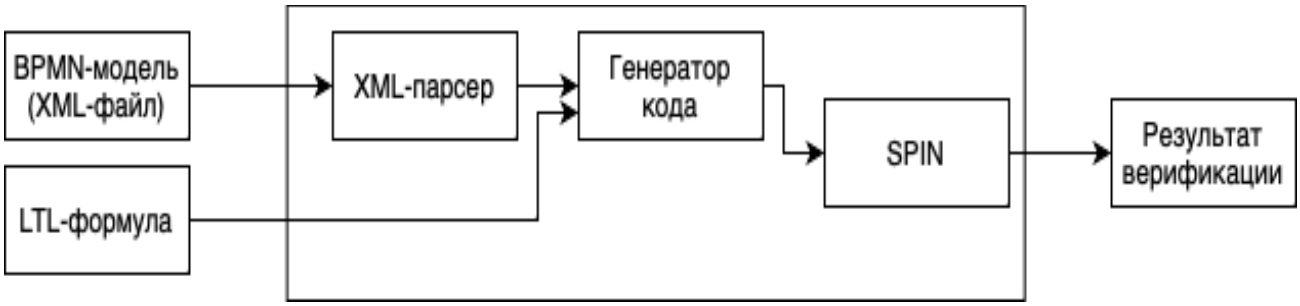


Рисунок 1 – Архитектура решения

В работе представлены подход верификации моделей BPMN 2.0 и модульная архитектура программного средства. На текущий момент предложенный подход находится на этапе реализации в программном средстве.

ЛИТЕРАТУРА

1. Cong L. Intelligent Operation and Maintenance Business Process Optimization Design of Integrated Energy System // 5th Asia Energy and Electrical Engineering Symposium (AEEES), Chengdu, China, 2023, pp. 1741-1746 – DOI: 10.1109/AEEES56888.2023.10114192.
2. Houhou S., Baarir S., Poizat P., Quéinnec P., Kahloul L. A First-Order Logic verification framework for communication-parametric and time-aware BPMN collaborations // Information Systems (104), 2022. – DOI: 10.1016/j.is.2021.101765.
3. Карпов Ю. Г. MODEL CHECKING. Верификация параллельных и распределенных программных систем. — СПб.: БХВ-Петербург, 2010. — 560 с.
4. Инкрементальный подход к технологии создания тестов для индустриальных проектов / П. Д. Дробинцев, В. П. Котляров, И. В. Никифоров, А. А. Летичевский // Моделирование и анализ информационных систем. – 2014. – Т. 21, № 6. – С. 144-154. – EDN TCCAKB.
5. Kherbouche O. M., Ahmad A., Basson H. Formal approach for compliance rules checking in Business Process Models // IEEE 9th International Conference on Emerging Technologies (ICET), Islamabad, Pakistan, 2013, pp. 1-6. DOI: 10.1109/ICET.2013.6743500.
6. 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. – EDN VDVCYX.
7. Mercer E., Butler K., Bahrami A. Model Checking Functional Integration of Human Cognition and Machine Reasoning // IEEE International Systems Conference (SysCon), Montreal, QC, Canada, 2022, pp. 1-8. DOI: 10.1109/SysCon53536.2022.9773873.
8. Yamasathien S., Vatanawood W. An approach to construct formal model of business process model from BPMN workflow patterns // Fourth International Conference on Digital Information and Communication Technology and its Applications (DICTAP), Bangkok, Thailand, 2014, pp. 211-215. DOI: 10.1109/DICTAP.2014.6821684.
9. Solaiman E., Sun W., Molina-Jimenez C. A Tool for the Automatic Verification of BPMN Choreographies // IEEE International Conference on Services Computing, New York, NY, USA, 2015, pp. 728-735. DOI: 10.1109/SCC.2015.103.

УДК 004.42

А. А. Шожат (4 курс бакалавриата),
Т. В. Коликова, ст. преподаватель

РАЗРАБОТКА МОБИЛЬНОГО ПРИЛОЖЕНИЯ НА ANDROID ДЛЯ ГРАМОТНОГО ВЕДЕНИЯ ЛИЧНЫХ ФИНАНСОВ

Современные пользователи все больше осознают важность эффективного управления своими финансами в удобном для них способе, ведь с каждым годом увеличивается количество финансовых операций. Поэтому люди ищут удобные способы отслеживания своих финансов, что делает мобильные приложения очень актуальными, так как, в основном, в наше время транзакции осуществляются с помощью мобильных устройств. Целью данной работы является разработка интуитивно понятного приложения с удобным интерфейсом, обладающее полезным функционалом.

Для достижения этой цели необходимо решение следующих задач:

1. Обзор существующих продуктов для учета финансов
2. Проведение сравнительного анализа существующих решений.
3. Определение ключевой функциональности приложения.
4. Реализация поставленной функциональности.

В качестве прямого аналога моего приложения существует приложение «Money Lover», которое позволяет планировать весь личный бюджет, учитывать траты по категориям, управлять кредитными и дебетовыми картами. Все доходы и расходы отображаются в простой и понятной диаграмме. Также здесь доступен экспорт операций в печатную версию. Но при этом здесь есть постоянная реклама, которая будет убрана только если подключить платную версию. Также отсутствует возможность отображения данных в различных форматах. Еще один аналог: приложение «Тяжеловато». Простейший функционал помогает следить за ежедневными тратами так, чтобы не превысить запланированный лимит. Достаточно задать период и определенную сумму, которую можно за него потратить. Приложение рассчитает количество денег, доступное для трат в день. Однако при этом приложение достаточно примитивно — оно не предполагает распределения трат по конкретным категориям, не собирает статистику за периоды, не отображает даже малейшую визуализацию и по сути представляет собой обычный калькулятор с довольно простой, но четкой логикой.

Одним из важнейших нефункциональных требований является соблюдение принципа Interface Segregation Principle[1] для архитектуры проекта. В основных функциональных требованиях были определены и должны быть реализованы такие возможности, как регистрация новых пользователей с использованием логина и пароля, а также их авторизация с возможностью восстановления пароля. В окне главного экрана должно отображаться текущее финансовое состояние пользователя вместе с основными категория расходов и доходов за выбранный по умолчанию период времени. В целом, при добавлении транзакции должен быть выбор между ее типом в виде доходов или расходов, выбор категории транзакции в зависимости от определенного ранее типа, возможность указать параметры суммы, даты, описания. В качестве механизма наглядности должно быть реализовано визуальное представление в виде таблиц, графиков и диаграмм с настройкой параметров дат, категорий и типов для отображения. Немаловажным являются функции напоминания о различных действиях, необходимых для грамотного ведения личных финансов, а именно напоминания о платежах, напоминания о получении процентов по вкладам, счетам, инвестициям, предупреждения и уведомления при приближении к предельным значениям бюджета, возможность установить финансовую цель. Поскольку приложение будет использоваться на мобильных устройствах, я выбрал один из популярнейших языков программирования для осуществления данной задачи — Java[2]. В качестве СУБД был выбран SQLite[3].

Java является кроссплатформенным языком программирования, что означает, что приложение, написанное на Java, может быть запущено на различных платформах, включая Android[4], который и был выбран целевой платформой, ведь она обладает обширной документацией, экосистемой библиотек и фреймворков и очень большим сообществом разработчиков, что облегчает разработку и поддержку приложений. Что касается SQLite – эта СУБД является легковесной, которая интегрируется прямо в приложение и считается идеальной для мобильных приложений с ограниченными ресурсами, что делает ее естественным выбором для хранения данных в таких системах.

ЛИТЕРАТУРА

1. Роберт Мартин, Чистая архитектура. Искусство разработки программного обеспечения. 2022, -с. 232-235
2. Кей Хорстманн, Гари Корнелл "Java. Библиотека профессионала. Том 1". 11-е издание
3. SQLite. [Электронный ресурс] Режим доступа: https://www.sqlite.org/index.html
4. Программирование под Android на Java // METANIT.COM, 2023. [Электронный ресурс] Режим доступа: https://metanit.com/java/android/

УДК 62-6

Ядхукришнан Си Джей (2 курс магистратуры)

АНАЛИЗ ГРАНИЦ ПАРЕТО ДЛЯ ПОВЫШЕНИЯ ПРОИЗВОДИТЕЛЬНОСТИ И ПРИНЯТИЯ РЕШЕНИЙ ПРИ ВЫБОРЕ КОТЛОВ

Промышленная революция 4.0 направлена на применение машинного обучения, искусственного интеллекта и внедрение промышленных роботов. Котельные агрегаты современных теплоэлектростанций (ТЭЦ) оснащены множеством датчиков, предоставляющими измерительную информацию, средствами накопления и хранения больших данных. На основе анализа данных интеллектуальные системы позволяют развивать процессы оптимизации производственных процессов. Решение задач оптимизации не всегда решаются просто. В частности, оптимизация паровых котлов ТЭЦ имеет несколько целей, которые включают максимизацию эффективности использования топлива, максимизацию производительности, сокращение выбросов вредных веществ в атмосферу. Исследовательская работа посвящена многокритериальной оптимизации паровых котлов с использованием исторических данных, хранящихся в озере данных, построения и анализа границ Парето, и нахождения оптимальных управляющих воздействий, реализующих стратегию оптимального управления.

В качестве цифровой модели парового котла использована нейронная сеть, обученная на исторических данных. Алгоритм сортировки без доминирования (NSGA-II) [1] использован для поиска недоминированных решений с сохранением разнообразия данных. Набор недоминированных решений использован для обоснования числа работающих котлов и конкретных экземпляров котлов. Поскольку каждый промышленный паровой котел характеризуется довольно большим количеством коррелированных параметров (30-60 параметров), было предложено использовать метод главных компонент.

Метод главных компонент [2] — это метод редукции данных, при котором исходное пространство исходных переменных более высокой размерности преобразуется в пространство более низкой размерности. Главные компоненты представляют собой линейные комбинации исходных переменных, а параметры организованы в последовательность с убывающей значимостью для модели котла. Сохраняя этот порядок при выборе значимых значений, первые несколько компонентов, как правило, способны сохранить большую часть вариативности, присутствующей для всех параметров. Выполнение операции PCA позволяет уменьшить количество параметров, используемых для моделирования нейронной сети, и убрать корреляцию между параметрами. Главные компоненты являются не связанными между собой компонентами. Их количество определяет количество нейронов во входном слое

нейронной сети. Метод PCA зависит от масштаба, что означает, что необходимо масштабировать исходные параметры паровых котлов, которые после масштабирования имеют нулевое среднее и единичную дисперсию. Для моделирования характеристик паровых котлов используется нейронная сеть прямого распространения сигнала [3]. Чтобы обеспечить лучшие результаты, необходимо настроить гиперпараметры нейронной сети, к которым относятся число слоев нейронов и число нейронов в каждом слое. Были проверены разные архитектуры нейронной сети, и выбрана архитектура, которая имеет входной слой, три скрытых слоя и выходной слой. Скрытые слои состоят из 32, 16 и 8 нейронов соответственно. Такая архитектура дала меньшую среднюю величину ошибки обучения. Котлы содержат данные наблюдений за один месяц для проведения анализа с использованием границы Парето. Были проанализированы три промышленные котла. Первым этапом обработки является удаление аномальных результатов измерения, восстановление пропущенных значений и нечисловых данных.

Генетический алгоритм недоминированной сортировки (NSGA-II) был использован вследствие его меньшей сложности по сравнению с другими генетическими алгоритмами и более высокого уровня элитарности. Управление паровыми котлами имеет две цели, которые должны быть максимизированы на основе 18 основных параметров [4]. Лица, принимающие решения, должны знать все Парето-оптимальные решения по всем значениям КПД и производительности. Метод возвращает результирующий объект, состоящий из множества недоминированных решений, а также может сохранять главные компоненты оптимальных решений Парето [5].

Алгоритм включает следующие пункты:

1. Чтение исторических данных из озера данных для котельных агрегатов;
2. Удаление выбросов и значений NaN из наборов данных;
3. Постановка задачи многокритериальной оптимизации с целями и ограничениями;
4. Выполнение операции PCA и выбор количества компонентов, которые необходимо использовать при моделировании;
5. Сохранение матрицы преобразования и параметров нормализации;
6. Обучение нейросетевой модели котлов на основе главных компонент;
7. Применение алгоритма NSGA-II и нахождение фронта Парето;
8. Восстановление исходных значений из главных компонентов.

Оптимальные решения Парето обеспечивают решение многокритериальной задачи, но эти решения представлены с использование главных компонентов. Для получения исходных значений необходимо выполнить обратную операцию PCA. Заключительная операция включает в себя принятие решения о выборе котла. Оптимальное решение для каждого котла представляет собой набор решений, поэтому лицо, принимающее решение, должно выбрать оптимальную точку, чтобы выбрать, какой котел должен работать. Это может быть один котел или несколько котлов. Если котлов несколько, то процесс решения будет сложнее. Общее решение является комбинированным на основании решений по Парето каждого из котлов по принципу иерархической оптимизации.

ЛИТЕРАТУРА

1. K. Deb, A. Pratap, S. Agarwal and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," in IEEE Transactions on Evolutionary Computation, vol. 6, no. 2, pp. 182-197, April 2002, doi: 10.1109/4235.996017
2. S. Sehgal, H. Singh, M. Agarwal, V. Bhasker and Shantanu, , "Data analysis using principal component analysis," *2014* International Conference on Medical Imaging, m-Health and Emerging Communication Systems (MedCom), Greater Noida, India, 2014, pp. 45-48, doi10.1109/MedCom.2014.7005973
3. Ganesh Arulampalam, Abdesselam Bouzerdoum, A generalized feedforward neural network architecture for classification and regression, Neural Networks, Volume 16, Issues 5–6, 2003, Pages 561-568, ISSN 0893-6080, https://doi.org/10.1016/S0893-6080(03)00116-3
4. Shuang Gao, Hong Zhao, Zhipeng Bai, Bin Han, Jia Xu, Ruojie Zhao, Nan Zhang, Li Chen, Xiang Lei, Wendong Shi, Liwen Zhang, Penghui Li, Hai Yu, Combined use of principal component analysis and

artificial neural network approach to improve estimates of PM2.5 personal exposure: A case study on older adults, Science of The Total Environment, Volume 726, 2020, 138533, ISSN 0048-9697, https://doi.org/10.1016/j.scitotenv.2020.138533.

5. Arsenyev, D.; Malykhina, G.; Shkodyrev, V. Industrial Process Control Using DPCA and Hierarchical Pareto Optimization. Processes 2023, 11, 3329. https:// doi.org/10.3390/pr11123329

УДК 004.4

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

РАЗРАБОТКА ВЕБ-ПЛАТФОРМЫ ДЛЯ ПОИСКА МАСТЕРОВ ПО РЕМОНТУ

Целью данного проекта является создание веб-платформы, предоставляющей пользователям возможность обсуждения неисправностей в различных устройствах, обмена опытом, поиска мастеров, а также предоставления своих услуг по ремонту. В современном информационном обществе, где технологии играют ключевую роль в организации повседневной жизни, создание такой платформы представляет собой важный и востребованный шаг в развитии онлайн-сервисов [1-3].

Перед разработкой платформы был проведен анализ сайтов, которые тем или иным способом взаимодействуют с пользователями для предоставления услуг мастеров по ремонту различных устройств. В результате были выявлены сильные и слабые стороны конкурентов, что позволило сформулировать требования к собственной платформе:

- регистрация пользователей с предоставлением личной информации;
- форум для обсуждения с удобным поиском;
- удобный интерфейс с возможностью просмотра профилей других пользователей (их рейтинг; темы форума, в которых делали записи и т.д.) [4];
- система обратной связи с поддержкой отзывов и оценок;
- чат для обсуждения;
- возможность добавления заказов под каждой темой форума.

Для реализации платформы был выбран стек PERN (P - PostgreSQL, E - Express, R - React, N - Node.js) [5,6]. PostgreSQL - это мощная объектно-реляционная система управления базами данных с открытым исходным кодом, в которой особое внимание уделяется расширяемости и соответствию стандартам, и которая использует и расширяет язык SQL в сочетании со многими функциями, позволяющими безопасно хранить и масштабировать самые сложные рабочие нагрузки по данным. Express (Back-End Framework) - это платформа веб-приложений для Node.js, используется для создания веб-приложений и особенно API, предоставляет тонкий слой фундаментальных функций веб-приложений, не скрывая при этом уже знакомые функции Node.js. React - библиотека JavaScript для создания пользовательских интерфейсов, поддерживается Facebook и сообществом отдельных разработчиков и компаний, позволяет создавать простые представления для каждого состояния приложения, эффективно обновлять и отображать нужные компоненты при изменении данных, используется для разработки одностраничных приложений или мобильных приложений. Node.js - среда выполнения JavaScript, созданная на базе JavaScript-движка Chrome V8 для разработки серверных и сетевых приложений, используется для создания быстрых и масштабируемых сетевых приложений.

Предлагаемая платформа отвечает нарастающей потребности рынка, внедряя инновационный подход к выбору и предоставлению услуг. Ее реализация обещает улучшение опыта клиентов и эффективность деятельности специалистов, что делает данный проект актуальным и важным в современном бизнес и технологическом контексте.

ЛИТЕРАТУРА

1. Логинов, А. В. Разработка веб-приложения сопровождения процесса отчетности сотрудников / А. В. Логинов, А. И. Тышкевич // 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. 42-44. – EDN DFYLXR.

2. Орлов, Л. О. Управляющая система для умного дома / Л. О. Орлов, А. И. Тышкевич // 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. 53-55. – EDN PILPYY.
3. Шушарин, А. В. Клиентское приложение на базе Android для литературного портала / А. В. Шушарин, А. И. Тышкевич // Современные технологии в теории и практике программирования: Сборник материалов научно-практической конференции студентов, аспирантов и молодых ученых, Санкт-Петербург, 26–27 апреля 2023 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2023. – С. 187-188. – EDN MEGIAA.
4. Зеленова, Д. Д. Автоматизация тестирования пользовательского интерфейса web-приложения / Д. Д. Зеленова, Н. В. Воинов, Т. В. Леонтьева // Современные технологии в теории и практике программирования: сборник материалов конференции, Санкт-Петербург, 19 апреля 2019 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2019. – С. 104-106. – EDN BVYPKM.
5. Чучин, Д. Ю. Разработка серверной части приложения для предоставления интерактивной среды клиентам HTTP API / Д. Ю. Чучин, В. Э. Шмаков // Современные технологии в теории и практике программирования: Сборник материалов конференции, Санкт-Петербург, 26 апреля 2022 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2022. – С. 112-114. – EDN EARKZS.
6. Цифровые технологии. Вычислительная техника. Программирование / В. Е. Баранов, Н. В. Ежова, Ю. В. Сотсков, В. Э. Шмаков. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2019. – 186 с. – ISBN 978-5-7422-6509-2. – EDN GKQIML.

УДК 681.3.06

С. Х. Якупов (4 курс бакалавриата),
В. А. Семенова-Тян-Шанская, к.т.н., доцент

ИСПОЛЬЗОВАНИЕ ПРОГРАММНОЙ СРЕДЫ R ДЛЯ АНАЛИЗА ДАННЫХ МОРСКОГО ФЛОТА

Часто исходные данные предприятий являются «грязными», то есть содержат факторы, мешающие их правильной обработке и анализу (пропуски, аномальные значения, дубликаты и противоречия). Открытые источники данных существуют, но доступ к ним затруднен из-за отсутствия удобных инструментов для сбора и обработки информации. Это приводит к необходимости использования специализированных «парсеров» или «скраперов» для загрузки данных из различных источников. Правильная обработка данных является ключевым элементом для успешного анализа и принятия решений в этой области. Анализ данных морского флота не только позволяет понять текущее состояние отрасли, но и помогает выявить тенденции, определить эффективность операций и прогнозировать будущие потребности и требования. Прежде чем приступать к анализу данных, необходимо выполнить ряд процедур, цель которых — доведение данных до приемлемого уровня качества и информативности, а также организовать их интегрированное хранение в структурах, обеспечивающих их целостность, непротиворечивость, высокую скорость и гибкость выполнения аналитических запросов [1].

Одним из ключевых преимуществ программной среды R является ее богатый набор пакетов и библиотек, предназначенных для выполнения различных задач анализа данных, включая статистическое моделирование, машинное обучение, визуализацию данных и многие другие. R также обладает мощными возможностями в работе с таблицами данных и векторами, что делает его идеальным инструментом для исследования и анализа разнообразных датасетов [2]. В нашей задаче на входе имеем очищенный от ошибок и неточностей датасет по судам морского флота. База данных морского флота представлена на сервере PostgreSQL[3] . Эта база данных содержит информацию о судах, их характеристиках, маршрутах, экипажах и перевозимых грузах. Схема базы данных представлена на рисунке 1

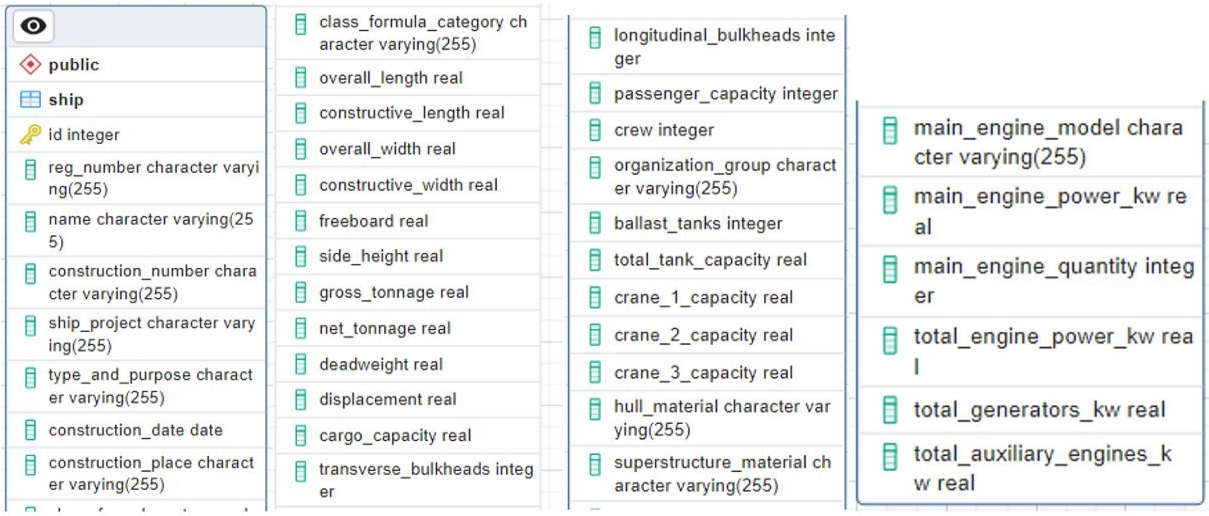


Рисунок 1 – Схема базы данных

С помощью языка R легко получить данные из базы данных PostgreSQL для дальнейшего анализа. Програмная среда R предоставляет удобные инструменты, такие как пакет

RPostgreSQL, который позволяет устанавливать соединение с базой данных и извлекать необходимую информацию для анализа. Это обеспечивает исследователям и аналитикам простой и эффективный способ работы с данными, хранящимися в PostgreSQL. Пример датасета, полученного из базы данных PostgreSQL, показан на рисунке 2

class_formula_category	overall_length	constructive_length	overall_width	constructive_width	freeboard
М-ПР2,5	43.18	40.0	7.86	7.4	1.080
М-ПР2,5	40.80	39.1	7.40	7.4	1.028
Р 1,2 II	116.40	100.0	31.00	30.1	0.200
Р1,2 II	90.00	90.0	26.00	26.0	0.410

Рисунок 2 – Пример датасета

Интеграция языка R с базой данных Postgres предоставляет возможность анализировать и обрабатывать данные напрямую из базы, используя мощные аналитические возможности R. С помощью специальных пакетов, таких как RPostgres, пользователи могут выполнять SQL-запросы из среды R и эффективно взаимодействовать с данными, хранящимися в Postgres, для выполнения различных аналитических задач и создания статистических моделей [4]. Эта интеграция упрощает процесс анализа данных и позволяет исследователям и аналитикам получать более полное понимание своих данных.

Матрица корреляции или correlation heatmap составлена на основе данных, полученных из базы данных PostgreSQL, также с использованием языка программирования R.

Матрица корреляции (рисунок 3) — это таблица, которая отображает коэффициенты корреляции между переменными в датасете. Коэффициент корреляции показывает, насколько сильно и в каком направлении реализована связь между двумя переменными.

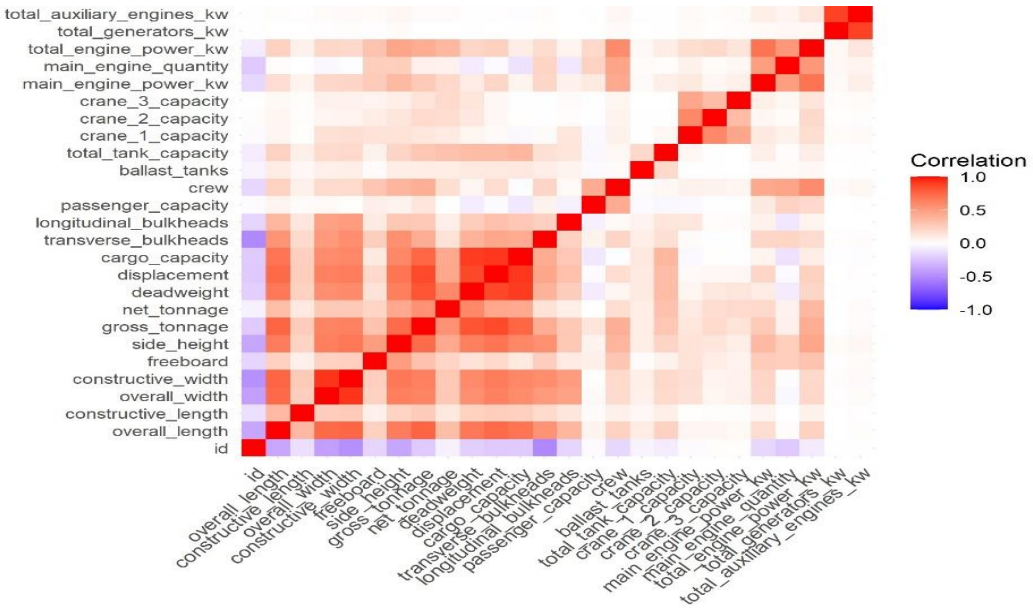


Рисунок 3 – Матрица корреляции

Если две переменные сильно коррелируют между собой, это может указывать на схожесть в их поведении или характеристиках. Используя информацию о сходстве между переменными, можно попытаться сделать кластеризацию для выявления групп или кластеров, в которых объекты (в данном случае переменные) схожи между собой.

Пример кластеризации данных (рисунок 4) морского флота с использованием языка R.

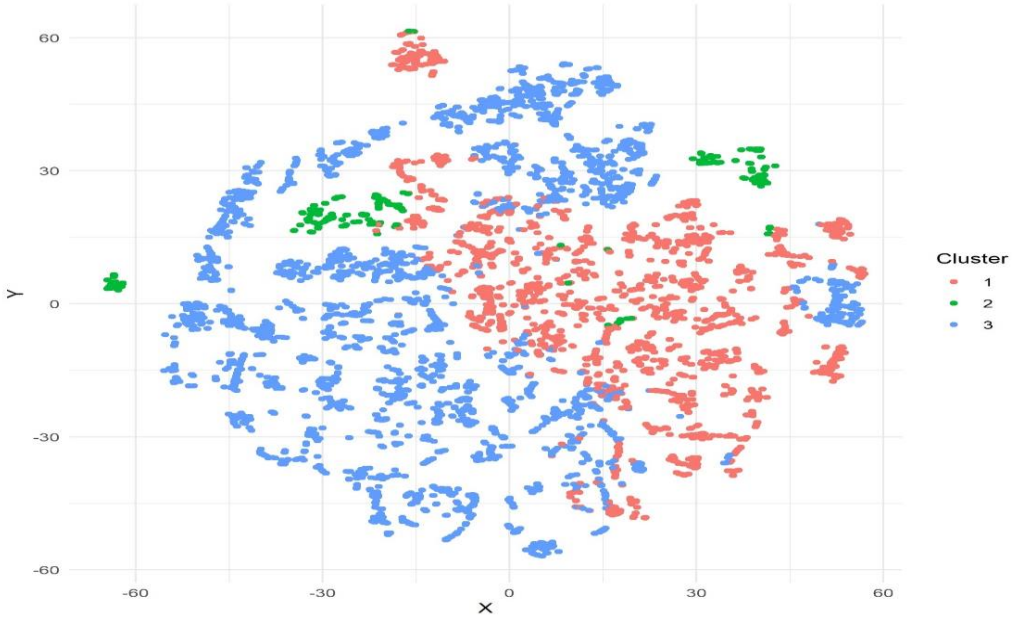


Рисунок 4 – Пример полученных кластеров

ЛИТЕРАТУРА

1. Шаблоны корпоративных приложений: Пер. с англ – М. : Издательский дом «Вильямс», 2011. – 544 с.
2. Кабаков Р. R в действии. Анализ и визуализация данных на языке R , М. ДМК Пресс, 2014. – 474 с.
3. Рогов Е. PostgreSQL изнутри, М.ДМК Пресс, 2023. – 660 с
4. Домбровская Г., Новиков Б., Бейликова А. Оптимизация запросов в PostgreSQL / пер. с англ. Д. А. Беликова. – М.: ДМК Пресс, 2022. – 278 с.: ил

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

УДК 004.65

Я. А. Алексеев (4 курс бакалавриата),
С. А. Нестеров, к. т. н., доцент

СРАВНИТЕЛЬНЫЙ АНАЛИЗ СРЕДСТВ ОБРАБОТКИ ПРОСТРАНСТВЕННЫХ ДАННЫХ В СУБД SQL SERVER И POSTGRESQL

Развитие геоинформационных систем (ГИС) в последние годы привело к повышенному вниманию со стороны разработчиков систем управления базами данных (СУБД) к работе с пространственными данными. Возрастающие требования ГИС к объему и надежности хранения данных привели к их интеграции с мощными универсальными СУБД [1].

В SQL Server поддержка пространственных данных впервые появилась в версии 2008. СУБД предоставляет пространственные типы данных geometry и geography и методы для работы с ними. Также при поддержке Майкрософт разработана коллекция инструментов с открытым исходным кодом для использования с пространственными типами в SQL Server. Инструменты предоставляют набор повторно используемых функций, в частности, для преобразования данных, которые могут использовать приложения [2].

В популярной СУБД с открытым исходным кодом PostgreSQL работа с пространственными данными реализуется при помощи расширения PostGIS [3], предназначенного для хранения в базе данных пространственной информации, включая поддержку пространственных индексов R-Tree/GiST и функций обработки геоданных [4].

Целью данной работы является анализ средств обработки пространственных данных в СУБД SQL Server и PostgreSQL, что может быть полезно при миграции данных и при разработке приложений, способных использовать любую из этих СУБД для хранения данных.

Для достижения этой цели необходимо решить следующие задачи:

1. Рассмотреть методы обработки пространственных данных в PostGIS и SQL Server.
2. Сравнить работу пространственных индексов в указанных СУБД.
3. Рассмотреть программы визуализации пространственных данных, которые могут работать с обеими СУБД.
4. Выполнить перенос тестового набора данных, включающего пространственные данные, с одной СУБД на другую и проанализировать результаты.

В настоящий момент рассмотрены предоставляемые СУБД типы данных, определены схожие по функциональности методы работы с ними. Надо отметить, что основные фукнции схожим образом реализованы в обеих СУБД, однако PostGIS предоставляет более широкий набор функций.

Проведен анализ средств визуализации пространственных данных, предоставляемых стандарными средствами работы с СУБД. Также рассмотрены фукнции популярной географической информационной системы с открытм исходным кодом QGIS [5] по работе с пространственными данными, хранимыми в СУБД.

В экспериментальной части работы пространственные данные информационной системы в области судоходства, представленые в базе данных PostgreSQL, с помощью популярного кросс-платформенного инструмента для администрирования баз данных DBeaver [6], были перенесены в таблицы базы данных SQL Server. Это позволит проводить дальнейшие эксперименты на одинаковых базах данных.

В ближайшее время будут рассмотрены особенности индексирования пространственных данных в указанных СУБД. С точки зрения синтаксиса, в PostGIS методы GiST, BRIN, SP-GiST применяются к индексам, тогда как в SQL Server используется команда CREATE SPATIAL INDEX с различными параметрами. В случае миграции баз данных, подобные синтаксические различия в диалектах SQL разных СУБД обязательно нужно учитывать.

Результаты работы планируется использовать при выполнении проекта по разработке информационной системы в области судоходства.

ЛИТЕРАТУРА

1. Шестаков Н. А. Индексирование пространственных данных в СУБД Microsoft SQL Server 2000 / Н. А. Шестаков // Известия Томского политехнического университета [Известия ТПУ]. — 2006. — Т. 309, № 4. — [С. 157-162].
2. Пространственные данные. Статья Learn Microsoft [Электронный ресурс] Режим доступа: https://learn.microsoft.com/ru-ru/sql/relational-databases/spatial/spatial-data-sql-server?view=sql-server-ver16
3. PostGIS [Электронный ресурс] Режим доступа: https://postgis.net/development
4. Черноморец, Д. А. Обзор пространственных баз данных на основе СУБД PostgreSQL и SQLite / Д. А. Черноморец, П. В. Васильев // Наука и образование: отечественный и зарубежный опыт : 19 междунар. науч.-практ. конф., Белгород, 19 апр. 2019 г. : сб. ст. / ред. кол.: С. И. Линник-Ботова, О. А. Гагауз. - Белгород, 2019. - С. 50-54. - Библиогр.: с. 53-54.
5. QGIS Documentation [Электронный ресурс] Режим доступа: https://www.qgis.org/en/docs/index.html
6. DBeaver Documentation [Электронный ресурс] Режим доступа: https://dbeaver.com/docs/dbeaver/

УДК 004.453

Е. Д. Андреева (2 курс магистратуры),
П. Д. Дробинцев, к.т.н., доцент

ОРКЕСТРАЦИЯ АВТОМАТИЗИРОВАННЫХ ТЕСТОВ НА WEB ПЛАТФОРМЕ

Автоматизация тестирования является неотъемлемой частью разработки программного продукта. Это один из инструментов обеспечения качества, который позволяет увеличить скорость тестирования и эффективность, а также упростить работу тестировщикам. Автоматизация особенно выгодна при использовании CI/CD подхода в разработке.

Однако недостаточно только написать набор тестов. Большую роль в автоматизации играет то, как мы запускаем тесты. Продуманное управление запуском автотестов позволяет гибко подходить к процессу тестирования, здесь мы говорим о создании и управлении самими запуском и остановкой автотеста, а также сбором метрик и составлением отчетов о результатах тестирования. Ускорение прохождения автотестов часто осуществляется за счет параллелизации запуска, но мы можем еще больше сократить время, если правильно отсортируем их.

Более того мы говорим про набор тестов на Web платформе, то есть про UI автотесты, а как известно они всегда нестабильные. Есть часть задач по стабилизации автотестов, которые мы можем передать «раннеру», например, автоматический перезапуск упавших автотестов сразу после прогона и включение в финальный отчет только последних результатов.

Таким образом целью данной работы является оптимизация запуска тестового набора на Web платформе за счет оркестрации автоматизированных тестов.

Для достижения поставленной цели необходимо решение следующих задач:

1. Анализ существующих решений;
2. Разработка архитектуры для проектируемой системы запуска автотестов;
3. Подключение в проект с Web автотестами необходимых инструментов;

4. Настройка «фермы» автотестов и взаимодействия с проектом автотестов;
5. Реализация логики запуска автотестов в проекте с Web тестами;
6. Проведение запуска автотестов с помощью нового «раннера»;
7. Анализ полученных результатов относительно старой реализации.

Один из способов управления запуском автотестов – это использование готовых фреймворков. Но того, что они предлагают, становится недостаточно, когда мы говорим о больших проектах с множеством собственных решений. Хочется иметь возможность легко добавлять новый функционал, поддерживать работоспособность, кроме того, использование своих собственных разработок обеспечивает нам повышение безопасности.

Архитектура системы представлена на рисунке 1.

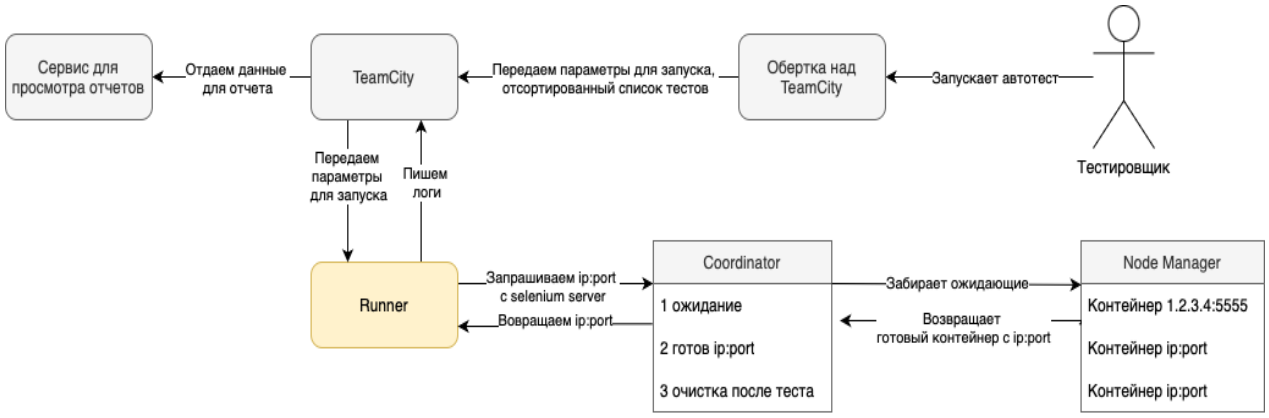


Рисунок 1 – Системный уровень разработки

Код для запуска находится в классе Runner. Он реализован с использованием API фреймворка JUnit5 [1] и написан на языке программирования Java 17, как и сами тесты.

Класс Runner берет на себя задачи от подготовки среды до формирования финального отчета. Мы задаем параметры через обертку над TeamCity, важно отметить, что на этом этапе выполняется алгоритм сортировки тестов, необходимый для корректного распределения тестов в параллельном запуске, учитывая их среднее время прохождения, далее перед запуском тестового набора находим свободные контейнеры, используя взаимодействие с внутренними сервисами, наподобие Kubernetes, каждый тест будет запущен в отдельном контейнере с помощью класса Runner.

В результате такого подхода управления запусками как показательные результаты ожидаем увидеть сокращение времени выполнения прогона автотестов до длительности самого долгого теста за счет правильной сортировки и параллельного запуска, а также увеличение стабильности автотестов за счет добавления автоматических перезапусков.

ЛИТЕРАТУРА

1. The JUnit Team, «JUnit 5 User Guide», [Электронный ресурс]. Режим доступа: https://junit.org/junit5/docs/current/user-guide/. [Дата обращения: 17.03.24].
2. CoduimAI, «Test Runner», [Электронный ресурс]. Режим доступа: https://www.codium.ai/glossary/test-runner/. [Дата обращения: 17.03.24].
3. E. Soares, «Refactoring Test Smells With JUnit 5: Why Should Developers Keep Up-to-Date?», [Электронный ресурс]. Режим доступа: https://ieeexplore.ieee.org/abstract/document/9769994. [Дата обращения: 17.03.24].
4. S. Gulati и R. Sharma, «JUnit 5 Extension Model», [Электронный ресурс]. Режим доступа: https://link.springer.com/chapter/10.1007/978-1-4842-3015-2_7. [Дата обращения: 17.03.24].
5. G. W. Bielesza, «Comparison of tools for creating and conducting automated tests», [Электронный ресурс]. Режим доступа: https://ph.pollub.pl/index.php/jcsi/article/view/3804. [Дата обращения: 17.03.24].

УДК 004.054

С. А. Аненко (2 курс магистратуры),
П. Д. Дробинцев, к. т. н., доцент

ОБЗОР СУЩЕСТВУЮЩИХ МЕТОДИК ОЦЕНКИ ТЕХНОЛОГИЧЕСКОЙ УСТОЙЧИВОСТИ ИТ-ЛАНДШАФТА КОМПАНИИ

В современном мире цифровые технологии играют все большую роль в деятельности людей, а также в деятельности предприятий. Хотя значимость степени влияния цифровых технологий на производственный процесс может разниться на различных предприятиях, нельзя отрицать, что общая тенденция говорит о все возрастающей роли, которую информационные системы (ИС) играют в деятельности компании. Сегодня невозможно представить хоть сколько-нибудь функционирующие предприятия без функционирующих ИС.

В этом контексте представляется актуальным вопрос о том, как именно устойчивость ИТ-ландшафта влияет на непрерывность деятельности предприятия.

Целью работы являлось проведение обзора существующих методик оценки технологической устойчивости ИТ-ландшафта компании.

ИТ-ландшафт компании представляет собой совокупность информационных систем, сервисов, услуг и продуктов, которые использует предприятие для поддержания и развития своего функционирования [1].

ИТ-ландшафт предприятия с технической точки зрения состоит из ряда ключевых компонентов:

– технологического слоя, включает в себя ИТ-инфраструктуру, которая является основой для функционирования слоя приложений и слоя сервисов и продуктов;

– слоя приложений, представляющим собой совокупность прикладного программного обеспечения, которое может поставляться вендорами (поставщиками) или разрабатываться самостоятельно;

– слоя сервисов и продуктов, который представляет собой верхнюю ступень данной системы и является тем слоем, с которым непосредственно сталкиваются заинтересованные стороны, поставку ценности которым осуществляет ИТ-ландшафт.

Рассмотрим теперь характеристики ИТ-ландшафта предприятия. В контексте настоящей работы ключевыми характеристиками для рассмотрения являются непрерывность и устойчивость ИТ-ландшафта предприятия. Существуют различные подходы к определению непрерывности ИТ-ландшафта (или ИТ-непрерывности) предприятия, акцентирующие внимание на различных аспектах. Так непрерывность может быть определена, как способность организации защитить свои данные в случае наступления запланированных или внезапных негативных событий и одновременно поддержка инициатив развития бизнеса, в основе которых лежит принятие управленческих решений на основе больших данных, а также цифровая трансформация бизнеса. Возможно также определение ИТ-непрерывности, как способности организации поддерживать приемлемый уровень сервиса в условиях воздействия негативных факторов на бизнес-процессы организации или её ИТ-ландшафт [2]. Также ИТ-непрерывность может пониматься, как состояние, при котором компания имеет возможности для защиты её данных и поддержании работоспособности своих информационных систем без перебоев максимально возможный период времени.

ИТ-непрерывность позволяет организации быть готовой к любым типам негативного воздействия и снижать риски нахождения системы в неработоспособном состоянии и возможных потерь данных [3].

Устойчивость ИТ-ландшафта предприятия или ИТ-устойчивость является более широкой категорией по сравнению с ИТ-непрерывностью и может быть определена в самом общем смысле как способность ИТ-ландшафта предприятия обеспечивать поставку ценности заинтересованным сторонам в долгосрочной перспективе [4].

Следует отметить, что в иностранных источниках в отношении ИТ-устойчивости используется не конструкция «готовность к воздействию негативных факторов», а слово «*disruption*», которое переводится, как разрушение. Т. е. имеется в виду готовность организации в том числе и к физическому разрушению элементов её архитектуры. Таким образом, ИТ-непрерывность может быть определена в рамках различных подходов, когда во главу угла ставится, как поддержание приемлемого уровня сервиса или сохранение способности организации двигаться в векторе цифровой трансформации, тем не менее все определения имеют один общий аспект – все они определяют ИТ-устойчивость, как способность ИТ-ландшафта организации противостоять различным угрозам, в том числе и физическим разрушениям.

Анализ открытых источников свидетельствует об отсутствии на данный момент разработанных комплексных методик для оценки непрерывности и устойчивости ИТ-ландшафта предприятия, что потребует составления собственных методик для оценки данных характеристик ИТ-ландшафта предприятия. Для разработки собственной комплексной методики были изучены различные источники, в частности опросник компании *Zerto* для оценки зрелости системы по обеспечению ИТ-непрерывности на предприятии [5], а также манифест ИТ-непрерывности Маккинси [6], работа «Стратегическое управление ИТ-ландшафтом» Инге Ханшке [2].

В результате проведенного анализа были выделены ключевые индикаторы и параметры, которые лягут в основу комплексной методики для оценки ИТ-непрерывности и ИТ-устойчивости.

ЛИТЕРАТУРА

1. Семенова, Л. В. ИТ-ландшафт современных компаний для поддержки бизнес-процессов / Л. В. Семенова, Т. А. Крамаренко // 102 Цифровизация экономики: направления, методы, инструменты : Сборник материалов IV всероссийской научно-практической конференции, Краснодар, 17–21 января 2022 года. – Краснодар: Кубанский государственный аграрный университет имени И.Т. Трубилина, 2022. – С. 93-95.
2. Hanschke, I. Strategic IT Management / I. Hanschke – Ottobrunn : Springer, 2009.
3. Zerto // IT Resilience A-to-Zerto Glossary of Terms [Электронный ресурс]. URL: https://www.zerto.com/resources/a-to-zerto/what-is-it-resilience/ (дата обращения: 16.03.2024).
4. Gartner is Special report for technology providers // Tech Providers 2025: Strategic Transformation Drives Growth, 2022.
5. Zerto // IT Resilience Maturity Benchmark Tool [Электронный ресурс]. URL: https://www.zerto.com/it-resilience-maturity-benchmark-tool/?inid=zerto_what-is-itresilience_gl_to_it-resilience-maturity-benchmark-tool_lp (дата обращения: 16.03.2024).
6. McKinseyDigital // IT resilience for the digital age [Электронный ресурс]. URL: https://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/techforward/it-resilience-for-the-digital-age (дата обращения: 16.03.2024).

УДК 004.455.2

А. А. Афонина (2 курс магистратуры),
А. Е. Кузькин, начальник отдела,
И. В. Никифоров, к.т.н., доцент

КОНЦЕПЦИЯ МОНИТОРИНГА ДЕГРАДАЦИИ, ОБНОВЛЕНИЯ И МОДЕРНИЗАЦИИ ОТДЕЛЬНЫХ КОНФИГУРАЦИОННЫХ ЕДИНИЦ ВЫЧИСЛИТЕЛЬНОГО ОБОРУДОВАНИЯ

Стандартом ИТ-области на текущий момент является организация защищенной системы для работы с собственными данными. Многие компании выбирают облачные решения, перенося свою ИТ-инфраструктуру на внешние сервера по PaaS-модели (платформа как сервис), не сразу получившей внимание со стороны бизнеса, но со временем прочно

закрепившейся в мировой практике [1]. Данная модель позволяет использовать только необходимый объем мощностей и соответственно оплачивать только его, а также соблюсти отказоустойчивость ИТ-кластера, так как большинство поставщиков облачных сервисов работает на территориально распределенных вычислительных сетях, тем самым обеспечивая георезервирование, которое сама компания не могла бы поддерживать [2].

В свою очередь крупные игроки российского рынка предпочитают организовывать собственную ИТ-инфраструктуру, максимально находящуюся под контролем внутреннего сервиса, арендуют или строят ЦОД (центры обработки данных). В ситуации, когда в приоритет ставится информационная безопасность [3] и сохранение коммерческой тайны, даже высокие показатели TCO (Total Cost of Ownership) [4] и TCS (Total Cost of Services) оправданы в глазах бизнеса. Главным условием со стороны бизнеса становится максимальная оптимизация утилизации собственных ИТ-ресурсов, как уже имеющихся, так и запланированных к приобретению, так как вопрос постоянной нелинейной деградации существующего стека ИТ-решений [5] заставляет закладывать закупку ресурсов с запасом мощностей для резервирования и возможного масштабирования.

Мониторинг состояния имеющейся ИТ-архитектуры ЦОД, в частности серверов и систем хранения данных, стал областью развития для современных технологий [6], а способность предсказать возможные отклонения в состоянии и предотвратить инциденты, способные приостановить работу инфраструктуры, а следовательно, и деятельность компании. В контексте построения алгоритма проверки текущей конфигурации вычислительного оборудования [7] и формирования концепции модернизации этой конфигурации становится важна стратегия консолидации информации об отдельных конфигурационных единицах и их возможных модернизациях.

Рассмотрим пример: сервера N и M одного производителя были закуплены в одной партии от одного поставщика, поставлялись в одинаковой конфигурации и работали бесперебойно в течение последних двух лет. Логично, что в таком случае для алгоритма они взаимозаменяемы, и их исторические данные сводятся к датам покупки и запуска. Мы понимаем, что вероятность вывода сервера из эксплуатации тем выше, чем большее количество инцидентов на нем произошло и большего возраста он достиг. Предположим, что на сервере N вышла из строя планка оперативной памяти, и это событие было зафиксировано как инцидент. Возникает вопрос, как поступить с сервером N как объектом в рамках работы алгоритма, если техническая проблема была именно в планке оперативной памяти и на ее место была установлена идентичная по характеристикам новая.

К рассмотрению предлагаются три подхода к решению вопроса о записи данных по обновлению отдельных элементов объекта. Рассмотрим каждый из них.

1. Моделецентричный подход – сервера N и M являются серверами одной модели, объекты, используемые алгоритмом практически идентичны и якорной характеристикой будет именно модель сервера. В описанном случае для одного из серверов мы увеличим счетчик инцидентов, и в случае установки планки оперативной памяти иного объема отмечаем это в соответствующем поле значения метрики. Ни характер замены, ни дата в этом случае не логируются, количество объектов не меняется.

2. Возможно ли считать такой объект абсолютным продолжением исходного сервера N? При подходе предок-потомок мы сохраняем в архиве один объект-предок – сервер N в конфигурации, на которой произошел инцидент, и создаем новый объект-потомок – сервер N с обновленным наполнением. Проблема данного метода в том, что, во-первых, с каждой заменой наполнения множатся объекты в архиве, которые тоже необходимо учитывать при работе алгоритма предсказания инцидентов, во-вторых, неясно, передавать ли в новые объекты информацию об инцидентах объекта-предка и если да, то в каком объеме.

3. Исторический метод. Он предполагает хранение в самом объекте логов как о заменах, так и об инцидентах с маркером комплектующей, на которой инцидент произошел. В этом случае также неясно, стоит ли хранить инцидент, если проблемную деталь заменили.

В целом, без апробации в условиях, приближенных к реальным, невозможно сделать вывод, какой из подходов является оптимальным. Вопрос выбора способа построения объектной модели для работы алгоритма верификации конфигурации вычислительных машин требует глубокой детальной проработки и может являться отдельным исследованием. В данном тезисе рассмотрены первичные подходы, проверка актуальности и доработка которых возможна только на большом объеме реальных конфигурационных данных.

ЛИТЕРАТУРА

1. D. S. Linthicum, "PaaS Death Watch?," // IEEE Cloud Computing, Jan.-Feb. 2017. – vol. 4, no. 1, pp. 6-9.
2. San Murugesan; Irena Bojanova, "Cloud Federation and Geo-Distribution," in Encyclopedia of Cloud Computing, IEEE, 2016, pp.178-190.
3. Зегжда, П. Д. Информационная безопасность и киберустойчивость цифровой экономики и цифрового производства / П. Д. Зегжда, Д. П. Зегжда // Региональная информатика и информационная безопасность, Санкт-Петербург, 01–03 ноября 2017 года. – Санкт-Петербург:Санкт-Петербургское Общество информатики, вычислительной техники, систем связи и управления, 2017. – С. 406-407.
4. Algorithm for calculating TCO and SCE metrics to assess the efficiency of using a data center / B. Denisenko, M. Tyanutov, I. Nikiforov, S. Ustinov // 2nd International Conference on Computer Applications for Management and Sustainable Development of Production and Industry (CMSD-II-2022), Dushanbe, 21–23 декабря 2022 года. – Washington: SPIE-SOC PHOTO-OPTICAL INSTRUMENTATION ENGINEERS, 2023. – P. 1256403. – DOI 10.1117/12.2669285. – EDN TSQXHS.
5. H. Yang, Y. Liang, C. Tan and J. Fu, "Detecting software aging of web servers with real-valued negative selection algorithm," 2011 IEEE 3rd International Conference on Communication Software and Networks, Xi'an, China, 2011, pp. 298-302.
6. Лялин Д. С., Мамадалиев Ш. Р., Никифоров И. В., Устинов С. М. Проектирование системы мониторинга и расчета метрик для эффективного использования ресурсов центров обработки данных // Современные технологии в теории и практике программирования : Сборник материалов конференции, Санкт-Петербург, 26 апреля 2022 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2022. – С. 194-197.
7. Афонина А. А., Кузькин А. Е., Никифоров И. В. Алгоритмизация конфигураций оборудования в цод как базис для автоматизации обновления оборудования // Современные технологии в теории и практике программирования : Сборник материалов конференции, Санкт-Петербург, 26 апреля 2023 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2023. – С. 115-117.

УДК 004.02

Н. И. Белоногов (2 курс магистратуры),
Г. Ф. Малыхина, д.т.н., профессор

РАЗРАБОТКА СИСТЕМЫ ДОСТУПА К НЕРЕЛЯЦИОННОЙ БАЗЕ ДАННЫХ TARANTOOL НА ОСНОВЕ SPRING DATA

В мире современной разработки программного обеспечения, где гибкость и расширяемость играют решающую роль, использование фреймворков, обеспечивающих удобство работы с данными, становится неотъемлемой частью создания высококачественных приложений. В этом смысле Spring Data [1], часть мощного семейства Spring Framework, занимает отдельное место, предоставляя удобные и эффективные средства для работы с различными источниками данных.

Однако, несмотря на свою популярность, Spring Data имеет реализацию для небольшого количества наиболее популярных хранилищ данных, представленных на рынке, такие как Redis или MongoDB, что накладывает определенные ограничения на работу отдельных разработчиков и даже целых организаций [1-2].

Tarantool, как нереляционная база данных, обладает своими особенностями и подходами к хранению данных, что требует тщательного проектирования и адаптации стандартных концепций Spring Data. Проблема разработки модуля, также включает в себя эффективное использование возможностей Tarantool [3], таких как возможность хранение и манипулирование данными, создание триггеров и хранимых процедур с использованием языка LUA, репликация и шардирование, и интеграцию подобной функциональности в концепции модуля Spring Data.

Таким образом, целью работы является разработка системы доступа к нереляционной базе данных Tarantool на основе Spring Data. Для достижения этой цели решены следующие задачи:

1. Проведение аналитического обзора методов и подходов к работе с базами данных в языке программирования Java.
2. Исследование спецификации Spring Data для дальнейшей разработки системы доступа [4].
3. Разработка компонентов для работы модуля Spring Data.

Разработка компонентов, основана на нескольких важных аспектах Spring: конфигурация контекста Spring, а также разработка компонентов, необходимых для работы с абстракцией Repository.

Разработанные компоненты настройки контекста Spring включают в себя конфигурацию на основе Java классов [5], конфигурации на основе аннотаций (аннотация @EnableTarantoolRepository) [5], конфигурация на основе файлов настройки (properties / yaml) [5]. Данные подсистемы позволяют настраивать клиенты распределенного подключения к базе данных (порты, хост-имена, количество и состав групп подключения и другие), а также выбрать вариант балансировки.

Каждый из вариантов настройки имеет свои плюсы и минусы. Настройка на основе конфигурационных файлов позволяет задавать профили работы, независимые друг от друга, компоненты контекста Spring которых, не пересекаются [6]. Настройка на основе классов Java позволяет задать настройки клиентов более низкоуровнево, в том числе и параметры, которые недоступны в настройке на основе конфигурационных файлов.

В купе с конфигурацией на основе классов Java, конфигурация на основе аннотации @EnableTarantoolRepositories позволяет подключить возможность использования абстракции Repository и ее преимущества - задание запросов через названия методов запроса, в том числе с пагинацией и сортировкой [7].

Разработка компонентов, ответственных за использование абстракции Repository, состоит в том, чтобы реализовать основные столпы этой технологии: отображение предметных сущностей в вид, который способен воспринимать низкоуровневый драйвер целевой базы данных, реализация шаблона доступа к базе данных (в моем случае - Tarantool), реализация синтаксического разбора [8] названий производных методов-запросов (рисунок 1).

```
public interface UserRepository extends Repository<User, Long> {

  List<User> findByEmailAddressAndLastname(String emailAddress, String lastname);
}
```

Рисунок 1 – Пример производного метода-запроса поиска записей-кортежей по электронному адресу и фамилии.

Учитывая особенности используемой нереляционной базы данных [3], а именно возможность создания и использования хранимых процедур и запросов на языке программирования LUA, отдельной задачей стоит реализация возможности использования

методов, помеченных аннотацией @Query [9]. Помеченные этой аннотацией методы производят вызов хранимой процедуры, название которой передается в аргумент самой аннотации (рисунок 2).

```
@Query("echo_procedure")
List<?> getEcho(String string, Boolean bool, Integer integer);
```

Рисунок 2 – Пример производного метода-запроса, который вызывает хранимую процедуру из базы данных с передачей аргументов.

Также важно предоставить пользователям возможность писать произвольные запросы, используя язык LUA прямо из разрабатываемой системы доступа. Для реализации этой особенности решено использовать аннотацию @Query - генерация запросов на языке LUA происходит посредством передачи LUA кода (скрипта) в аннотацию. В итоге, скрипт передается в базу данных на исполнение.

Таким образом, система доступа к нереляционной базе данных Tarantool была реализована на языке программирования Java и позволяет осуществлять распределенный доступ на базе модуля Spring Data с интеграцией к компонентам Spring Framework. Стоит заметить, что такой подход позволяет уменьшить количество написанного разработчиком кода, а также увеличить переносимость системы при выборе другой базы данных, для которой также реализована система доступа на основе Spring Data.

ЛИТЕРАТУРА

1. Mark Pollack, Oliver Gierke, Thomas Risberg Spring Data. - O'Reilly Media, Inc, 2012. - 288 с.
2. Гутьеррес Фелипе Spring Boot 2: лучшие практики для профессионалов. - "Издательский дом ""Питер""", 2020. - 464 с.
3. Tarantool Documentation. [Электронный ресурс] Режим доступа: https://clck.ru/39WKVb
4. Spring Data Commons. [Электронный ресурс] Режим доступа: https://clck.ru/39WKdR
5. Catalin Tudose, Christian Bauer, Gavin King Java Persistence with Spring Data and Hibernate. - Manning, 2023. - 562 с.
6. Ю. Козмина, Р. Харроп, К. Шефер, К. Хо. Spring 5 для профессионалов. : Пер. с англ. - СПб. : ООО "Диалектика", 2019. - 1120 с. : ил. - Парал. тит. англ. ISBN 978-5-907114-07-4G.
7. EnableRedisRepositories. [Электронный ресурс] Режим доступа: https://clck.ru/39WK9U/
8. PartTree. [Электронный ресурс] Режим доступа: https://clck.ru/39WKHu
9. Query. [Электронный ресурс] Режим доступа: https://clck.ru/39WKPS

УДК 004.052.42

А.О. Бушкина (4 курс бакалавриата),
П.Д. Дробинцев, к.т.н., доцент

АВТОМАТИЗАЦИЯ СОЗДАНИЯ ТЕСТОВОГО НАБОРА НА ПЛАТФОРМЕ ANDROID С ПОМОЩЬЮ ИСПОЛЬЗОВАНИЯ СТАТИЧЕСКОГО АНАЛИЗА КОДА

Сложность и большой объем тестирования программного обеспечения в развивающемся проекте приводит к необходимости использования подходов, позволяющих ускорить данный процесс. Автоматизация регрессионного набора тестов позволяет повысить скорость проверки функционала, так как с большой точностью может указать место ошибки [1] и поддержать уровень качества кодовой базы при наличии вносимых изменений. Наличие автоматизированных тестов на проекте экономит ресурсы команды тестирования на проверку функционала, который не должен был быть изменен, но добавляет расходы на поддержание актуальности такого набора. Так как множество разработчиков ежедневно вносят изменения в код приложения, то внесение исправлений в код автоматизированных тестов также является постоянной составляющей цикла разработки, в связи с чем актуальна задача минимизации

времени, затрачиваемого на поддержание тестов в рабочем состоянии.

В процессе создания или исправления автоматизированного теста частой практикой является инспекция нового кода, вносимого в проект. Ревью кода [2, с. 99], производимое другими разработчиками, способствует поддержанию единообразия кодовой базы и исключению попадания потенциально проблемного кода в основную ветку проекта, но значительно увеличивает время, добавляя итерации исправлений ошибок и ожидания повторной проверки ревьюером. Кроме того, процесс проверки кода осложняется наличием человеческого фактора: комментарии могут быть неясны или неточно описывать проблему, разработчик может не иметь возможность за короткий промежуток времени перепроверять внесенные изменения [3].

Ускорения данного процесса позволяет достичь внедрение статического анализа кода [4], так как при использовании предложенного подхода сразу после написания кода разработчику предоставляется отчет об обнаруженных проблемах и производится их автоматическая перепроверка после исправления. Для обеспечения статического анализа используются линтеры, которые дают возможность настроить правила проверки кода и указать на ошибки без участия человека. Линтер выявляет несоответствия заранее заданным конструкциям, после чего формирует отчет о найденных ошибках, их расположении в коде и, в некоторых случаях, предлагаемых исправлений для устранения проблем. Инструмент статического анализа не гарантирует локализацию все дефектов кода [5], но способен уменьшить их количество, поэтому добавление описанного этапа проверки позволяет упростить последующий процесс инспекции кода разработчиками и сократить количество итераций исправлений.

Так как рассматриваемый для улучшения Android проект имеет на данный момент тестовый набор, состоящий примерно из 1500 тестов на языке Java, CI/CD осуществляется на с помощью инструментов, предоставляемых Bitbucket, то линтером, подходящим под специфику проекта, был выбран PMD [6], как гибкий в настройке инструмент статического анализа.

PMD содержит встроенные категории правил, которые специализируются на определенном аспекте кода, например проверка стиля кода, обнаружение потенциальных уязвимостей безопасности, проблемы документирования кода и другие. Изучение существующих категорий позволяет составить необходимую структуру линтера и определить набор пользовательских правил, которыми нужно дополнить существующие выбранные стандартные проверки. Уязвимые ситуации, на которые следует обращать внимание анализатору, выявляется в ходе обсуждения с разработчиками, участвующими в ревью кода для получения актуальных и часто встречаемых проблем. Таким образом в ходе реализации формируются конфигурационные файлы для линтера в формате xml, которые позволяют локально получать отчет о вносимых изменениях в код.

Для внедрения статического анализатора в CI/CD создаются конфигурации в TeamCity, в которых настраиваются триггеры для автоматического запуска проверки кода после каждого коммита в ветку. Такие настройки позволяют получать информацию о состоянии кода при внесении изменений и отображать ее в pull-request в виде подробного настроенного отчета с разделением найденных проблем по уровню приоритета.

Таким образом, внедрение в цикл разработки автоматизированных тестов статического анализатора кода позволяет ускорить процесс ревью, сделать код более качественным вследствие чего упростить автоматизацию тестового набора.

ЛИТЕРАТУРА

1. Надыкто, М. О. Разработка автотестов для обеспечения качества программного обеспечения / М. О. Надыкто, С. В. Белим // Образование. Транспорт. Инновации. Строительство : Сборник материалов V Национальной научно-практической конференции, Омск, 28 апреля – 29 2022 года. – Омск: Сибирский государственный автомобильно-дорожный университет (СибАДИ), 2022. – С. 588-594.
2. Тестирование программного обеспечения. Базовый курс / С.С. Куликов. — 3-е изд. — Минск:

Четыре четверти, 2020. — 312 с.

3. Бахтин, И. В. Шесть ошибок при проверке программного кода / И. В. Бахтин // Научный электронный журнал Меридиан. – 2020. – № 15(49). – С. 27-29.
4. Воротникова, Т. Ю. Надежный код: статический анализ программного кода как средство повышения надежности программного обеспечения информационных систем / Т. Ю. Воротникова // Информационные технологии в УИС. – 2020. – № 2. – С. 22-27.
5. Технический долг в жизненном цикле разработки ПО: запахи кода / В. В. Качанов, М. К. Ермаков, Г. А. Панкратенко [и др.] // Труды Института системного программирования РАН. – 2021. – Т. 33, № 6. – С. 95-110.
6. PMD. [Электронный ресурс] Режим доступа: https://pmd.github.io/ (дата обращения: 17.03.2024).

УДК 004.4

Д. Р. Белошицкий (4 курс бакалавриата),
А. П. Маслаков, ст. преподаватель

РАЗРАБОТКА РАЗДЕЛА «ОБСУЖДЕНИЯ» ДЛЯ ПРИЛОЖЕНИЯ ОДНОКЛАССНИКИ ПОД ОПЕРАЦИОННУЮ СИСТЕМУ IOS

Одноклассники [1] – одна из крупнейших социальных сетей в России и странах ближнего зарубежья. Значительное число пользователей площадки используют мобильные устройства на базе операционной системы iOS.

С развитием мобильных технологий и широким распространением смартфонов, социальные сети стали неотъемлемой частью повседневной жизни миллиардов людей по всему миру.

В этом контексте раздел комментариев играет ключевую роль в пользовательском взаимодействии, а именно, предоставляет уникальную площадку для обмена мнениями. Пользователи могут делиться впечатлениями и обсуждать актуальные события, что способствует формированию разностороннего общественного диалога. Раздел комментариев также является ценным инструментом для создателей контента. Он предоставляет им возможность получать мгновенную обратную связь от аудитории, а также взаимодействовать с ней напрямую. Это способствует улучшению контента и создает более тесные связи между создателями и их аудиторией.

В работе был реализован переработанный экран публикации, в котором появилась возможность варьировать отображаемый пользователю контент в зависимости от ответа с сервера. Например, одним пользователям показывать рекомендации и комментарии, другим только комментарии, а третьим дополнительные виджеты помимо основного контента.

На первом этапе разработки со стороны команды бэкенда был реализован метод supply [2], задачей которого было в зависимости от серверного контекста возвращать набор флагов на конечный клиент. По контракту, клиент в свою очередь, на основании этих флагов должен был принять решение о показе интерфейса в том или ином виде.

На втором этапе разработки был реализован универсальный контейнер публикации, в который планировалось добавлять элементы, тем самым меняя общий вид интерфейса. Он состоял из двух основных частей: секции публикации и секции дополнительного контента, где могли располагаться комментарии, блоки рекомендаций, рекламы и т.д. (см. рисунок 1)

На последующих этапах разработки на основании требований бизнеса добавлялись различные типы контента, которые контейнер может отобразить, например, закрепленный комментарий.

Программный код контейнера писался с использованием языков программирования Swift и Objective-C, стандартных фреймворков Foundation [3] и UIKit [4], архитектурного шаблона проектирования MVVM с использованием паттернов проектирования: делегат, фабрика, адаптер.

Итоговая версия контейнера стала доступна для пользователей вместе с запуском обновленного сервиса «Увлечения» [5] и в данный момент имеет поддержку нескольких дополнительных виджетов, что позволяет уменьшать TTM (time-to-market) при новых продуктовых запусках.

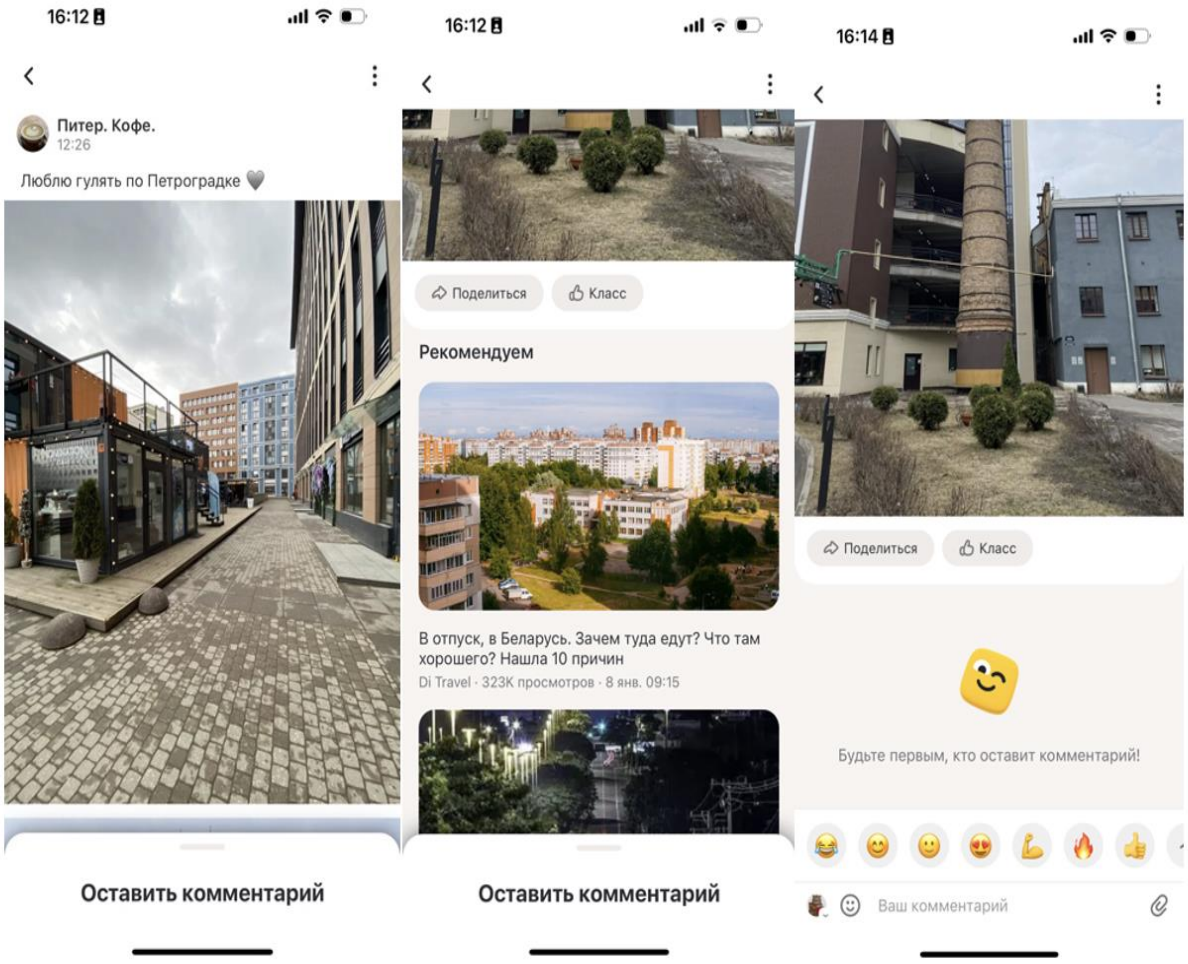


Рисунок 1 – Примеры разных отображений контейнера

ЛИТЕРАТУРА

1. История и этапы развития соцсети Одноклассники. [Электронный ресурс]. Режим доступа: https://insideok.ru/info/
2. API OK Documentation. [Электронный ресурс]. Режим доступа: https://apiok.ru
3. Foundation Documentation. [Электронный ресурс]. Режим доступа: https://developer.apple.com/documentation/foundation
4. UIKit Documentation. [Электронный ресурс]. Режим доступа: https://developer.apple.com/documentation/uikit
5. Одноклассники обновили «Увлечения». [Электронный ресурс]. Режим доступа: https://insideok.ru/blog/odnoklassniki-obnovili-uvlechenija-obshhenie-po-interesam-voprosy-professionalam-i-jekspertnye-materialy

УДК 32.973

С. С. Гаврилов (3 курс бакалавриата),
Е. А. Муравьев, к.т.н., доцент

ВЕРИФИКАЦИЯ UML СПЕЦИФИКАЦИЙ НА ЛОГИЧЕСКОМ ПРОЦЕССОРЕ RDF+SPARQL.

В докладе обсуждается применение процессора логического вывода RDF+SPARQL стека семантических программ онтологического моделирования для верификации UML диаграмм. Цель - верификация качественных функциональных требований предметной области (ПрО) к спецификациям программного обеспечения (ПО).

В работе использованы технологии: StarUML[1]; Библиотека API StarUML[2] - применялась для доступа из программ на C# к UML диаграммам и реализации компилятора диаграмм в формат RDF[4]; Билиотека dotNetRDF[3] - применялась для работы с UML диаграммами в формате RDF а также для реализации алгоритмов верификации с применением языка запросов SPARQL[5].

Верификация состоит в разборе структуры UML и распознавании семантических дефектов спецификации без моделирования работы специфицированного алгоритма на какой либое виртуальной машине. Тип верификации близок к статическому анализу кода.

На рисунке 1 приведена корректная спецификация алгоритма.

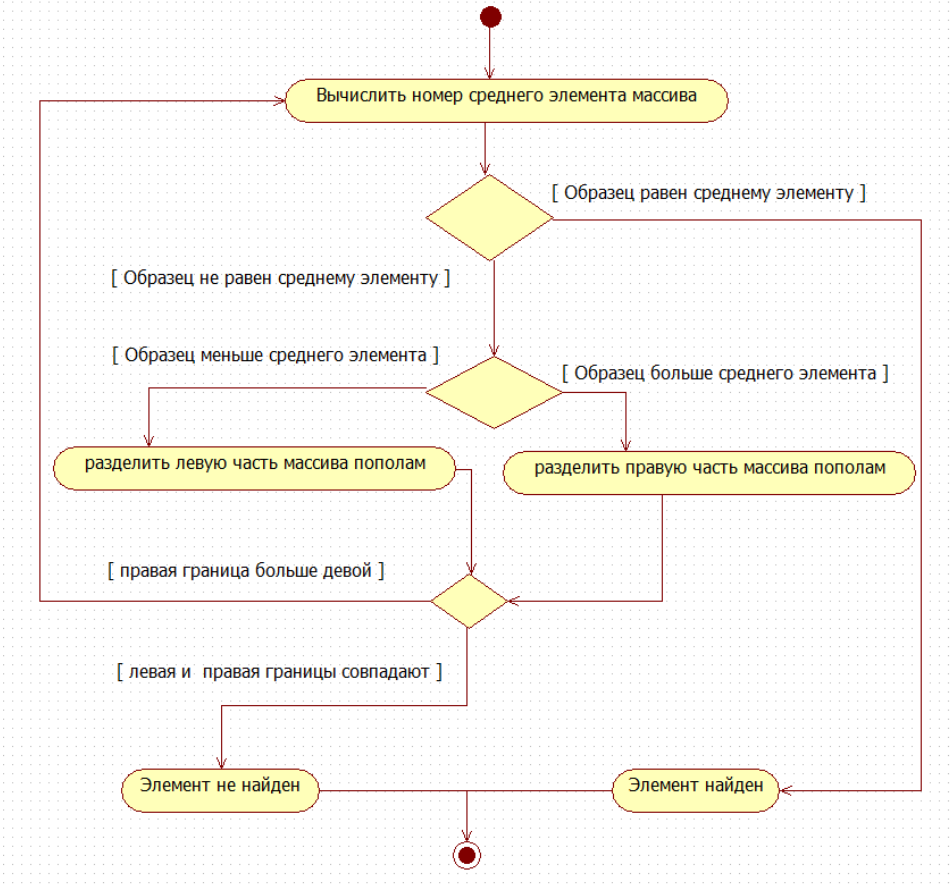


Рисунок 1 – корректный алгоритм

На рисунке 2 приведена спецификация с дефектами, внесенными в спецификацию рисунка 1. Дефекты получены удалением некоторых связей операторов диаграммы. В результате получены недостижимые и незавершаемые операторы. Дефекты были выявлены алгоритмом верификации.

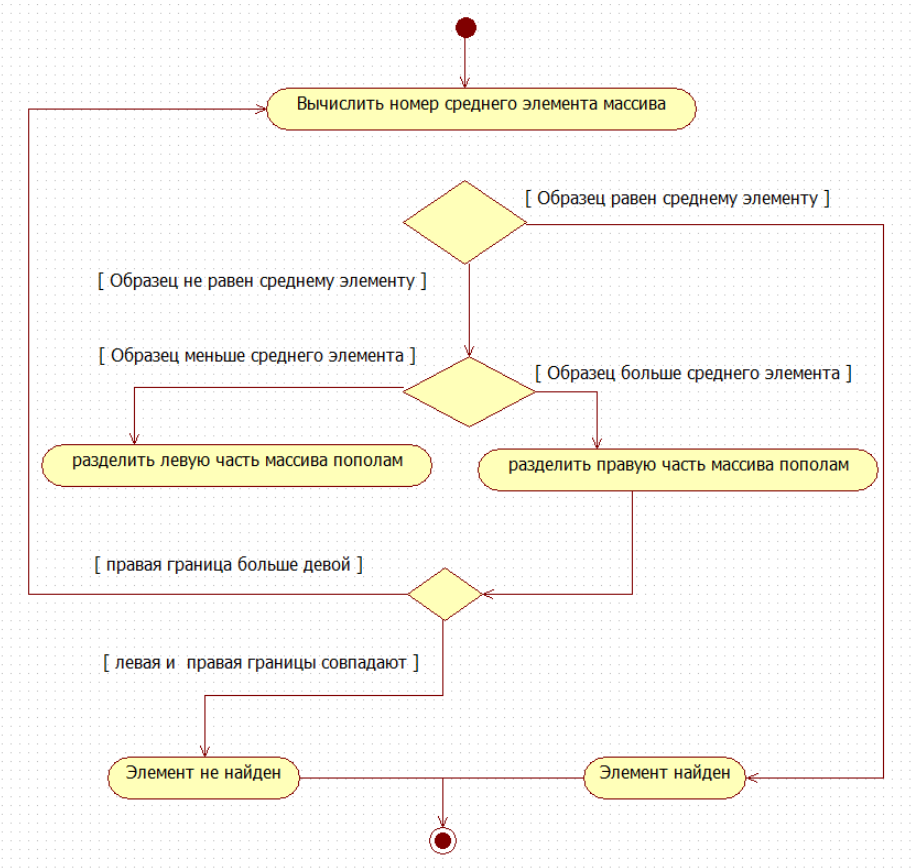


Рисунок 2 – некорректный алгоритм

Следует отметить, что возможно некоторые реализации UML выполняют проверки, которые мы привоим в качестве иллюстрации. Наша задача – показать возможность применение стандартных средств логического вывода, поддерживаемых сообществом W3C, для решения задач верификации ПО.

Примеры классов семантических ошибок, которые могут выявляться таким подходом:

1. Незавершаемость алгоритма, распознавание областей спецификации алгоритма, ответственных за такую ошибку выполнения.
2. Обнаружение недостижимых операторов алгоритма.
3. Нарушение частичного порядка следования во вовремени операций и групп операций алгоритма.
4. Отсутствие поддержки определенного функционала ПрО, например – контроля корректности данных с точки зрения требований ПрО.
5. Отсутствие обработки ошибок определенного предметного типа.
6. И подобные качественные свойства алгоритмов.

Общие свойства обрабатываемых семантических ошибок - качественные, не количественные свойства алгоритма, дефекты, определяемые суждениями о свойствах бесконечных множеств значений объектов ПрО, семантические дефекты с точки зрения функциональных требований ПрО.

Алгоритмы верификации реализуются на языке программирования C#. Обсуждаемые в докладе результаты применяются в учебном процессе при изучении дисциплины “Технология разработки программного обеспечения”.

ЛИТЕРАТУРА

1. StarUML. Руководство пользователя. https://staruml.sourceforge.net/docs/user-guide(ru)/user-guide.pdf1.
2. StarUML. Руководство разработчика. https://staruml.sourceforge.net/docs/developer-guide(ru)/developer-guide.pdf
3. dotNetRDF. https://dotnetrdf.org/

4. Учебное пособие по RDF http://www.w3.org/TR/rdf11-primer/
5. SPARQL 1.1 Query Language. https://www.w3.org/TR/sparql11-query/

УДК 004.43

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

СОЗДАНИЕ НЕРЕЛЯЦИОННОЙ РАСПРЕДЕЛЕННОЙ СУБД С ИСПОЛЬЗОВАНИЕМ JAVA 21 API

В современном мире, где данные становятся всё более ценным активом, базы данных выступают ключевыми инструментами для их хранения и анализа. Среди многих решений Apache Cassandra зарекомендовала себя как одна из лидеров среди распределённых NoSQL баз данных [3]. В этой статье мы фокусируемся на разработке собственной базы данных, используя последние достижения в Java.

В эпоху быстрого роста объёмов информации становится всё более важным разрабатывать технологии, которые позволяют эффективно управлять данными. Проект по созданию распределённой системы баз данных, основанный на использовании memory segments Java 21, является шагом вперёд в этом направлении [2]. Этот подход позволяет работать с объемами памяти вплоть до 2^{64} байт вне кучи, обещая значительные улучшения производительности за счёт оптимизации операций ввода-вывода и снижения нагрузки на сборщик мусора. Проект включает в себя этапы от анализа существующих решений до разработки архитектуры и комплексного тестирования, целью которых является оценка нового API для создания готовых к эксплуатации продуктов [4]. Внедрение принципов работы системы вроде Cassandra с новейшими технологическими решениями Java 21 позволяет разрабатывать масштабируемые и надёжные системы хранения, способные удовлетворить растущий спрос на производительные и одновременно отказоустойчивые базы данных [3].

В нашем проекте мы выбрали стек технологий, который включает Java 21, Memory Segments, HTTP [6], Consistent Hashing, ConcurrentSkipListMap[5], ChunkedTransferEncoding [8], Async-Profiler [7] и WRK 2 [9]. Этот выбор отражает нашу стремление к использованию современных и эффективных инструментов для создания надежной и производительной системы.

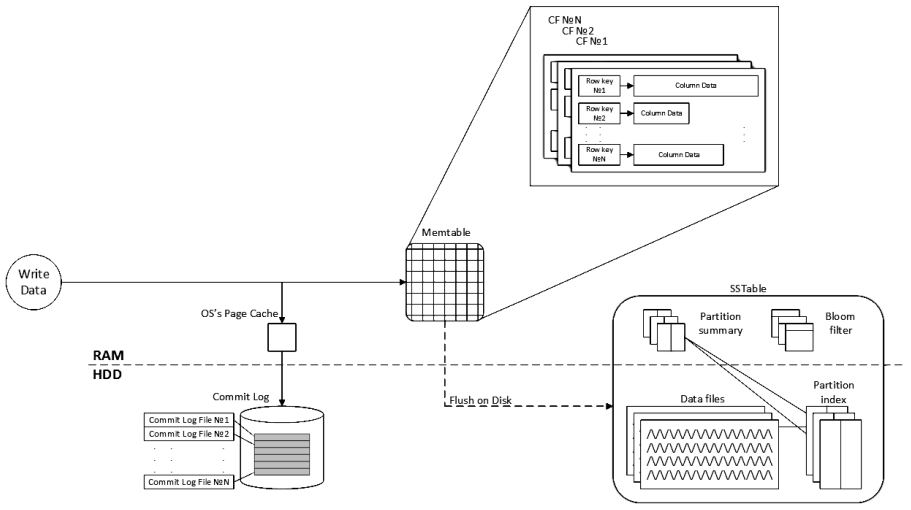


Рисунок 1 – Схема работы

Внутренняя организация хранения данных в одной ноде базы данных, вдохновленной архитектурой Cassandra, представляет собой сложную и эффективную систему, способную обеспечивать высокую производительность и отказоустойчивость (см. рисунок 1). В центре этой системы лежат SSTable (Sorted String Table) файлы, Memtable структуры и несколько

важных механизмов, таких как журнал записи (commit log), механизм сборки мусора (compaction), индексы и bloom filters, которые вместе обеспечивают эффективное управление данными [3].

SSTable файлы, неизменяемые и содержащие данные, отсортированные по ключу, являются основой системы хранения, что упрощает поиск и доступ к данным. Они организованы в блоки со сжатыми строками для экономии пространства, облегчая сбор мусора и управление версиями. Данные сначала хранятся в Memtable, skip list map для быстрой записи, а затем переносятся в SSTable на диске по достижении Memtable определённого размера, что обеспечивает эффективное управление хранением данных.

Для обеспечения устойчивости данных при сбоях каждая запись сначала фиксируется в журнале записи (commit log), который является постоянным и гарантирует, что данные не будут потеряны в случае аварии системы.

Со временем рост числа SSTable файлов может замедлять чтение данных, но механизм сборки мусора, объединяя файлы и удаляя устаревшие данные, ускоряет доступ. Каждый SSTable содержит индекс для упрощения поиска по ключу и включает bloom filter, что позволяет быстро проверять наличие ключа без полной загрузки файла, сокращая необходимость в операциях чтения с диска.

В совокупности эти механизмы формируют основу для эффективного и надежного хранения данных внутри одной ноды распределенной базы данных. Это позволяет не только эффективно управлять большими объемами данных, но и обеспечивает высокую доступность и отказоустойчивость хранилища.

Тестирование играет центральную роль в обеспечении качества нашего продукта. Для этого мы разработали 38 тестов для проверки сетевого взаимодействия между нодами и 98 тестов для тестирования взаимодействия внутри нод, обеспечивая общее покрытие 89% кода. Эти меры гарантируют высокую надежность и эффективность работы нашей системы.

Внедрение Foreign Memory API в Java 21 знаменует собой маленький прорыв в управлении памятью в java, предоставляя разработчикам инструменты для более эффективного выделения и использования памяти вне кучи, что открывает новые горизонты для производительности и безопасности. Это API позволяет выделять большие объемы памяти, при этом уменьшая накладные расходы, что ранее было сложно достижимо с использованием Byte Buffers, и обеспечивает более точное и низкоуровневое управление памятью, упрощая работу с нативным кодом. Однако, превью статус этого API, текущее состояние поддержки в экосистеме и потенциальная волатильность будущих обновлений Java вводят определенные риски и вызовы для разработчиков, а также вопросы относительно его готовности к использованию в производственных средах [2]. Тем не менее, проект, нацеленный на создание высокомасштабируемой и отказоустойчивой распределенной системы баз данных, достиг требуемых результатов, включая тщательный анализ существующих решений[1], четкое определение системных требований, разработку устойчивой архитектуры и воплощение продукта, который отвечает этим требованиям, а также проведение комплексного тестирования. В итоге, это обеспечило важное понимание применимости нового API Java для создания продуктов, готовых к непосредственной эксплуатации и внедрению в производственные процессы.

ЛИТЕРАТУРА

1. Lakshman, A., & Malik, P. (2010). Cassandra: A Decentralized Structured Storage System. ACM SIGOPS Operating Systems Review, 44(2), 35-40.
2. Tilmann, R. (2021). Java 21: The Comprehensive Guide. Manning Publications.
3. Hewitt, E. (2010). Cassandra: The Definitive Guide. O'Reilly Media, Inc.
4. Richards, M. (2015). Software Architecture Patterns. O'Reilly Media, Inc.
5. Sutter, H. (2020). Mastering ConcurrentSkipListMap for High-Performance Java Applications. Journal of Computer Science and Technology.
6. McGrath, R. E., & Bates, B. (2021). HTTP: The Definitive Guide. O'Reilly Media, Inc.

7. Gil Tene, M. Nitsan, B. (2019). Continuous Profiling in Production: What, Why and How. Communications of the ACM.
8. Kalid, A. (2021). BetterExplained Series: Chunked Transfer Encoding. BetterExplained Books.
9. Zoeller, M. (2022). WRK 2: Benchmarking Tool for Modern Web Services. ACM Transactions on Internet Technology.

УДК 004.453

Го Синь (2 курс магистратуры),
Г. Ф. Малыхина, д.т.н., профессор

СОСТЯЗАТЕЛЬНОЕ ОБУЧЕНИЕ НЕЙРОСЕТЕВЫХ МОДЕЛЕЙ С ИСПОЛЬЗОВАНИЕМ КООПЕРАТИВНЫХ И КОНКУРЕНТНЫХ СТРАТЕГИЙ В ТЕОРИИ ИГР ДЛЯ СЕГМЕНТАЦИИ МЕДИЦИНСКИХ ИЗОБРАЖЕНИЙ

Сегментация медицинских изображений играет важнейшую роль в области обработки медицинских изображений, разделяя сложные медицинские изображения на отдельные области или объекты. Такая сегментация помогает врачам точно определять очаги поражения, что обеспечивает точную диагностику и эффективное планирование лечения. Точность сегментации существенно влияет на принятие медицинских решений, что делает ее одной из наиболее важных областей исследований [1]. Использование моделей теории игр и нейросетевых методов может улучшить процесс сегментации и повысить точность анализа медицинских изображений.

Медицинские изображения, такие как рентгеновские снимки, позитронно-эмиссионная томография и гамма-томография, часто страдают от шума. Традиционные методы сегментации медицинских изображений сталкиваются с такими проблемами, как чрезмерная подгонка, нехватка данных и плохая обобщенность на новые изображения, что ограничивает эффективность моделей глубокого обучения [2]. Для решения этих проблем мы предлагаем новый подход, сочетающий теоретико-игровые модели с нейронными сетями, в частности генеративными адверсарными сетями (GAN) и сверточными сетями для сегментации биомедицинских изображений (U-Net). Этот подход направлен на повышение точности и согласованности сегментации путем оптимизации принятия решений и использования GAN для подавления шума, а затем U-Net для точной сегментации медицинских изображений [3-5]. Решая эти задачи, наш метод не только устраняет ограничения традиционных методов, но и повышает качество и точность результатов сегментации, обеспечивая надежную поддержку медицинской диагностики.

Автор предлагает инновационную методологию сегментации медицинских изображений, сочетающую модели теории игр и нейросетевые модели. Для этого использованы генеративные состязательные сети для уменьшения шума на медицинских изображениях, стремясь минимизировать помехи. Затем очищенные от шума изображения были введены в нейронную сеть U-Net для точной сегментации отдельных областей, содержащий объекты интереса. В процессе сегментации был реализован механизм принятия решений, основанный на теории игр, для оптимизации стратегии сегментации [6]. Этот механизм позволяет непрерывно вносить коррективы, вовлекая как алгоритм, так и участников оценки, что повышает точность и согласованность.

Алгоритм, использующий теорию игр, включает следующие пункты:

1. Инициализацию Q-таблиц. Сначала инициализируют Q-значение для каждой возможной пары состояние-действие.

2. Выбор действий. На каждом временном шаге алгоритм сегментации выбирают действие, основываясь на текущем состоянии.

3. Наблюдение результата выбранного действия.

4. Обновление Q-таблицы. Обновление Q-таблицы на основе наблюдаемых результатов и текущего Q-значения.

5. Повторение шагов 2–4 алгоритма. Алгоритм повторяет процесс выбора действия, выполнения действия, наблюдения за результатом и обновления Q-таблицы до тех пор, пока не будет достигнуто заданное количество раундов обучения или не будут выполнены другие условия остановки.

6. Применение оптимальной стратегии. В конце обучения Q-таблица может быть использована для реализации оптимальной стратегии сегментации.

Основной алгоритм сегментации медицинских изображений с использованием нейросетевого модуля реализует следующие действия:

1. Повышение качества изображений за счет подавления шума и помех с использованием GAN. Модель GAN состоит из генератора, который создает новые изображения, и дискриминатора, который оценивает подлинность сгенерированных изображений [7]. Такой подход в итоге повышает точность последующих задач сегментации изображений.

2. Сегментация изображений с использованием нейронной сети U-Net. В процессе сегментации на вход нейронной сети подают изображения с пониженным уровнем шума для точной сегментации. U-Net представляет собой особый тип сверточной нейронной сети, которая использует структуру кодировщика-декатировщика для достижения точной сегментации. Кодер извлекает особенности изображения, а декодер восстанавливает его и выполняет сегментацию. Такой подход позволяет точно сегментировать различные области или объекты на изображении на основе очищенного входного изображения.

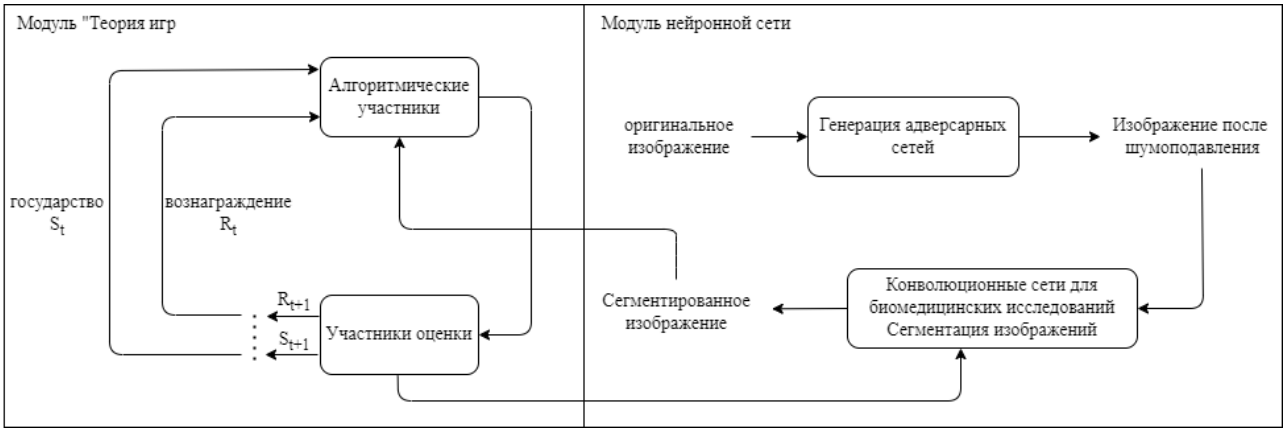


Рисунок 1 – Архитектура системы

Первая серия сравнительных экспериментов показала, что эффект от сегментации с использованием нейронной сети в сочетании с модулем теории игр значительно выше, чем эффект от использования только модуля нейронной сети.

Во второй серии сравнительных экспериментов была использовали нейронная сеть GAN в сочетании с нейронной сетью U-Net. Эксперименты показали, что лучшие результаты могут быть получены при использовании модуля теории игр и нейронной сети U-Net, при этом модуль теории игр остался неизменным.

ЛИТЕРАТУРА

1. Азад, Р., и др. (2022). Обзор сегментации медицинских изображений: Успех U-Net. Предварительный отчет arXiv arXiv:2211.14830.
2. Ху, М., Цзян, Ж., Маткович, Л., Лиу, Т., & Ян, С. (2023). Использование обучения с подкреплением в анализе медицинских изображений: концепции, применения, проблемы и будущие направления. Журнал прикладной клинической медицинской физики, 24, e13898.
3. Ван, Ю., Луо, С., Ма, Л., & Хан, М. (2023). RCA-GAN: Улучшенный алгоритм деноизинга изображений на основе генеративно-состязательных сетей. Электроника, 12(22), 4595.
4. Роннебергер, О., Фишер, П., & Брок, Т. (2015). U-Net: Сверточные сети для сегментации медицинских изображений. В MICCAI.
5. Лиу, Л., Чжэн, Ж., Кван, Ц., Ву, Ф.Х., Ван, Ю.П., & Ван, Ц. (2020). Обзор сетей в форме U в сегментации медицинских изображений. Неуроинформатика, 409, 244–258.

6. Чжэн Ц, Ян Х, Фанг В, Лиу Л, Гао Х, Ван Ц. Теория игр встречает глубокое обучение: критический обзор. Транзакции по искусственному интеллекту, 2021; 3: 123–135.
7. Гудфеллоу И, Пуже-Абади Ж, Мирза М, Су Б, Варде-Фарлей Д, Озаир С, и др. Генеративно-состязательные сети. В: Продвинутые нейронные информационные системы. 2014; 2672–2680.

УДК 004.65

Д. С. Дрелин (2 курс магистратуры),
С. А. Нестеров, к.т.н., доцент

МИГРАЦИЯ БАЗЫ ДАННЫХ С ORACLE DATABASE НА POSTGRES PRO STANDARD НА ПРИМЕРЕ ГОСУДАРСТВЕННОЙ ИНФОРМАЦИОННОЙ СИСТЕМЫ САНКТ-ПЕТЕРБУРГА «ТЕРРИТОРИАЛЬНАЯ ОТРАСЛЕВАЯ РЕГИОНАЛЬНАЯ ИНФОРМАЦИОННАЯ СИСТЕМА»

В настоящее время российские компании активно переходят на отечественное программное обеспечение (ПО). Причинами этого является принятие регуляторных актов, обязывающих организации переходить на отечественное ПО [1], а также сниженная защищенность ПО ввиду отсутствия обновлений.

Информационная система «Жилищный надзор Санкт-Петербурга» Государственной информационной системы Санкт-Петербурга «Территориальная отраслевая региональная информационная система» (далее АС ЖН) была разработана для автоматизации процессов Государственной жилищной инспекции Санкт-Петербурга (ГЖИ). Она включает в себя широкий спектр функций: осуществление регионального жилищного надзора, осуществление лицензирования деятельности по управлению многоквартирными домами, ведение делопроизводства и документооборота ГЖИ. Структура программного обеспечения АС ЖН реализована в рамках трехуровневой клиент-серверной архитектуры. База данных (БД) этой системы обеспечивает хранение и использование взаимосвязанных типов учетных объектов (сущностей) и справочников.

Целью представляемой работы является миграция БД АС ЖН с Oracle Database 11g Release 2 Enterprise Edition на Postgres PRO Standard 15. Размер БД составляет 315 Гб.

Для достижения этой цели необходимо решение следующих задач:

1. Анализ исходной среды: необходимо провести детальный анализ текущей базы данных, работающей под управлением Oracle Database, используемой в АС ЖН. Этот анализ должен включать: детальное описание структуры данных и существующих объектов в базе данных, их объем.
2. Анализ проблем и инструментов миграции: необходимо выявить возможные проблемы при миграции, связанные с несовпадением типов данных, функций, а также провести анализ существующих инструментов для миграции данных.
3. Планирование миграции: разработать поэтапный план миграции, включая этапы извлечения данных, устранения несоответствий, загрузки в новую базу данных.
4. Выполнение миграции и тестирование полученного результата.

Postgres Pro Standard — это объектно-реляционная система управления базами данных (ОРСУБД, ORDBMS), которая была разработана Postgres Professional в рамках проекта Postgres Pro на основе PostgreSQL [2]. Данная ОРСУБД находится в реестре отечественного ПО [3], что является важной характеристикой для государственных информационных систем.

Миграция данных является ETL (Extract, Transform, Load) процессом, подразумевающим извлечение, преобразование и загрузку данных [4]. ETL – это трехэтапный процесс, применяемый для извлечения данных из различных источников, очистки и загрузки в целевую базу данных.

Миграция данных из одной базы данных в другую представляет собой процесс, который состоит из поочередных этапов: анализ текущей базы данных, планирование миграции, подготовка данных, выполнение миграции, последующая оптимизация и тестирование. Это

требует тщательной оценки и адаптации схемы базы данных, данных, хранимых процедур и триггеров, а также проверки совместимости.

Миграция баз данных с Oracle Database на Postgres Pro Standard (PostgreSQL) осложняется тем, что эти ОРСУБД имеют различные архитектуры, функциональные возможности и синтаксис SQL [5]. Основные трудности заключаются в разных типах данных и отличиях синтаксиса.

Существуют готовые инструменты для миграции данных, которые позволяют существенно упростить процесс переноса данных. Для миграции данных был выбран инструмент Ora2Pg, который позволяет автоматизировать процесс переноса схемы базы данных, данных, хранимых процедур и функций [6]. Этот инструмент предоставляет гибкие настройки миграции, поддерживая большое количество типов данных и объектов базы данных Oracle, что существенно упрощает процесс миграции и снижает вероятность возникновения ошибок. Однако данный инструмент требует ручного вмешательства при преобразовании PL/SQL в PL/pgSQL.

Наиболее быстрый вариант миграции, осуществленный в ходе тестов, - копирование данных с помощью инструкции COPY. Примерное время выполнения заняло 8 часов. Этот способ имеет недостаток: тип данных Oracle BLOB можно конвертировать только в тип данных Postgres BYTEA. Поэтому следующим шагом после загрузки необходимо выполнить конвертацию и перенос таких данных в поля типа OID, время этой операции так же приблизительно 8 часов.

Порядок миграции:

1. Создание структуры БД с помощью скрипта.
2. Запуск миграции с помощью скрипта.
3. Запуск скрипта, вносящего необходимые изменения по окончании импорта данных, в том числе:
   - изменение типа данных bytea на oid;
   - создания индексов;
   - создание ограничений.
4. Выгрузка и загрузка последовательностей:
   - выгрузка текущих значений;
   - загрузка определений;
   - загрузка текущих значений.

На данном этапе БД перенесена на ОРСУБД Postgres Pro Standard и успешно развернута. Также стоит отметить увеличившееся быстродействие БД, представленное ниже в Таблица .

Таблица 1 – Сравнение времени выполнения запросов в разных СУБД

SQL-запрос	Время выполнения в Oracle Database	Время выполнения в Postgres Pro Standard
select * **from** ACCOUNT **join** CLS on CLS.CREATOR_ID = ACCOUNT.ID	0,103 с	0,021 с
select * **from** ATTRIBUTE **where** ID = 464	0.004 с	0.002 с

ЛИТЕРАТУРА

1. Министерство цифрового развития, связи и массовых коммуникаций РФ [Электронный ресурс] Режим доступа: https://digital.gov.ru/ru/documents/7342/
2. Документация к Postgres Pro Standard 15.5.1 [Электронный ресурс] Режим доступа: https://repo.postgrespro.ru/doc//pgpro/15.5.1/ru/postgres-A4.pdf
3. Реестр программного обеспечения [Электронный ресурс] Режим доступа: https://reestr.digital.gov.ru/reestr/301574/

4. Reis J., Housley M. Fundamentals of Data Engineering: Plan and Build Robust Data Systems // O'Reilly Media. – 2022. – 422pp.
5. Новиков Б. А. Основы технологий баз данных: учебное пособие // Б. А. Новиков, Е. А. Горшкова, Н. Г. Графеева; под ред. Е. В. Рогова. — 2-е изд. — М.: ДМК Пресс, 2020. — 582 с.
6. Документация ora2pg [Электронный ресурс] Режим доступа: https://ora2pg.darold.net/documentation.html

УДК 004.054

Д. С. Дюрдева (4 курс бакалавриата),
А. П. Маслаков, ст. преподаватель

РАЗРАБОТКА ПОДХОДА ДЛЯ РАБОТЫ С FEATURE TOGGLE В АВТОМАТИЗИРОВАННЫХ ТЕСТАХ ДЛЯ ANDROID ПРОЕКТА ОДНОКЛАССНИКИ

Feature toggle (или feature-switcher, или рубильник, или флаг) — это механизм, позволяющий управлять функциональностью системы без необходимости повторной компиляции кода и выпуска новой версии [1]. Реализуется с помощью добавления в код if-оператора, который дает возможность управлять поведением. Значение feature toggle, не меняющее текущее поведение системы, называется default или «по умолчанию». Сам рубильник может существовать в файле конфигурации или в базе данных. Частой практикой является использование отдельной системы – Portal Management System (PMS), для быстрого и легкого управления feature toggle. Использования флагов в разработке предоставляет ряд преимуществ:

1. Возможность включать и отключать функциональность, описанную флагом в любое время.

2. Возможность проводить A/B тестирование. Рубильник позволяет сделать изменение доступным только для определенных пользователей, провести эксперимент, оценить его влияние и сделать правильный выбор для бизнеса.

3. Возможность без выпуска новой версии вернуть приложение в стабильное состояние, если изменение не работает или работает не так, как ожидалось.

4. Уменьшение количества веток, их времени жизни, а также сложность их слияния.

5. Исключение влияния изменений одной команды на разработку и тестирование другой команды.

Постоянное проведение экспериментов не может гарантировать стабильность как тестового, так и production окружения. Эта проблема имеет большое влияние на автоматизированное тестирование приложения. Так как, написание тестовых сценариев происходит под определенную функциональность, при постоянном включении/отключении feature toggle тесты становятся нестабильными. Это приводит к тому, что тестирование средствами автоматизации становится не качественным, а также не может быть использовано в инструментах CI/CD.

В настоящее время в проекте Android-автотестов компании «Одноклассники» для достижения стабильности тестов в условиях постоянных экспериментов, было принято решение отключить синхронизацию с системой PMS и зафиксировать окружение для запуска автотестов. Это означает, что автотесты запускаются на значениях feature toggle «по умолчанию». Для того, чтобы использовать в тестах значение рубильника отличное от default, необходимо прописать его в специальный файл конфигурации – mock_pms_settings и исправить появившиеся падающие тесты. Однако это подход обладает существенными недостатками:

1. Из-за отсутствия «чистки кода» от рубильников автотесты запускаются на неактуальных default настройках. Доказательством тому является то, что при прогоне тестов с включением синхронизации с PMS падает около 500 тестов. Кроме того, желание

актуализировать тесты, привело к тому, что файл mock_pms_settings со временем стал увеличиваться и на данный момент содержит уже 1000 строк.

2. Неявное включение feature toggle. Файл mock_pms_settings применяет рубильник ко всему коду автотестов, поэтому при анализе теста, поиск существующих флагов, влияющих на его поведение, занимает очень продолжительное время.

3. Нет возможности проводить аудит рубильников. Для актуализации тестов необходимо вручную сравнить все существующие default настройки в Android приложении с настройками из PMS, а также учитывать значения из файла mock_pms_settings. Учитывая количество существующих feature toggle, и отсутствия специальных инструментов для аудита, такая процедура займет продолжительное время.

Таким образом целью работы является разработка новых инструментов для работы с feature toggle в проекте Android-автотестов компании «Одноклассники» с учетом текущих процессов. Для достижения цели в первую очередь были выделены основные требования к новому подходу:

1. Тесты должны оставаться стабильными. Эксперименты на портале не должны приводить к падениям автотестов, так как они используется в CI/CD для запрета merge в pull request.

2. Необходима возможность проверять функционал, закрытый флагами. При этом необходимо использовать feature toggle явно, чтобы обеспечить прозрачную работу и возможность быстро и однозначно идентифицировать падение теста, по причине несоответствия или конфликта флагов, а также уметь легко определить функциональность, которой управляет рубильник.

3. Иметь возможность писать автотесты для всех возможных значений рубильника. Такой инструмент может понадобиться при проведении долгих и серьёзных экспериментов.

4. Реализовать инструмент для проведения аудита автотестов, для выявления не актуальных значений рубильников.

На основе этих требований было решено отказаться от файла mock_pms_settings. Без явного указания нужного рубильника использовать значение «по умолчанию» — это позволит сохранить обратную совместимость, для большинства текущих тестов, а также позволит тестам оставаться стабильными во время проведения экспериментов на портале. Для включения feature toggle были написаны специальные аннотации, позволяющие включать настройку как из теста, так и из методов внутри Page Object [2]. Также была реализована возможность создавать костюмные аннотации MockScope, для указания набора рубильников в одном месте и переиспользовании скоупов в разных местах [3].

Для проведения аудита автотестов, был написан скрипт на языке Python позволяющий получить выгрузку всех актуальные feature toggle и их значения из PMS [4]. Далее была реализована конфигурация (так называемая job) в TeamCity позволяющая выполнять этот скрипт и сравнивать выгрузку с текущими default флагами и явно включенными флагами в тестах [5]. На основе сравнения составляется файл разницы. Это файл может быть использован для разных целей: заведение задач на разработку автотестов для функционала, доступного в рамках эксперимента чувствительному кол-во пользователей, заведение задач на удаление логики флагов, закрывающих функционал, который доступен на 100%, а также для выявления автотестов проверяющих неактуальное состояние системы. Job запускается раз в неделю, результаты отправляются дежурному автоматизатору тестирования для заведения нужных задач или актуализации тестов.

Таким образом, в ходе работы был реализован новый подход для работы с feature toggle в Android-автотестах проекта «Одноклассники».

ЛИТЕРАТУРА

1. Feature toggle и их применение. История одного проекта. [Электронный ресурс] URL: https://habr.com/ru/companies/simbirsoft/articles/756864/ (дата обращения:18.03.2024).
2. Паттерны проектирования в автоматизации тестирования. [Электронный ресурс] URL: https://habr.com/ru/companies/jugru/articles/338836/ (дата обращения:18.03.2024).

3. Creating a Custom Annotation in Java. [Электронный ресурс] URL: https://www.baeldung.com/java-custom-annotation (дата обращения:18.03.2024).
4. Разработка надежных Python-скриптов. [Электронный ресурс] URL: https://habr.com/ru/companies/ruvds/articles/462007/ (дата обращения:18.03.2024).
5. Мощная непрерывная интеграция для команд, ориентированных на DevOps. [Электронный ресурс] URl: https://www.jetbrains.com/teamcity/ (дата обращения:18.03.2024).

УДК 004.415

В. Ю. Зыбкин (2 курс магистратуры),
П. Д. Дробинцев, к.т.н., доцент

СОКРАЩЕНИЕ ВРЕМЕНИ ВЫХОДА НА РЫНОК С ПОМОЩЬЮ РАЗРАБОТКИ СЕРВИСА ДЛЯ АДМИНИСТРИРОВАНИЯ СИСТЕМЫ ЗАПУСКА ЭКСПЕРИМЕНТОВ

С каждым годом появляется все больше и больше сервисов в интернете, которые предоставляют те или иные услуги: доставка еды, доставка мебели, платформы для просмотра фильмов и сериалов, социальные сети, банки и еще много чего другого. При этом важно понимать, что растет не только количество этих сервисов, но и то количество людей, которое им пользуется, а значит увеличивается количество запросов на сервисы.

Большие нагрузки диктуют свои правила, с ними мы уже не можем использовать те подходы и алгоритмы, которые работали на низких и средних нагрузках [1], поэтому обычно во всех крупных компаниях используются следующие подходы:

1. К сервисам предъявляются более жесткие критерии по отказоустойчивости, uptime’у. Все сервисы начинают хостится в нескольких дата-центрах, падение одного из них – не должно приводить к недоступности какой-либо функциональности на портале, так как недоступность приводит к потере денег и потенциальной потере части аудитории, потому что она может утечь к конкуренту.

2. Появляется система конфигурации. Весь новый код заносится под feature toggle [2], при обновлении сервисов все новые toggle’ы выключены, и система после обновления на новую версию работает также, как и до обновления. Весь новый код включается в runtime через систему конфигурации.

3. Появляется система A/B экспериментов, которая позволяет измерить эффективность нового функционала путем сравнения его со старым.

Система конфигурации и система A/B экспериментов зачастую связаны, потому что аудитория, на которую включается новая функциональность, задается именно через систему конфигурации. Однако порог входа в систему конфигурации относительно высокий:

1. Для разработчика – система понятна и логична.

2. Для аналитика – система понятна и логична, однако количество изменяемых свойств в конфигурации в большинстве случаев ограничена. Также аналитики прибегают к помощи разработчиков в сложных свойствах или просят сделать понятный интерфейс для конкретного случая где-нибудь в сервисы для ад.

3. Для проектных и продуктовых менеджеров – система слишком сложная, количество настроек пугает. Для изменения настроек они используют разработчиков.

Таким образом, разработчик в такой ситуации тратит много времени на то, чтобы пойти поменять аудиторию в экспериментах, к которым он имеет отношения, однако такое простое действие как изменение аудитории можно снять с разработчика и переложить на плечи менеджера.

Целью данной работы является разработка новой системы, которая:

1. Понизит порог входа в конфигурацию экспериментов.

2. Сократит время выхода на рынок.

3. Визуально сможет отображать то, как компания продуктово видит свой сервис.

4. Сможет валидировать эксперименты и будет не позволять запускать два разных эксперимента, которые могут влиять друг на друга, на одних и тех же группах пользователей

5. Фильтровать эксперименты по: статусу, настройкам целевой аудитории, создателю, feature toggle’ам, участвующим в эксперименте, названию.

Для достижения цели необходимо решить следующие задачи:

1. Проанализировать существующие решения и подходы.

2. Определить то, как визуально будут отображаться эксперименты, чтобы было просто валидировать эксперименты, не влияющие на друг друга эксперименты, могли включаться на одну и ту же аудиторию, а влияющие друг на друга эксперименты не пересекались по аудитории.

3. Выбрать базу данных для хранения всей информации, которая будет создаваться и использоваться в сервис для администрирования экспериментов, с учетом критериев доступности и отказоустойчивости системы.

4. Разработать архитектуру нового сервиса.

5. Проработать синхранизацию данных сервиса для администрирования экспериментов с данными в системе конфигурации при запуске или откате эксперимента.

6. Разработать сам сервис.

ЛИТЕРАТУРА

1. Fabian Brosig, Nikolaus Huber, Samuel Kounev, Automated extraction of architecture-level performance models of distributed component-based systems. // Conference: 26th IEEE/ACM International Conference on Automated Software Engineering (ASE 2011), Lawrence, KS, USA, November 6-10, 2011
2. Louis-Philippe Querel, Peter C. Rigby, Bram Adams, Feature toggles: practitioner practices and a case study // MSR '16: Proceedings of the 13th International Conference on Mining Software Repositories. – 2016.

УДК 004.89:007.3

В. А. Ивлев, Г. В. Мироненков (3 курс аспирантуры),
И. В. Никифоров, доцент, к.т.н.,
С. М. Устинов, д.т.н., профессор

ИСПОЛЬЗОВАНИЕ МОДЕЛИ GPT-3 ДЛЯ ГЕНЕРАЦИИ ИНФОРМАЦИОННО-ТЕХНОЛОГИЧЕСКОЙ ИНФРАСТРУКТУРЫ ПРОЕКТА НА ОСНОВЕ НЕФОРМАЛИЗОВАННЫХ ТРЕБОВАНИЙ

Исследования в области автоматизации настройки информационно-технической инфраструктуры на основе Natural Language Processing (NLP) представляют собой актуальное направление [1, 2], оказывающее значительное влияние на современные процессы разработки программного обеспечения. Развитие инновационных методов в этой области не только способствует повышению эффективности разработки, но и увеличивает производительность команд разработки [3], что является критически важным в динамично развивающейся сфере информационных технологий и позволяет в конечном счете сократить Time To Market (TTM) программного продукта.

При этом использование импульсных нейронных сетей [4] или Generative Pre-trained Transformer [5] (далее GPT) в качестве NLP инструмента для автоматизации настройки информационно-технической инфраструктуры предоставляет уникальные возможности, т.к. GPT обладает способностью понимать и обрабатывать естественный язык, что позволяет генерировать набор команд для настройки ИТ-инфраструктуры на основе человекочитаемого описания в виде набора поставленных задач.

Опираясь на вышеперечисленную актуальность темы и уникальность подхода для генерации команд с использованием GPT, для дальнейшей работы, необходимо провести обзор существующих GPT моделей для выявления покрытия ряда критериев, необходимых для реализации NLP model service [3] системы для генерации команд для настройки информационно-технической инфраструктуры на основе неформальных требований в виде человекочитаемого набора задач.

Таблица 1 – Таблица анализа существующих GPT моделей

Тип модели	**Большой размер модели**	**Высокое качество генерации текста**	**Контроль за контекстом**	**Высокая скорость генерации текста**	**Понимание естественного языка**	**Гибкость использования**	**Доступность предобученных моделей**
GPT-1	+	+	-	+	-	-	-
GPT-2	+	+	-	+	+	+	+
GPT-3	+	+	+	+	+	+	+

Исходя из проведенного анализа моделей GPT можно сделать вывод, что GPT-3 [5] значительно улучшает контроль за контекстом по сравнению с более ранними версиями, что позволяет более точно управлять выходными результатами. Данный критерий является одним из важнейших при генерации команд для настройки информационно-технической инфраструктуры. При этом, по остальным критериям GPT-3 не уступает более ранним версиям, что делает ее идеальным кандидатом для использования при реализации модуля архитектуры NLP model service [3].

Далее, после выбора типа модели GPT, необходимо провести анализ описанной ранее архитектуры [1] и внести в схему более детальное описания модуля NLP с учетом использования внутри него GPT-3.

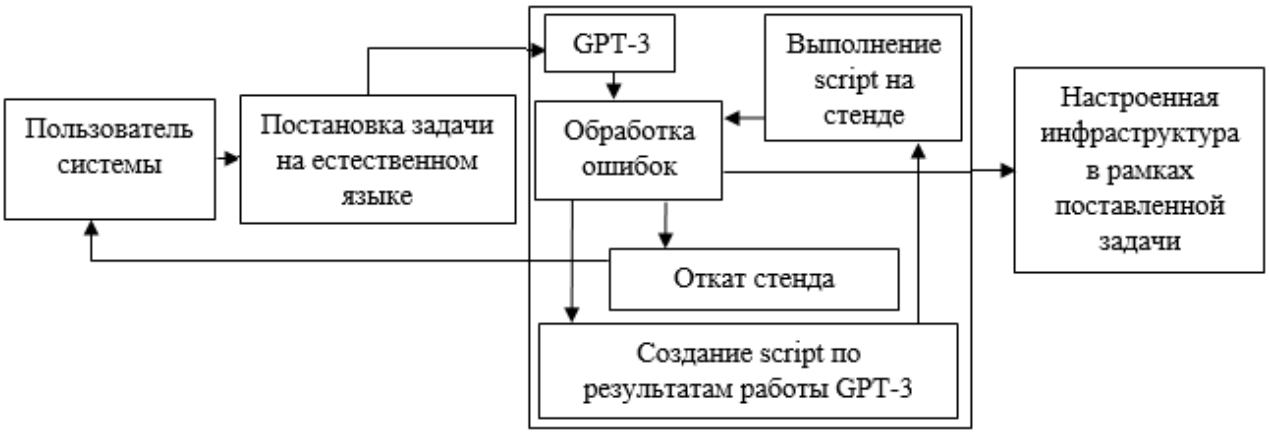


Рисунок 1 – Архитектурная схема с использованием GPT-3 и обработкой ошибок

Важной особенностью представленной выше архитектурной схемы является использование реализации GPT-3 в виде модуля NLP [6], что предоставляет возможность более точного и естественного взаимодействия с пользователем. Поскольку GPT-3 в виде своей реализации имеет мощную модель генерации текста, интеграция этого модуля позволит системе более эффективно понимать человекочитаемые описания задач и преобразовывать их в конкретные команды для настройки ИТ-инфраструктуры. Это может сделать процесс взаимодействия с системой более интуитивным и удобным для пользователей, так как им не придется следовать строгому синтаксису команд, а можно будет использовать естественный язык для передачи своих инструкций. Таким образом, архитектура системы после интеграции

GPT-3 станет более гибкой и адаптивной к потребностям пользователей, повышая уровень ее интеллектуальности и эффективности.

ЛИТЕРАТУРА

1. Генерация информационно-технологической инфраструктуры проекта на основе неформализованных требований / В. А. Ивлев, Г. В. Мироненков, И. В. Никифоров, А. Д. Ковалев // Современные технологии в теории и практике программирования : Сборник материалов научно-практической конференции студентов, аспирантов и молодых ученых, Санкт-Петербург, 26–27 апреля 2023 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2023. – С. 242-244. – EDN CENYIP.
2. Ивлев, В. А. Актуальность автоматизированной настройки инфраструктуры ит-проекта / В. А. Ивлев, И. В. Никифоров // Современные технологии в теории и практике программирования : Сборник материалов конференции, Санкт-Петербург, 26 апреля 2022 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2022. – С. 79-80. – EDN QQAVXT.
3. Ivlev, V. A. Automation method for configuring IT infrastructure for IT projects / V. A. Ivlev, I. V. Nikiforov, O. A. Yusupova // International Conference on Digital Transformation: Informatics, Economics,and Education (DTIEE2023) : Proceedings of SPIE - The International Society for Optical Engineering, Fergana, Uzbekistan, 20–22 марта 2023 года. – Washington: SPIE-SOC PHOTO-OPTICAL INSTRUMENTATION ENGINEERS, 2023. – P. 126370D. – DOI 10.1117/12.2680779. – EDN BYZHIK
4. Мироненков, Г. В. Решение задачи N тел с использованием импульсной нейронной сети / Г. В. Мироненков, В. А. Ивлев, Н. В. Воинов // Современные технологии в теории и практике программирования : Сборник материалов научно-практической конференции студентов, аспирантов и молодых ученых, Санкт-Петербург, 26–27 апреля 2023 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2023. – С. 209-210. – EDN AIURXH.
5. Pre-trained image processing transformer / H. Chen, T. Guo, C. Xu [et al.] // Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Virtual, Online, 19–25 июня 2021 года. – Virtual, Online, 2021. – P. 12294-12305. – DOI 10.1109/CVPR46437.2021.01212. – EDN FSDUXX.
6. Тюрюмина, В. А. Разработка нейросети GPT-3 на базе NLP / В. А. Тюрюмина // СОВРЕМЕННЫЕ ДОСТИЖЕНИЯ МОЛОДЕЖНОЙ науки : сборник статей Международного научно-исследовательского конкурса, Петрозаводск, 11 мая 2021 года. – Петрозаводск: Международный центр научного партнерства «Новая Наука» (ИП Ивановская Ирина Игоревна), 2021. – С. 14-18. – EDN OHQFWK.

УДК 004.4

А. А. Ивлева (2 курс магистратуры),
А. В. Алёшкин,
П. Д. Дробинцев, к.т.н., доцент

МЕТОДИКА РАСЧЕТА ДОСТУПНОСТИ И ПРОИЗВОДИТЕЛЬНОСТИ СЕРВИСА

Современному бизнесу требуется стабильная работа ИТ-инфраструктуры. Компании заинтересованы в том, чтобы в кратчайшие сроки узнавать о существующих проблемах и устранять их, поскольку это может привести к значительным финансовым потерям и нанести репутационный ущерб предприятию. Система расчета доступности и производительности позволяет отслеживать работу сервисов [1].

В совокупности, страницы состояния и расчеты доступности и производительности являются ключевыми элементами для обеспечения стабильности, надежности и

эффективности работы ИТ-инфраструктур. Они позволяют не только своевременно реагировать на текущие проблемы, но и прогнозировать потенциальные вызовы, оптимизируя тем самым работу и обеспечивая высокий уровень удовлетворенности пользователей [2].

Однако готовые решения расчета доступности и производительности не всегда покрывают потребности компании. Можно выделить несколько факторов, влияющих на это: некоторые страницы состояния могут предоставлять очень обобщенную информацию, не давая достаточно деталей о конкретных проблемах или причинах сбоев, иногда обновления страниц состояния могут происходить с задержкой, не отражая реального времени возникновения сбоев или восстановления сервисов, а в некоторых случаях автоматизированные системы, управляющие страницами состояния, могут не распознавать или неправильно классифицировать некоторые виды проблем, что приводит к неполному или некорректному отображению информации [3]. Вследствие перечисленных факторов требуется создание универсального инструмента, который может быть адаптирован под любые типы веб-сервисов, независимо от их масштаба и специфики работы. Результаты исследования могут быть использованы для проектирования новых систем, а также для оптимизации уже существующих, что в конечном итоге способствует повышению их конкурентоспособности и качества обслуживания пользователей [4].

Целью работы является разработка методики расчета доступности и производительности системы для снижения времени на диспетчеризацию и повышения продаваемости продукта за счет улучшения репутации компании.

На рисунке 1 представлен предлагаемый алгоритм методики создания страницы состояния, который позволяет рассчитывать доступность и производительность сервиса для любых потребностей бизнеса.

Рисунок 1 – Схема методики создания страницы состояний

После сбора данных, выполняется их анализ и рассчитываются необходимые метрики. Таким образом, для формирования единого отчета о производительности и доступности ИТ-инфраструктуры необходимо наличие метрик и параметров, по которым проводится выборка и расчет данных. Итоговый отчет оформляется в любом удобном для пользователя виде [5].

В будущем планируется реализовать страницу состояний для ООО «ГАЗПРОМНЕФТЬ ИТО» на основе предлагаемой методики. Апробировать систему на реальных данных. Оценить снижение времени на диспетчеризацию, а также повышения продаваемости продукта, и посредством этого реализовать цель работы.

ЛИТЕРАТУРА

1. Huber, S., Zoupanos, S., Uhrin, M., Talirz, L., Kahle, L., Häuselmann, R., ... & Pizzi, G. (2020). AiiDA 1.0, a scalable computational infrastructure for automated reproducible workflows and data provenance. Scientific Data, 7(1), 300. DOI: 10.1038/s41597-020-00638-4.
2. Urgaonkar, B., Shenoy, P., & Roscoe, T. (2005). Resource overbooking and application profiling in a shared Internet hosting platform. ACM SIGOPS Operating Systems Review, 39(1), 239-254.
3. Yaqoob, I., Hashem, I. A. T., Ahmed, A., Kazmi, S. A., & Hong, C. S. (2021). Reliability and high availability in cloud computing environments: a reference roadmap. Human-centric Computing and Information Sciences, 11(1), 14. DOI: 10.22967/HCIS.2021.11.014.
4. Liu, Z., Squillante, M. S., & Wolf, J. L. (2001). On maximizing service-level-agreement profits. In Proceedings of the 3rd ACM conference on Electronic Commerce (pp. 213-223).
5. Menasce, D. A., & Almeida, V. A. F. (2002). Capacity planning for web services: metrics, models, and methods. Prentice Hall PTR.

УДК 004.428.4

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

РАЗРАБОТКА БИБЛИОТЕКИ ДЛЯ УДАЛЕННОГО РАСЧЕТА КЭШЕЙ

Разработчики программного обеспечения весьма часто сталкиваются с проблемами, связанными с распространением набора данных, который не относится к категории "большие данные". На практике чаще всего используются два способа решения этой проблемы:

– Хранение данных в централизованном хранилище (например, в RDBMS, хранилище данных NoSQL [1] или кластер memcached [2]), чтобы потребители могли удаленно получить доступ к ним.

– Сериализация данных (например, в формате json, XML, protobuf и т.д.) и их распространение среди потребителей, которые имеют локальную копию.

Однако возникают различные проблемы при масштабировании каждого из этих подходов. Централизация данных может позволить вашему набору данных неограниченно расти, но это сопряжено с ограничениями по задержке и пропускной способности при взаимодействии с данными. Удаленное хранилище данных никогда не будет настолько надежным, как локальная копия данных, а георасределенность серверов станет причиной увеличенных задержек при обращении к хранилищу

С другой стороны, хранение копии данных в оперативной памяти может значительно сократить задержку и увеличить скорость доступа к данным. Но при использовании данного подхода могут возникнуть проблемы с масштабированием при увеличении размера:

– Потребление памяти приложением начинает расти

– Получение и десериализация набора данных дополнительно использует сеть и ресурсы сервера

– Замена текущего набора данных на новый требует значительных ресурсов процессора и памяти

Предложенное решение библиотеки для удаленного расчета кэшей объединяет лучшие аспекты двух подходов. Архитектура приложения представлена на рисунке 1.

Общий сценарий работы библиотеки:

1. На одном из серверов генерируется кэш.
2. Происходит сериализация protobuf [3] и сохранение в s3.
3. Сохраняется новая версия кэша в zookeeper [4].
4. Сервисы получают уведомление о новой версии кэша: перед скачиванием происходит ожидание в диапазоне от 0 до заданного времени, чтобы не появилась большая мгновенная нагрузка на s3 [5].
5. Сервисы скачивают кэш и десериализуют его.
6. Потребители переключаются на новую версию кэша.

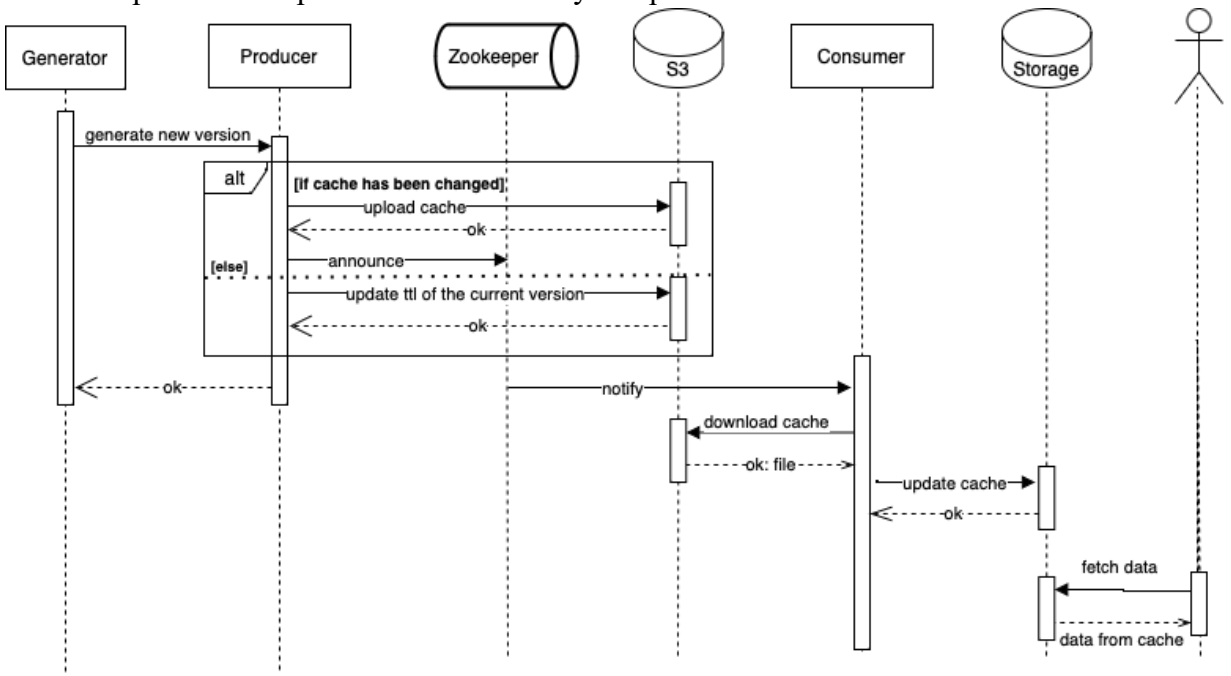


Рисунок 1 – Верхнеуровневая архитектура генерации кэша

Обработкой сгенерированного кэша занимается Producer, который вычисляет хэш кэша, сравнивает его с последней версией существующего: если они различны, то загружает в S3. Далее анонсирует новое событие, а Retriever скачивает сериализованные данные с обновленной версией. Затем Consumer применяет кэш.

Таким образом библиотека сочетает в себе высокую производительность и надежность. Ее использование позволяет значительно упростить процесс разработки и сделать систему более гибкой и масштабируемой, что является важным фактором в современном мире вычислений.

ЛИТЕРАТУРА

1. Нереляционные базы данных NoSQL [Электронный ресурс] Режим доступа: https://aws.amazon.com/ru/nosql/
2. Высокопроизводительное хранилище данных [Электронный ресурс] Режим доступа: https://aws.amazon.com/ru/memcached/
3. Protocol Buffer Documentation [Электронный ресурс] Режим доступа: https://protobuf.dev
4. Открытая программная служба для координации распределённых систем Apache ZooKeeper [Электронный ресурс] Режим доступа: https://zookeeper.apache.org
5. Amazon S3 Электронный ресурс [Электронный ресурс] Режим доступа: https://aws.amazon.com/s3/

УДК 004.415.25

П. Б. Коваленко (4 курс бакалавриата),
О. В. Прокофьев, ст. преподаватель

ПОВЫШЕНИЕ ПРОИЗВОДИТЕЛЬНОСТИ NOSQL СУБД С ИСПОЛЬЗОВАНИЕМ ONE-NIO RPC

Современные крупные информационные компании активно используют базы данных и большие объемы информации [1]. Существует огромное множество протоколов передачи данных, которые используются в СУБД. Среди них - протокол one-nio RPC, выделяющийся, с одной стороной, малоизученностью и отсутствием сообщества, а с другой - заявлениями его разработчиков о том, что он превосходит свои аналоги. Взаимодействие по такому протоколу подразумевает Remote Procedure Call (RPC), то есть удаленный вызов процедур таким образом, чтобы локально все выглядело как обычный вызов функции.

В связи с большим потреблением памяти базами данных принято их размещать на разных машинах, вводя разграничения по какому-то ключу в данных, чтобы координатор запросов понимал, где какие данные лежат и куда надо обращаться в случае необходимости. Зачастую запросы к такой СУБД отличаются большим размером данных, которыми обмениваются между собой узлы топологии как для формирования полного ответа, так и для принятия решения на основании того, какая машина какие данные успела записать. К таким системам очень часто выдвигаются требования по скорости выдачи ответа. Таким образом, *решением* было бы использование такого протокола передачи данных, который должен иметь минимум накладных расходов и поддерживать сжатие данных для ускорения их передачи. Графики нагрузки по каждому хосту позволят определить, насколько изменились основные характеристики запросов при их передаче.

Для разработки использовались некоторые технологии, в частности:

1. Языком программирования для разработки данного решения был выбран Java, так как весь проект NoSQL СУБД, для которой используется протокол one-nio RPC, написан именно на этом языке, который является стандартом в разработке современного качественного ПО.
2. Для реализации механизма удаленного вызова процедур (RPC) выбрана библиотека one-nio [2], которая представляет собой интерес с точки зрения улучшения текущей производительности NoSQL СУБД.
3. Выбор узла, ответственного за определенную порцию данных, происходит с помощью балансировщика нагрузки, реализующего механизм консистентного хэширования [3].
4. Для сбора и просмотра статистики по нагрузке используется wrk2 - высокопроизводительный расширяемый генератор нагрузки [4].
5. Тестирование алгоритма будет производиться с помощью фреймворка JUnit версии 5 [5].

Исходная NoSQL СУБД хранит в себе записи в качестве полей “ключ” - “значение”. Исходя из этого будут генерироваться запросы следующих типов:

– опрос одного и того же существующего ключа
– опрос существующих случайных ключей
– опрос несуществующих ключей
– смешанный опрос (существующих и несуществующих в равной доле)
– запись ключей фиксированного размера
– запись ключей случайного размера

График, показывающий нагрузку на систему в целом при выбранном типе запроса (опрос одного и того же существующего ключа), представлен на рисунке 1. На нем по оси X - перцентили, по оси Y - задержка ответа, то есть фактически показывает максимальное время ответа данного хоста, достигнутое для заданной доли запросов. Как видно по графику, начиная с определенной доли запросов задержка ответа стремительно растет.

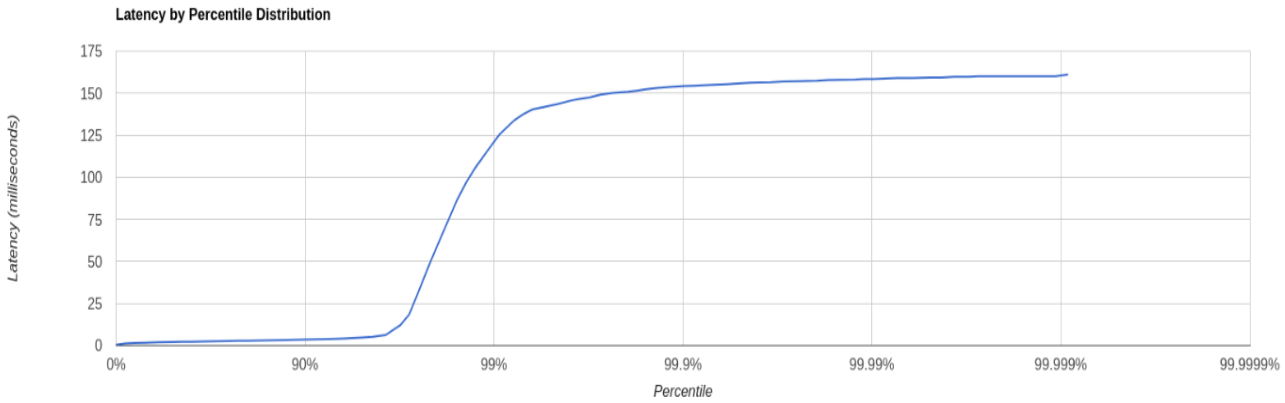


Рисунок 1 – График нагрузки

Критерием эффективности алгоритма будет являться уменьшение доли запросов (каждого типа) с задержкой более 50 миллисекунд.

Архитектура системы с взаимодействием её основных компонентов изображена на рисунке 2 (Node<N> - N-ый узел данных СУБД):

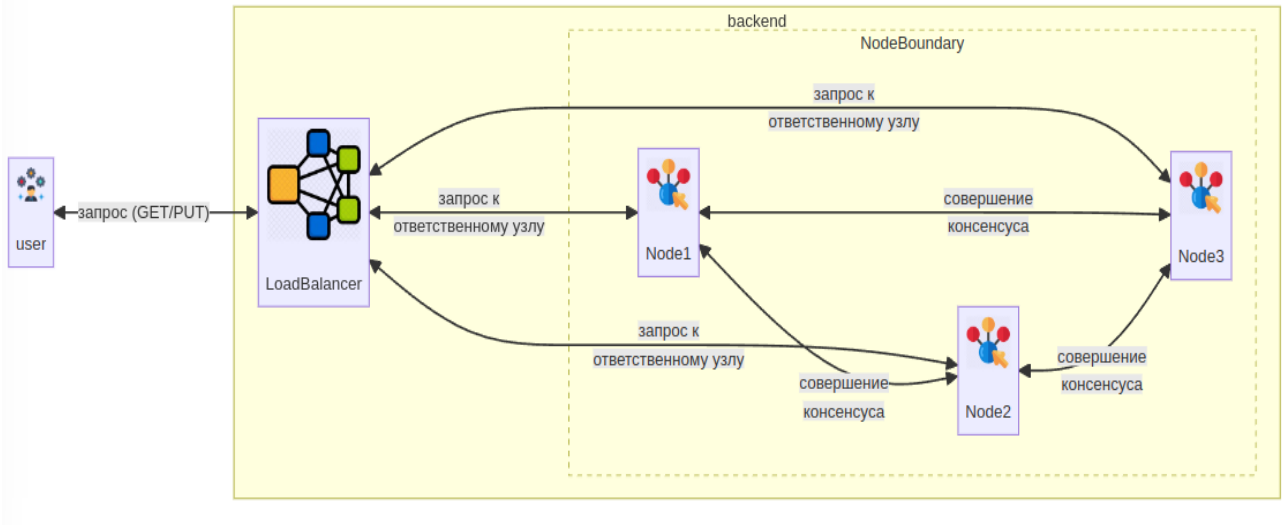


Рисунок 2 – Архитектура системы

ЛИТЕРАТУРА

1. Цыбулько В.В. ИНФОРМАЦИОННЫЕ СИСТЕМЫ НА ПРЕДПРИЯТИИ // Теория и практика современной науки. 2016. №3 (9). URL: https://cyberleninka.ru/article/n/informatsionnye-sistemy-na-predpriyatii-1 (дата обращения: 17.03.2024).
2. One-Nio Documentation [Электронный ресурс]. Режим доступа: https://github.com/odnoklassniki/one-nio/wiki (дата обращения: 17.03.2024).
3. Зернов А.С., Ожиганов А.А. Горизонтальное масштабирование базы данных с использованием консистентного хеширования // Приборостроение. 2017. №3. URL: https://cyberleninka.ru/article/n/gorizontalnoe-masshtabirovanie-bazy-dannyh-s-ispolzovaniem-konsistentnogo-heshirovaniya (дата обращения: 18.03.2024).
4. wrk2 Documentation [Электронный ресурс]. Режим доступа: https://github.com/giltene/wrk2 (дата обращения: 18.03.2024).
5. JUnit Documentation [Электронный ресурс]. Режим доступа: https://github.com/junit-team/junit5 (дата обращения: 18.03.2024).

УДК 004.453

А. С. Курдиков (4 курс бакалавриата),
Л. П. Котлярова, ст. преподаватель

РАЗВЁРТЫВАНИЕ SELF-HOSTED GITHUB ACTIONS RUNNER CONTROLLER С ПОМОЩЬЮ KUBERNETES

Разработка современного программного обеспечения (ПО) требует эффективное управление и автоматизацию процессов разработки. Методология непрерывной интеграции и доставки – CI/CD (Continuous Integration/Continuous Delivery) [1, 2], предоставляющая набор практик и инструментов, направлена на улучшение качества кода, сокращение времени разработки и ускорение поставки ПО конечному пользователю.

CI (непрерывная интеграция) автоматизирует компиляцию, сборку, тестирование и интеграцию кода в основную ветвь, освобождая разработчиков от этих рутинных задач, тем самым сокращая временные затраты на разработку ПО [3, 4].

CD (непрерывная доставка) автоматизирует развертывание ПО после успешного завершения цикла интеграции, обеспечивая непрерывную автоматизированную поставку готового ПО в выбранное окружение проекта, сокращая время развертывания [3, 4].

Веб-сервис GitHub имеет систему автоматизации – GitHub Actions [6]. Этот инструмент позволяет создавать индивидуальные рабочие процессы непрерывной интеграции и доставки, а также предоставляет широкий выбор готовых сценариев от разработчиков GitHub и сообщества, упрощая создание необходимого CI/CD процесса [6].

Политика GitHub позволяет бесплатно использовать CI/CD для open source (открытых) проектов, однако проекты с закрытым кодом должны либо использовать бесплатный тариф с большими ограничениями по времени и памяти, либо выбрать платные тарифы [6]. Это может вызвать проблемы, такие как несоответствие тарифов потребностям проекта, высокие затраты на платные тарифы и ограничения по безопасности проекта, требующие использование self-hosted решения (размещения стороннего ПО на собственных серверах).

Цель данной работы – развёртывание self-hosted GitHub Actions Runner Controller (ARC) [7] при помощи Kubernetes (k8s) для запуска процессов CI/CD в собственном рабочем окружении. Данное решение позволит сэкономить на использовании платных подписок GitHub Actions и выполнить требования по безопасности для проектов с закрытым кодом.

Предлагаемое решение требует навыков работы с контейнерами и k8s, однако существует более простое, но менее гибкое решение. Ниже приведена таблица 1, в которой сравниваются эти решения. В ходе сравнения было установлено, что стандартное решение на основе self-hosted actions runner менее предпочтительно – менее гибкое и не использует технологии виртуализации, что накладывает большие ограничения на автоматическое масштабирование и управление ресурсами, в то время как ARC позволяет гибко настраивать и масштабировать self-hosted actions runners, равномерно распределяя нагрузку между каждым запущенным процессом с помощью Kubernetes.

Таблица 1 – Сравнение методов развёртывания self-hosted CI/CD runners

Название решения	Контейнеризация	Масштабирование	Управление ресурсами	Сложность установки
ARC (Kubernetes)	Да	Автоматическое	Да	Сложно
Self-hosted actions runner	Нет	Нет	Нет	Просто

Из анализа следует необходимость создания одноузлового кластера Kubernetes с использованием ARC, удовлетворяющий следующим основным требованиям:

– должен быть развёрнут менеджер сертификатов для шифрования сообщений между GitHub и компьютером (сервером);

– должен быть развёрнут экземпляр GitHub ARC;

– должны быть созданы манифесты runner deployment и horizontal runner autoscaler.

Для создания кластера с одной нодой будет использоваться Minikube [5]. В основе ноды лежит экземпляр ARC, который будет управлять жизненным циклом runner (исполнителем процессов CI/CD). Манифест runner deployment управляет количеством реплик runner, а манифест horizontal runner autoscaler автоматически изменяет количество запущенных реплик в соответствии с указанной метрикой. В данном случае будет использоваться метрика GitHub - total number of queued and in progress workflows run (общее число планируемых и исполняемых запусков рабочих процессов).

Итоговая архитектура полученной ноды minikube представлена ниже на рисунке 1:

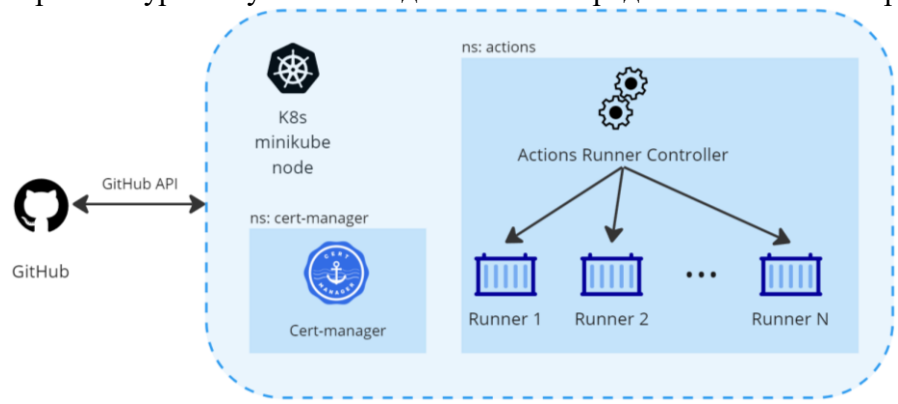


Рисунок 1 – Архитектура ноды minikube

Таким образом, были рассмотрены существующие способы развёртывания GitHub self-hosted actions runners, предложена архитектура, технологический стек и требования реализации. В ходе работы был создан одноузловой кластер на основе Kubernetes, в котором был развёрнут экземпляр ARC и менеджера сертификатов. Также для данной ноды было создано два манифеста, которые позволяют автоматически управлять жизненным циклом runner и увеличивать их количество в зависимости от количества запланированных запусков рабочих процессов. Далее планируется тщательное тестирование и демонстрация работы полученного решения.

ЛИТЕРАТУРА

1. Луценко Д. Ю. Сборка (CI/CD) проектов, не использующих JVM с помощью Gradle / Kotlin // Информационные технологии в управлении и экономике / Санкт-Петербургский политехнический университет Петра Великого, 2021. – С. 61-67.
2. Мадаев С. М., Алиевич А. Н. Современные тренды разработки программного обеспечения: Agile, DevOps, CI/CD // Тенденции развития науки и образования / Чеченский государственный университет им. А. А. Кадырова, 2023. – С. 58-60.
3. Никифоров И. В., Кольцов А. А. Автоматизация конфигурации системы непрерывной поставки и интеграции по - Jenkins // НЕДЕЛЯ НАУКИ СПбПУ: материалы научной конференции с международным участием, Санкт-Петербург, 19–23 ноября 2019 года / Санкт-Петербургский политехнический университет Петра Великого. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2019. – С. 56-59.
4. M. Shahin, M. Ali Babar and L. Zhu, "Continuous Integration, Delivery and Deployment: A Systematic Review on Approaches, Tools, Challenges and Practices," in *IEEE Access*, vol. 5, pp. 3909-3943, 2017, doi: 10.1109/ACCESS.2017.2685629.
5. Установка Minikube [Электронный ресурс] / Kubernetes. Режим доступа: https://kubernetes.io/ru/docs/tasks/tools/install-minikube/. (дата обращения: 12.03.2024)
6. GitHub Actions documentation [Электронный ресурс] / GitHub Docs. Режим доступа: https://docs.github.com/en/actions/. (дата обращения: 12.03.2024)
7. Quickstart for Actions Runner Controller [Электронный ресурс] / GitHub Docs. Режим доступа: https://docs.github.com/en/actions/hosting-your-own-runners/managing-self-hosted-runners-with-actions-runner-controller/quickstart-for-actions-runner-controller/. (дата обращения: 12.03.2024)

УДК 004.58

Е. С. Максина (1 курс магистратуры),
И. В. Никифоров, к.т.н., доцент

РЕАЛИЗАЦИЯ ПРОГРАММНОГО СРЕДСТВА ДЛЯ ОПТИМИЗАЦИИ ПРОЦЕССА УПРАВЛЕНИЯ ИЗМЕНЕНИЯМИ

Изменение – это добавление, изменение или удаление любых элементов, которое может прямо или косвенно повлиять на услуги. Управление изменениями – это практика в сфере ИТ, позволяющая свести к минимуму нарушения в предоставлении ИТ-услуг при внесении изменений в критически важные системы и сервисы [1]. Процесс управления изменениями важен для любой ИТ-компании. Внедрение изменений в информационные технологии может повлиять на работоспособность систем, безопасность данных, производительность сотрудников и качество обслуживания клиентов. Правильно организованный процесс управления изменениями помогает минимизировать риски и обеспечивает эффективное внедрение изменений в ИТ-среде компании [2]. Схема рабочего процесса управления изменениями приведена на рисунке 1.

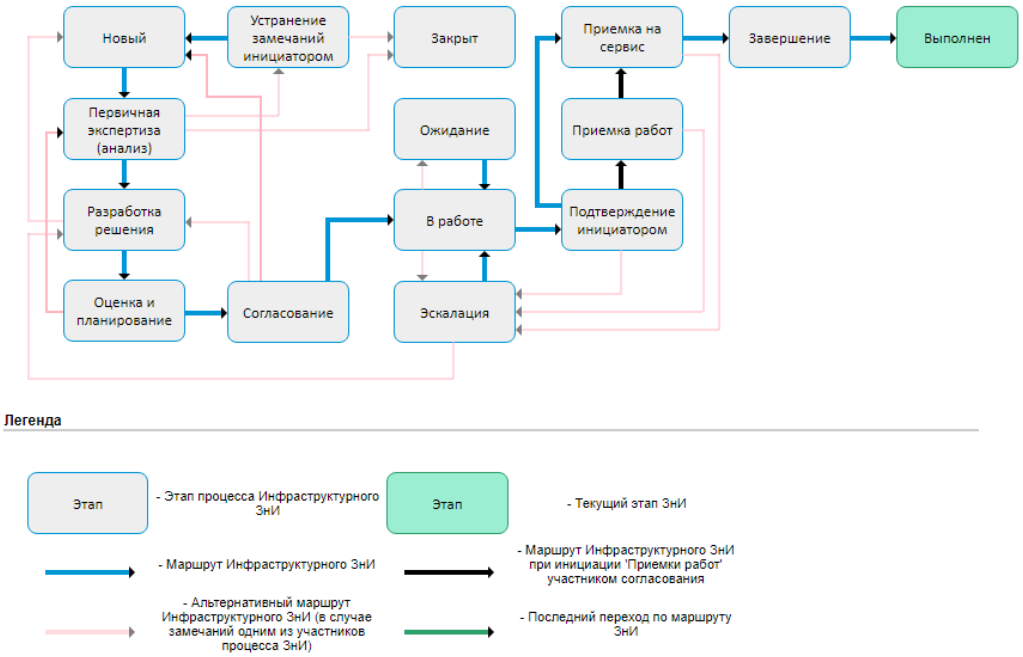


Рисунок 1 – Схема рабочего процесса управления изменениями в ИТ-инфраструктуре

Существует проблема долгого перехода запроса на изменение (ЗнИ) в состояние «В работе». Это связано с тем, что заявка может проходить многократно один и тот же цикл. Запрос на изменение проходит этапы оценки и согласования. Если в аналитической записке (АЗ) есть ошибки, которые нужно исправить, то создается наряд на корректировку АЗ. Инициатор АЗ формирует новую версию АЗ в соответствии с замечаниями и закрывает наряд. После чего цикл согласования/оценки повторяется. Таких циклов может быть большое количество, и они требуют времени, так как зависят от разных людей и требуется время на коммуникацию.

Цель данной работы – оптимизация процесса управления изменениями. Для достижения этой цели ставится задача сократить время, которое требуется на согласование заявки. Это реализуется с помощью программного инструмента, который позволит составителю аналитической записки проверять её автоматически, до отправки на оценку и согласование. Программное средство выявит ошибки, и инициатор сможет исправить их в тот же момент. Это позволит сократить количество циклов и, соответственно, время, необходимое для перехода ЗнИ в состояние «В работе». Большое количество ошибок связаны с опечатками,

невнимательным и неточным заполнением, невыполнением требований заполнения, которые прописаны в шаблонах. Нахождение таких ошибок можно автоматизировать.

В качестве вводных данных для исследования и разработки были взяты данные по запросам на изменение в компании за предыдущие года. Туда входят отклоненные и принятые аналитические записки, а также замечания, где указана ошибка в аналитической записке, ставшая причиной отклонения.

Проверка правильности заполнения аналитических записок является трудной задачей, так как работа ведется неформализованными данными. Для реализации используются регулярные выражения [3] совместно с методами машинного обучения, в частности метод обработки естественного языка Doc2Vec [4-6], который позволяет представлять документы в виде векторов. Схема работы разрабатываемого инструмента представлена на рисунке 2.

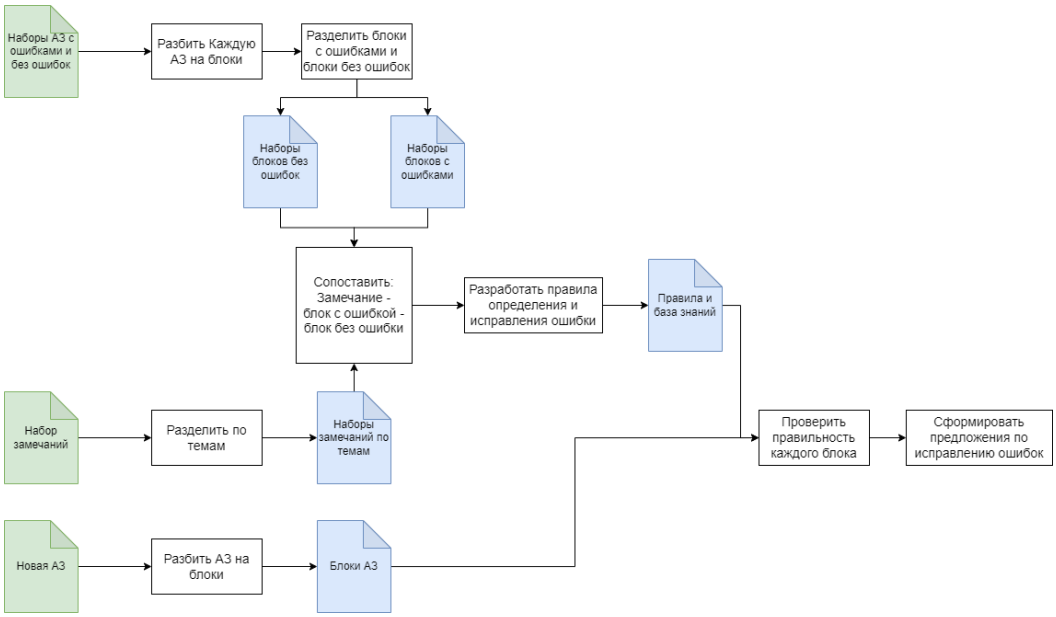


Рисунок 2 – Схема работы программного средства

На данный момент система находится на этапе разработки. Реализована часть, связанная с обработкой набора замечаний и предварительной обработкой новой аналитической записки. Для классификации замечаний использовалась модель Doc2Vec и расчет косинусного расстояния [4-6] для определения принадлежности классу.

Работа выполняется при поддержке компании Газпромнефть ИТО.

ЛИТЕРАТУРА

1. Geada N. Change Management Science Innovation Methodologies //Encyclopedia of Data Science and Machine Learning. – IGI Global, 2023. – С. 1614-1626.
2. Астафьева О. Е., Гончаров И. Л., Моисеенко Н. А. Анализ опыта управления изменениями в организациях //Управление. – 2020. – Т. 8. – №. 3. – С. 24-32.
3. Kaur G. Usage of regular expressions in NLP //International Journal of Research in Engineering and Technology IJERT. – 2014. – Т. 3. – №. 01. – С. 7.
4. Hanifi M. et al. Problem formulation in inventive design using Doc2vec and Cosine Similarity as Artificial Intelligence methods and Scientific Papers //Engineering Applications of Artificial Intelligence. – 2022. – Т. 109. – С. 104661.
5. 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. – EDN CFRNSJ.
6. Ковалев, А. Д. Автоматизированный подход к семантическому поиску по программной документации на основе алгоритма Doc2Vec / А. Д. Ковалев, И. В. Никифоров, П. Д. Дробинцев //

УДК 004.415.2

И. И. Малинин (2 курс магистратуры),
С. А. Молодяков, д.т.н., профессор

ФРЕЙМВОРК SERVICE WEAVER ДЛЯ ЯЗЫКА GO

Разработка приложений с микросервисной архитектурой представляет собой задачу с высокой степенью сложности, связанной с необходимостью эффективного управления распределенными системами, обеспечением их масштабируемости, безопасности и производительности. Существующие подходы часто требуют значительных усилий для настройки и интеграции различных компонентов инфраструктуры. Цель работы состоит в анализе и демонстрации возможностей фреймворка Service Weaver для языка программирования Go, который предлагает элегантное решение вышеперечисленных проблем за счет автоматизации части процессов разработки, развертывания и управления микросервисами.

Service Weaver — это новый фреймворк для разработки, развертывания и управления распределенными приложениями на Go представленный компанией Google 1 марта 2023 года. При разработке микросервисных приложения компания выявила ключевые трудности, которые попыталась решить: медленная скорость разработки из-за необходимости поддержки множества микросервисных исполняемых файлов с уникальными конфигурациями и сетевыми настройками, ограничения на изменения в исполняемых файлах, усложняющие внедрение нововведений и обновление форматов данных. Также, строгая фиксация на определенном наборе микросервисов ограничивала эволюцию API, делая развитие и внесение изменений в существующие интерфейсы чрезмерно трудоемким. Для решения этой задачи и был разработан данный фреймворк.

Работая с Service Weaver, разработчик создает приложение практически также как обычное монолитное приложение на Go. Приложение с помощью стандартных интерфейсов языка и специальных аннотаций фреймворка разбивается на модули с фокусом на бизнес логику, компоненты которых могут быть вызваны в коде без использования сетевых запросов, при развертывании в облако это берет на себя фреймворк, оборачивая вызовы в передачу сообщений с использованием современного формата Protobuf по протоколу RPC (Remote procedure call). Затем, при развертывании в облаке, фреймворк трансформирует код приложения в набор связанных микросервисов, упаковывает их в контейнеры и интегрирует их с облачным провайдером, обеспечивая мониторинг, трассировку и логирование.

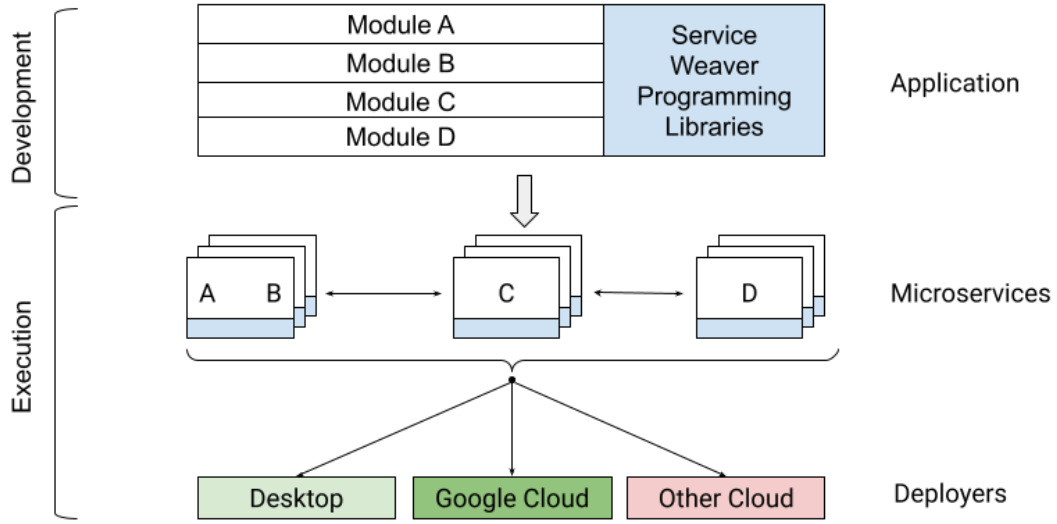


Рисунок 1 – схема работы Service Weaver

Применение фреймворка Service Weaver при переработке приложения "Online Boutique" на Google Cloud Platform, состоящего из 11 микросервисов, демонстрирует заметное

повышение производительности. В условиях нагрузки в 10000 запросов в секунду, использование Service Weaver обеспечило девятикратное снижение нагрузки на процессор и десятикратное уменьшение задержки (99-й процентиль). Эти результаты, в сочетании с другими преимуществами фреймворка, значительно увеличивают экономическую эффективность проектов, позволяя сократить затраты на разработку и поддержание облачной инфраструктуры.

Несмотря на ощутимые преимущества, у фреймворка Service Weaver есть и определенные ограничения. В частности, Google описывает архитектуру, создаваемых с его помощью приложений, как "модульный монолит" (Modular monolith), что может лишать их некоторых преимуществ традиционной микросервисной архитектуры, таких как гибкость в использовании различных языков программирования и технологий, а также независимость масштабирования и развертывания отдельных компонентов. Кроме того, относительная новизна фреймворка и ограниченный опыт его применения в крупных проектах пока не позволяют полностью оценить его надежность.

Фреймворк Service Weaver для Go предлагает новый подход к разработке приложений, существенно упрощая и ускоряя процесс за счёт автоматизации развертывания и управления. Это достигается благодаря способности фреймворка трансформировать традиционные приложения Go в комплекс микросервисов, интегрируя их с облачной инфраструктурой и предоставляя необходимые инструменты для мониторинга, трассировки и логирования. Применение Service Weaver демонстрирует значительное улучшение производительности и экономической эффективности, позволяя разработчикам сосредоточиться на бизнес-логике, минимизируя затраты на поддержку и разработку.

ЛИТЕРАТУРА

1. Go Programming Language Docs. [Электронный ресурс] URL: https://go.dev/doc/ (Дата обращения: 10.03.2024)
2. Service Weaver. [Электронный ресурс] URL: https://serviceweaver.dev/ (Дата обращения: 10.03.2024)
3. Google Open Source Blog: Introducing Service Weaver: A Framework for Writing Distributed Applications. [Электронный ресурс] URL: https://opensource.googleblog.com/2023/03/introducing-service-weaver-framework-for-writing-distributed-applications.html (Дата обращения: 10.03.2024)
4. Github: microservices-demo. [Электронный ресурс] URL: https://github.com/GoogleCloudPlatform/microservices-demo (Дата обращения: 10.03.2024)
5. Github: weaver. [Электронный ресурс] URL: https://github.com/ServiceWeaver/weaver (Дата обращения: 10.03.2024)

УДК 004.415.2

Д. О. Козлов (2 курс магистратуры),
И. В. Никифоров, к.т.н., доцент,
О. А. Юсупова, ассистент

АВТОМАТИЗАЦИЯ ПОСТРОЕНИЯ СИСТЕМЫ МНОГОАГЕНТНОГО МОДЕЛИРОВАНИЯ ДЛЯ КОНТЕЙНЕРИЗИРОВАННОЙ СРЕДЫ

Контейнеризация повсеместно используется для передачи проектов заказчикам и обеспечения масштабируемости. Процесс сборки образов после внесения изменений в кодовую базу является трудоемким и рутинным. В связи с этим задача автоматизации процесса контейнеризации является актуальной, так как позволит снять часть задач с программиста и снизить влияние человеческого фактора.

Целью работы является сокращение времени и трудоемкости работы программиста, затрачиваемые на создание, загрузку на удаленный репозиторий образов компонентов системы многоагентного моделирования для контейнеризованной среды [1].

Наиболее популярными технологиями автоматизации процесса контейнеризации являются Docker, Podman, Kubernetes [2]. Они обладают широким набором функций, позволяющих упростить процесс работы с контейнерами, такими как, автоматизация сборки, работа с удаленными репозиториями хранения образов, балансировка нагрузки [3], автоматическое масштабирование. Docker - это платформа, предназначенная для того, чтобы помогать разработчикам создавать контейнерные приложения, предоставлять к ним общий доступ и запускать их. Podman является альтернативой Docker, главными преимуществами которого являются поддержка различных форматов образов, работа без постоянного фонового процесса и возможность работы без root – доступа. Kubernetes является технологией для автоматизации развертывания, масштабирования и управления контейнерными приложениями, состоящими из нескольких взаимосвязанных контейнеров. Kubernetes позволяет автоматизировать процесс развертывания контейнеров, однако является сложной в использовании технологией.

В результате анализа существующих решений был сделан вывод, что ни одно из них не позволяет полностью автоматизировать процесс контейнеризации системы многоагентного моделирования без значительного увеличения сложности процесса сборки.

В рамках работы предполагается разработать средство автоматизации процесса контейнеризации, которое позволит автоматизировать создание, загрузку и скачивание образов из удаленного репозитория, а также оптимизировать размер образов.

Процесс состоит из 7 этапов. Традиционных подход предполагает, что лишь 3 из 7 этапов являются полностью автоматизированными. Разработанное средство автоматизации оставляет неавтоматизированным только этап тестирования созданных образов компонентов системы. На рисунке 1 представлены традиционный и предлагаемый подходы к процессу.

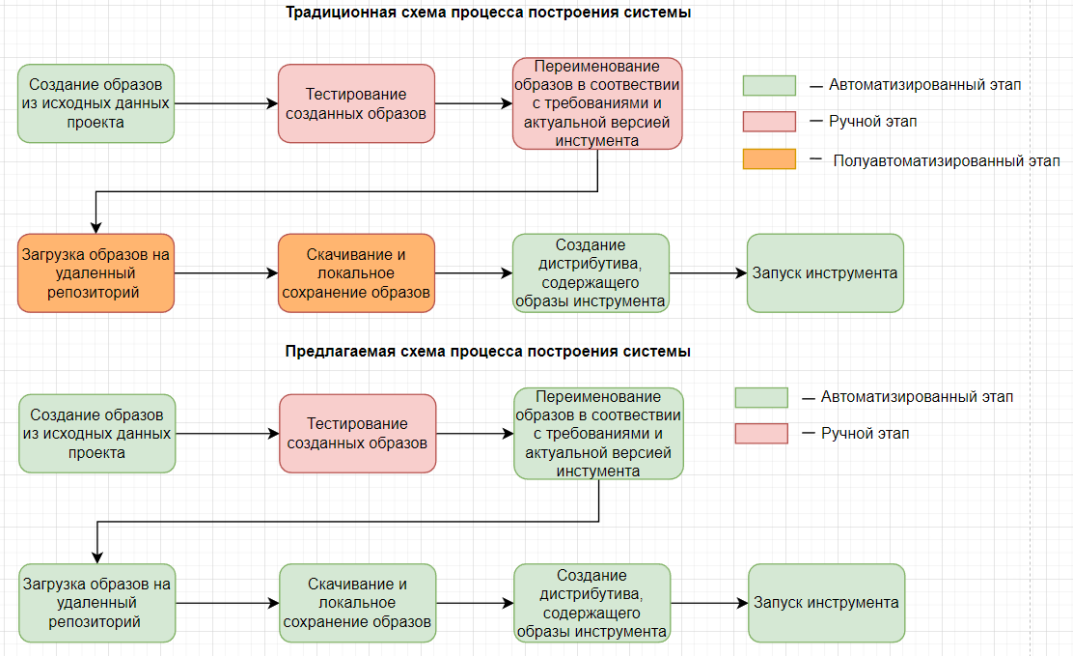


Рисунок 1 – Схемы традиционного и предлагаемого подхода к процессу построения системы многоагентного моделирования

Данное средство автоматизации реализовано в виде консольного приложения с использованием языков Python и Bash, технологии docker. Приложение позволяет взаимодействовать с удаленным хранителем образов – GitLab Container Registry.

Оценка эффективности разработанного решения будет проводиться по сокращению затрачиваемого программистом времени на контейнеризацию, а также по сокращению размера образов.

ЛИТЕРАТУРА

1. Имитационное моделирование поведения сложных многоагентных систем с использованием вероятностной модели / А. М. Сабуткевич, Д. А. Вихляев, И. В. Никифоров, А. В. Самочадин //

Современные технологии в теории и практике программирования : Сборник материалов конференции, Санкт-Петербург, 26 апреля 2022 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2022. – С. 98-100. – EDN ACJXEE.

2. Konev, I. Algorithm for Containers' Persistent Volumes Auto-scaling in Kubernetes / I. Konev, I. Nikiforov, S. Ustinov // Conference of Open Innovations Association, FRUCT. – 2022. – No. 31. – P. 89-95. – DOI 10.23919/FRUCT54823.2022.9770916. – EDN SJPFOH.
3. Сафронов, Д. Автоматическая балансировка нагрузки между потоковой обработкой данных и внутренними задачами кластера с использованием Kubernetes / Д. Сафронов, К. М. Стоноженко, И. В. Никифоров // Современные технологии в теории и практике программирования : сборник материалов конференции, Санкт-Петербург, 23 апреля 2020 года / Санкт-Петербургский политехнический университет Петра Великого; Dell Technologies; EPAM Systems. – Санкт-Петербург: ПОЛИТЕХ-ПРЕСС, 2020. – С. 165-167. – EDN VOXGIM.

УДК 004.457

И. Р. Молчанов (4 курс бакалавриата),
Б. М. Медведев, к.т.н., доцент

РАЗРАБОТКА ПРОГРАММНЫХ СРЕДСТВ РАСПОЗНАВАНИЯ OFDM СИГНАЛОВ В ШИРОКОПОЛОСНОЙ СПЕКТРОГРАММЕ

Системы радиоконтроля предназначены для выявления нарушений норм использования радиочастот, организации частотного планирования, учета и контроля радиоэлектронных средств [1]. Существующие беспроводные телекоммуникационные системы используют очень большой диапазон радиочастот (от очень низких частот до гипервысоких частот), и последовательное узкополосное сканирование радиоэфира не обеспечивает обнаружение сигналов в реальном времени. В связи с этим актуальной является разработка широкополосных средств радиомониторинга, позволяющих обнаружить и классифицировать множество сигналов с неизвестными параметрами (частота несущей, полоса занимаемых частот, мощность сигнала и метод модуляции).

Современные беспроводные локальные сети WiFi, а также сотовые сети 4G и 5G для передачи информации используют OFDM сигналы [2]. При анализе сигналов в широкой полосе частот будут присутствовать как OFDM сигналы, так и сигналы с другими методами модуляции, такими, например, как фазовая и квадратурая амплитудная модуляции. Для определения параметров OFDM сигналов необходимо обнаружить все сигналы в анализируемой полосе частот, а также определить какие из них принадлежат классу OFDM сигналов. Для решения задач обнаружения и классификации перспективным является использование нейронных сетей, демонстрирующих высокую вероятность правильной классификации до 90% для некоторых типов сигналов [3].

Таким образом, целью данной работы является разработка программных средств распознавания OFDM сигналов при помощи нейронной сети.

Для достижения поставленной цели необходимо решение следующих задач:

1) Анализ существующих решений для поиска и классификации сигналов с использованием нейронных сетей.
2) Разработка генератора тестовых сигналов для обучения нейронной сети.
3) Обучение нейронной сети YOLOv8.
4) Разработка программных средств оценки результатов обнаружения и классификации.

Анализ существующих систем классификации показал перспективность двух моделей нейронных сетей: нейронная сеть с продолжительной короткой памятью [4] и нейронная сеть YOLO [3], использующаяся для решения задач детектирования объектов на изображениях. Основным преимуществом сети YOLO является возможность реализации как обнаружения

неизвестных сигналов, так и классификации сигналов по используемому методу модуляции. В качестве входных данных такая сеть использует спектрограмму – изображение, показывающее изменения спектра сигнала во времени.

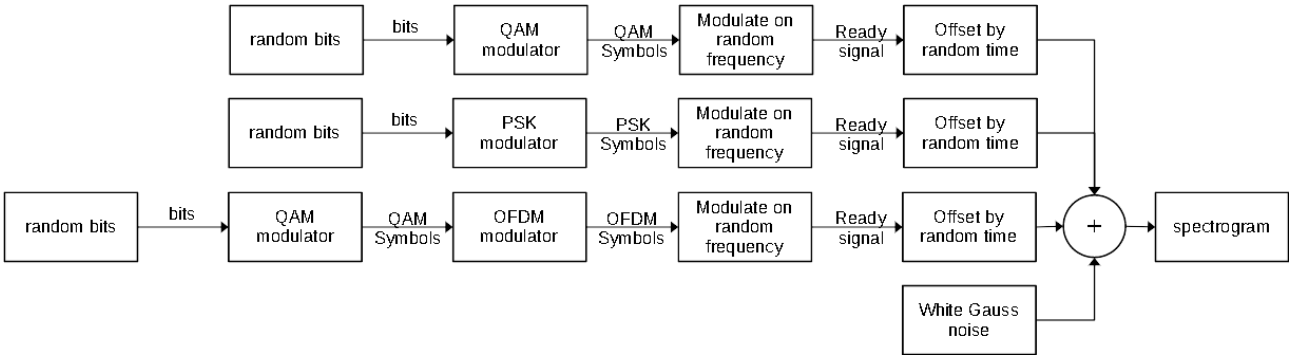


Рисунок 1 – Функциональная схема генератора тестовых сигналов

Для генерация тестовых данных, используемых в обучении нейронной сети, разработаны программные средства в среде Matlab (см. рисунок 1). Генератор создает спектрограмму длительностью до 26 мс с шириной полосы частот 640 МГц. При генерации спектрограммы используются 3 вида модуляции: квадратурно-амплитудной модуляция (QAM), фазовая модуляция (PSK) и мультиплексирование с ортогональным частотным разделением каналов (OFDM). Параметры OFDM сигналов, такие как: общее количество подканалов, количество защитных каналов и номера синхронизирующих подканалов (pilot channels) соответствуют требованиям стандарта IEEE 802.11ac [5]. Генератор формирует сигналы, разнесенные по частоте и времени со случайным смещением, но без перекрытия. Обучающая и тестовая выборки создаются при отношении сигнал/шум, задаваемым пользователем. Спектрограмма представлена в виде изображения JPG формата. Для обучения и анализа результатов классификации генератор также создает информационный файл, в котором содержатся типы и положения сигналов в изображении.

Для детектирования и классификации сигналов в спектрограмме была выбрана нейронная сеть YOLOv8 [6]. Программные средства обучения нейронной сети, запуска и анализа результатов разработаны средствами язык программирования Python.

Разработанные программные средства могут быть использованы в качестве прототипа встраиваемых программных средств в системах радиомониторинга.

ЛИТЕРАТУРА

1. Рудницкий В. Д., Медведев Б. М. Обнаружение неизвестных радиосигналов в частотном спектре. Современные технологии в теории и практике программирования: сборник материалов научно-практической конференции студентов, аспирантов и молодых ученых, 26–27 апреля 2023 г. – СПб. : ПОЛИТЕХ-ПРЕСС, 2023.
2. IEEE SA – IEEE 802. [Электронный ресурс] Режим доступа: https://standards.ieee.org/featured/ieee-802/#:~:text=The%20most%20widely%20used%20IEEE,providing%20focus%20for%20each%20area.
3. Adela Vagollari, Viktoria Schram, Wayan Wicke, Martin Hirschbecky, and Wolfgang Gerstacker. Joint Detection and Classification of RF Signals Using Deep Learning // IEEE 93rd Vehicular Technology Conference (VTC2021-Spring). – 2021 – DOI: 10.1109/VTC2021-Spring51267.2021.9449073
4. Mario Bkassiny. A Deep Learning-based Signal Classification Approach for Spectrum Sensing using Long Short-Term Memory (LSTM) Networks // 2022 6th International Conference on Information Technology, Information Systems and Electrical Engineering (

ICITISEE). – 2022 - DOI: 10.1109/ICITISEE57756.2022.10057728
5. Matthew S. Gast. 802.11ac: A Survival Guide // Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol – 2013 – CA 95472; ISBN: 978-1-449-34314-9
6. Ultralytics YOLOv8 Docs [Электронный ресурс] Режим доступа: https://docs.ultralytics.com/ru

УДК 004.4'2

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

ПОДХОД К СОПРОВОЖДЕНИЮ СИСТЕМЫ ОБРАБОТКИ И ХРАНЕНИЯ ИНФОРМАЦИИ

Целью работы является разработка системы обработки информации и построение хранилища данных. В рамках данного подхода используется отечественное программное обеспечение, которое адаптируется и модернизируется под специфику проекта. Для организации сопровождения системы используется следующий стек технологий: продукты российской компании ArenaData для развертывания платформы данных, GitLab как средство для хранения кода и выполнения CI/CD, Ansible для автоматизации подготовки ресурсов и развертывания приложений, bash для написания скриптов для бэкапирования, развертки приложений и автоматизации, технология docker-контейнеров для эксплуатации приложений. Вся система находится в облаке, там же осуществляется бэкапирование.

Система используется для хранения и обработки больших объемов данных, поэтому должна обеспечивать безопасность в хранении, масштабируемость в случаях увеличения количества данных, отказоустойчивость такие, как дублирование данных и восстановление после сбоев, высокую производительность и мониторирование ресурсов для контроля производительности. Благодаря разработанному методу сопровождение системы будет автоматизироваться, что уменьшает временные затраты на выполнения рутинных операций, повышает эффективность в разработке системы и позволяет предотвращать появление инцидентов.

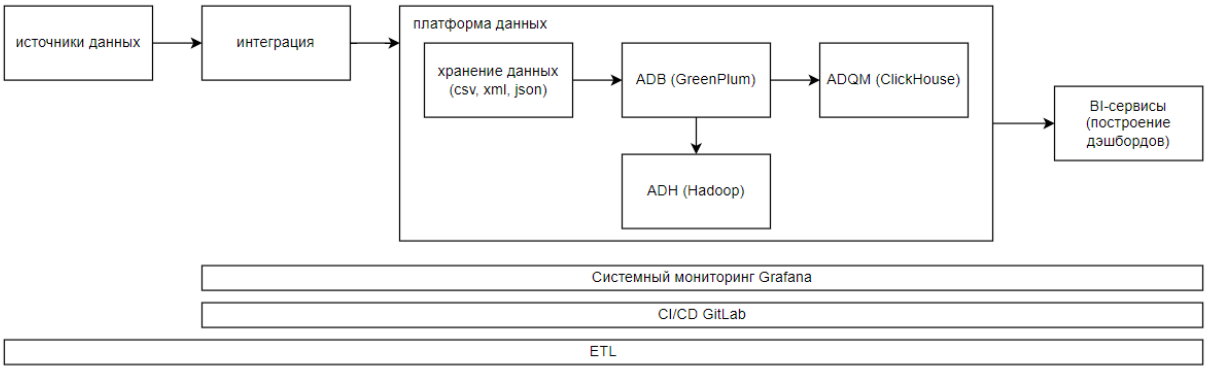


Рисунок 1 – Компонентная архитектура системы обработки и хранения данных

Данная система включается в себя компоненты хранилища данных:

- источники данных;
- модуль интеграции - различные API, которые либо уже существуют, либо специально разработаны;
- платформа данных, которая включается в себя прием и хранение данных в различных форматах, передачу “сырых” данных в базу данных GreenPlum, обработку данных в Hadoop и хранение обработанных данных в ClickHouse;
- BI-сервисы;
- мониторинг;
- управление и менеджмент данных.

Основными приоритетами в сопровождении систем хранения данных являются масштабируемость - при увеличении хранения и обработки данных необходимо увеличивать и ресурсы, и производительность, при которой осуществляется корректная и быстрая обработка больших объемов данных. Первое автоматизируется за счет написания скриптов и конфигураций, при которых удается в быстро осуществить настройку окружения и

развертывание компонентов системы. Также это позволяет сохранять идентичность сред на разных контурах, что упрощает интеграцию. Надежность, отказоустойчивость и производительность достигаются путем оценки требуемых ресурсов, выбора средств для реализации и настройки мониторирования системы. В данной ситуации отдается предпочтение хранению данных с дублированием сегментов для обеспечения целостности данных. Резервное копирование осуществляется и хранится в гипервизоре для гибкого управления данными.

В ходе работы был реализован подход к сопровождению системы обработки и хранения информации: проведен анализ средств для реализации, спроектирована система и реализована система хранения данных. На данный момент продолжается разработка скриптов и конфигураций для

ЛИТЕРАТУРА

1. Antonio Capizzi, Salvatore Distefano, Manuel Mazzara. From DevOps to DevDataOps: Data Management in DevOps Processes // 2020 Software Engineering Aspects of Continuous Development and New Paradigms of Software Production and Deployment (pp.52-62)
2. Boon Keong Seah; Nor Ezam Selan. Design and implementation of data warehouse with data model using survey-based services data // Fourth edition of the International Conference on the Innovative Computing Technology (INTECH 2014). Режим доступа: https://ieeexplore.ieee.org/document/6927748
3. Oras Baker; Chuong Nguyen Thien. A New Approach to Use Big Data Tools to Substitute Unstructured Data Warehouse // 2020 IEEE Conference on Big Data and Analytics (ICBDA). Режим доступа: https://ieeexplore.ieee.org/document/9289757
4. Hong-Mei Chen; Rick Kazman; Serge Haziyev. Requirements for DataOps to foster Dynamic Capabilities in Organizations - A mixed methods approach // 2022 IEEE 24th Conference on Business Informatics (CBI). Режим доступа: https://ieeexplore.ieee.org/document/9944751

УДК 004.415

Е. И. Морева (2 курс магистратуры),
П. Д. Дробинцев, к.т.н., доцент

ПОВЫШЕНИЕ ПРОИЗВОДИТЕЛЬНОСТИ СИСТЕМЫ ИНТЕЛЛЕКТУАЛЬНОГО АНАЛИЗА ДАННЫХ ЗА СЧЕТ ИСПОЛЬЗОВАНИЯ МЕТОДОВ ИЗЯЩНОЙ ДЕГРАДАЦИИ

В социальных сетях ежедневно публикуются невообразимые объёмы текста, изображений и видео, при этом весь контент необходимо отслеживать на предмет соответствия правилам платформы и законодательства.

Система интеллектуального анализа пользовательского контента является критически важной для социальной сети, поскольку она напрямую влияет на доход и посещаемость сайта и приложений. Этот сервис, как и вся социальная сеть является распределенной системой.

В распределенных системах крайне сложно гарантировать, что все сервисы будут доступны все время [1]. Компоненты могут выходить из строя, изменяться, быть временно недоступными или замедляться [2]. Особенно опасна проблема каскадного отказа системы [3]. При наличии отказоустойчивых стратегий и шаблонов можно обеспечить доступность платформы даже в случае сбоя.

Целью данной работы является модификация системы таким образом, чтобы она обнаруживала сбои и автоматически корректировала свое поведение для их компенсации путем использования подходов изящной деградации.

Graceful degradation (изящная деградация) — ряд мер, направленных на улучшение доступности системы при отказе ее части. В системе, в которой реализована изящная деградация [4], за счет снижения эффективности работы, скорости доставки, доступности

функций [5] даже одновременная потеря нескольких компонентов не приводит к простоям. Вместо этого функциональность снижается до уровня, который могут поддерживать остальные рабочие компоненты.

Для достижения цели необходимо решить следующие задачи:

1) Проанализировать существующие решения и подходы, а также текущую архитектуру сервиса

2) Определить наиболее нагруженные компоненты, неэффективное использование ресурсов, наиболее критические функции

3) Исправить архитектурные ошибки, приводящие к неэффективному использованию ресурсов

4) Реализовать модуль мониторинга состояния системы и модуль управления обработкой данных, позволяющий автоматически корректировать функциональность системы с учетом текущей нагрузки

Архитектура сервиса представлена на рисунке 1. Взаимодействие между клиентом, а также другими сервисами необходимыми для обработки контента реализовано с использованием One-nio и Apache Kafka. Приложение развернуто в собственном облаке One-cloud в нескольких датацентрах, что гарантирует наличие резервных компонентов и автоматического переключения между ними. В качестве поставщика данных может выступать Kafka кластер, файлы в HDFS или другой микросервис с NoSQL или SQL базой данных.

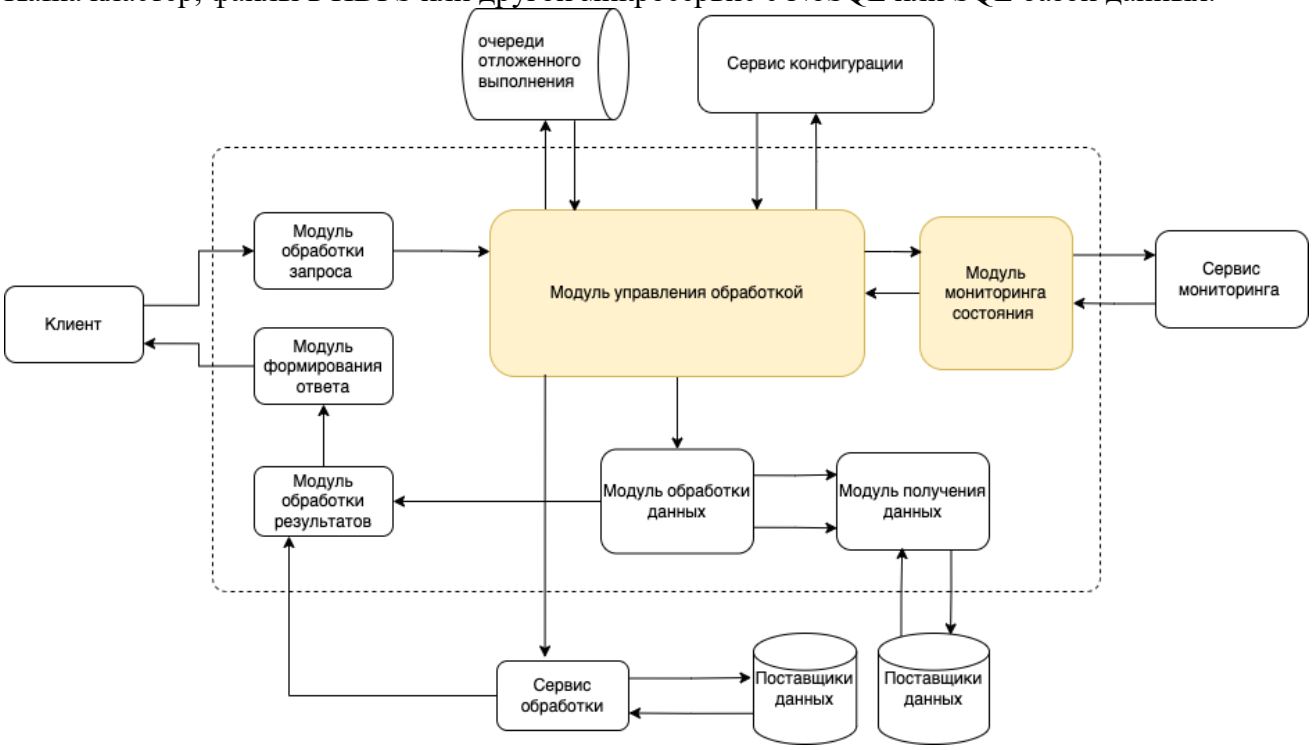


Рисунок 1— Архитектура сервиса

Для реализации включения и отключения функциональности приложения во время выполнения используются специальные флаги функций, настраиваемые в отдельном сервисе конфигураций.

Для предотвращения проблемы исчерпания ресурса потоков при обращении к внешней системе или базе данных применение используются подходы реактивного программирования, что позволяет добиться неблокирующего и асинхронного выполнения программы, обеспечивающего более эффективное управление потоками.

Модуль управления откладывает или отбрасывает выполнение той или иной задачи по обработке выходящих данных в зависимости от их приоритета, вероятности наличия неприемлемого контента и текущего состояния сервиса.

Благодаря разработанным модулям может быть повышена доступность сервиса, а также сокращены расходы, за счет улучшения контроля некритичного трафика во внепиковые периоды.

ЛИТЕРАТУРА

1. van Steen, M., Tanenbaum, A.S. A brief introduction to distributed systems. — DOI https://doi.org/10.1007/s00607-016-0508-7 // Computing. — 2016. — pp. 967–1009.
2. Y. Liu, X. Zhang and M. Auguston. Modeling and Analyzing Timed Security Protocols Using Extended Timed CSP. – DOI 10.1109/SSIRI.2010.29 // Secure System Integration and Reliability Improvement. — Singapore, Singapore. — 2010. — pp. 217–226.
3. Addeen H.H. A Dynamic Fault Tolerance Model for Microservices Architecture. — DOI https://api.semanticscholar.org/CorpusID:201905190 // Semantic Scholar. — 2019
4. Edwards T., Lee P.U. Designing graceful degradation into com- plex systems: The interaction between causes of degradation and the association with degradation prevention and recovery. — DOI https://doi.org/10.2514/6.2018-3510 // Aviation Technology, Integration, and Operations Conference. — 2018
5. Falahah, Surendro K., Sunindyo W.D. Circuit Breaker in Mi- croservices: State of the Art and Future Prospects. – DOI https://doi.org/10.1088/1757-899x/1077/1/012065 // IOP Conference Series: Materials Science and Engineering. — 2021

УДК 004.932

С. Мхаммад (1 курс магистратуры),
С. А. Молодяков, д.т.н., профессор

ПРИМЕНЕНИЕ НЕЙРОННЫХ АЛГОРИТМОВ ДЛЯ РАСШИРЕНИЯ ВОЗМОЖНОСТЕЙ ВИДЕОРЕДАКТОРА OPENSHOT

В последние несколько лет огромными темпами растет использование мультимедийных данных. Это привело к увеличению потребности в разработке новых мощных программ, особенно видеоредакторов. Последние исследования в области видеоредакторов были направлены на расширение их функциональности, в том числе на использование нейронных алгоритмов. Нейронные алгоритмы предлагают использовать для анализа контента, поиска объектов, отслеживания движения, распознавания лиц и др.

Известны профессиональные видеоредакторы, такие как Adobe Premiere pro, PowerDirector и др., в которых используются нейронные алгоритмы. Эти редакторы в основном работают под управлением операционных систем Windows или MacOS. С другой стороны, большинство видеоредакторов, работающих в системе Linux, по-прежнему используют традиционные алгоритмы в видеопроцессах. В данном исследовании рассматриваются вопросы применения нейронных алгоритмов в видеоредакторе OpenShot, который может работать в операционных системах Linux, Mac и Windows.

OpenShot Video Editor - это бесплатный видеоредактор с открытым исходным кодом. Он написан на языке Python с использованием PyQt5 для интерфейсов и библиотек C++ для редактирования видео и аудио [1]. OpenShot также использует FFmpeg, который представляет собой набор бесплатных библиотек с открытым исходным кодом, позволяющих записывать, конвертировать, передавать цифровые аудио- и видеозаписи в различных форматах [2, 3]. Он предоставляет такие основные возможности, как импорт видео-, аудио- и графических файлов, интерфейс временной шкалы для организации и редактирования медиаклипов, создание и настройка текстовых накладок для видео и т. д.

Таким образом, целью данной работы является расширение функциональности видеоредактора OpenShot. Разрабатывается новая версия видеоредактора, в котором с помощью нейронных алгоритмов будут решены следующие задачи:

1. Извлечение звука: эта задача содержит извлечение аудиопотока из мультимедийного файла с помощью FFmpeg.

2. Препроцессинг: на этом этапе качество аудио будет улучшено с помощью понижающей дискретизации, фильтрации и шумоподавления.

3. Извлечение признаков: Из предварительно обработанных аудиоданных извлекаются аудиофункции, имеющие отношение к распознаванию речи, такие как Mel-Frequency Cepstral Coefficients (MFCCs) или спектрограммы. Эта задача важна для идентификации языка по речевому сигналу [4].

4. Распознавание речи: на основе скрытых марковских моделей (HMM) распознается и транскрибируется речевой контент.

5. Транскрипция: распознанный речевой контент транскрибируется в текстовый формат, преобразуя звуковые характеристики в человекочитаемый текст.

6. Постобработка: эти этапы обработки будут применены к транскрибированному тексту, который будет содержать текст перевода с русского на английский / с английского на русский с использованием моделей нейромашинного перевода (NMT)[5].

Каждая из предыдущих задач будет реализована в OpenShot как независимые функции, поэтому пользователь сможет использовать их как новые возможности. Но в то же время эти функции будут работать вместе, чтобы достичь результата, как показано на рисунке 1.

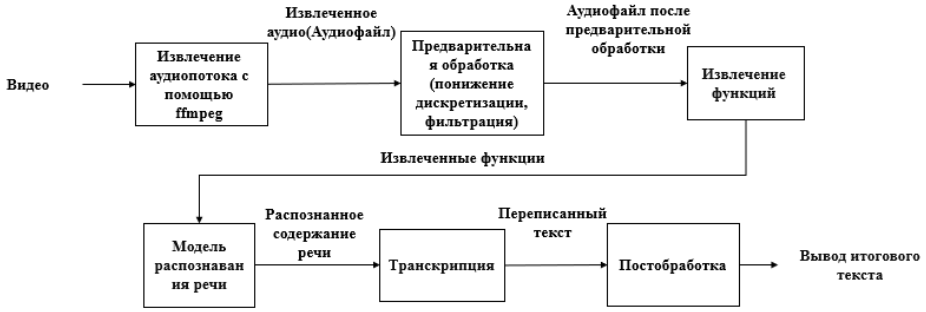


Рисунок 1 – Рабочие алгоритмы исследования

Для применения распознавания речи в видеоредакторе используются скрытые марковские модели (HMM). Эта модель является одним из методов автоматического распознавания устной речи. Основные компоненты такой системы автоматического распознавания речи показаны на рисунке 2. После процесса извлечения признаков речь (входной сигнал) преобразуется в последовательность акустических векторов фиксированного размера Y = y1,.....,yT. Затем декодер пытается найти последовательность слов w = w1,.....,wL, которая с наибольшей вероятностью порождает Y [6].

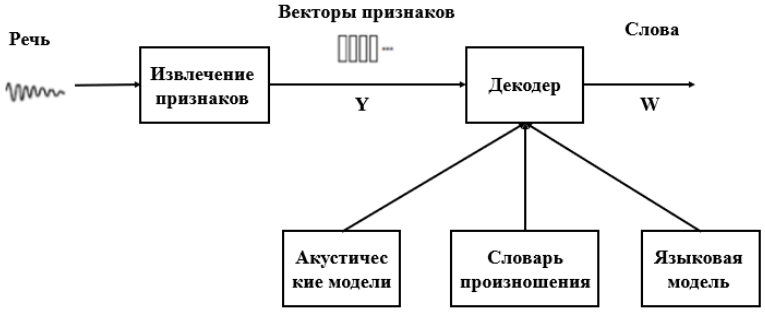


Рисунок 2 – Архитектура распознавателя на основе ЧММ [6].

Ожидается, что обновленная версия OpenShot, используя предложенный алгоритм, сможет удалять шумы из аудио, извлекать из него первичные признаки, генерировать транскрибированный текст и переводить его с английского на русский и наоборот. Функции будут работать по отдельности, а затем объединяться для получения окончательного переведенного текста. В заключение следует отметить, что данное исследование посвящено

улучшению видеоредактора OpenShot в системе Linux путем применения нейронных алгоритмов.

ЛИТЕРАТУРА

1. Информация об OpenShot (2023) URL: https://www.openshot.org/about/ (15.02.2024).
2. Молодяков С. А. Применение функций FFmpeg в мультимедийных приложениях (100 примеров на Python). СПб. : ПОЛИТЕХ-ПРЕСС. – 2023. – 514 с. DOI 10.18720/SPBPU/2/i23-44
3. Молодяков С.А., Милицын А. В. Алгоритмы работы с мультимедийными данными в telegram-боте (100 примеров на Python). Санкт-Петербург : ПОЛИТЕХ-ПРЕСС. – 2024. – 586 с. ISBN 978-5-7422-8448-2
4. Abdul Z. K., Al-Talabani A. K. Mel frequency cepstral coefficient and its applications: A review //IEEE Access. – 2022. – Т. 10. – С. 122136-122158.
5. Grygoryev N. T. et al. English-Russian Data Augmentation for Neural Machine Translation //Proceedings of the 15th biennial conference of the Association for Machine Translation in the Americas (Workshop 2: Corpus Generation and Corpus Augmentation for Machine Translation). – 2022. – С. 1-10.
6. Deshmukh A. M. Comparison of hidden markov model and recurrent neural network in automatic speech recognition //European Journal of Engineering and Technology Research. – 2020. – Т. 5. – №. 8. – С. 958-965.

УДК 004.65

К. В. Победоносцев (4 курс бакалавриата),
А. П. Маслаков, ст. преподаватель

ПОВЫШЕНИЕ ОТКАЗОУСТОЙЧИВОСТИ И ЭФФЕКТИВНОСТИ СИСТЕМЫ ЗА СЧЕТ МИГРАЦИИ В NEWSQL ХРАНИЛИЩЕ ДАННЫХ

Социальная сеть Одноклассники [1], известна как одна из самых популярных платформ. На данный момент занимает второе место в России и одиннадцатое в мире по популярности. Только новогодние праздники 2023-2024 годов через Одноклассники было отправлено более миллиарда виртуальных подарков [2], что подчеркивает значимость этой функции в сети. Однако, такая высокая популярность также означает большие нагрузки на систему. Необходимо обеспечивать их полную доступность и работоспособность даже в периоды высокой нагрузки. На данный момент для хранения подарков используется Microsoft SQL Server [3] и это несет в себе несколько проблем:

1. При росте количества записей в SQL хранилище снижается скорость вставки. При таком количестве подарков, которое используется в Одноклассниках достичь удовлетворительной скорости добавления новых подарков не удается.
2. Добавление новых серверов можно проводить лишь вручную. А при добавлении новых шардов, реплики не обслуживают запросы.
3. Microsoft SQL Server требует использования Windows Server, что усложняет управление серверами.
4. Microsoft SQL Server является проприетарным решением, которое несет за собой коммерческие расходы и не включено в Единый Реестр ПО [4].

Исследование и оптимизация компонента подарков в социальной сети "Одноклассники" является востребованным и актуальным направлением развития. Перенос в NewSQL хранилище данных повысит отказоустойчивость, эффективность и скорость работы, а также позволит избавиться от затрат на использование коммерческого ПО и уменьшить расходы на сопровождение, а повышение скорости работы в свою очередь позволит создавать новую функциональность.

Целью данной работы является повышение эффективности и отказоустойчивости компонента подарков за счет миграции в NewSQL хранилище данных.

Для выполнения цели необходимо выполнить следующие задачи:

1. Изучить основные принципы работы традиционных SQL баз данных и NewSQL систем.
2. Провести анализ характеристик текущей SQL-системы, включая производительность, надежность и отказоустойчивость.
3. Изучить схему данных в текущем SQL хранилище и перестроить ее под NewSQL хранилище.
4. Написать код на языке Java для проведения миграции.
5. Сравнить характеристики системы до и после миграции.
6. Сделать анализ использования NewSQL хранилища данных для повышения отказоустойчивости и эффективности информационных систем.

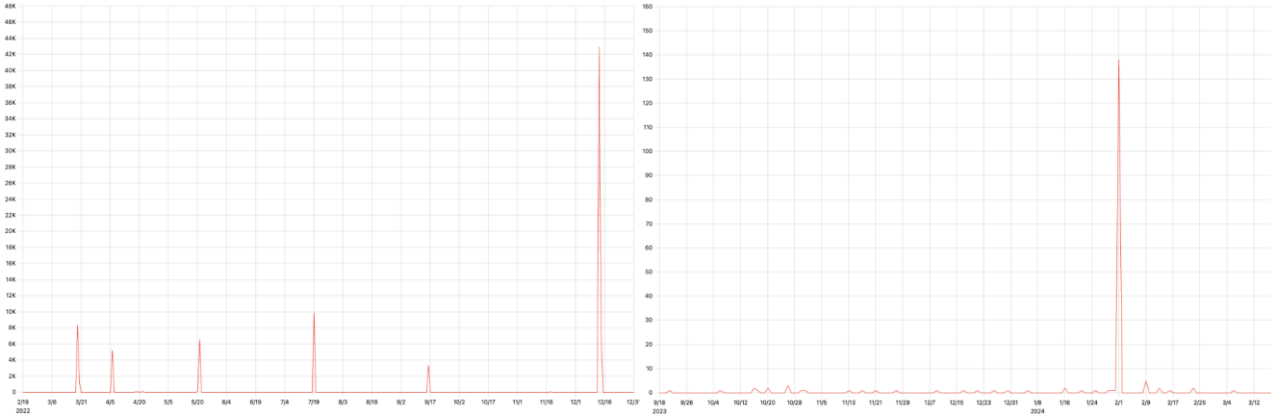


Рисунок 1 – Количество ошибок при использовании MSSQL и после миграции в C*One

На первом этапе работы было проведено исследование на рынке баз данных. Поскольку подобные хранилища нужны только в случае большого количества данных, компании предпочитают писать такие базы под свои конкретные цели. Под наши задачи была выбрана СУБД C*One - разработанная в Одноклассниках распределенная база данных, поддерживающая ACID транзакции. C*One основана на Cassandra, в связи с этим просто перенести данные 1 к 1 не представлялось возможным. В частности, в Cassandra нет возможности проверки конкретных битов в числе, в SQL на этом был основан поиск по категориям подарков. Поэтому схема данных была переработана под термины C*One, а на языке программирования Java был написан код, выполняющий миграцию данных с учетом новой структуры. В то время как данные переносились в C*One, записи о новых подарках записывались параллельно в оба хранилища, а после копирования дополнительно шел этап валидации. Только после этого, постепенно чтение переключалось на использование C*One.

Благодаря проведению миграции все проблемы, связанные с недоступностью данных, пропали. На рисунке 1 приведены графики количества ошибок при исполнении запросов к базе данных. После переноса данных ошибки возникают крайне редко (в основном 0–2 в день), а самый большой зафиксированный всплеск не превышал 150 ошибок за день. Также удалось избавиться от дорогостоящего оборудования, которое использовалось для поддержания доступности MS SQL кластера, а новый запас производительности открыл путь к реализации новой функциональности.

ЛИТЕРАТУРА

1. Top Websites Ranking - SimilarWeb. [Электронный ресурс]. – URL: https://www.similarweb.com/top-websites/computers-electronics-and-technology/social-networks-and-online-communities/ (дата обращения 18.03.2024).
2. ОК подвели итоги Нового года - Insideok. [Электронный ресурс]. – URL: https://insideok.ru/blog/ok-podveli-itogi-novogo-goda-milliard-podarkov-sotni-millionov-otkrytok-i-20-mln-polzovatelej-momentov/ (дата обращения 18.03.2024).
3. SQL Server 2022 - Microsoft. [Электронный ресурс]. – URL: https://www.microsoft.com/en-us/sql-server/sql-server-2022-pricing (дата обращения 18.03.2024).

4. Методические рекомендации АНО "ЦКИТ" по подготовке заявок на включение ПО в Единый реестр - ЦКИТ. [Электронный ресурс]. – URL: https://ru-ikt.ru/metodic#!/tab/595058081-1 (дата обращения 18.031.2024).
5. Забродин И.М., Круг П.Г. NewSQL - новый шаг в развитие BigData // Евразийский научный журнал – 2017. – №3. – С. 215-116.

УДК 004.9

А. А. Поздняков (4 курс бакалавриата),
И. А. Шемякин, ст. преподаватель

МОКИРОВАНИЕ СЕРВИСОВ, ЗАЩИЩЕННЫХ ЦИФРОВЫМ СЕРТИФИКАТОМ, ПРИ ИНТЕГРАЦИОННОМ ТЕСТИРОВАНИИ

Обычным решением при проведении интеграционного тестирования веб-приложений является мокирование внешних сервисов – запуск собственного сервера-заглушки, имитирующего поведение конкретного сервиса. Однако, при использовании клиентских библиотек, упрощающих взаимодействие с внешними сервисами, зачастую адрес сервиса записывается напрямую в исходный код библиотеки, вследствие чего изменение данного адреса и конфигурация приложения для использования сервера-заглушки в тестовом окружении представляются невозможными. Повсеместное использование цифровых сертификатов и протокола HTTPS для защиты сайтов [1] защищает пользователей от подмены доменов и усложняет мокирование внешних сервисов при интеграционном тестировании.

Разработка решения для мокирования сервисов, защищенных цифровым сертификатом, позволит упростить интеграционное тестирование приложений, использующих внешние сервисы. Целью данной работы является разработка подхода к интеграционному тестированию приложений, взаимодействующих с внешним API по протоколу HTTPS.

Защита от подмены домена в протоколе HTTPS обеспечивается посредством использования цифровых сертификатов. При установлении соединения, стороны осуществляют так называемое рукопожатие, в ходе которого сервер передаёт клиенту сертификат, подписанный приватным ключом центра сертификации (Certificate Authority - CA) [2]. Для успешной установки соединения необходимо, чтобы сертификат CA находился в списке доверенных сертификатов клиента.

В данной работе интеграционное тестирование проводилось в окружении Docker Compose. Мокирование внешнего сервиса осуществлялось с помощью Wiremock – фреймворка для создания заглушек API, работающих по протоколу HTTP [3]. Wiremock не позволяет использовать собственный сертификат при запуске сервера-заглушки в docker-контейнере, для решения этой проблемы в тестовое окружение был добавлен контейнер Nginx, выступающий в роли прокси между приложением и сервером-заглушкой и отвечающий за терминацию TLS-соединения. Для подмены домена внешнего сервиса использовалась встроенная DNS-служба Docker Compose [4].

Для установления доверия между приложением и прокси необходимо выпустить сертификат на доменное имя внешнего сервиса. При этом приватный ключ данного сертификата должен быть добавлен в контейнер Nginx для работы по HTTPS.

Вследствие публичности тестового окружения, возникает возможность утечки приватного ключа оконечного сертификата. Для решения данной проблемы в данной работе использовалось 2 ключа: корневой сертификат собственного CA, добавленный в список доверенных сертификатов приложения, и оконечный сертификат на доменное имя внешнего сервиса, подписанный корневым сертификатом. Оконечный сертификат имеет ограничение “CA:FALSE”, поддерживаемое стандартом X.509v3, которое не позволяет использовать

данный сертификат для подписи других сертификатов [5]. На рисунке 1 приведена схема разработанного решения.

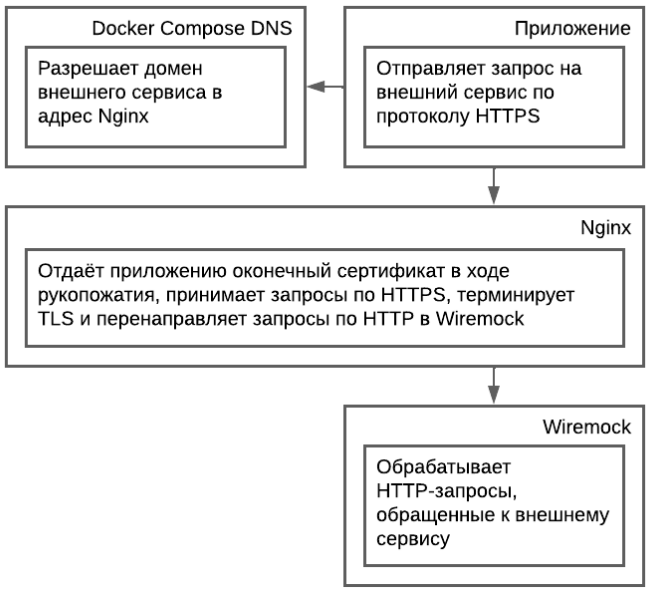


Рисунок 1 – Блок-схема разработанного решения

Таким образом, использование прокси-сервера, терминирующего TLS, и подмена домена с помощью Docker Compose DNS позволяет решить проблему мокирования сервисов, защищенных цифровым сертификатом, при интеграционном тестировании. При этом использование ограничений, описанных в стандарте X.509v3, позволяет дополнительно защитить приложение от возможной утечки тестовых сертификатов.

ЛИТЕРАТУРА

1. «Let's Encrypt Stats,» [В Интернете]. Available: https://letsencrypt.org/stats/. [Дата обращения: 19 Март 2024].
2. E. Rescorla, RFC 8446. The Transport Layer Security (TLS) Protocol Version 1.3, 2018.
3. «WireMock User Documentation,» [В Интернете]. Available: https://wiremock.org/docs/. [Дата обращения: 19 Март 2024].
4. «Docker Networking overview,» [В Интернете]. Available: https://docs.docker.com/network/. [Дата обращения: 19 Март 2024].
5. D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley и W. Polk, RFC 5280. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile.

УДК 004.622

М. А. Потапова (2 курс магистратуры),
О. Г. Малеев, к. т. н., доцент,
А. В. Ерошкин, ст. преподаватель

АНАЛИЗ И ПРИМЕНЕНИЕ АЛГОРИТМОВ ДЕДУПЛИКАЦИИ ДАННЫХ В MDM-СИСТЕМАХ

Целью работы является анализ и применение алгоритмов дедупликации мастер-данных для повышения эффективности управления ими в рамках разрабатываемого сервиса дедупликации в MDM-системе.

Дедупликация данных является одним из методов решения проблемы хранения ненужной, дублирующейся информации. Процесс дедупликации отвечает за идентификацию и удаление дубликатов данных [1]. Его применение позволяет снизить необходимые объемы

для хранения данных, ускорить процессы, связанные с анализом, резервным копированием и восстановлением данных, а также повысить качество данных.

В качестве общего алгоритма дедупликации данных использовалась следующая последовательность действий:

1. Нормализация данных, приведение их к общему формату
2. Поиск и определение дубликатов
3. Выделение «золотой записи» среди выделенных дублей
4. Удаление дублей

Не существует единого алгоритма для нормализации данных и поиска дублей, поэтому выбор как шагов нормализации, так и метода поиска дубликатов обуславливается спецификой данных и потребностями заказчика.

Для выделения каких-либо групп, к которым может быть применен один и тот же алгоритм дедупликации, были определены бизнес-поля. К примерам таких бизнес-полей можно отнести: последовательность цифр, адрес и ФИО.

В качестве алгоритмов поиска дублей рассматривались методы нечеткого поиска, основанные на символьном подходе, которые считаются наиболее точными при сравнении двух строк [2]. К таким методам можно отнести:

1. Метод, основанный на определении расстояния Левенштейна
2. Метод, основанный на определении расстояния Хэмминга
3. Метод Джаро-Винклера, основанный на определении расстояния Джаро
4. Метод, основанный на определении косинусного сходства для строк

Для выбора метода поиска дублей основными критериями выступали скорость и точность работы для определенных ранее бизнес-полей. На основании этих критериев был выбран метод, основанный на определении расстояния Левенштейна.

Для нормализации данных не существует стандартного решения, поэтому для каждого бизнес-поля была выбрана своя последовательность шагов. Таких как, например, удаление запятых, преобразование сокращений по определенным правилам, исправление наиболее частых опечаток, удаление окончаний и суффиксов и т.д. [4-5]

По итогам проведенного анализа для каждого бизнес-поля был выбран и протестирован алгоритм дедупликации данных. Пример работы дедупликации для бизнес-поля ФИО приведен на рисунке 1.

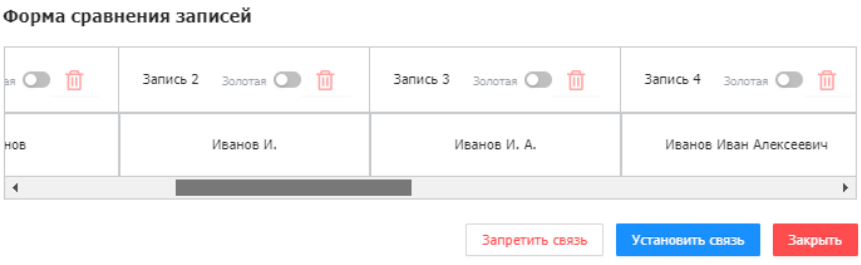


Рисунок 1 – Пример работы дедупликации для бизнес-правила "ФИО"

ЛИТЕРАТУРА

1. Sasi Kumar Raju Addepalli. Deduplication and Data Stewardship Process in MDM. [Электронный ресурс] Режим доступа: https://dzone.com/articles/de-duplication-and-data-stewardship-process-in-mdm
2. Чиркин Е.С., Лопатин Д.В. Подходы к нечеткому поиску нежелательного контента на веб-странице // ВЕСТНИК ТАМБОВСКОГО УНИВЕРСИТЕТА. СЕРИЯ: ЕСТЕСТВЕННЫЕ И ТЕХНИЧЕСКИЕ НАУКИ – Тамбов: Тамбовский государственный университет им. Г.Р. Державина, 2016 г. – С. 2358-2365

3. Нечеткое сравнение строк с помощью rapidfuzz. [Электронный ресурс] Режим доступа: https://habr.com/ru/articles/733492
4. Neri Van Otten. How To Use Text Normalization Techniques In NLP With Python. [Электронный ресурс] Режим доступа: https://spotintelligence.com/2023/01/25/text-normalization-techniques-nlp
5. Бакаев И.И., Шафиев Т.Р. Методы построения алгоритмов стемминга для естественных языков // ПРОБЛЕМЫ ВЫЧИСЛИТЕЛЬНОЙ И ПРИКЛАДНОЙ МАТЕМАТИКИ, 2020 г. – С. 146-153

УДК 004.8

А. И. Пухальский, А. Н. Тихонов (4 курс бакалавриата),
Д. Ф. Дробинцев, ст. преподаватель,
А. В. Гончаров, ассистент

РАЗРАБОТКА ПРИЛОЖЕНИЯ С ИСПОЛЬЗОВАНИЕМ АЛГОРИТМОВ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА ДЛЯ УПРАВЛЕНИЯ ЗАДАЧАМИ

В современном мире трудно соблюдать баланс между учебой, работой, личными делами и досугом. Это происходит за счет специфики нынешнего темпа жизни, который с каждым годом растет. Данное явление наиболее распространено среди людей, проживающих в больших городах. Они спешат, пытаясь успеть выполнить поставленные задачи на день, но не всегда получается достигнуть поставленных целей, поскольку время в сутках фиксировано, а методами тайм-менеджмента владеют не все. И если над временем, в общем его понимании, мы не властны, то повысить свою продуктивность путем грамотного тайм-менеджмента – способны. Но в данном случае возникает другой вопрос: что делать, если количество задач растет, а человеческая память не безгранична? Тут необходим инструмент, который способен помочь в структурировании, определении приоритетов, отслеживании сроков, а также сохранении и отображении поставленных задач. Неотъемлемым решением в данном случае являются приложения для планирования и отслеживания задач, которые признаны решить эту проблему. Следует учесть тот факт, что не всегда есть возможность запланировать задачу, а также записать некоторую важную информацию. В решении этого вопроса может помочь искусственный интеллект. В частности, голосовой помощник Алиса.

Цель работы заключается в разработке приложения для планирования задач, включающего: функциональную часть, интуитивно понятный пользовательский веб-интерфейс, интеграцию с голосовым помощником Алиса для взаимодействия пользователя с планировщиком задач, в частности для автоматизации записи задач и создания заметок.

В рамках системы (рисунок 1), для реализации серверной части выбран язык программирования Java+Spring Boot [2, 3, 4, 5].

Для реализации пользовательского интерфейса выбраны следующие инструменты: HTML, TypeScript, Tailwind [6], React [7].

Для хранения данных используется СУБД PostgreSQL [1]. Это объектно-реляционная система управления базами данных, которая отличается открытым исходным кодом и активным сообществом разработчиков. Достоинства PostgreSQL:

- Обеспечивает возможность горизонтального и вертикального масштабирования, что позволяет работать с большими объемами данных и высокими нагрузками.
- Распространяется под открытой лицензией, что делает его доступным для использования и модификации без ограничений.
- Активно поддерживает стандарты SQL и предлагает богатые возможности для работы с данными, включая поддержку геоданных, JSON и многие другие.
- Предоставляет широкий набор встроенных функций и возможностей расширения, что позволяет разработчикам создавать решения под свои нужды.

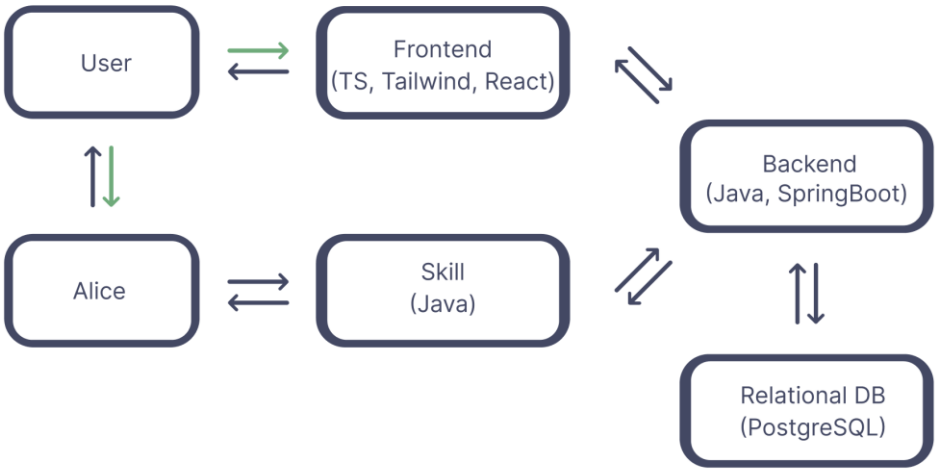


Рисунок 1 – Архитектура системы

ЛИТЕРАТУРА

1. PostgreSQL [Электронный ресурс]. URL: https://www.postgresql.org/ (Дата обращения: 10.03.2023)
2. Spring projects [Электронный ресурс]. URL: https://spring.io/projects (Дата обращения: 09.03.2023)
3. Walls, C. Spring in Action / C. Walls. — Moscow: DMK Press, 2019. — 624 p. — ISBN 978-5-97060-624-3.
4. Проектирование API [Электронный ресурс]. URL: https://systems.education/api-design (Дата обращения: 10.03.2023)
5. REST API [Электронный ресурс]. URL: https://habr.com/ru/articles/483202/ (Дата обращения: 10.03.2023)
6. Официальная документация Tailwind [Электронный ресурс]. URL: https://tailwindcss.com/ (Дата обращения 10.03.2023)
7. Официальная документация React [Электронный ресурс]. URL: https://ru.legacy.reactjs.org/ (Дата обращения: 09.03.2023)

УДК 004.415.2

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

РАЗРАБОТКА МИКРОСЕРВИСОВ ВЗАИМОДЕЙСТВИЯ С ПОЛЬЗОВАТЕЛЯМИ В РАМКАХ БИЛЛИНГОВОЙ СИСТЕМЫ

Целью работы является создание микросервисов для поддержки взаимодействия с пользователями биллинговой системы. Биллинговая система в телекоммуникационной отрасли - это процесс, используемый компанией для выставления счетов клиентам [1,2]. Основная задача заключается в автоматизации процедуры обработки звонков. Автоматизированный биллинг позволяет уменьшить количество ошибок при расчете стоимости. Благодаря такой системе появляется возможность обрабатывать огромное количество клиентов.

Компоненты программного обеспечения, которые планируется создать в ходе работы, необходимы для реализации бизнес-логики биллинговой системы. Данный программный продукт включает в себя множество микросервисов [3-6], но в рамках проекта будут реализованы следующие:

- сервис для обработки и хранения персональных данных пользователей;
- сервис для управления тарифами связи;
- сервис для обработки звонков пользователей;

- сервис для биллинга - расчёта стоимости звонка для определенного тарифа под конкретного пользователя;
- сервис для тарификации - реализация обработки входных данных для начала процесса выставления счета.

Схема взаимодействия микросервисов представлена на рисунке 1.

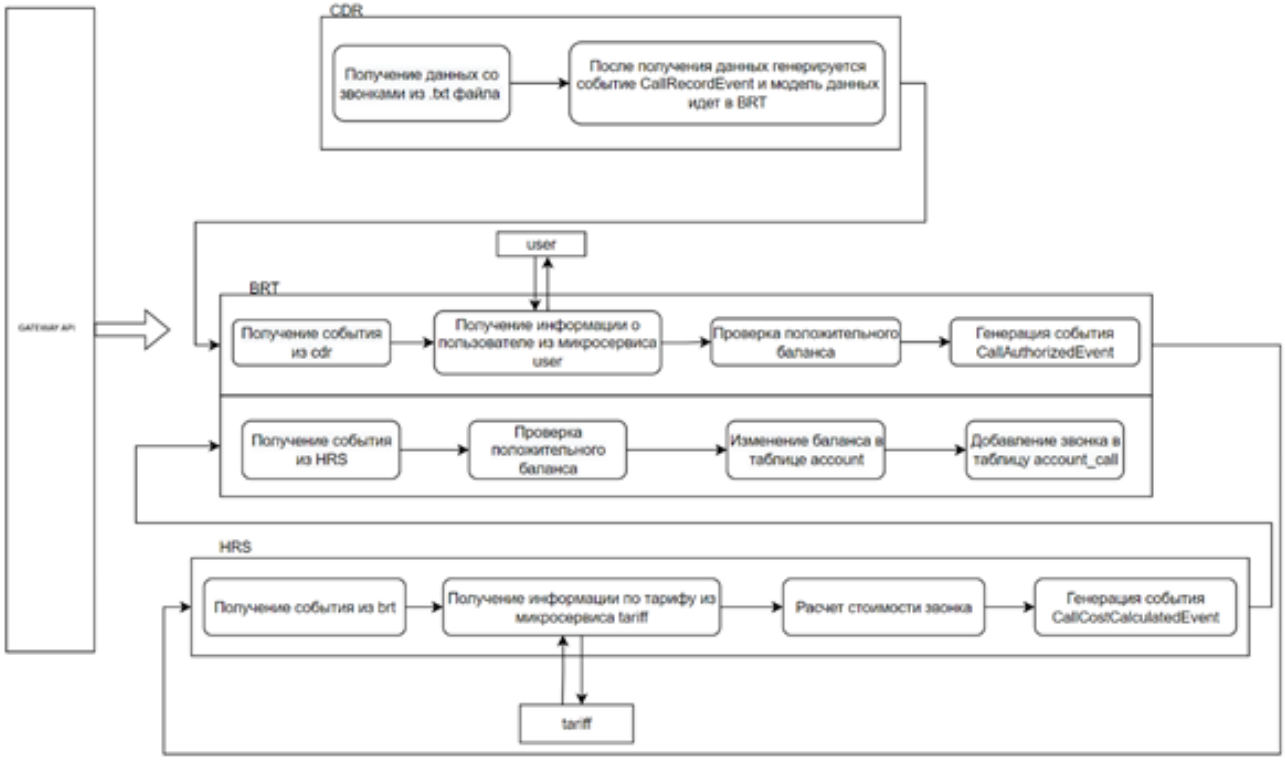


Рисунок 1 – Схема взаимодействия микросервисов

Логика сервисов реализована на языке программирования Java при помощи Spring Boot. Для увеличения отказоустойчивости системы используется асинхронное общение между сервисами посредством брокера сообщений [7]. В качестве СУБД используется PostgreSQL.

ЛИТЕРАТУРА

1. Dr.N. Baggyalakshmi, M.S. Harrsini, Dr.R. Revathi. Smart Billing Software // International Academic Journal of Innovative Research. February 2024 [Электронный ресурс] Режим доступа: https://www.researchgate.net/publication/378569303_Smart_Billing_Software
2. K. K. Deepika, Ronanki Ravi Sankar, A. V. Satyanarayana. RESTful API Integrations in Telecom Billing System Management // Proceedings of the International Conference on Cognitive and Intelligent Computing. November 2022 [Электронный ресурс] Режим доступа: https://www.researchgate.net/publication/365024776_RESTful_API_Integrations_in_Telecom_Billing_System_Management
3. Microservice architecture. [Электронный ресурс] Режим доступа: https://microservices.io/tags/microservice%20architecture
4. Tom Černý, Michael J. Donahoo, Michal Trnka. Contextual understanding of microservice architecture: current and future directions // ACM SIGAPP Applied Computing Review. January 2018 [Электронный ресурс] Режим доступа: https://www.researchgate.net/publication/322842819_Contextual_understanding_of_microservice_architecture_current_and_future_directions
5. Чучин, Д. Ю. Разработка серверной части приложения для предоставления интерактивной среды клиентам HTTP API / Д. Ю. Чучин, В. Э. Шмаков // Современные технологии в теории и практике программирования: Сборник материалов конференции, Санкт-Петербург, 26 апреля 2022 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение

высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2022. – С. 112-114. – EDN EARKZS.

6. Томилин, И. С. Развертывание микросервисной архитектуры с использованием consul, vault, nomad / И. С. Томилин, С. А. Федоров, В. Э. Шмаков // Современные технологии в теории и практике программирования : Сборник материалов научно-практической конференции студентов, аспирантов и молодых ученых, Санкт-Петербург, 26–27 апреля 2023 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2023. – С. 255-256. – EDN FUYZVW.
7. Message brokers – use cases & model AWS implementation using SNS & SQS [Электронный ресурс] Режим доступа: https://tsh.io/blog/message-broker/

УДК 004.422

В. Е. Разукрантов (4 курс бакалавриата),
А. В. Петров, ст. преподаватель

СОЗДАНИЕ СЕРВИСА ДЛЯ ПЕРЕХОДА С МОНОЛИТНОЙ АРХИТЕКТУРЫ НА МИКРОСЕРВИСНУЮ

Выбор подходящей архитектуры является основополагающим решением при разработке приложения. От выбора архитектуры зависит его успех при эксплуатации. Микросервисная архитектура представляет собой подход к разработке приложения, при котором используется несколько отдельных независимых компонентов с ограниченным функционалом – микросервисов [1]. Монолитной архитектуре при этом соответствует единая структура приложения, где все компоненты связаны между собой, а клиентский и серверный код являются частью одного процесса.

Существуют ситуации, когда использование монолитной архитектуры является более разумным и оправданным решением. К таким случаям можно отнести следующие: компания является небольшой и команды разработчиков также небольшие; отсутствует возможность тщательной проработки структуры каждого микросервиса; текущая монолитная реализация стабильно функционирует и проблем с выполнением требуемых бизнес-процессов нет [2]. Важно понимать, что использование микросервисов стоит рассматривать не как цель, а как инструмент для улучшения процессов в компании.

Целью данной работы является создание сервиса и внедрение его в процесс оформления полисов ОСАГО компании АО «Совкомбанк Страхование» для перехода с монолитной архитектуры на микросервисную. Также данный сервис должен обеспечить гибкое управление последовательностью вызываемых микросервисов.

Необходимость в данном переходе возникла из-за трудностей при использовании монолитной архитектуры. У компании возникла потребность в масштабировании – добавление нового функционала при использовании монолитной архитектуры означало масштабирование всего приложения, а не его части. Возможность частых релизов отсутствовала: любые изменения могли повлиять на функционал всего приложения и с каждым релизом это влияние становилось все сложнее отслеживать и тестировать. Таким образом, все изменения требовали больших затрат. Одним из основных желаний со стороны компании была возможность гибкого управления процессом при оформлении полиса. Изменение порядка выполнения функционала было практически не осуществимо, поскольку разные части приложения зависели друг от друга. Желание изменить бизнес-процесс требовало серьезных доработок и анализа текущих взаимосвязей. Следовательно, гибкое управление последовательностью выполнения функционала не предоставлялось возможным.

Для решения перечисленных проблем был реализован сервис для разделения монолитного приложения на несколько сервисов, которые могут изменяться и разрабатываться независимо друг от друга.

В качестве языка программирования для создания сервиса по переходу с монолитной архитектуры на микросервисную выбран Java [3]. Данный язык программирования используется практически во всех проектах компании АО «Совкомбанк Страхование». Более того, монолитная архитектура также была написана на нем. Для взаимодействия между создаваемым сервисом и клиентом и между сервисом и микросервисами использовался набор правил API для взаимодействия друг с другом. В качестве протокола был выбран REST API [4], который основан на принципах архитектуры REST и использует стандартные HTTP методы: GET, POST, DELETE, PUT.

В качестве системы управления базами данных (СУБД) выбрана PostgreSQL. PostgreSQL является объектно-реляционной СУБД, что позволяет поддерживать пользовательские объекты, включая типы данных, функции, индексы [5]. В базе хранится следующая информация: данные об используемых микросервисах, критичность и приоритет их выполнения, все вызовы сервисов, их взаимосвязь и порядок выполнения, информация по статусам ответа и по источникам обращения к сервису.

В результате, был разработан сервис, с помощью которого удалось реализовать переход процесса оформления полиса ОСАГО в компании АО «Совкомбанк Страхование» с монолитной архитектуры на микросервисную. Из монолитной архитектуры было выделено 10 независимых микросервисов, которые можно изменять отдельно от всего приложения и независимо от других микросервисов. Часть из выделенных микросервисов представлена на рисунке 1.

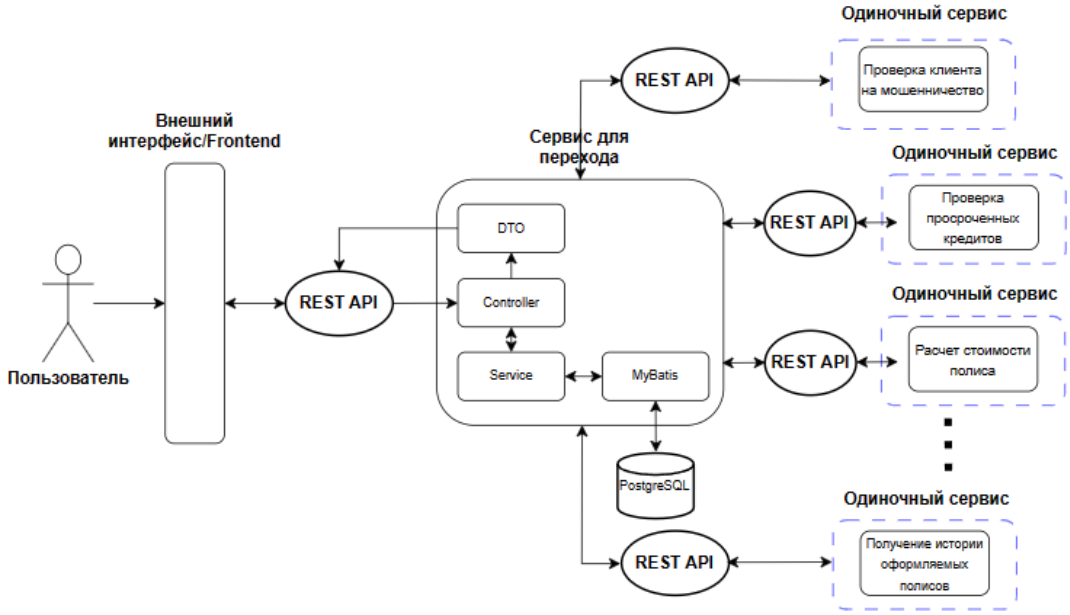


Рисунок 1 – Выделенные микросервисы из монолитной архитектуры

Таким образом, удалось удовлетворить потребность компании в масштабируемости, возможности частых релизов и гибкости при управлении процессом оформления полиса посредством реализации сервиса для перехода на микросервисную архитектуру.

ЛИТЕРАТУРА

1. Кравченко Д. А. Микросервисная архитектура // Технические науки, № 4 (69), 2022. – С. 1-2.
2. Никитин И. В., Гриценко Т. Ю. Сравнение подходов монолитной архитектуры и микросервисной архитектуры при реализации серверной части веб-приложения // Дневник науки, № 3 (39), 2020. – С. 3-8.
3. Java programming language. [Электронный ресурс] Режим доступа: https://dev.java/

4. Никонова Е. З, Королев Р. И. Анализ архитектурного стиля REST API // В сборнике: Современные вопросы устойчивого развития общества в эпоху трансформационных процессов. Сборник материалов VIII Международной научно-практической конференции. Москва, 2023. – С. 176-179.
5. Сорокин В. Е. Об эффективности наследования таблиц в СУБД PostgreSQL // Программные продукты и системы, № 3, 2016. – С. 15-23.

УДК 004.272

М. А. Рукавишников (2 курс магистратуры),
Э. С. Соколова, д.т.н., профессор

АВТОМАТИЗАЦИЯ РАЗВЕРТЫВАНИЯ, СБОРКИ И ТЕСТИРОВАНИЯ ПРИЛОЖЕНИЙ В ЛОКАЛЬНОМ КЛАСТЕРЕ MINIKUBE

Использование облачных технологий крупными IT-компаниями для разворачивания и масштабирования кластеров требует значительных финансовых вложений, связанных с обслуживанием высокопроизводительного оборудования, сложностью конфигурации и настройки развертывания Kubernetes, разворачиванием тестового стенда для ежедневного использования командами при разработке новых возможностей продукта. В то же время активный переход на виртуализацию, с изолированием приложений между виртуальными машинами позволяет снизить затраты, увеличить производительность, эффективность, повысить уровень безопасности системы.

Анализ работ показал отсутствие автоматизированных инструментов, покрывающих функционал для разработки приложений на локальных машинах пользователей. Документация Kubernetes предлагает Minikube в качестве инструмента для установки на собственную машину [1, 2], при этом разработчик, не имеющий экспертизы в области DevOps не сможет установить и настроить все необходимые компоненты, реализовать CI/CD-процесс. Идея решения состоит в том, чтобы автоматизировать реализацию CI/CD процесса [3], настроить его запуск по изменениям в ветке репозитория для надежного развертывания изменений программного обеспечения с использованием инструментария Gitlab CI, устанавливающего на виртуальную машину Minikube и все необходимые компоненты, запускающего сборку приложения и тесты на локальном стенде. Создание такого универсального инструмента с простым и интуитивно понятным механизмом значительно облегчит и ускорит процесс разработки программных продуктов, минимизирует затраты разработчиков на сборку, тестирование и развертывание приложений. Автоматическая сборка и тестирование минимизируют возможные ошибки и проблемы, которые возникают из-за человеческого фактора [4]. В качестве инструмента сборки выбран Docker, оркестрация контейнеров на виртуальных машинах реализована за счет инструмента Minikube. Процессы сборки, тестирования и разворачивания настроены с помощью Gitlab CI-конвейеров.

Архитектура разработанной системы автоматизации процессов разработки приложений включает набор существующих технологий для создания новых продуктов (рисунок 1). Ядром архитектуры является Minikube, инструмент, позволяющий легко запускать одноузловой кластер Kubernetes внутри виртуальной машины. Архитектура разделена на две среды: среду разработки (слева) и виртуальную машину, на которой происходит запуск приложений (справа). Среда разработки включает инструмент Gitlab CI, запускающий сборку исходного кода, тесты и линтеры, затем собранный docker-образ сохраняется в Image Registry – хранилище образов [5]. На виртуальную или локальную машину разработчика с помощью bash-скрипта устанавливаются все необходимые компоненты для запуска его приложений [6]. Внутри Minikube содержатся средства логирования и мониторинга, которые также устанавливаются при запуске скрипта, а также реализуется внешний доступ (ingress) для просмотра интерфейса приложений с любого устройства. Само приложение разворачивается в Minikube с помощью docker-образа, который уже собран и хранится в Image Registry.

Для оценки эффективности разработанной системы проведено комплексное исследование [7]. Оценка общей производительности выполнена с помощью инструментов Pytest и Junit [5]. Сравнительный анализ качества кода осуществлялся с использованием метрик – цикломатической сложности (использовались статические анализаторы кода SonarQube и pylint), покрытия кода тестами (инструменты тестирования кода Pytest, Junit) и уровня соблюдения стандартов кодирования.

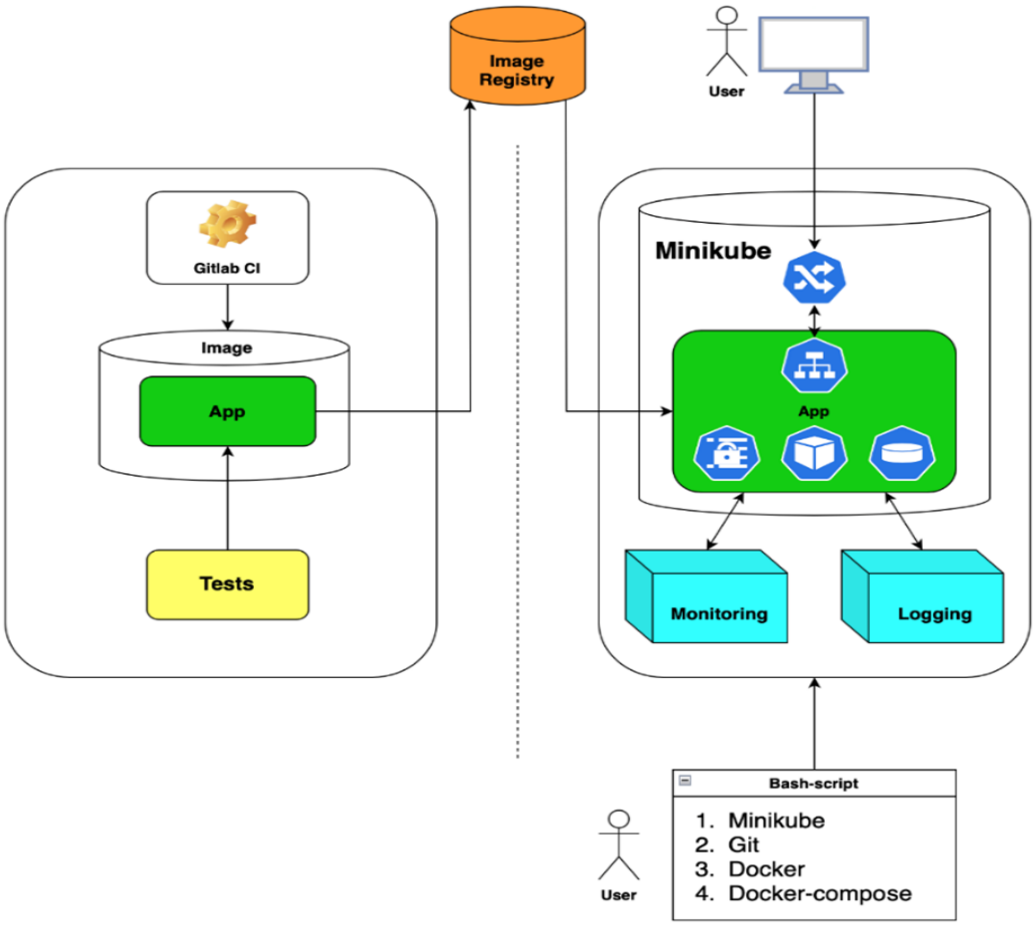


Рисунок 1 – Архитектура автоматизированной системы разработки приложений

Разработанный инструмент сборки, тестирования и развертывания приложения инициируется одним нажатием кнопки, что позволит разработчикам сосредоточиться на творческих, сложных задачах, повышая общую производительность команды. Разработанная система показала хорошую эффективность при ее сравнении с системой разработки и отладки продуктов на кластерах облачной инфраструктуры. Внедрение автоматизированного процесса развертывания на локальном кластере создает предсказуемую и изолированную среду для разработчиков, что снижает возможность конфликтов и ошибок, связанных с различиями в окружениях. Дальнейшие исследования могут включать в себя расширение функциональности продукта и его адаптацию для различных сценариев разработки программного обеспечения

ЛИТЕРАТУРА

1. Работа с контейнерами в Minikube // Хабр URL: https://habr.com/ru/companies/otus/articles/714612/ (дата обращения: 07.03.2024).
2. Что такое Minikube // Cloud4y URL: https://www.cloud4y.ru/blog/what-is-minikube/ (дата обращения: 22.01.2024).
3. Когда и зачем нужен CI/CD // Хабр URL: https://habr.com/ru/companies/slurm/articles/649027/ (дата обращения: 17.02.2024).

4. Joseph T. Automation Testing Technologies: What’s Trending in 2020 // QASource. 2020. URL: https://blog.qasource.com/automation-testing-technologies-whats-trending-in-2020 (дата обращения 26.01.2024).
5. Полное практическое руководство по Docker: с нуля до кластера на AWS [Электронный ресурс] // Хабр: сайт – URL: https://habr.com/ru/post/310460/ (дата обращения: 03.02.2024).
6. Валади, Дж. 100% самоучитель Linux / Дж. Валади. - М.: Технолоджи-3000, 2006. - 336 с.
7. О линтерах, качестве кода, качестве вообще и управлении качеством // Хабр URL: https://habr.com/ru/articles/440414/ (дата обращения: 14.02.2024).

УДК 004.4

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

РАЗРАБОТКА ПРОГРАММНЫХ СРЕДСТВ ПОДУРОВНЯ АДАПТАЦИИ ПРОТОКОЛА GEONETWORKING К IPV6 ДЛЯ ИНТЕЛЛЕКТУАЛЬНОЙ ТРАНСПОРТНОЙ СИСТЕМЫ

Интеллектуальные транспортные системы (ИТС) решают задачи повышения безопасности движения, а также задачи управления и мониторинга транспортных потоков на дорогах общего пользования. ИТС включают транспортные средства с бортовыми устройствами (OBU – on board unit) и стационарные средства связи, установленные вдоль дорог (RSU – road side unit). Все устройства могут обладать функциональностью маршрутизаторов в беспроводной ah-hoc сети, в которой для маршрутизации сообщений используется географическое положение устройств (GeoNetworking) с идентификацией по MAC адресам сетевых контроллеров [1]. Протокол GeoNetworking удовлетворяет требованиям служб ИТС, область применения которых ограничена сетями, отключенными от Интернета. Однако некоторые приложения ИТС требуют интеграции станций ИТС с более крупными сетями, такими как частные транспортные сети или Интернет. Чтобы соединить сети, основанные на GeoNetworking, с сетями, использующими Интернет-протокол (IP), необходимо позволить станциям GeoNetworking ИТС действовать как интернет-хосты или маршрутизаторы [2]. При этом достигаются следующие два основных преимущества: покрытие, предлагаемое точками подключения к Интернету, такими как RSU, расширяется за счет географической маршрутизации; и многоадресный трафик IPv6 может быть адресован и доставлен всем станциям ИТС, расположенным в данный момент в пределах географической области.

Целью данной работы является разработка программных средств подуровня адаптации протокола Geonetworking к протоколу IPv6 для придорожной станции RSU интеллектуальной транспортной системы.

Для достижения этой цели необходимо решение следующих задач:

1) Разработка архитектуры ПО на базе стандартов ETSI.

2) Разработка алгоритма работы с виртуальным интерфейсом TUN/TAP

3) Разработка программных средств сервиса GN6ASL

4) Провести тестирование разработанных программных средств

Разрабатываемые программные средства необходимы для распространения предупреждений или общих информационных сообщений в географически ограниченных районах, используя пересылки сообщений между устройствами OBU и RSU.

Передача пакетов IPv6 по протоколу GeoNetworking [3] происходит через подуровень адаптации протокола, называемый GN6ASL. В результате подключение, обеспечиваемое точками подключения к сетям IPv6, расширяется с помощью мобильных ретрансляционных узлов. Методы данного слоя должны позволять осуществлять геокастинг (передача пакетов в зависимости от географического местоположения) многоадресных пакетов IPv6.

Разрабатываемые средства должны обеспечить:

1) Обмен пакетами IPv6 OBU со станциями RSU.

2) Обмен данными с произвольным хостом Интернета.

3) Выполнять операции для мобильной маршрутизации.

Архитектура разрабатываемых программных средств представлена на рисунке 1. Подуровень GN6ASL использует виртуальные сетевые устройства типа TAP для взаимодействия с сервисами других подуровней с использованием универсального драйвера TUN/TAP для Linux [4]. Программные средства GN6ASL реализуют протокол обнаружения соседей (ND), которая включает в себя такие функции, как обнаружение маршрутизатора.

Для уровня IPv6+Mobility Extensions подуровень GN6ASL представлен протоколом канального уровня на основе протокола GeoNetworking. Обеспечивается связь типа «точка-точка», «точка-многоточка», а также адресация с географическим охватом, включая GeoAnycast и GeoBroadcast. На рисунке 2 приведена схема адресации на каждом уровне в архитектуре системы. Для идентификации физического интерфейса связи используется 48- разрядный MAC-адрес.

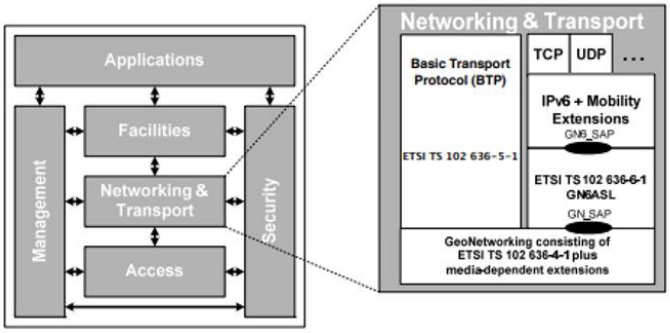


Рисунок 1 – Архитектура системы

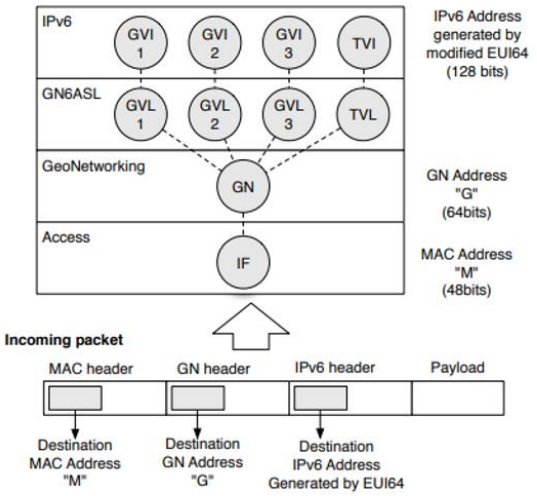


Рисунок 2 – Схема адресации

В реализации GN6ASL предусмотрены следующие каналы: географический виртуальный канал (GVL) (локальный виртуальный канал с возможностью многоадресной рассылки, охватывающий несколько физических каналов в пределах географических границ зоны обслуживания), а также топологический виртуальный канал (TVL) (локальный виртуальный канал с возможностью многоадресной рассылки, охватывающий несколько физических каналов в пределах топологических границ).

Разработка программных средств осуществляется на языке программирования C/C++ на базе стандарта POSIX [5] для операционной системы Ubuntu 20.04.

Разработанные программные средства позволят распространять информацию о дорожных происшествиях в географически отдаленных от RSU местах, что позволит другим водителям знать состояние трафика заранее.

ЛИТЕРАТУРА

1. Ткачук А. С., Медведев Б. М. Модификация стека V2X для одновременной работы с радиорежимами ITS G5 и LTE-CV2X. Современные технологии в теории и практике программирования : сборник материалов научно-практической конференции студентов, аспирантов и молодых ученых, 26–27 апреля 2023 г. – СПб. : ПОЛИТЕХ-ПРЕСС, 2023.
2. ETSI EN 302 636-6-1 V1.2.1 (2014-05) Intelligent Transport Systems (ITS); Vehicular Communications; GeoNetworking; Part 6: Internet Integration; Sub-part 1: Transmission of IPv6 Packets over GeoNetworking Protocols
3. ETSI - EN 302 636-4-1 Intelligent Transport Systems (ITS); Vehicular Communications; GeoNetworking; Part 4: Geographical addressing and forwarding for point-to-point and point-to-multipoint communications; Sub-part 1: Media-Independent Functionality

4. Universal TUN/TAP device driver [Электронный ресурс] https://docs.kernel.org/networking/tuntap.html
5. У. Р. Стивенс, Б. Феннер, Э. М. Рудофф. UNIX. Разработка сетевых приложений, 3-е издание.

УДК 004.415.2

Н. А. Созыкин (2 курс магистратуры),
Б. М. Медведев., к.т.н., доцент

РАЗРАБОТКА ПРОГРАММНЫХ СРЕДСТВ ТЕСТИРОВАНИЯ РАБОТОСПОСОБНОСТИ ОБОРУДОВАНИЯ УСТРОЙСТВ ЦИФРОВОЙ ОБРАБОТКИ СИГНАЛОВ

С ростом сложности цифровых устройств обработки сигналов и повышением требований к их надежности и эффективности, необходимость в развитых средствах тестирования становится все более актуальной [1]. Программные средства тестирования не только обеспечивают проверку функциональности оборудования, но и выявляют ошибки на этапах верификации дизайна и серийного производства продукции [2].

В качестве аппаратной платформы для разработки новых устройств цифровой обработки сигналов целесообразно использовать системы на кристалле на базе многоядерной ARM архитектуры [3], которые содержат интерфейсные модули для взаимодействия с компонентами системы, такие как USB, SATA, PCIe, Ethernet, а также специализированные интерфейсы, например JESD204, для подключения высокоскоростных цифро-аналоговых и аналогово-цифровых преобразователей. В процессе разработки новых устройств необходимо создать и встраиваемые программные средства тестирования модулей и интерфейсов этого оборудования.

Таким образом, целью работы является создание программных средств, обеспечивающих определение работоспособности модулей аппаратной платформы и диагностику ошибок.

Для достижения этой цели необходимо решение следующих задач:

– Обзор существующих подходов к реализации тестирования работоспособности оборудования устройств цифровой обработки сигналов.

– Разработка сценариев тестирования и протокола управления с применением сетевого протокола Telnet.

– Реализация графического интерфейса пользователя на C++ с применением фреймворка QT 5.

– Тестирование разработанных программных средств.

Структурная схема разрабатываемой системы тестирования представлена на рисунке 1.

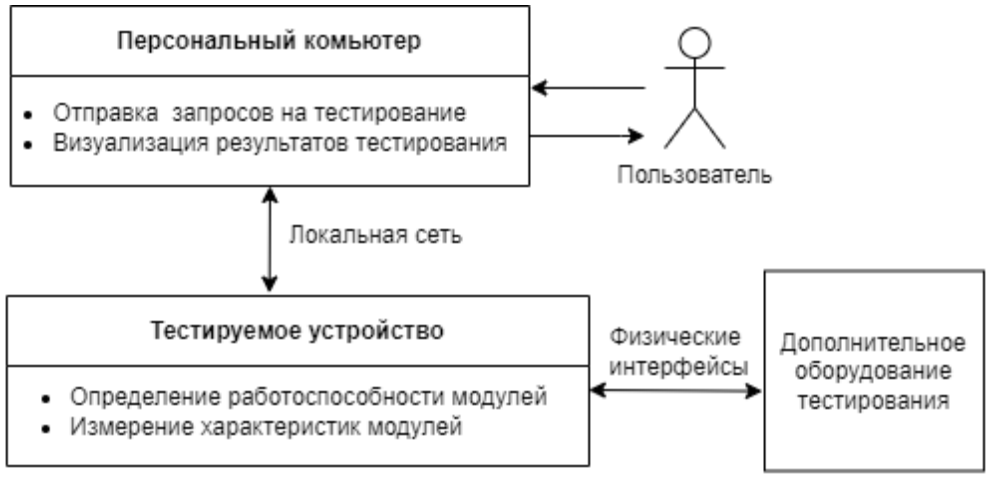


Рисунок 1 – Структурная схема системы тестирования

Разрабатываемые программные средства состоят из двух компонентов:

– пользовательский интерфейс для взаимодействия с тестируемым устройством запускается на ПК пользователя, так как не ко всем тестируемым устройствам могут быть подключены средства для взаимодействия с человеком;

– встраиваемые программные средства тестирования, поддерживающие удаленное тестирование через локальную сеть.

Разработаны требования к протоколу управления и сценарии тестирования, позволяющие:

– однократно или циклически выполнять тесты модулей оборудования;

– следить за состоянием процессов;

– проводить нагрузочное тестирование, включая параллельное выполнение тестов для оценки взаимного влияния и проверки встроенных модулей питания при большой нагрузке.

Разработан протокол дистанционного управления тестированием с использованием протоколов UDP и TelNet [4]. Разработаны встраиваемые программные средства тестирования следующих интерфейсов: USB, Ethernet, HDMI, COM-порт, Bluetooth.

Для запуска тестов, просмотра журналов тестов и данных разработано кроссплатформенное приложение на языке программирования C++ [5] и фреймворке Qt5 [6], которое возможно запускать на различных платформах, таких как Windows и Linux [7]. Предусмотрена возможность формирования временных и спектральных диаграмм для измерения характеристик аналого-цифрового преобразователя.

Разработанные программные средства могут быть использованы в составе системы тестирования новых аппаратных платформ цифровой обработки сигналов, построенных на базе AMD Ultrascale+.

ЛИТЕРАТУРА

1. G. Papadimitriou, et al., “Exceeding Conservative Limits: A Consolidated Analysis on Modern Hardware Margins,” IEEE Transactions on Device and Materials Reliability, vol. 20, no. 2, pp. 341-350, 2020.
2. В. В. Гаврилова, Б. М. Медведев. Разработка программных средств тестирования встраиваемых систем обработки информации. Современные технологии в теории и практике программирования: сборник материалов научно-практической конференции студентов, аспирантов и молодых ученых, 26–27 апреля 2023 г. – СПб. : ПОЛИТЕХ-ПРЕСС, 2023.
3. Калачев Александр Мультиядерные процессоры ARM-архитектуры // Компоненты и Технологии. 2010. №104. URL: https://cyberleninka.ru/article/n/multiyadernye-protsessory-arm-arhitektury
4. Telnet [Электронный ресурс] Режим доступа: https://www.telnet.org/
5. C++ [Электронный ресурс] Режим доступа: https://cplusplus.com/
6. Qt5. [Электронный ресурс] Режим доступа: https://doc.qt.io/qt-5/qt5-intro.html
7. Integra Sources [Электронный ресурс] https://www.integrasources.com/blog/qt-c-embedded-development-pros-cons-alternatives/

УДК 004.896

А. А. Степанов (4 курс бакалавриата),
О. Г. Малеев, доцент

ИСПОЛЬЗОВАНИЕ СВЕРТОЧНЫХ НЕЙРОННЫХ СЕТЕЙ ДЛЯ КЛАССИФИКАЦИИ ИЗОБРАЖЕНИЙ ЗЕРНА

Классификация зерен сейчас как никогда актуальна в агрокультуре. Производители зерна хотят знать, что конкретно и в каком процентном соотношении они вырастили. Покупатели в свою очередь тоже хотят знать качество покупаемого зерна. Однако на сегодняшний день процесс классификации практически не автоматизирован: зерна перебираются вручную с увеличительным стеклом специально обученными людьми.

В связи с этим компания ООО «ЭКАН» создала прибор «Анализатор САПФИР» [1], который пропускает через себя зерна, делает их изображения, а затем с помощью нейронных сетей определяет их категории. «САПФИР» значительно ускоряет процесс классификации и избавляет от необходимости услуг экспертов по зерну. Так как сверточные нейронные сети являются одним из самых мощных инструментов при работе с классификацией изображений, было принято решение остановиться на них. Таким образом, целью данной работы является обучение сверточной нейронной сети для классификации изображений зерна для ее интеграции в «САПФИР».

Задача классификации зерен с помощью сверточной нейронной сети уже рассматривалась бангладешскими учеными [2]. В данной работе предложена сверточная нейронная сеть для классификации 5 бангладешских зерновых культур. При сборе данных для обучения авторы отдельно снимали каждое зерно. Всего было собрано по 250 изображений объектов из каждой категории. В результате была получена сеть с точностью 87-89% на тренировочной выборке и 90-93% на тестовой. Подход в данной работе был наиболее близок к условиям «САПФИРА» с учетом того, что здесь идет классификация каждого отдельного зерна. Однако с использованием «САПФИРА» можно собрать датасет значительно больших размеров, так как он автоматически снимает изображения всех зерен, проходящих через него. Как следствие, с увеличением датасета ожидается увеличение качества классификации.

Была рассмотрена еще одна работа, где раскрывалась заданная проблема. [3]. Здесь авторы рассматривали классификацию кучек различного зерна. Однако, с учетом того, что многим заказчикам хочется знать процентное соотношение той или иной категории в общей куче, такой подход при решении заданной проблемы не рассматривался.

На рисунке 1 представлена предложенная для решения рассматриваемой задачи архитектура нейронной сети.

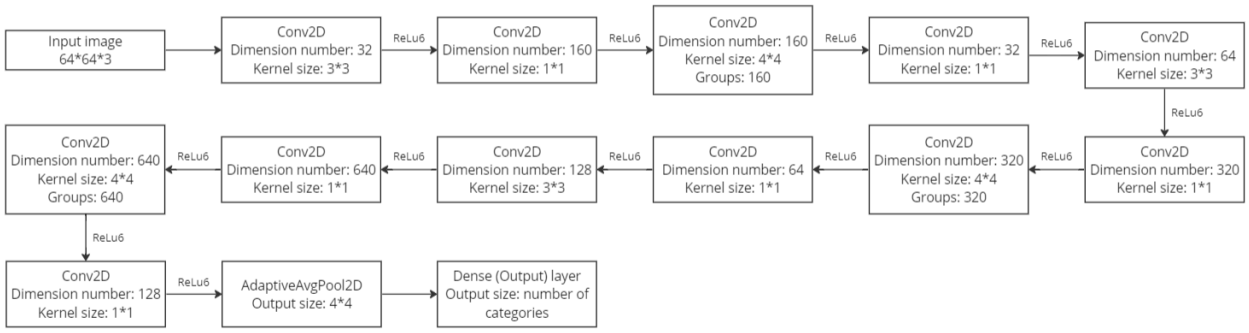


Рисунок 1 – Архитектура нейронной сети

Для заявленных 16 категорий, в общей сумме составляющих около 1000000 изображений, точность на тренировочной выборке составила около 92%, на валидационной около 84% и на тестовой тоже около 84%. Малая точность классификации была обусловлена тем, что многие заявленные категории входили в одну большую категорию. Например, «Пшеница проросшая» и «Пшеница, больная альтернариозом», входят в общую категорию «Пшеница». Более того, некоторые категории можно определить только по изображению конкретной стороны зерна, и может возникнуть ситуация, когда «САПФИР» сделает изображение больного зерна со здоровой стороны и пометит его, как больное зерно.

В связи с этим для изучения классификации больших категорий малые подкатегории были сгруппированы в большие, и для них была обучена новая нейронная сеть на той же архитектуре. В результате точность на тренировочной выборке составила 96.9%, на валидационной 96.8% и на тестовой 96.45%, что является значительно лучшим результатом. На рисунке 2 представлена матрица ошибок нейронной сети для тестовой выборки. На ней в строках располагаются классифицируемые категории, а в столбцах категории, в которые изображения из категорий строк были определены.

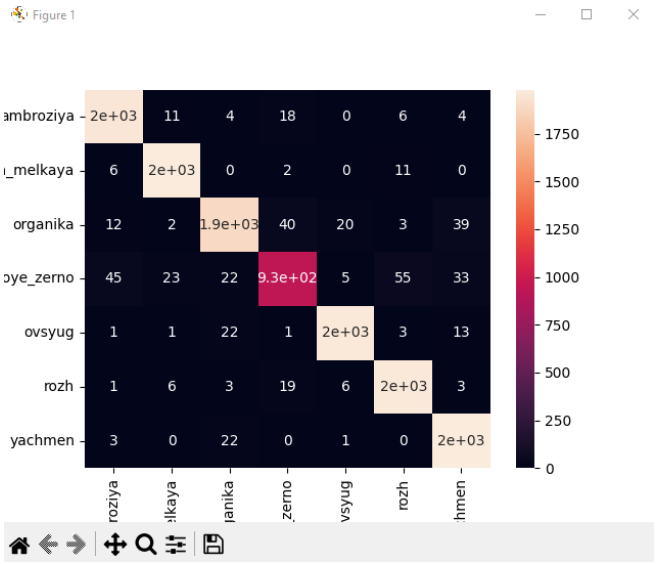


Рисунок 2 – Матрица ошибок нейронной сети для тестовой выборки

Таким образом, было выявлено, что с большим объемом датасета и использованием сверточных нейронных сетей можно достичь большой точности классификации различных видов зерна. В дальнейшем планируется разработать подход, с помощью которого с таким же успехом можно будет качественно классифицировать подкатегории больших категорий.

Полученная нейронная сеть была реализована на языке Python [4], так как на нем реализованы мощные библиотеки для работы с данными и машинным обучением, с использованием фреймворка PyTorch [5]. Выбор данного фреймворка обусловлен тем, что код «САПФИРА» написан на языке C++, и у PyTorch существует расширение libtorch, позволяющее запускать обученные с помощью PyTorch нейронные сети в коде C++.

ЛИТЕРАТУРА

1. Анализатор САПФИР. [Электронный ресурс] Режим доступа: https://ekan.spb.ru/produktsiya/pribory-ekspress-analiza/analizator-sapfir
2. A Robust Deep Learning Segmentation and Identification Approach of Different Bangladeshi Plant Seeds Using CNN. [Электронный ресурс] Режим доступа: https://www.researchgate.net/publication/344693609_A_Robust_Deep_Learning_Segmentation_and_Identification_Approach_of_Different_Bangladeshi_Plant_Seeds_Using_CNN
3. A Convolution Neural Network-Based Seed Classification System. [Электронный ресурс] Режим доступа: https://www.mdpi.com/2073-8994/12/12/2018
4. Python. [Электронный ресурс] Режим доступа: https://www.python.org/
5. PyTorch. [Электронный ресурс] Режим доступа: https://pytorch.org/

УДК 004.054

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

РАЗРАБОТКА ПРОЕКТА АВТОМАТИЗИРОВАННЫХ ТЕСТОВ ДЛЯ САЙТА СОЦИАЛЬНОЙ СЕТИ «ОДНОКЛАССНИКИ»

За последние десятилетия IT-сфера пережила небывалый рост. Выпускаемые программные продукты становятся все масштабнее и сложнее, в том числе и веб-приложения. Из-за повсеместного распространения сети «Интернет» все большее количество людей получают доступ к ним доступ. Компании должны более тщательно следить за качеством выпускаемых приложений. Одним из инструментов для этого является автоматизированное тестирования.

Внедрение и развитие данного рода тестирования требует больших вложений, но в

перспективе оно дает преимущества перед аналогичной ручной работой: высокая скорость проверок, расширения тестового покрытия, выполнение без участия наблюдателя, меньшие затраты на поддержку, независимость результатов, уменьшение влияния человеческого фактора, формирование автоматических отчетов и так далее [1].

Одним из наиболее распространенных языков программирования для автоматизации тестирования является Java [2], который имеет множество инструментов в этой сфере. Самый популярный тестовый фреймворк – JUnit, а именно его последняя версия JUnit 5 [3][4]. Более всего интересует новый внедренный во фреймворк инструмент - Extensions, который позволяет управлять жизненным циклом тестов [5]. Необходимо реализовать взаимодействие с браузером посредством веб-драйвера, в этом поможет Selenium [6].

Таким образом, целью работы является реализация проекта автоматизированных тестов для сайта социальной сети «Одноклассники» на основе тестового фреймворка JUnit 5 и инструмента для взаимодействия с браузером Selenium, используя современные решения в вопросах построения архитектуры проектов такого типа.

Для выполнения поставленной цели будет необходимо решить следующие задачи

1. Обзор существующих архитектурных решений.
2. Проведение анализа найденных подходов и сравнение их между собой.
3. Реализация выбранных решений, позволяющих значительно упросить работу с проектом, расширить количество покрываемых тестовых сценариев, увеличить стабильность, скорость и независимость тестов.
4. Демонстрация полученных результатов

Одним из главных вопросов при построении проекта автоматизированных тестов является выбор стратегии работы со страницами веб-приложения и элементами на ней. Мной был выбран принцип DRY, заключающийся в том, что каждый фрагмент логики, каждая функция приложения и элемент кодируются единожды [7]. Самые распространённые представители данной архитектуры – это паттерны Page Object, Page Element, Factory, Decorator, Value Object. Помимо них в проекте реализуются паттерны Promise, Object Pool, Data Registry, Matchers, Builder, Chain of Invocation, Loadable Component. Все они позволяются значительно упростить работу с автоматизированными тестами (сделать более понятную структура проекта), расширить тестовое покрытие и ускорить разработку. Помимо перечисленного для достижения цели данной работы необходима интеграция API социальной сети в разрабатываемый проект. Это уменьшит время прохождения тестов за счет более быстрой подготовки начальных условий до запуска основного сценария.

Проблемы нестабильности тестов, отсутствия независимости их друг от друга и нехватка возможностей проекта для увеличения покрытия автоматизированными тестами продукта смогут решить следующие инструменты [8]. Внедрение внешних сервисов для генерации и хранения используемых в тестах данных и файлов (фото, видео, пароли, строки и иное) через Dependency Injections средствами Spring. Данный шаг увеличивает возможные сценария покрытия, повышает гибкость и скорость написания кода. Это происходит из-за разграничения мест хранения кодовой базы и данных. Таким образом, нет необходимости проходить все этапы ревью кода, чтобы поменять определенные начальные условия для теста.

Чтобы ещё больше повысить независимость автоматизированных тестов необходимо исключить использование одних и тех же пользователей (ботов). Эта проблема стоит остро, так как почти в каждом тесте необходимо проходить аутентификацию. Динамическое создание ботов практически полностью помогает в данной ситуации, но сильно нагружает инфраструктуру и инструмент для их создания, так как ежедневно тысячи тестов запускаются множество раз. Решением является хранилище статических пользователей, из которого в начале каждого тестового сценария берется случайный бот, а в конце происходит его очистка через API от всевозможных изменений, которые с ним могли случиться. Таким образом, значительно повышается атомарность тестов, а также появляется возможность их параллельного запуска, что уменьшает время прохождения.

На момент начала работы уже существовал некий проект автоматизированного

тестирования, в который и проводилось внедрение всего вышеописанного. Поэтому одной из задачей является взаимодействие со множеством легаси-кода. В этот список входят устаревшие библиотеки, самописные обертки, статические данные, дублирующие функции друг друга классы и так далее. Есть три стратегии по работе с таким типом кода: единоразовое удаление (малое количество использований), пометка его как устаревшего и удаление в течении нескольких итераций (среднее количество использований), помечание как устаревшего, информирование всех заинтересованных о переходе на новую технологию и глобальная длительная реструктуризации той части проекта, которую задевает удаляемый код (большое количество использований).

В ходе выполнения работы удалось перевести проект автоматизированного тестирования на более новые технологии с применением современных архитектурных решений. Таким образом, появились возможности для увеличения тестового покрытия специфическими и сложными сценариями, уменьшилось время прохождения тестов, повысились их атомарность, стабильность, качество и простота восприятия.

ЛИТЕРАТУРА

1. Берегейко О. П., Дубовский А. С. Автоматизация тестирования веб-приложений // Научный журнал “Вестник магистратуры” 2016 12–4 (63), Йошкар-Ола, 15 декабря 2016 года / ООО “Коллоквиум” – с. 39–41.
2. Java | Oracle [Электронный ресурс] Режим доступа: https://www.java.com/ru/
3. JUnit 5 [Электронный ресурс] Режим доступа: https://junit.org/junit5/
4. Boni Garcia. Mastering Software Testing with JUnit 5: Comprehensive guide to develop high quality Java applications. – M: Packt Publishing, 2017. – 350 с.: ил. – ISBN 978-1787285736
5. Boni García, Carlos Delgado Kloos, Carlos Alario-Hoyos, Mario Munoz-Organero. Selenium-Jupiter: A JUnit 5 extentions for Selenium WebDriver // Journal of Systems and Software.– 2022. – Vol. 189. - Article 111298
6. Ю.В. Балашова. Автоматизированное тестирование веб-приложений: роль Selenium в инновационных подходах // Научный журнал “ Вестник приднестровского университета. Серия: Физико-математические и технические науки. Экономика и управление”, Преднестровье, 2023 / Преднестровский государственный университет им Т. Г. Шевченко – с. 217–226.
7. Л. А. Сазанова. Паттерны тестирования как инструмент разработки программного обеспечения // Журнал “Вестник Воронежского института высоких технологий”, Екатеринбург, 2022 / Уральский государственный экономический университет – с. 113–118.
8. С. С. Кириллов. Решение проблемы нестабильности автоматизированных тестов пользовательского интерфейса web-приложений // Журнал “Современные наукоемкие технологии”, 2022 / ООО “Издательский дом ‘Академия естествознания’” – с. 23–28.

УДК 004.891

О. М. Сысоева (2 курс магистратуры),
Т. В. Леонтьева, к.т.н., доцент

АЛГОРИТМ КЛАССИФИКАЦИИ ТРАНСПОРТНЫХ СРЕДСТВ С ИСПОЛЬЗОВАНИЕМ ДАННЫХ НОСИМЫХ УСТРОЙСТВ

В современных городах городская мобильность играет ключевую роль, оказывая влияние на развитие и функционирование городской инфраструктуры. Изучение данных об использовании различных видов транспортного средства (ТС) становится ключевым аспектом исследований в областях городской мобильности и обеспечения безопасности на дорогах. Важность определения вида ТС также проявляется в системах "умного страхования". Приложения автоматически регистрируют поездки пользователей при обнаружении движения. Распознавание вида ТС позволит отсечь неавтомобильные поездки, такие как пешеходные прогулки или поездки на велосипеде, что поможет улучшить аналитику и оценку рисков и повысит эффективность и надежность таких систем.

Большинство существующих решений используют данные с датчиков мобильных устройств, которые позволяют точно определить вид ТС. Они опираются на различные методы, такие как использование акселерометра и алгоритма AdaBoost [3], случайного леса и анализа показателей гироскопа и статистических характеристик ускорения [4]. Например, была предложена трехступенчатая система классификации, которая сначала определяет тип движения (пешком/в покое), затем определяет находится ли пользователь неподвижно или в транспорте, и далее определяет вида ТС. Точность моделей составляет около 85%.

Для реализации нового подхода классификации вида транспортного средства необходимо использовать иерархическую систему классификации, то есть перехода от общих классов к частным.

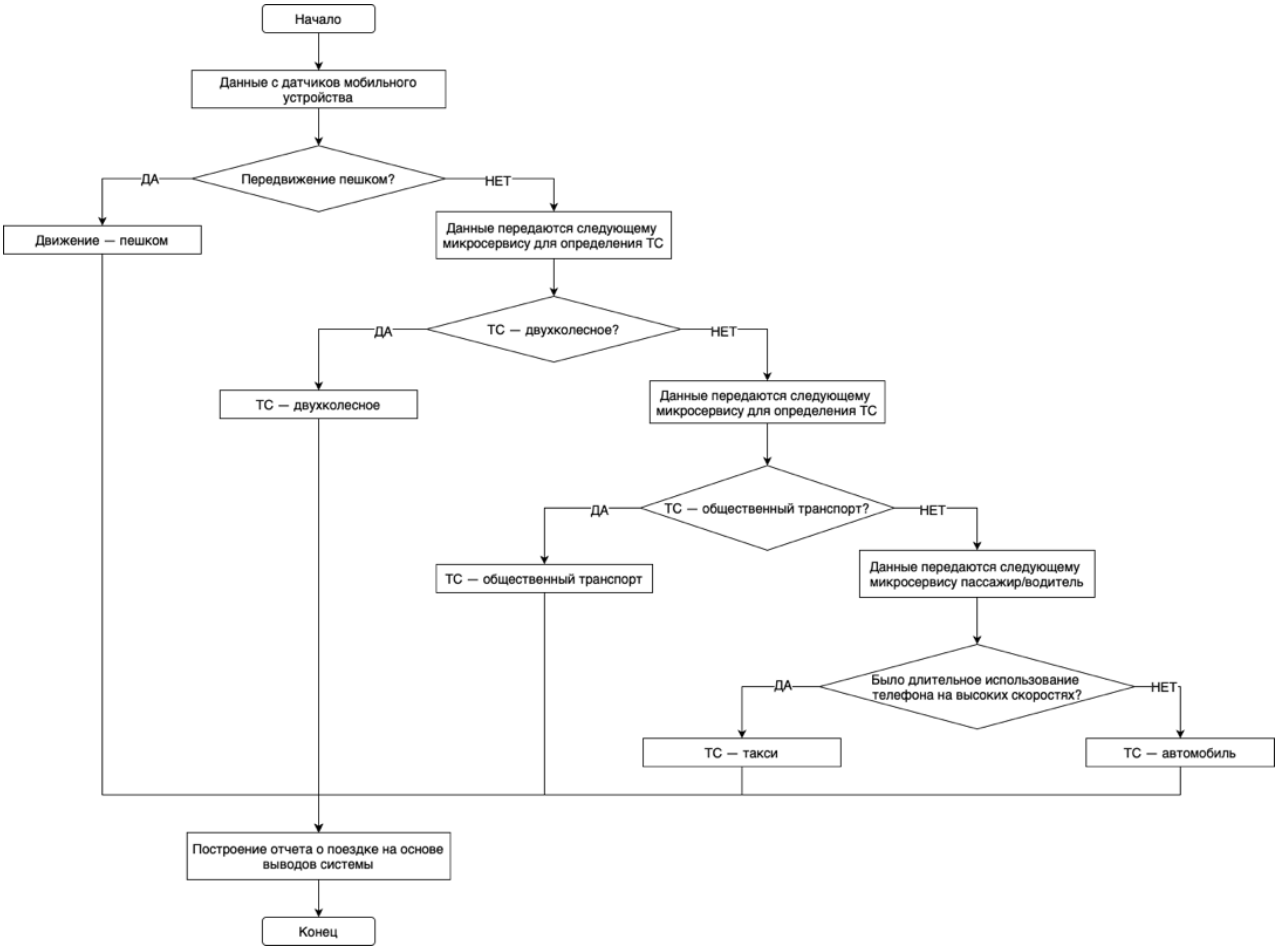


Рисунок 1 – Алгоритм иерархической классификации.

Предлагаемое решение включает в себя создание алгоритма трехступенчатой классификацией:

1. алгоритм определения вида передвижения (пешком/с использованием ТС);
2. алгоритм определения вида ТС:
   – классификация четырехколесное/двухколесное;
   – классификация автомобиль/общественный транспорт;
3. алгоритм определения пассажир (такси) или водитель.

Преимущество решения состоит в разнообразии определяемых видов транспортов и возможности внедрения других алгоритмов, которые могут улучшить алгоритм классификации ТС, например, разделение автомобилей на личные и коммерческие.

Для определения вида передвижения и классификации четырехколесного и двухколесного ТС используются данные акселерометра и метод машинного обучения — случайный лес (Random Forest) [2]. В результате обученные модели были протестированы, и доля правильных предсказаний моделей оказалась равной ~96 %.

При проведении работ по классификации вида транспорта на классы четырехколесные и двухколесные было обнаружено, что использование данных с датчиков мобильного устройства недостаточно для более детальной классификации ТС. Алгоритмы машинного обучения по данным акселерометра плохо разделяют похожие между собой виды транспорта. Поэтому определение общественного вида транспорта реализовано по GPS данным, то есть обнаружение остановок транспорта во время поездки около автобусных/троллейбусных остановок. Аналогично общественному транспорту реализована классификация поездок на поездах. Информация о местоположении остановок общественного транспорта и железнодорожных станций была получена из открытого источника — OpenStreetMap [5]. Среднее значение точности модели по определению общественного транспорта составила 93%, по определению поездов — 90%.

Алгоритм определения пассажир или водитель основывается на модели определения использования мобильного устройства. Данная модель обучена на кватернионах с помощью метода опорных векторов и доля правильных предсказаний модели оказалась равной ~95 % [1]. Подразумевается, что водитель не часто использует телефон во время езды за рулем, а также у него нет возможности использовать его длительное время и на большой скорости. Поэтому с помощью установления необходимых порогов по использованию телефона во время поездки производится последний этап алгоритма классификации ТС.

Разработанный алгоритм классификации ТС с использованием методов машинного обучения обеспечивает определение вида ТС и статуса пассажира или водителя. Полученные результаты показывают высокую точность классификации. В дальнейшем будут улучшены модель определения общественного транспорта путем проверки начальных и конечных точек на наличие остановок, а также модель определения поездов с помощью фильтрации данных GPS.

ЛИТЕРАТУРА

1. Сысоева О.М., Селин И.А. Разработка алгоритма детектирования использования мобильного устройства в автомобиле с помощью данных акселерометра мобильного устройства // Современные технологии в теории и практике программирования: сборник материалов конференции, 26 апреля 2022 года. - СПб: ПОЛИТЕХ-ПРЕСС, 2022. - С. 179-180.
2. Машинное обучение // Home page of Lev V. Utkin URL: https://levutkin.github.io/teaching/machine-learning (дата обращения: 17.03.2024).
3. Hemminki S., Nurmi P., Tarkoma S.: "Accelerometer- based transportation mode detection on smartphones SenSys 2013 Conf., Roma (Italy), 11-15 Nov. 2013.
4. Shafque M.A., Hato E.: Travel mode detection with varying smartphone data collection frequencies. Sensors, 16(5), pp. 716-739, 2016.
5. OpenStreetMap URL: https://www.openstreetmap.org (дата обращения: 17.03.2024).

УДК 004.62

Б. Т. Талипова (4 курс бакалавриата),
А. В. Самочадин, к.т.н., доцент

РАЗРАБОТКА И ВНЕДРЕНИЕ СИСТЕМЫ ОВЕРБУКИНГА С ПРИМЕНЕНИЕМ МЕТОДОВ МАШИННОГО ОБУЧЕНИЯ

Сейчас неотъемлемой частью привлечения клиентов, в частности в онлайн школы, стали бесплатные вводные мастер-классы. Однако, отделы продаж сталкиваются с такой проблемой, как неявка записанных учеников на занятие. Это приводит к простою методистов, повышению издержек и потере потенциальных клиентов.

В рамках работы эта проблема будет решаться с помощью создания системы овербукинга [2], то есть «сверхбронирования».

Овербукинг - стратегия сбыта товаров или услуг, при которой поставщик принимает на себя больше обязательств по поставке товара или предоставлению услуг, чем может

выполнить, в расчете на то, что не все из взятых обязательств действительно придется выполнять.

В контексте онлайн-школы система овербукинга направлена на решение следующих проблем:

- Снижение издержек
- Увеличение количества продаж
- Улучшение опыта клиентов и методистов

Компания ранее предпринимала попытки внедрения овербукинга на примере другой школы [1], записывая по различной логике больше клиентов. Попытки не решали проблему в полной мере. Они не учитывали сезонность, особые признаки клиентов, источник рекламного трафика и еще много других факторов.

Таким образом, целью моей работы стала разработка и внедрение в имеющуюся в компании систему онлайн записи клиентов (далее - букинг) функциональность овербукинга с применением модели машинного обучения (далее - ML-модели).

Для достижения поставленной цели необходимо выполнить следующие задачи:

1. Проведение нескольких итераций сбора обучающего набора данных (датасета), в которые входят анализ и сбор метрик из корпоративного хранилища данных (DWH, Data Warehouse), исправление ошибок в данных и уменьшение кардинальности категориальных признаков.
2. Выбор ML-модели и ее обучение.
3. Тестирование каждого датасета и выявление наиболее важных метрик
4. Создание сервиса оценки (скоринга) потенциальных клиентов (лидов), который проводит прогнозирование выходимости каждого записанного клиента и интерпретирует результат в нужный для овербукинга формат
5. Создание системы овербукинга, которая получает на вход результат сервиса скоринга и в зависимости от него, закрывает или открывает слоты под запись. Также система отправляет результаты в базу данных и на frontend для визуализации.

Backend сервиса Букинга в компании реализован на языке программирования Python с использованием фреймворка FastAPI [3] и пакета Pydantic [4]. Архитектура решения представлена на рисунке 1:

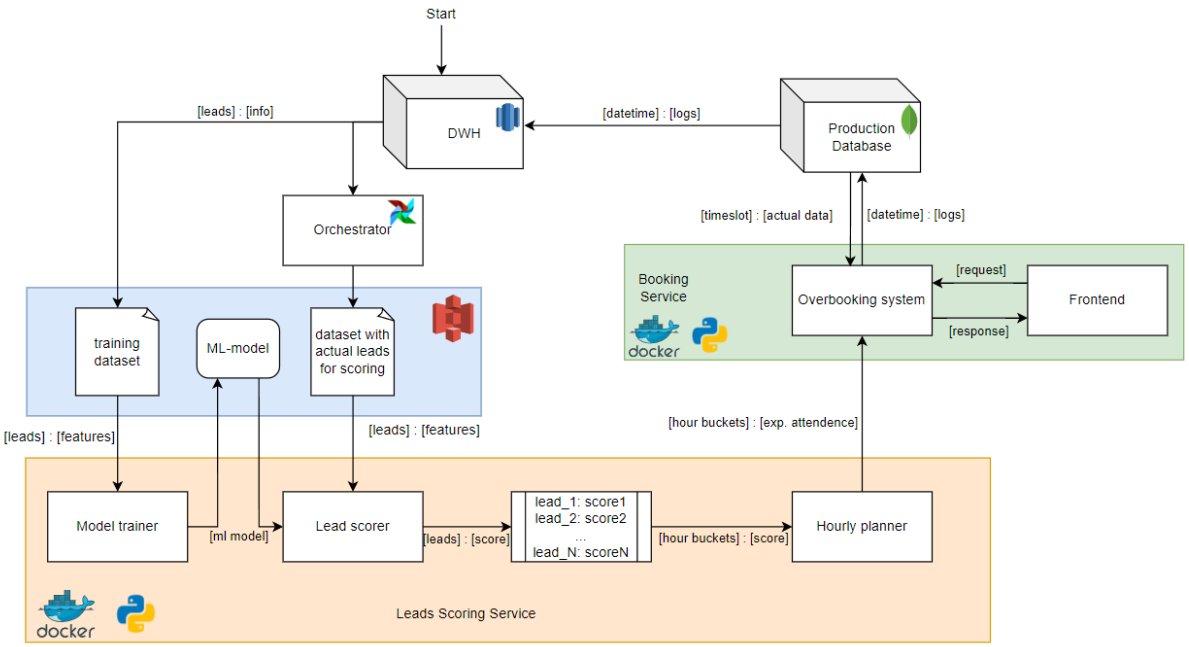


Рисунок 1 – Архитектура решения

Датасет для обучения и датасет для прогнозирования формируются SQL запросами из корпоративного DWH на базе Amazon Redshift [5]. Для ежечасного формирования актуального датасета с текущими лидами используется оркестратор Airflow. Оба датасета хранятся в Amazon Simple Storage Service (S3)

Leads Scoring Service (сервис оценки лидов) — для обучения ML модели и прогнозирования (скоринга) будет написан также на Python. Model trainer получает из S3 обучающий датасет, обучает модель и отправляет файл в S3. Scorer должен получать на вход файл с обученной моделью, датасет с актуальными данными из S3 и каждый час прогнозировать выходимость клиентов. Hourly planner интерпретирует результаты скоринга и отдает их в нужном формате сервису Букинга.

В сервисе Букинга будет реализована система овербукинга, которая принимает на вход прогноз выходимости клиентов и в зависимости от него закрывает или открывает слоты в Букинге для записи клиентов. Актуальную информацию о таймслотах система будет получать из MongoDB [6]. Также будут логироваться все действия системы и храниться в той же базе.

ЛИТЕРАТУРА

1. Как мы вдвое сократили издержки на вводный урок, подсмотрев решение у авиакомпаний. URL: https://habr.com/ru/companies/skyeng/articles/452052/
2. Овербукинг. URL:
2. https://ru.wikipedia.org/wiki/%D0%9E%D0%B2%D0%B5%D1%80%D0%B1%D1%83%D0%BA%D0%B8%D0%BD%D0%B3
3. FastAPI Documentation. URL: https://fastapi.tiangolo.com
4. Pydantic Documentation. URL: https://docs.pydantic.dev/latest/
5. Amazon Redshift Documentation. URL: https://docs.aws.amazon.com/redshift/
6. MongoDB Documentation. URL: https://www.mongodb.com/docs/

УДК 004.932

Т. В. Тельнова (2 курс магистратуры),
С. А. Молодяков, д.т.н., профессор

ПРИМЕНЕНИЕ НЕЙРОННЫХ СЕТЕЙ ДЛЯ БОРЬБЫ С ШУМОМ НА ИЗОБРАЖЕНИЯХ

За последние годы значительно расширилась область применения компьютерного зрения. Возросла популярность технологий, связанных с ним. Особенно выделяются методы глубокого обучения, которые обеспечивают возможности повышения качества и надежности передачи и приема потока изображений. За счет применения нейронных сетей можно найти и убрать паразитные шумы на изображении.

Существуют алгоритмы добавления шума на изображение таким образом, что модель компьютерного зрения поменяет прогноз относительно него – оно будет восприниматься как принадлежащее неверному классу. Пиксели исходного изображения изменяются на сотые и тысячные доли, поэтому шум визуально незаметен [1].

Примерами алгоритмов наложения шума являются FGSM, PGD, DeepFool, C&W. Принцип основан на наложения возмущение для подавления пикселей, на основе которых модель принимает правильное решение, и увеличении влияния в финальном решении пикселей, приводящих к неправильному ответу [2-3]. Результат – новое изображение, которое распознается моделью максимально некорректно.

Одним из способов борьбы с шумом является использование нейронных сетей, в частности, автоэнкодера [4-5]. В работе рассматривается использование нейронных сетей с данной архитектурой с возможностью улучшения показателей эффективности на примере набора данных MS COCO [6].

Целью данной работы является разработка наиболее универсального метода подавления шума на изображениях с применением нейронных сетей для повышения устойчивости систем компьютерного зрения к дефектам входных данных.

Для достижения поставленной цели необходимо решить следующие задачи:

1. Рассмотреть влияние шума на примере задачи классификации.
2. Изучить способы борьбы с шумом на изображениях.
3. Реализовать метод борьбы с шумом.
4. Оценить эффективность полученного решения.

В рамках работы рассмотрено предположение, что чем лучше обобщающая способность нейросети, тем устойчивее она будет к изображениям с шумом и тем проще минимизировать эффект шума для нее.

Метод подавления реализован с помощью нейронных сетей с архитектурой кодер-декодер для увеличения обобщающей способности модели компьютерного зрения и уменьшения привязки решения к выбору ответа по отдельным пикселям. Его использование должно делать метод компьютерного зрения более устойчивым к влиянию шумов, это является основным и наиболее важным требованием к разрабатываемому методу.

ЛИТЕРАТУРА

1. Grosse K., Bieringer L., Besold T. R., Biggio B. Krombholz K. Machine Learning Security in Industry: A Quantitative Survey // IEEE Transactions on Information Forensics and Security. – 2023. – Vol. 18. – P. 1749-1762. DOI 10.1109/TIFS.2023.3251842.
2. Akhtar N., Mian A., Kardan N., Shah M. Advances in Adversarial Attacks and Defenses in Computer Vision: A Survey // IEEE Access. – 2021. – Vol. 9. – P. 155161-155196. – DOI 10.1109/ACCESS.2021.3127960.
3. Muhammad S., Mahum N., Theocharis T., Christos K., Onur M., Lois O., Jungwook C. Robust Machine Learning Systems: Challenges, Current Trends, Perspectives, and the Road Ahead // IEEE Design & Test. – 2020. – Vol. 37. – No. 2. – P. 30-57. DOI 10.1109/MDAT.2020.2971217.
4. Pawlicki M., Choraś R.S. Preprocessing Pipelines including Block-Matching Convolutional Neural Network for Image Denoising to Robustify Deep Reidentification against Evasion Attacks // Entropy. – 2021. – Vol. 23. – No. 10. – P. 1304. – DOI 10.3390/e23101304.
5. Jalalipour S., Rekabdar B. Noisy-Defense Variational Auto-Encoder (ND-VAE): An Adversarial Defense Framework to Eliminate Adversarial Attacks // 2023 Fifth International Conference on Transdisciplinary AI (TransAI), 25-27 Sept. 2023. – P. 50-57. - DOI 10.1109/TransAI60598.2023.00018.
6. Набор данных MS COCO. [Электронный ресурс] Режим доступа https://cocodataset.org/#home.

УДК 004.4

А. Н. Тихонов, А. И. Пухальский (4 курс бакалавриата),
Д. Ф. Дробинцев, ст. преподаватель,
А. В. Гончаров, ассистент

ИНТЕГРАЦИЯ МЕЖДУ BACKEND И MQTT БРОКЕРОМ ЧЕРЕЗ APACHE KAFKA: ОПТИМИЗАЦИЯ ОБМЕНА ДАННЫМИ И СНИЖЕНИЕ НАГРУЗКИ

В последние годы Интернет вещей (IoT) стал одним из самых быстро развивающихся сегментов технологической индустрии. Он представляет собой сеть взаимосвязанных устройств, способных собирать, обрабатывать и обмениваться данными без прямого участия человека. Однако для эффективной работы IoT-систем требуется надежный и масштабируемый канал связи. MQTT [1] — это протокол связи, который эффективно решает эту задачу. В данном контексте разработка MQTT брокера сообщений становится важным шагом в создании современной IoT-инфраструктуры. Брокер MQTT обеспечивает передачу сообщений между устройствами в реальном времени, обеспечивая надежность и масштабируемость системы. При работе с MQTT и Backend одной из ключевых проблем является отсутствие

унифицированного интерфейса для взаимодействия с MQTT брокером. MQTT, как протокол передачи сообщений, обеспечивает эффективное и легковесное обмен сообщениями, но его специфика не всегда соответствует ожиданиям Backend. В таких случаях разработчикам приходится разрабатывать специальные интерфейсы на стороне Backend, чтобы обеспечить совместимость и понимание между двумя системами. Этот процесс может быть сложным и трудоемким, поскольку требует глубокого понимания как MQTT, так и специфики Backend. Кроме того, каждая реализация интерфейса может отличаться в зависимости от потребностей конкретного приложения, что может привести к разрозненности и недостаточной стандартизации в интеграционных процессах. Использование Apache Kafka [2] в качестве посредника между Backend и MQTT брокером может решить эту проблему, предоставляя единый и унифицированный интерфейс для обмена данными. Вместо того чтобы каждый раз создавать и поддерживать собственные интерфейсы, разработчики могут использовать Kafka для структурирования и маршрутизации сообщений между Backend и MQTT брокером, что упрощает интеграцию и снижает сложность разработки.

Целью данной работы является разработка и реализация масштабируемого MQTT брокера сообщений, который будет поддерживать протоколы MQTT и WebSockets [3], а также интеграцию с распределенной платформой для обработки и передачи потоковых данных - Kafka. Основной целью является создание эффективного и надежного канала связи между устройствами IoT, который обеспечит передачу данных в режиме реального времени. Реализация MQTT брокера позволит управлять потоком сообщений между различными устройствами, обеспечивая высокую производительность, масштабируемость и надежность передачи данных.

Существует несколько готовых решений, таких как EMQX, Mosquitto, NanoMQ и т.д. [4]. Все они не имеют поддержки интеграции с Apache Kafka, за исключением EMQX, но данная функция доступна только в корпоративной версии и не предоставляет большой гибкости в формате сообщений [5].

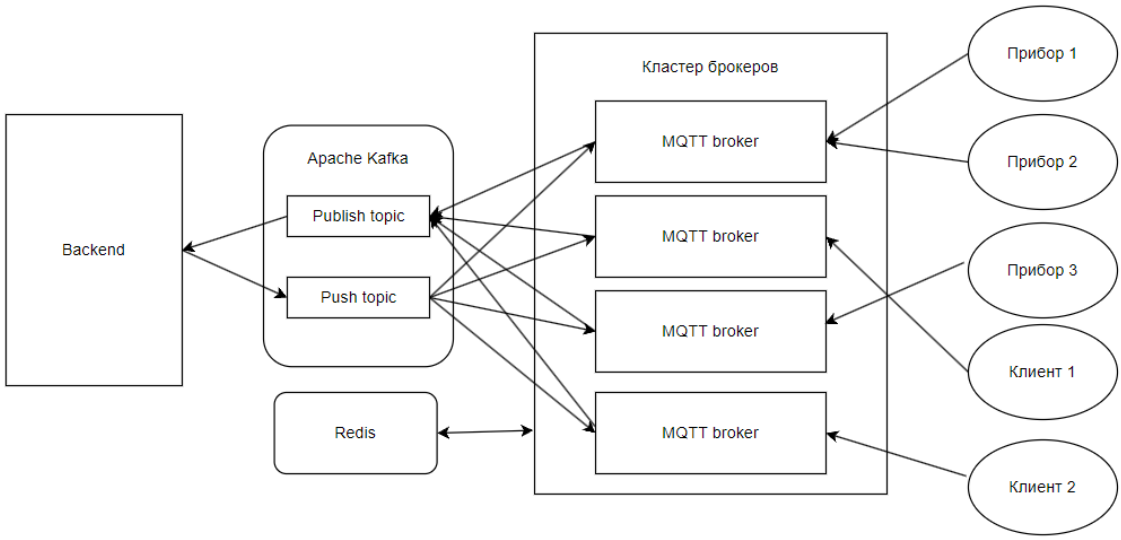


Рисунок 1 – Архитектура системы

Для реализации брокера был выбран фреймворк Netty [6]. Netty — это мощный сетевой фреймворк для Java, который позволяет создавать высокопроизводительные и масштабируемые сетевые приложения. Он поддерживает многопоточность и асинхронность, что позволяет обрабатывать большое количество одновременных соединений (рисунок 1). Netty предоставляет богатый набор функций для работы с сетевыми протоколами. Фреймворк гибкий и позволяет настроить и расширить его функциональность для решения специфических задач. Для общения между MQTT брокером и Kafka будет использоваться клиентская библиотека Apache Kafka. Для авторизации пользователей при подключении к брокеру будет использоваться NoSQL база данных Redis.

ЛИТЕРАТУРА

1. What is MQTT [Электронный ресурс] – URL: https://aws.amazon.com/ru/what-is/mqtt (дата обращения: 01.03.2024)
2. Apache Kafka [Электронный ресурс]. URL: https://kafka.apache.org (Дата обращения: 03.03.2023)
3. MQTT vs. WebSocket - Key differences and when to use them together [Электронный ресурс] – URL: https://ably.com/topic/mqtt-vs-websocket (дата обращения: 01.03.2024)
4. Comparison of Open Source MQTT Brokers 2023 [Электронный ресурс] – URL: https://www.emqx.com/en/blog/a-comprehensive-comparison-of-open-source-mqtt-brokers-in-2023 (дата обращения: 02.03.2024)
5. Comparison of MQTT implementations [Электронный ресурс] – URL: https://en.wikipedia.org/wiki/Comparison_of_MQTT_implementations (дата обращения: 02.03.2024)
6. Netty project [Электронный ресурс]. URL: https://netty.io (Дата обращения: 0403.2023)

УДК 004.912

А. А. Ткаченко (1 курс, магистратуры),
С. М. Устинов, д.т.н., профессор,
И. В. Никифоров, к.т.н., доцент

ПОДХОД ПОВЫШЕНИЯ ЭФФЕКТИВНОСТИ ПРОЦЕССА ОБРАБОТКИ ЗАПРОСОВ НА ИНФРАСТРУКТУРНЫЕ ИЗМЕНЕНИЕ В ИНФОРМАЦИОННО-ТЕХНОЛОГИЧЕСКОЙ СРЕДЕ ОРГАНИЗАЦИИ

Исследование, посвященное оптимизации процесса управления изменениями, актуально: анализируемая корпоративная среда сталкивается с замедлением хода процесса управления изменениями, что уменьшает лояльность бизнес-пользователей [1]. Проблема связана с неэффективным взаимодействием между участниками процесса, длительными процессами согласования и частыми корректировками, что уменьшает скорость внедрения изменений и приводит к повторным работам и задержкам в реализации (продемонстрировано на рисунке 1). В условиях быстро меняющегося рынка изменения становятся неотъемлемой составляющей успешного функционирования компаний, поэтому основное значение исследования заключается в увеличении эффективности и скорости бизнес-процессов, что является стратегически важным фактором для корпораций, позволяющим оперативно адаптироваться к новым требованиям и конкурентным условиям. Оптимизация управления изменениями помогут сократить временные ресурсы на согласование и реализацию изменений, что, в свою очередь, ускорит внедрение нововведений и повысит общую производительность, обеспечивая оперативную адаптацию и обеспечивая конкурентное преимущество на рынке в долгосрочной перспективе.

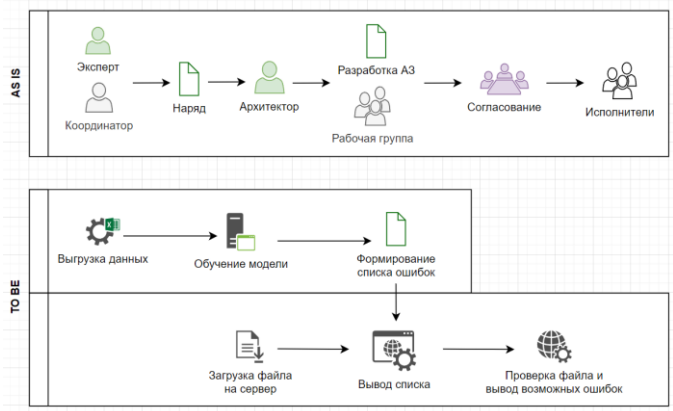


Рисунок 1 – Описание модели бизнес-процессов As Is и To Be

Библиотека ITIL (IT Infrastructure Library) широко используется в современном мире как набор передового опыта в области управления IT-услугами и стандартизации, чья область охватывает не только технические вопросы, но и человеческие и экономические аспекты управления IT-услугами [2, 3]. Ряд исследований рассматривают опыт и методы применения ITIL в различных сферах, включая управление изменениями и создание систем управления инцидентами (IMS) совместимыми с ITIL. Так, исследование [4] ищет ответ на вопрос: какие типы требований следует учитывать при построении системы управления инцидентами. Системы управления объектами ЦЕРН реализовывает процессы управления изменениями [5] и в исследовании описывают опыт и методы, которые применялись для настройки инструмента Service-Now в соответствии с требованиями CERN. В статье [6] группа авторов описывают подход, который заключается в использовании многокритериальной оптимизации на основе моделирования для выбора оптимальных стратегий процесса управления изменениями ITIL, которые помогают ИТ-менеджерам достичь эффективности процессов как критического фактора успеха (CSF).

Цель данного исследования состоит в повышении эффективности процесса управления изменениями в корпоративной среде с использованием анализа текстовых комментариев в АЗ (аналитических записках). Оно направлено на выявление наиболее часто встречающихся замечаний в процессе согласования изменений и разработку инструмента для автоматизированной превентивной проверки новых изменений на основе собранной базы знаний, то есть нужно создать систему, способную проводить семантический анализ текстовых комментариев и предоставлять рекомендации для улучшения качества АЗ, с целью ускорения процесса утверждения изменений и уменьшения количества ошибок при их реализации. Желаемый результат – снижение времени, затрачиваемого на утверждение изменений, на 30% за первые 6 месяцев внедрения разработанной системы. Ключевым показателем эффективности (KPI) будет являться увеличение процента успешных изменений без дополнительных корректировок, достигнутый благодаря автоматизированной проверке на основе аналитической базы знаний.

Задачи исследования, необходимые для достижения цели, перечислены ниже.

1. Исследование содержания АЗ с целью выявления с целью классификации замечаний по группам замечаний;
2. Сравнительный анализ существующих решений и качественный анализ текстовых комментариев в рамках согласования изменений для определения наиболее частых аспектов, требующих корректировки или доработки;
3. Предложение решения в виде разработки инструмента, способного анализировать АЗ на основе базы знаний, выявляя соответствие уже известных замечаний новым изменениям и предоставляя рекомендации по их улучшению.
4. Реализация разработанного инструмента для проведения семантического анализа комментариев в АЗ и интеграции его в процесс управления изменениями;
5. Демонстрация результатов, включающая работу разработанного инструмента, обзор его эффективности прототипа при внедрении в рабочую среду.

Для достижения поставленной цели использовались различные методы исследования, включая машинное обучение, кейс-стади, экспертные интервью и опросы сотрудников, а также анализ данных. Были созданы скрипты Python для парсинга текста, позволяющие выявлять проблемы в заполненных шаблонах и предотвращать их отправку для дальнейшего ручного анализа на соответствие шаблонам и актуальным данным, то есть осуществлено превентивное предупреждение вхождения некорректного документа в цепочку согласования и затягивание изменения. Для анализа данных использовались статистические методы для выявления общих тенденций, распределения частоты различных типов замечаний и других характеристик изменений и кластерный анализ для выявления общих тенденций и группировки изменений по схожим характеристикам. Построение моделей включало в себя как классические методы (ансамбли, бустинг), так и использование предобученных моделей, таких как BERT, для классификации текста. Нынешняя фаза разработки рассматривает

потенциал подхода RAG для LLM (Large Language Models) [7], что может дать дополнительные новые знания и улучшить результаты исследования. Метод является частью модернизированного подхода к NLP с использованием LLM и представляет собой архитектуру, сочетающую в себе возможности поиска (Retrieve), расширения (Add) и генерации (Generate) текста на основе баз данных текстов. Подход обладает потенциалом в контексте управления изменениями, поскольку позволит работать с обширными данными и генерировать релевантные рекомендации на их основе [8, 9]. Это ускорит процесс утверждения изменений и снизит количество ошибок в их реализации.

Работа выполняется при постановке задачи от компании Газпромнефть-ИТО. В настоящей работе проведено качественное исследование данных, написаны скрипты проверки соответствия АЗ шаблонам и предложен подход в сфере машинного обучения, который бы позволил. В дальнейшем планируется проработка архитектуры программного средства, реализация и проведение экспериментальных исследований, которые должны показать повышение эффективности за счет снижения времени, затрачиваемого на утверждение изменений, на 30% за первые 6 месяцев внедрения.

ЛИТЕРАТУРА

1. Киринович И. Ф., Антюшеня Д. В., Яшин К. Д. Исследование зависимости индекса потребительской лояльности веб-приложения от эргономических параметров и удобства использования интерфейса //Доклады Белорусского государственного университета информатики и радиоэлектроники. – 2018. – №. 1 (111). – С. 30-36.
2. Библиотека ITIL [Элекстронный источник]. Режим доступа: https://www.atlassian.com/itsm/itil
3. Thomas Schaaf; "The IT Infrastructure Library (ITIL) - An Introduction for Practitioners and Researchers", 2007.
4. Marko Jäntti; "Defining Requirements for An Incident Management System: A Case Study", 2009 FOURTH INTERNATIONAL CONFERENCE ON SYSTEMS, 2009.
5. Zhechka Toteva; R Alvarez Alonso; E Alvarez Granda; M-E Cheimariou; I Fedorko; J Hefferman; S Lemaitre; D Martin Clavo; P Martinez Pedreira; O Pera Mira; "Service Management at CERN with Service-Now", 2012.
6. Mercedes Ruiz; Javier Moreno; Bernabé Dorronsoro; Daniel Rodríguez-García; "Using Simulation-based Optimization in The Context of IT Service Management Change Process", DECIS. SUPPORT SYST., 2018.
7. Генерация с дополненной выборкой (RAG) [Элекстронный источник]. Режим доступа: https://aws.amazon.com/ru/what-is/retrieval-augmented-generation/
8. Ковалев, А. Д. Автоматизированный подход к семантическому поиску по программной документации на основе алгоритма Doc2Vec / А. Д. Ковалев, И. В. Никифоров, П. Д. Дробинцев // Информационно-управляющие системы. – 2021. – № 1(110). – С. 17-27. – DOI 10.31799/1684-8853-2021-1-17-27. – EDN SBRUMH.
9. 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. – EDN CFRNSJ.

УДК 004.932.2

М. Т. Толчёнов (3 курс бакалавриата),
П. А. Шагалова, к.т.н., доцент

РАЗРАБОТКА СИСТЕМЫ ФОТОФИКСАЦИИ И АНАЛИЗА ИЗОБРАЖЕНИЙ МИКРОСКОПИИ НА БАЗЕ ОДНОПЛАТНОГО КОМПЬЮТЕРА

Развитие алгоритмов компьютерного зрения предоставляет широкие возможности в области автоматизации задач в различных областях человеческой деятельности. Одним из актуальных и перспективных направлений является обработка и анализ изображений, полученных с использованием микроскопа [1]. Данные изображения обладают высокой визуальной сложностью и нередко анализируются специалистами вручную. Разработка

автоматизированных систем анализа изображений микроскопии позволит сократить временные затраты и повысить точность и эффективность выполнения исследований.

В данной работе использован usb-микроскоп с датчиком изображения 5 МПикс и разрешением 2592x1944 и одноплатный компьютер MangoPi MQ PRO D1 [2] на базе архитектуры RISC-V [3, 4]. Выбор одноплатного компьютера в качестве вычислительного устройства позволяет создать компактное и портативное решение, не требующее дополнительных настроек, а использование открытой архитектуры увеличивает потенциал и перспективы дальнейшего развития проекта.

В процессе разработки системы фотофиксации и анализа изображений микроскопии были выполнены следующие шаги: установка операционной системы, подключение и настройка usb-микроскопа, получение изображения, выбор инструментов анализа снимков, получаемых с микроскопа, разработка программного обеспечения, позволяющего выполнить поиск объектов на изображении, кросс-компиляция и апробация программы.

При установке операционной системы был выявлен ряд проблем, включающий перегрев платы, бесконечный цикл загрузки, проблемы с подключением устройств через стандартные интерфейсы. После апробации нескольких сборок операционных систем для дальнейшей работы был выбран образ Armbian 22.08.0-trunk Trixie с ядром Linux 6.1.0-rc3-d1, как наиболее стабильный в использовании [5].

Операционная система идентифицирует usb-микроскоп как веб-камеру. С учетом этого обстоятельства для фотосьемки было подобрано программное обеспечение fswebcam, выполнены настройки параметров съемки, а также разработан и реализован сценарий фотофиксации на скриптовом языке bash.

Разработано программное решение для поиска объектов на изображениях, полученных с usb-микроскопа, включающее фильтрацию изображения с использованием фильтра Гаусса, бинаризацию с использованием алгоритма Оцу [6], применение морфологических операций сужения и расширения, поиск контуров объектов.

Реализация программного решения выполнена на языке C++ с использованием библиотеки алгоритмов компьютерного зрения с открытым исходным кодом OpenCV [7]. Поскольку для систем с низкой производительностью и небольшим объемом памяти компиляция программного кода на целевой плате нецелесообразна, сборка производилась на персональном компьютере с использованием кросс-компилятора RISC-V GNU Compiler Toolchain [8]. Для копирования файлов с основной машины на MangoPi выполнено подключение к плате по протоколу ssh. Программное решение апробировано на изображениях фармацевтических препаратов.

В настоящее время проводится проектирование программно-управляемой подвижной платформы для предметного стекла микроскопа с подключением через интерфейс ввода-вывода общего назначения, а также проектирование и сборка корпуса системы.

ЛИТЕРАТУРА

1. Л. Шапиро, Дж. Стокман. Компьютерное зрение. – М. – Бином. – Лаборатория знаний. – 2015. – С.763
2. MangoPi [Электронный ресурс] Режим доступа: https://mangopi.org/
3. Фролов В.А., Галактионов В.А., Санжаров В.В. Исследование технологии RISC-V // Труды ИСП РАН. – 2020. – 32(2). – С. 81-98. DOI: 10.15514/ISPRAS–2020–32(2)–7
4. RISC-V [Электронный ресурс] Режим доступа: https://riscv.org/
5. Armbian [Электронный ресурс] Режим доступа: https://www.armbian.com/
6. Otsu N. A threshold selection method from gray-level histograms //Automatica. – 1975. – Т. 11. – №. 285-296. – С. 23-27
7. OpenCV [Электронный ресурс] Режим доступа: https://opencv.org/
8. RISC-V GNU Compiler Toolchain [Электронный ресурс] Режим доступа: https://github.com/riscv/riscv-gnu-toolchain

УДК 004.031.43

Д. А. Тяпуев (2 курс магистратуры),
Г. Ф. Малыхина, д.т.н., профессор

РАЗРАБОТКА АЛГОРИТМОВ УПРАВЛЕНИЯ РАСПРЕДЕЛЕННЫМИ ТРАНЗАКЦИЯМИ В МИКРОСЕРВИСНОЙ АРХИТЕКТУРЕ НА БАЗЕ JAVA

В современном мире тенденция разработки все больше и больше сводится к использованию микросервисных архитектур. В таких условиях разработчики неизбежно сталкиваются с проблемами управления и обработки распределенных транзакций. С появлением микросервисной архитектуры и распределенных систем, сложность приложений значительно возросла. Управление распределенными транзакциями стало намного сложнее из-за необходимости согласования изменений в нескольких сервисах. Создание системы из нескольких микросервисов позволяет исследовать различные алгоритмы управления распределенными транзакциями и выбрать наиболее подходящий подход [1].

Управление транзакциями в распределенных системах имеет очень большое значение для надежности и целостности данных. Правильный выбор алгоритмов управления транзакциями может значительно улучшить надежность и целостность данных в распределенных системах [2]-[3].

Таким образом разработка системы из нескольких микросервисов с различными алгоритмами управления распределенными транзакциями позволит провести различные исследования в одинаковых условиях и выбрать оптимальный алгоритм. Так же в будущем в данную систему можно будет внедрить новые алгоритмы для их тестирования [4].

Целью данной работы является разработка единой распределенной системы с тремя микросервисами, где управление распределенными транзакциями будет осуществляться по алгоритмам: 2PC, компенсирующих транзакций и event sourcing. А также необходимо провести различные исследования данной системы. Для достижения цели работы требуется выполнить следующие задачи:

- Разработка трех микросервисов (сервисы заказов, инвентаря и доставки)
- Осуществление системы сообщений между микросервисами с помощью Apache Kafka
- Упаковка всех микросервисов в изолированный Docker-контейнер
- Разработка трех алгоритмов управления распределеными транзакциями
- Провести исследования в полученной распределенной системе.

Двухфазная фиксация (2PC) – это метод обработки данных, который предполагает два этапа: в первом этапе данные блокируются для избежания конфликтов при одновременном доступе, а затем они фиксируются, чтобы другие процессы могли увидеть изменения и работать с актуальными данными. Этот метод обеспечивает безопасность и целостность данных, предотвращая возможные ошибки и конфликты при работе с множеством процессов одновременно [5].

Алгоритм компенсации транзакций используется для отмены изменений, сделанных транзакцией, если происходит сбой или ошибка в процессе выполнения. При возникновении ошибки транзакция откатывается, и на ее место запускается компенсационная транзакция, которая возвращает данные к предыдущему состоянию. Это обеспечивает целостность данных и избегает нежелательных изменений в случае возникновения проблем во время выполнения операции [6].

Алгоритм event sourcing представляет собой метод хранения данных в виде серии событий, которые отражают все изменения состояния системы. Вместо того чтобы хранить текущее состояние объекта, система хранит только события, которые привели к текущему состоянию, а также их порядок. Это позволяет восстанавливать состояние системы на любом этапе путем повторного применения всех событий. Event sourcing обеспечивает полную

трассируемость изменений данных, возможность анализировать историю и откатывать изменения, а также улучшает производительность и масштабируемость системы [7].

Spring Boot – это популярный фреймворк для создания веб-приложений с использованием Java. Это часть фреймворка Spring, которая представляет собой набор инструментов и библиотек для создания приложений корпоративного уровня. Благодаря данному фреймворку были развернуты три микросервиса и сервис-координатор транзакций. К каждому микросервису была подключена база данных PostgeSQL [8].

За обмен транзакционными сообщениями между сервисами отвечает Apache Kafka. Это распределённая система, предназначенная для обработки потоков данных в режиме реального времени [9]. Для автоматизации развертывания всей системы все сервисы были упакованы в автономные контейнеры с использованием Docker [10]. Схема получившейся распределенной системы представлена на рисунке 1.

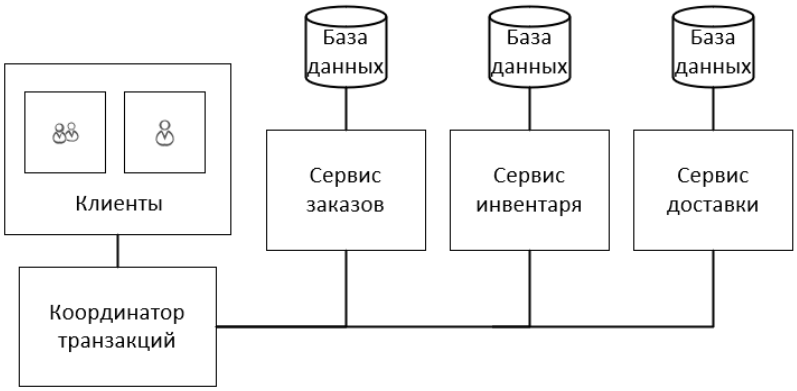


Рисунок 1 – Схема системы.

Полученная распределенная система позволяет проводить различные исследования над разработанными алгоритмами (нагрузочное тестирование, пропускная способность, максимальная производительность и т.д.). В будущем данную систему можно будет расширить, дополнив ее новыми алгоритмами управления распределенными транзакциями.

ЛИТЕРАТУРА

1. Unmesh Joshi. Patterns of distributed systems, 2023.
2. Распределенные данные. Алгоритмы работы современных систем хранения информации – СПб.: Питер, 2021. — 336 с.: ил. — (Серия «Бестселлеры O’Reilly»).
3. Принципы организации распределенных баз данных / пер. с англ. А. А. Слинкина. – М.: ДМК Пресс, 2021. – 672 с.: ил.
4. Распределенное управление конкурентностью [Электронный ресурс]. – 2024. – URL: - https://habr.com/ru/articles/784750/ (дата обращения 08.01.2024).
5. Двухфазная фиксация в распределенных транзакциях [Электронный ресурс]. – 2023. – URL: - https://otus.ru/nest/post/2945/ (дата обращения 10.11.2023).
6. Паттерн Saga в микросервисной архитектуре [Электронный ресурс]- 2023. – URL: - https://habr.com/ru/companies/otus/articles/751134/ (дата обращения 22.11.2023).
7. Lima S. et al. Improving observability in event sourcing systems //Journal of Systems and Software. – 2021. – Т. 181. – С. 111015.
8. Spring Boot [Электронный ресурс] – 2022 – URL: - https://spring.io/projects/spring-boot (дата обращения 20.01.2024).
9. Apache Kafka: основы технологии [Электронный ресурс]. – 2021. – URL: - https://habr.com/ru/companies/slurm/articles/550934/(дата обращения 10.01.2024).
10. Docker docs [Электронный ресурс] – 2022 – URL: - https://docs.docker.com/ (дата обращения 04.02.2024).

УДК 004.453

Фам Ван Ньян (4 курс бакалавриата),
Л. П. Котлярова, ст. преподаватель

ПРИМЕНЕНИЕ НАБОРА ТЕХНОЛОГИЙ СТЕКА MERN ДЛЯ ДИНАМИЧЕСКОЙ ВЕБ-РАЗРАБОТКИ

В современном мире образование играет важную роль в индивидуальном и общественном развитии. С появлением Интернета и цифровых технологий появились новые возможности для обучения и самообразования. Такие веб-приложения имеют ряд преимуществ перед традиционными формами обучения, таких как гибкость, доступность, персонализация, интерактивность и т. д. Поэтому возникла идея спроектировать и реализовать веб-приложение для поиска и изучения курсов, подходящее для разных групп пользователей. Для разработки своего веб-приложения я выбрал стек технологий MERN [1] (MongoDB, Express.js, React.js и Node.js). Стек MERN — это не только сочетание мощных технологий, но и предоставляет оптимальные и гибкие решения для разработчиков.

Архитектура MERN [2] позволяет легко создавать веб-приложения с трехуровневой архитектурой (интерфейсная часть, серверная часть, база данных) полностью с использованием JavaScript и JSON.

- Интерфейс (React.js): это часть пользовательского интерфейса приложения, созданная с помощью React.js для создания компонентов и интерактивных интерфейсов для пользователей.
- Серверная часть (Express.js + Node.js): эта часть обрабатывает логику и запросы со стороны внешнего интерфейса, используя Express.js для управления маршрутизацией и обработкой HTTP-запросов, а также Node.js для выполнения операций на стороне сервера.
- База данных (MongoDB): хранит данные веб-приложений и управляет ими, предоставляя необходимые запросы и операции с базой данных через API из серверной части.

Во-первых, мы понимаем, что использование JavaScript как на стороне сервера, так и на стороне клиента помогает обеспечить единообразие языка и среды. Это уменьшает фрагментацию процесса разработки и повышает опыт разработчиков.[3]

Во-вторых, использование React.js в стеке MERN обеспечивает эффективный подход к пользовательскому интерфейсу (UI). React.js создан для повышения производительности благодаря гибкому повторному использованию, Virtual DOM и богатой экосистеме плагинов.

Кроме того, в стеке MERN важную роль играют методы оптимизации исходного кода.[4]

- Minification và Compression: Минимизация удаляет ненужные символы, такие как пробелы и комментарии, из кода JavaScript, а сжатие уменьшает размер кода за счет замены повторяющихся выражений более короткими выражениями. Сочетание этих двух методов значительно уменьшает размер кода JavaScript.
- Lazy Loading: Lazy loading помогает загружать код только при необходимости, а не загружать весь код сразу. Это особенно полезно для приложений с большим исходным кодом, помогая сократить время начальной загрузки.
- Server-Side Rendering (SSR): SSR — это метод, который сначала отображает HTML-контент на сервере, а не на клиентском JavaScript. SSR сокращает время начальной загрузки приложения и снижает нагрузку на браузер пользователя.

На стороне клиента сокращение количества HTTP-запросов за счет объединения и сжатия файлов CSS и JavaScript, использования спрайтов изображений и отложенной загрузки, а также использования сети доставки контента (CDN) повышает производительность приложения.

При оптимизации базы данных важными мерами повышения производительности являются использование индексов, отказ от использования запросов OR и сокращение количества необходимых запросов.

По сравнению с другими технологическими стеками, такими как SERN и MEAN, MERN Stack имеет свои выдающиеся преимущества. Что касается SERN [5], у MERN Stack есть преимущество использования MongoDB вместо SQLite. MongoDB — популярная база данных NoSQL, отличающаяся от баз данных SQL, имеющих табличную и реляционную структуру. С другой стороны, базы данных NoSQL используют гибкую модель данных, в которой данные хранятся различными способами, например, в виде пар ключ-значение, документов или графиков. Это позволяет выполнять более быстрые и гибкие запросы, особенно для больших наборов данных со сложной или изменяющейся структурой. Что касается MEAN [6], MERN Stack использует React.js вместо Angular.js, React.js имеет компонентную архитектуру, более гибок и прост в управлении, чем Angular.js, особенно в больших и сложных приложениях. Кроме того, благодаря сочетанию React.js и Node.js стек MERN обычно обеспечивает более высокую производительность при создании одностраничных приложений и приложений реального времени, чем стек MEAN.

Суммируя, использование стековой технологии MERN не только обеспечивает хорошую производительность, но также помогает оптимизировать разработку веб-приложений и управление ими. Гибкое сочетание мощных технологий в стеке MERN играет решающую роль в создании качественных динамичных веб-продуктов и хорошего пользовательского опыта.

ЛИТЕРАТУРА

1. Sumangala A. Bafna, Pratiksha D. Dutonde, Shivani S. Mamidwar, Monali S. Korvate, Prof. Dhiraj Shirbhare, “Review on Study and Usage of MERN Stack for Web Development” International Journal for Research in Applied Science & Engineering Technology (IJRASET) ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538 Volume 10 Issue II Feb 2022.
2. Yogesh Baiskar, Priyas Paulzagade, Krutik Koradia, Pramod Ingole, Dhiraj Shirbhate, “MERN: A Full-Stack Development”, International Journal for Research in Applied Science & Engineering Technology (IJRASET) ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538 Volume 10 Issue I Jan 2022
3. Prof. Yogesh Kadam, Akhil Goplani, Shubit Mattoo, Shashank Kumar Gupta, Darshan Amrutkar, Prof. Dr. Jyoti Dhanke, “Introduction to MERN Stack & Comparison with Previous Technologies” European Chemical Bulletin 12(Special Issue 4):14382-14386.
4. Optimizing MERN Stack Performance. [Электронный ресурс]. – URL: https://dev.to/clickysoft/optimizing-mern-stack-performance-techniques-for-high-traffic-applications-628 (дата обращения 15.03.2024)
5. SERN Stack. [Электронный ресурс]. – URL: https://perpetualnoobblog.wordpress.com/2017/11/04/sernsean-stack/ (дата обращения 15.03.2024)
6. Обзор по изучению и использованию MEAN Stack для веб-разработки. [Электронный ресурс]. – URL: https://nvjournal.ru/article/Obzor_po_izucheniju_i_ispolzovaniju_MEAN_Stack_dlja_veb-razrabotki/ (дата обращения 15.03.2024)

УДК 004.415.25

А. А. Трофимов (4 курс бакалавриата),
И. А. Шемякин, ст. преподаватель

РАСПРЕДЕЛЕНИЕ МОДЕЛЕЙ НЕЙРОННЫХ СЕТЕЙ ПО СЕРВЕРАМ В СЕРВИСЕ «НЕЙРОПЛАТФОРМА» В «ОДНОКЛАССНИКАХ»

Современные крупные информационные компании активно используют большие данные и машинное обучение [1]. Существует огромное множество технологий, позволяющих создавать модели нейронных сетей, обучать их и запускать. Одноклассники не исключение и тоже разрабатывает алгоритмы, позволяющие решать различные задачи, начиная от определения объектов на фото, заканчивая рекомендательными системами в ленте новостей. Для этих целей существует платформа Neural Network Platform. Данный сервис позволяет производить так называемый «инференс» нейронных сетей на python в синхронном и

асинхронном режиме, то есть либо через прямой запрос по RPC (Remote Procedure Call), либо через брокер сообщений Kafka.

В связи с большим потреблением памяти нейронными сетями, запустить все модели на одном сервере невозможно, поэтому существует кластер из большого количества машин, в котором модели распределяются по хостам. Сейчас распределение происходит с помощью подхода хореографии, когда каждый хост сам определяет, что на себя загружать. При таком подходе происходит неравномерное распределение запросов по серверам, что может приводить к чрезмерной нагрузке на одни сервера и простою других. *Решением* было бы создание алгоритма на основе подхода оркестратора, представляющего собой центральный сервис, который эффективно распределяет по серверам модели в зависимости от нагрузки и других ограничений. Графики нагрузки по каждому хосту позволят определить, насколько изменилось распределение запросов.

Для разработки использовались некоторые технологии, в частности:

1. Языком программирования для разработки данного алгоритма был выбран Java, так как весь проект Neural Network Platform, для которого разрабатывается алгоритм, написан именно на этом языке, который является стандартом в компании.
2. Для реализации механизма удаленного вызова процедур (RPC) выбрана библиотека one-nio, которая принята в компании в качестве стандарта [2]. Это обеспечивает эффективную коммуникацию между компонентами системы.
3. Выбор лидера в распределенной системе происходит с помощью Apache Zookeeper [5]
4. Для сбора и просмотра статистики по нагрузке используется Apache Druid - высокопроизводительная распределенная система для обработки и анализа больших объемов данных в реальном времени [6].
5. Тестирование алгоритма будет производиться с помощью фреймворка JUnit версии 5 [7]

График, показывающий нагрузку на каждый сервер, представлен на рисунке 1. На нем по оси X - время, по оси Y - суммарное время обработки вызовов каждой моделей на данном хосте, то есть фактически показывает, насколько интенсивно используется данный хост. Как видно по графику, разница между хостами может быть в несколько раз.

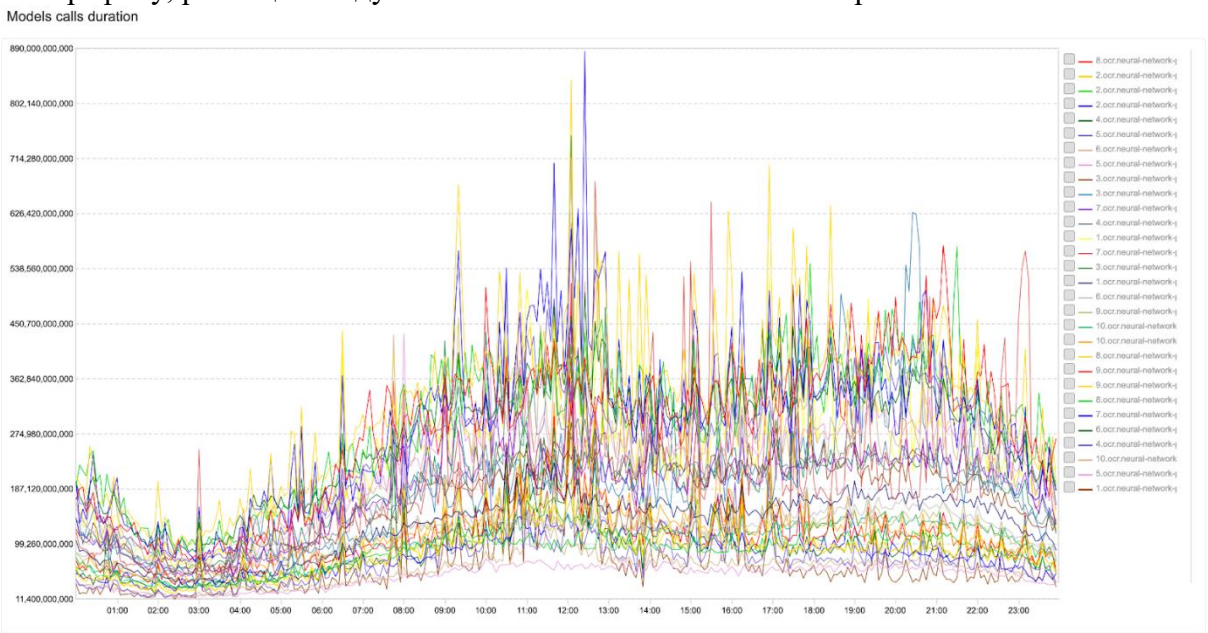


Рисунок 1 – График нагрузки

Критерием эффективности алгоритма будет являться уменьшение разницы между самым нагруженным хостом и ненагруженным в каждый момент времени.

Архитектура системы с взаимодействием её основных компонентов изображена на рисунке 2 (NNP - neural network platform):

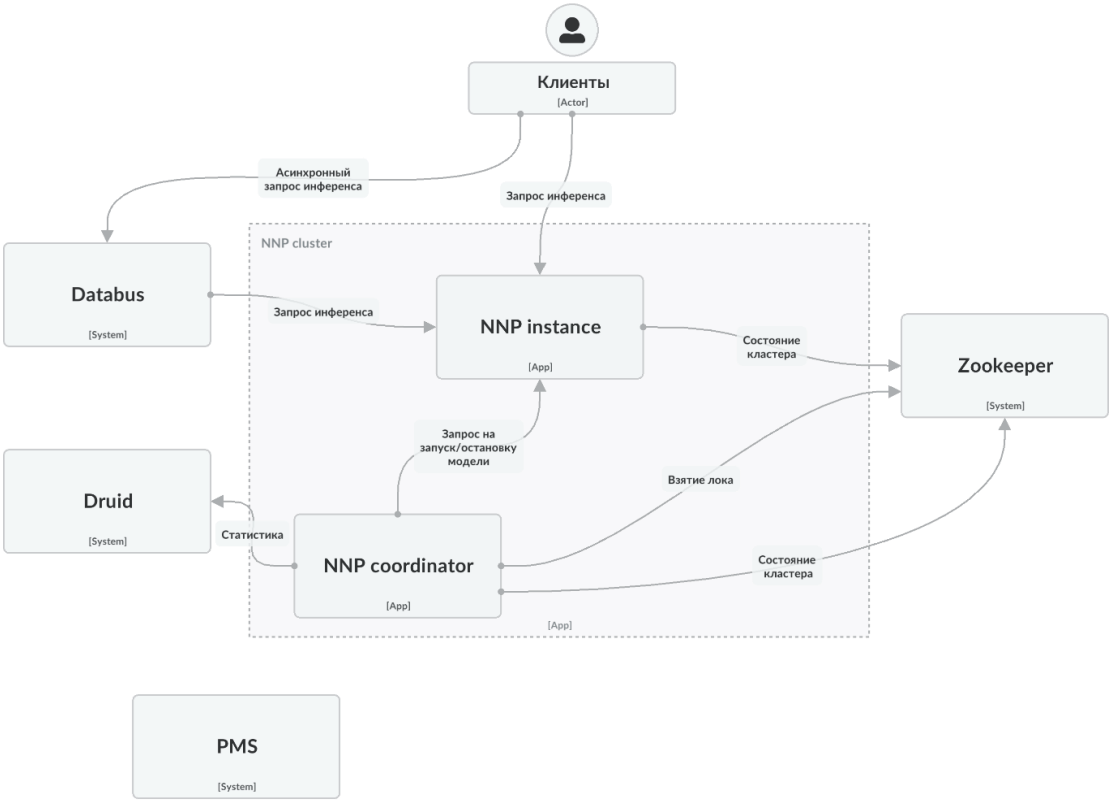


Рисунок 2 – Архитектура системы

ЛИТЕРАТУРА

1. Коротеев Михаил Викторович Обзор некоторых современных тенденций в технологии машинного обучения // E-Management. 2018. №1. URL: https://cyberleninka.ru/article/n/obzor-nekotoryh-sovremennyh-tendentsiy-v-tehnologii-mashinnogo-obucheniya (дата обращения: 19.01.2024).
2. One-Nio Documentation [Электронный ресурс]. Режим доступа: https://github.com/odnoklassniki/one-nio/wiki (дата обращения: 19.01.2024).
3. Арпачи-Дюссо, Р. Х. Операционные системы: Три простых элемента / Р. Х. Арпачи-Дюссо, А. К. Арпачи-Дюссо. – Москва : ДМК Пресс, 2021. – 730 с.
4. S. Mustafa, K. Bilal, S. A. Madani, N. Tziritas, S. U. Khan and L. T. Yang, "Performance Evaluation of Energy-Aware Best Fit Decreasing Algorithms for Cloud Environments," 2015 IEEE International Conference on Data Science and Data Intensive Systems, Sydney, NSW, Australia, 2015, pp. 464-469, doi: 10.1109/DSDIS.2015.104.
5. Zookeeper 3.9 Documentation : сайт. – URL: https://zookeeper.apache.org/doc/r3.9.1/index.html (дата обращения: 19.01.2024)
6. Apache Druid : сайт. – URL: https://druid.apache.org/ (дата обращения: 19.01.2024)
7. JUnit : сайт. – URL: https://junit.org/junit5/ (дата обращения: 19.01.2024)

УДК 004.42

М. В. Ферапонтов (4 курс бакалавриата),
О. В. Прокофьев, ст. преподаватель

РАЗРАБОТКА POSTGRESQL РАСШИРЕНИЯ ДЛЯ ПОДДЕРЖКИ ГРАФОВЫХ ЗАПРОСОВ

В современном информационном обществе объем данных постоянно увеличивается, делая эффективное управление информацией важной приоритетной задачей для различных сфер деятельности. Графовые базы данных предоставляют мощные инструменты для представления и анализа сложных взаимосвязей между данными, где объекты представлены узлами, а связи – ребрами [1]. Такие структуры данных находят свое применение в различных

областях, включая социальные сети, биоинформатику, транспортные системы и многие другие.

В последнее время специализированные графовые базы данных приобретают все большую популярность [2], однако различные графовые базы данных предлагают разнообразные языки запросов, что затрудняет использование этих систем и, также, переносимость запросов между ними. Была предпринята попытка введение стандарта для графового языка запросов, но на данный момент он находится на этапе рассмотрения и пока не реализован. В свою очередь, реляционные СУБД, обладая более долгой историей и широким использованием, являются более зрелой технологией. Важно заметить, что скорость графовых запросов специализированных графовых баз данных может быть достигнута стандартными запросами реляционных СУБД [3]. Таким образом, существует актуальная потребностьв механизмах интеграции графовых структур в реляционные СУБД для обеспечения универсальности и эффективности анализа данных.

Целью данной работы является разработка расширения для реляционной СУБД PostgreSQL для обеспечения поддержки графовых запросов. Графовые запросы позволяют эффективно выражать и извлекать информацию о взаимосвязях между данными, что является критически важным в контексте анализа сложных сетевых структур. Специализированный синтаксис графовых запросов понятнее базового языка SQL, поэтому его использование сокращает время на разработку. Улучшение понимания и ускорение разработки, также является одной из целей данной работы.

Для достижения этой цели необходимо решение следующих задач:

1. Обзор существующих методов реализации поддержки графовых запросов в реляционных базах данных
2. Проведение сравнительного анализа найденных методов.
3. Разработка расширения.
4. Демонстрация работы расширения.

В литературе встречается два основных подхода обработки графовых запрсов в реляционных СУБД. Работа [4] предлагает называть данные подходы, как встроенное графовое ядро и встроенное реляционное ядро. При разработке расширения используется второй способ. Данный подход хранит графы внутри реляционных таблиц в специфичном виде. Затем, приложение, реализованное поверх СУБД переводит графовые запросы в их SQL представление, чтобы СУБД их исполнило. Например Grail [4] может транслировать графовые запросы на поиск кратчашего пути в SQL, а SQLGraph [5] может транслировать запросы Gremlin в SQL без побочных эффектов. Рисунок 1 показывает базовую архитектуру метода встроенного реляционного ядра.

Главная проблема данного подхода заключается в том, что графовые операции представляют собой серию реляционных операций, например через оператор JOIN, что может быть более дорогостоящей операцией, чем обработка графового представления данных в специализированных системах, но современные оптимизаторы запросов могут снизить влияние данной проблемы.

В рамках данной работы, выбор PostgreSQL обоснован его открытым исходным кодом, выбранным методом, и его встроенной поддержкой рекурсивных запросов. Рекурсивные запросы в PostgreSQL обеспечивают уникальную возможность использования языка SQL для обработки и навигации по графовым структурам, что является ключевым аспектом при разработке расширения для поддержки графовых запросов. Проблема с рекурсивными запросами в PostgreSQL заключается в их сложном синтаксисе, из-за чего разработка приложений с их использованием сильно затрудняется.

Для разработки расширений для PostgreSQL обычно используется язык программирования C, что обосновано не только историческими аспектами, но и рядом выдающихся преимуществ данного языка в контексте системного программирования и расширения функционала СУБД.

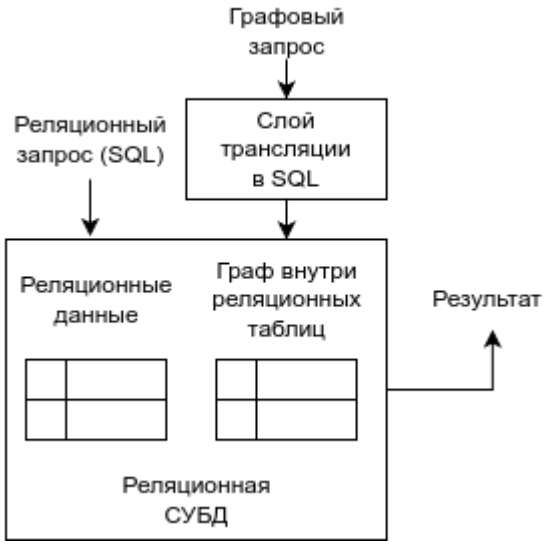


Рисунок 1 —Архитектура метода встроенного реляционного ядра

ЛИТЕРАТУРА

1. Что такое графовая база данных? [Электронный ресурс] Режим доступа: https://aws .amazon.com/ru/nosql/graph/
2. Andlinger P. Graph DBMS increased their popularity by 500% within the last 2 years. [Электронный ресурс] Режим доступа: https://db-engines.com/en/blog_post/43
3. Fan J., Raj A. G. S., Patel J. M. The Case Against Specialized Graph Analytics Engines // Seventh Biennial Conference on Innovative Data Systems Research, CIDR 2015, Asilomar, CA, USA, January 4-7, 2015, Online Proceedings. — 2015.
4. Extending In-Memory Relational Database Engines with Native Graph Support / M. Hassan [и др.] //. — OpenProceedings.org, 2018.
5. SQLGraph: An Efficient Relational-Based Property Graph Store /W. Sun [и др.] // Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data. — New York, NY, USA : Association for Computing Machinery, 2015. — С. 1887—1901.

УДК 004.415.2

Н. В. Фролов (2 курс магистратуры),
С. А. Молодяков, д.т.н,, профессор

РАЗРАБОТКА SDK ДЛЯ ПРОСМОТРА ВЕРТИКАЛЬНЫХ ВИДЕО ПОД iOS

В настоящее время интенсивно развиваются все направления представления, преобразования и обработки мультимедийного контента. Все большей популярностью стала пользоваться лента с вертикальными видео. Так социальная сеть TikTok предоставляет такой функционал пользователям, аудитория социальной сети Tiktok насчитывает около 1.5 миллиарда пользователей. У маркетплейса Ozon есть вертикальная лента с отзывами пользователей на товары. Пользователи публикуют видео-отчет и он попадает в общую ленту товара. Также подобную технологию используют внутри приложения VK Музыка, добавив ленту с фрагментами из клипов исполнителей, что помогает продвигать авторов и их творчество в удобном для пользователя формате.

Анализ подобных систем показывает, что технология использования вертикального видео актуальна, однако известные решения закрытые и используются внутри компаний. И их нельзя масштабировать на какие-то свои проекты.

Целью данной работы является создание библиотеки для просмотра вертикальных видео под устройства с операционной системой iOS.

В рамках данной работы предполагается изучить известные библиотеки и подходы для работы с видео и разработки SDK, а также разработать собственную библиотеку под устройства с операционной системой iOS [1-3]. Данная библиотека должна отвечать следующим требованиям:

1) Возможность отобразить свой пользовательский или стандартный интерфейс поверх видео.
2) Возможность использовать стандартный или пользовательский источник данных.

Разработанное решение будет поставляться в виде SDK под операционные системы iOS разделенные на несколько уровней в соответствии с требованиями. Предполагаемый проект будет разработан на языке программирования Swift с применением IDE Xcode под операционную систему MacOS.

ЛИТЕРАТУРА

1. SwiftBook [Электронный ресурс]. Режим доступа: https://swiftbook.ru/ (17.03.2024)
2. Молодяков С. А. Применение функций FFmpeg в мультимедийных приложениях (100 примеров на Python). СПб.: ПОЛИТЕХ-ПРЕСС. – 2023. – 514 с. DOI 10.18720/SPBPU/2/i23-44
3. Swift Cookbook [Электронный ресурс]. Режим доступа: https://www.kodeco.com/ books/swift-cookbook/v1.0 (17.03.2024)

УДК 004.4'242

К. А. Шишин (2 курс бакалавриата),
Е. К. Куликов, к.ф.-м.н., доцент

ГЕНЕРАЦИЯ МОДУЛЬНЫХ ТЕСТОВ С КОНКРЕТИЗАЦИЕЙ ТИПОВ ДЛЯ SPRING-СПЕЦИФИЧНЫХ ПРИЛОЖЕНИЙ

Последние десятилетия активно разрабатываются решения для автоматизации тестирования. Они призваны помочь существенно повысить покрытие программы тестами, многократно снизив время, потраченное на их написание. Довольно известным является эксперимент [1] над проектом Coreutils, показывающий, насколько эффективной и полезной может быть автоматизация тестирования. Авторам удалось значительно увеличить покрытие кода тестами всего за несколько часов и узнать о дефектах, некоторые из которых существовали более пятнадцати лет.

Среди автоматических генераторов тестов хотелось бы выделить проект с открытым исходным кодом UnitTestBot Java [2]. Это инструмент командной строки и плагин для IntelliJ IDEA, предназначенный для генерации модульных тестов для Java приложений. Методологической основой проекта являются две техники анализа кода: символьное исполнение и фаззинг.

UnitTestBot Java способен создавать довольно неплохие тесты для “чистой” Java (что подтверждается, в частности, результатами соревнований генераторов тестов [3]), однако для этого языка написано множество фреймворков и библиотек, накладывающих дополнительные требования к анализу пользовательского кода и виду генерируемых тестов. Один из самых популярных [4] на сегодняшний день фреймворков – Spring [5]: он используется при разработке большинства промышленных проектов на Java. В связи с этим важно, чтобы UnitTestBot Java умел генерировать качественные тесты для приложений, использующих этот фреймворк.

Spring – многообразный инструмент. Однако его главными задачами являются внедрение зависимостей и инверсия управления. В Spring имеется контейнер DI/IoC, хранящий в себе управляемые объекты – бины. Конфигурация этого контейнера может определяться различными способами: на основе аннотаций в пользовательском коде, а также посредством специальных конфигурационных классов или xml файлов. Кроме того, в фреймворке предложен механизм реализации паттерна MVC, а подходы к тестированию

сервисов и контроллеров при написании тестов вручную, как правило, отличаются. Все это делает автоматизированное тестирование Spring-приложений весьма трудоемкой задачей.

Хотя на момент начала этого исследования UnitTestBot Java уже генерировал какие-то тесты для Spring-приложений, эти тесты сложно было назвать Spring-специфичными. В них, как и при тестировании обычных приложений на Java, мокируется поведение всех объектов, которые находятся вне тестируемого класса или пакета (в зависимости от выбранной стратегии мокирования). Однако тестирование Spring-приложений подразумевает использование меньшего количества моков благодаря информации о реальном поведении программы (в частности, конкретизации абстрактных типов), полученной в ходе анализа конфигурации приложения.

Таким образом, *целью* данной работы является поддержка генерации модульных тестов с конкретизацией типов для Spring-специфичных приложений в UnitTestBot Java.

Также важно отметить, что, поскольку модульные тесты (в отличие, например, от интеграционных) не предполагают запуск приложения и инициализацию его контекста в процессе их исполнения, а тестируют компонент в изоляции, то ожидается, что и в процессе генерации тестов, который включает в себя анализ пользовательской конфигурации, инициализация контекста приложения производиться не будет. В противном случае генерация тестов, сопряжённая с инициализацией контекста, может иметь непредсказуемые сайд-эффекты и быть опасной для пользовательских данных. Это создало дополнительные трудности при решении задачи генерации Spring-специфичных тестов.

В рамках проведенной работы был разработан безопасный инструмент анализа пользовательских конфигураций, а также осуществлена модернизация символьного движка.

Суть модернизации состоит в изменении механизма выбора типов символьных объектов: если раньше выбирался произвольный тип, который определялся SMT-решателем как удовлетворяющий ограничениям символьного пути, то теперь рассматриваются только те типы, которые используются в той конфигурации пользовательского приложения, которая выбрана для генерации тестов.

Spring-специфичную информацию для конкретизации типов движок получает из анализатора конфигураций пользовательского приложения. Сбор информации осуществляется с использованием инструментов самого Spring, во время инициализации контекста Spring-приложения.

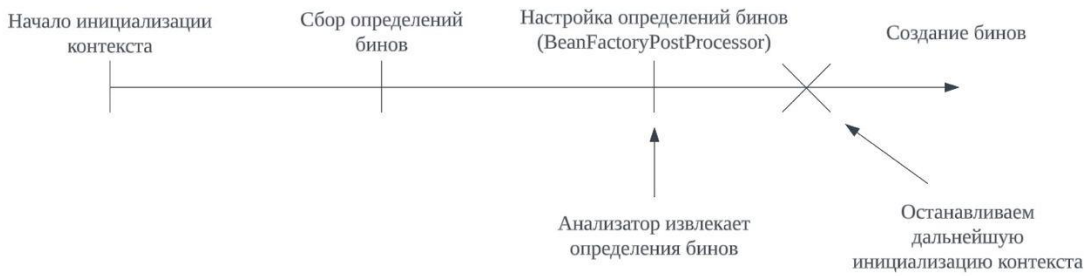


Рисунок 1 – Этапы инициализации контекста Spring-приложения

Инициализация контекста происходит в несколько этапов. И если этапы сбора и настройки определений бинов (bean definitions) являются безопасными для пользовательского приложения, то этап создания бинов может повлечь изменения в пользовательских данных. В связи с этим было принято решение реализовать собственный постпроцессор фабрики определений бинов (BeanFactoryPostProcessor), который на этапе настройки определений бинов собирает всю нужную анализатору информацию о типах, а затем останавливает дальнейшую инициализацию контекста, удаляя определения бинов из контекста, не допуская создания самих бинов.

Поскольку использование этого постпроцессора не может происходить на стороне пользовательского приложения, было решено динамически создавать свое Spring-приложение, эмулирующее конфигурацию пользовательского, на стороне UnitTestBot Java.

Однако в таком случае Spring не может напрямую получить доступ к конфигурациям из пользовательского приложения. Поэтому для запуска анализатора конфигураций используется расширенный classpath, содержащий помимо classpath'а самого анализатора еще и classpath пользовательского приложения.

В результате в инструменте UnitTestBot Java был поддержан механизм генерации модульных тестов с конкретизацией типов, позволяющий избегать излишнего мокирования и в ряде случаев генерировать тесты, которые более точно отражают реальное поведение программы и больше похожи на те, которые могли бы быть написаны вручную.

ЛИТЕРАТУРА

1. Cadar Cristian, Dunbar Daniel, Engler Dawson. KLEE: Unassisted and Automatic Generation of High-Coverage Tests for Complex Systems Programs // Proceedings of the 8th USENIX Conference on Operating Systems Design and Implementation. — OSDI'08. — USA : USENIX Association, 2008. – – P. 209–224.
2. UnitTestBot/UTBotJava: Automated unit test generation and precise code analysis for Java. [Электронный ресурс] Режим доступа: https://github.com/UnitTestBot/UTBotJava
3. D. Ivanov et al. UTBot at the SBFT 2023 Tool Competition // 2023 IEEE/ACM International Workshop on Search-Based and Fuzz testing — 2023 — P. 68-69
4. "Java Programming - The State of Developer Ecosystem in 2023 Info-graphic" — JetBrains: Developer Tools for Professionals and Teams. — 2023. [Электронный ресурс] Режим доступа: https://www.jetbrains.com/lp/devecosystem-2023/
5. Spring. [Электронный ресурс] Режим доступа: https://spring.io/

УДК 004.455.2

В. И. Шишкина (2 курс, магистратуры),
П. Д. Дробинцев, к. т. н., доцент

ПРОЦЕСС МИГРАЦИИ ОБЪЕКТОВ С СОХРАНЕНИЕМ СВЯЗЕЙ МЕЖДУ СЛУЖБАМИ КАТАЛОГОВ НА ПРИМЕРЕ ИНСТРУМЕНТОВ MS ACTIVE DIRECTORY И FREEIPA

В компаниях из различных областей производства, управления и коммерции используются программные инструменты для организации и управления правами доступа пользователей к общим информационным, техническим, аппаратным и другим ресурсам. Эти программные средства называются службами каталогов (*directory service*) [1].

Служба каталогов централизованно хранит информацию, необходимую для использования и управления ресурсами, упрощая процесс поиска и настройки для пользователей. Она управляет идентификацией и отношениями между ресурсами, позволяя им работать вместе.

На рынке служб каталогов доминирующее положение занимает продукт компании *Microsoft – MS Active Directory* [2]. Однако в отечественных компаниях использование этой системы может быть небезопасным из-за ее зарубежного происхождения, что может привести к отсутствию возможности своевременного обновления ПО, а также к рискам блокировки использования системы из-за отзыва лицензий или невозможности оплаты и продления.

Поэтому сейчас у отечественных компаний актуальной задачей является переход от многофункциональной, годами развивавшейся инфраструктуры домена на экосистеме зарубежных продуктов к устойчивым к санкционным рискам отечественным решениям.

Однако не все компании имеют финансовую возможность использовать лицензируемое ПО от российских производителей. В этом случае рассматривается вариант перехода на *opensource* решения.

Для того, чтобы перейти к рассмотрению процессов миграции от существующей службы каталогов к альтернативной, необходимо выбрать новую службу каталогов. В данной работе примем за используемый программный комплекс открытый программный продукт *FreeIPA* [3].

Задача ручного перехода и переноса данных – это крайне трудоемкий процесс, который не свободен от ошибок человека по невнимательности [4]. Для компании среднего размера (более 1000 человек) в общем случае трудоемкость ручного процесса переноса данных может оцениваться в 2-3 человеко-месяца. И еще 1-2 человеко-месяца на тестирование и проверку работоспособности всех информационных систем компании. Также необходимо учесть, что важна проработка вопроса миграции пользователей с сохранением прав доступа, полученных путем включения в доменные группы.

Несмотря то, что между схемой *LDAP Active Directory* [5] и *389 Directory Server LDAP*-схемой, используемой *FreeIPA*, существуют значительные различия, существует много атрибутов, которые одинаковы. Эти атрибуты просто синхронизируются между элементами *Active Directory* и *FreeIPA*, без изменений в имени атрибута или в формате значений. Некоторые атрибуты имеют разные названия, но все же их можно ассоциировать между собой.

В данной работе проведен краткий обзор служб каталогов *MS Active Directory* и *FreeIPA* и рассмотрен процесс синхронизации существующего «леса» в *MS AD* и нового – во *FreeIPA*.

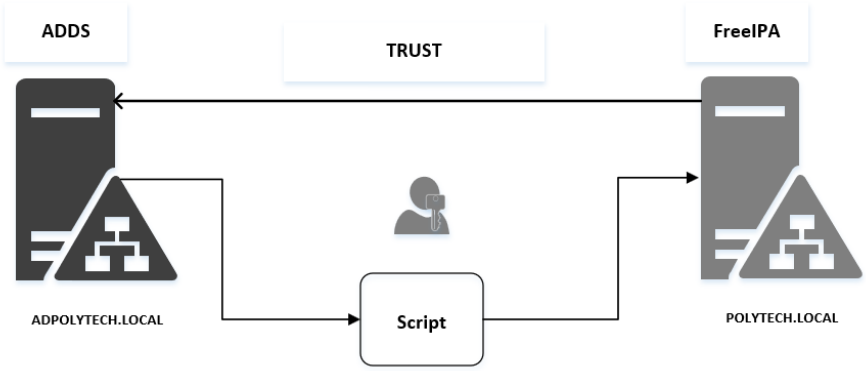


Рисунок 1 – Схема миграции пользователя

Из всего вышесказанного можно сделать вывод, что процесс замены действующей службы каталогов на новую можно построить следующим образом: необходимо организовать новый домен параллельно с существующим, перенести в него объекты из существующего, а все новые записи вести только в целевой системе.

Глобальной целью исследования является сокращение трудозатрат на работы по миграции данных между разными службами каталогов за счет разработки набора скриптов [6] для копирования объектов и их атрибутов.

Реализовав в программном средстве представленный подход можно ожидать снижения трудоемкости миграции данных в 10-15 раз. При этом трудоемкость тестирования качества миграции данных тоже снизится, но все равно останется существенной.

В ходе реализации данного подхода планируется проведение следующих экспериментов:

– копирование учетных записей пользователей в наполненный сторонними объектами домен FreeIPA;

– копирование учетных записей пользователей и групп доступа, в которые они включены, проверка связей;

– копирование целой ветки объектов, проверка связей.

ЛИТЕРАТУРА

1. Sheresh B., Sheresh D. Understanding Directory Services. System Research Corporation, 2002. ISBN 0-672-32305-2.
2. Коробко И. Каталог Active Directory. Поиск объектов с помощью PowerShell // Системный администратор – 2019. – № 4(197). – С. 36-38.
3. Кантер, Л. Построение корпоративной системы управления идентификационной информацией на базе FreeIPA / Л. Кантер // Пятнадцатая конференция разработчиков свободных программ: Тезисы докладов, Калуга, 28–30 сентября 2018 года / Ответственный редактор В.Л. Черный. – Калуга: ООО "МАКС Пресс", 2018. – С. 44-46.

4. Хитев А. Ю. Программа автоматической развертки сервера SAMBA и миграции записей из Active Directory в SAMBA // Бизнес-информатика: состояние, проблемы, перспективы: Сборник материалов Международной школы-семинара, Санкт-Петербург, 19–21 сентября 2013 года. – Санкт-Петербург: Санкт-Петербургский государственный университет, 2013. – С. 179-182.
5. Manav A. Thakur, Rahul Gaikwad. User identity & lifecycle management using LDAP directory server on distributed network. – International Conference on Pervasive Computing (ICPC), 2015.
6. Лавлинская О. Ю., Кузнецов В. И., Петрученко М. К. Скриптовый язык PowerShell как средство автоматизации работы с объектами службы active directory // Информационные технологии в строительных, социальных и экономических системах. – 2019. – № 1(15). – С. 55-58. – EDN CIRQCG.

УДК 681.3.06

А. Н. Шаровагин (4 курс бакалавриата),
В. А. Семенова-Тян-Шанская, к.т.н., доцент

РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ КОНСОЛИДАЦИИ РАЗНОРОДНЫХ ДАННЫХ О ТРАНСПОРТНЫХ СУДАХ

В стремительно развивающейся цифровой отрасли консолидация данных играет неотъемлемую роль. Ведь любая база данных это по факту консолидация десятков полей, взятых из разных, в том числе и книжных источников. Очищение и унификация данных, достигаемые в ходе консолидации, не только упрощают поиск нужной информации в сети, но и дают возможность её анализа, с целью получение из этого экономической прибыли [1].

В данной работе проведена консолидация данных с трёх открытых электронных ресурсов: сайт Морского регистра, сайт Речного регистра и сайт Водного транспорта.

Для этой цели были разработаны парсеры на языке программирования Python, с помощью которых данные извлекались из указанных ресурсов и сохранялись в виде текстовых .csv файлов [2].

Далее происходило объединение .csv файлов также с использованием языка Python, а также неотъемлемой для подобной задачи библиотеки pandas. Pandas – библиотека специализирующая на чтении, обработке, структуризации, моделировании и последующем анализе данных. В программе были использованы функции чтения и записи в файл, причём как в csv, так и xlsx формат, а также гибко настраиваемое объединение файлов данных с помощью функции merge. При объединении баз данных было выявлено большое количество ошибок и неточностей. Например, многочисленные повторяющиеся поля, имеющие схожее, но слегка отличающиеся название. Однако даже объединенные данные из трёх источников были далеки от своей конечной цели быть чистыми, упорядоченными, чтобы в конечном итоге отправиться в хранилище, закончив тем самым ETL процесс [3].

Процесс очистки данных сопровождался следующими подзадачами:

1. Разделение полей на более мелкие, в виду того, что поле само в себе содержит другие (рисунок 1)

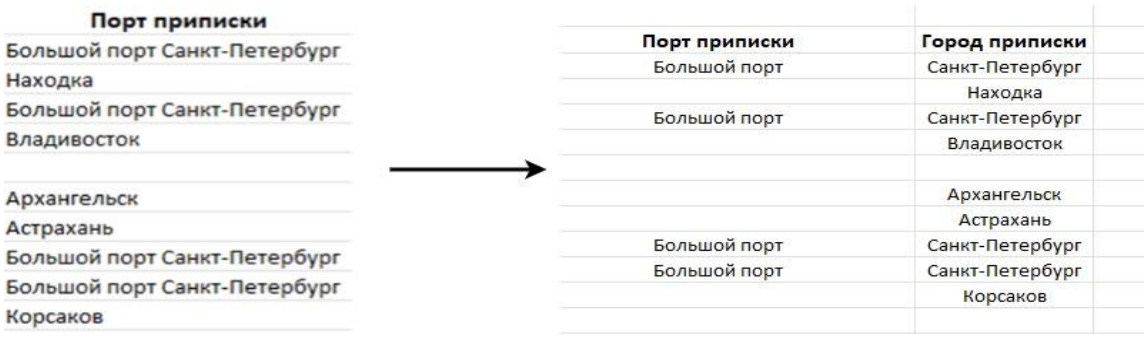


Рисунок 1 – Разделение полей на более мелкие части

2. Конвертация значения поля из строчного типа и типа даты в цифровой тип (рисунок 2)

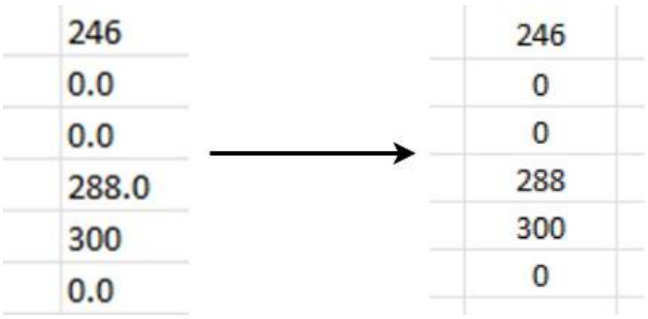


Рисунок 2 – Конвертация типа поля из строчного в цифровой

3. Превращение формул, записанных в поле в конечное значение (рисунок 3)

Рисунок 3 – Замена формул на значение

После произведенной очистки данные стали готовыми для загрузки в СУБД PostgreSQL, выступающую хранилищем в нашей задаче.

На рисунке 4 представлена общая схема работы:

Рисунок 4 – Общая схема ETL процесса

В результате работы программы был получен структурированный датасет, готовый к созданию на его основе витрин данных, а также к проведению анализа с целью оценить общую целесообразность постройки того или иного типа судна в определенном районе плавания.

ЛИТЕРАТУРА

1. Цехановский, В. В. Управление данными : учебник / В. В. Цехановский, В. Д. Чертовской. — Санкт-Петербург: Лань, 2022. — 432 с. — ISBN 978-5-8114-1853-4. — Текст: электронный // Лань: электроннобиблиотечная система. — URL: https://e.lanbook.com/book/212084 - страницы 105-106.
2. Введение в машинное обучение с помощью Python. Руководство для специалистов по работе с данными [Книга]. – Москва: Вильямс, 2017.
3. Белореченский Д.А., Сараев М.И., Семенова-Тян-Шанская В.А. Разработка веб-приложения для анализа характеристик судов с использованием методов ETL. Материалы научно-практической конференции студентов, аспирантов и молодых ученых «Современные технологии в теории и практике программирования», 26 апреля 2022г. СПб, Санкт-Петербургский Политехнический Университет, с.67-69.

УДК 004.93

А. В. Шпак (4 курс бакалавриата),
А. С. Луценко (4 курс бакалавриата),
В. С. Тутыгин, к.т.н., доцент

РАЗРАБОТКА МОБИЛЬНОГО ПРИЛОЖЕНИЯ НА PYTHON ДЛЯ ИДЕНТИФИКАЦИИ ЛИЦ С ПРИМЕНЕНИЕМ АЛГОРИТМА ВИОЛЫ-ДЖОНСА

В данной работе объединены два популярных направления: мобильная разработка на Python и идентификация человека.

Мобильная разработка стала особенно популярна в последнее время из-за развития мобильных устройств. Каждый год выпускаются новые модели телефонов, которые превосходят по мощности и другим характеристикам своих предшественников. Разработка мобильных приложений чаще всего осуществляется на Java или Swift, но тенденция может измениться, и Python будет также активно использоваться в мобильной разработке [1]. Этот язык программирования завоевал свою популярность низким порогом вхождения для новичков и простым синтаксисом, помимо мобильной разработки его применяют во многих других отраслях [2].

Решение задачи идентификации человека важна для общества. Любой вклад в развитие этого направления это вклад в повышение уровня безопасности и снижение уровня преступности. Идентифицировать человека можно разными способами, например, по отпечаткам пальцев, по радужной оболочке глаз или по голосу. Но идентификация по изображению имеет преимущество перед этими способами. В отличие от более надёжных способов – анализ отпечатков пальцев или радужной оболочки глаз, идентификация по фото не требует близкого контакта аппаратуры с проверяемым человеком.

Цель работы – реализовать алгоритм идентификации человека по изображения лица. Для систем идентификации на основе анализа изображения, где мы заранее можем предсказать положение лица в кадре и ракурс, можно не тратить время на этап поиска лица на изображении. В нашем случае без этого этапа не обойтись, поскольку мы не можем знать заранее как будет располагаться лицо на фото для анализа.

Для нахождения лиц на фото будет использоваться алгоритм Виолы-Джонса. Этот алгоритм работает с чёрно-белыми изображениями, сперва он ищет светлую вертикальную линию – нос человека, а затем – тёмную горизонтальную линию, которая является глазами. Наиболее эффективно это алгоритм работает с лицами, которые повёрнуты прямо к камере или с небольшим отклонением, примерно до 30 градусов [3]

На рисунке 1 представлена Use case диаграмма, по ней можно проследить работу приложения.

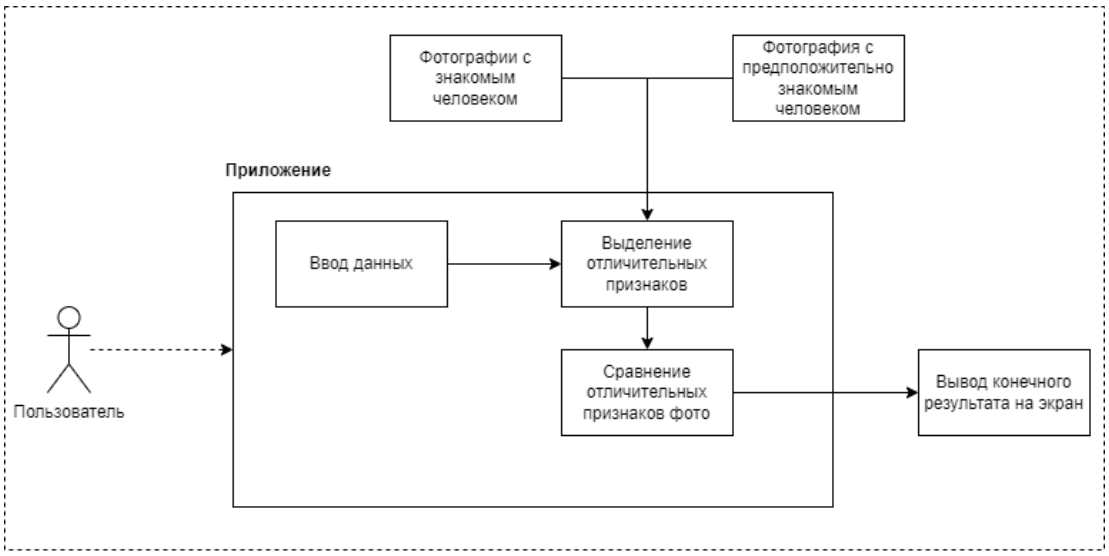


Рисунок 1 – Use case диаграмма

На вход приложения поступает имя знакомого человека, для подписи его на изображении, и несколько фотографий: одна для анализа и одна или более фотографий для сбора уникальных признаков человека, по которым его можно идентифицировать. Каждая фотография проходит следующие шаги: перевод в черно-белую палитру, нахождение лица алгоритмом Виолы-Джонса, выделение вектора особых признаков для идентификации. Далее вектора отличительных признаков всех фотографий сравниваются, и на основе этого сравнения делается вывод: один и тот же человек на фото или нет.

Для разработки этого приложения была выбрана библиотека Kivy. Библиотека Kivy в отличие от SL4A не ограничена средствами HTML-разметки для создания графического пользовательского интерфейса [4]. Приложения, написанные с помощью библиотеки Kivy, являются кроссплатформенными, то есть программное обеспечение пишется один раз, и работает на разных платформах без внесения изменений в код. Свойством кроссплатформенности также обладают приложения, написанные с помощью библиотеки BeeWare, но она до сих пор находится в разработке, что является существенным недостатком, так как это подразумевает наличие ошибок и отсутствие некоторых базовых функций [5].

В результате было получено мобильное приложение, с помощью которого можно идентифицировать человека по изображению его лица. Решение было реализовано с помощью языка программирования Python, библиотеки Kivy и OpenCV.

ЛИТЕРАТУРА

1. Буистов, В. В. Возможности и перспективы использования языка программирования Python в мобильной разработке / В. В. Буистов, В. С. Гречко, А. А. Андрейченко // НАУКА и ОБРАЗОВАНИЕ в ЭПОХУ ПЕРЕМЕН: ПЕРСПЕКТИВЫ РАЗВИТИЯ, НОВЫЕ ПАРАДИГМЫ : Материалы X Всероссийской научно-практической конференции, Ростов-на-Дону, 15 июля 2022 года. Том Часть 1. – Ростов-на-Дону: Общество с ограниченной ответственностью "Манускрипт", 2022. – С. 27-28. – EDN AAYDRK.
2. Python стали использовать для обучения программированию шире, чем Java // Открытые системы. СУБД. – 2014. – № 6. – С. 3-7m. – EDN SJKFLB.
3. Надежкин, А. И. Перспективы совершенствования современного алгоритма распознавания лиц на основе метода Виолы - Джонса / А. И. Надежкин, В. В. Мокеев // Новое слово в науке: перспективы развития. – 2015. – № 4(6). – С. 156-157. – EDN XXXRZX.
4. Изукаев, Е. В. Сравнительный анализ разработки мобильных приложений на Python / Е. В. Изукаев, Д. А. Шуклин // Альманах научных работ молодых ученых Университета ИТМО : в 5 т., Санкт-Петербург, 02–06 февраля 2016 года. Том 2. – Санкт-Петербург: Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики, 2016. – С. 229-230. – EDN ZOOTWP.

5. Суворова, М. В. Разработка мобильных бизнес-приложений на Python / М. В. Суворова // Научное обеспечение агропромышленного комплекса : Сборник статей по материалам XII Всероссийской конференции молодых ученых, Краснодар, 05–08 февраля 2019 года / Отв. за вып. А.Г. Кощаев. – Краснодар: Кубанский государственный аграрный университет имени И.Т. Трубилина, 2019. – С. 95-96. – EDN ZBPHDF.

УДК 004.4'22

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

ГЕНЕРАЦИЯ BAZEL BUILD ФАЙЛОВ ИЗ ИСХОДНЫХ ФАЙЛОВ НА ЯЗЫКЕ ПРОГРАММИРОВАНИЯ SCALA

В современном мире, когда проекты становятся все более объемными и сложными, ручное создание и поддержание файлов сборки становится все более трудоемким и подверженным ошибкам. Управление обширными зависимостями в современных проектах является сложной задачей. В разработке ПО сегодня требуется быстрое взаимодействие с другими продуктами или проектами, а длительный процесс настройки и сборки может серьезно замедлить этот процесс. Большие проекты часто содержат общие компоненты, используемые в разных частях, и разработчикам приходится тщательно отслеживать переиспользование кода при создании файлов сборки. Использование разных библиотек, фреймворков и платформ в проектах добавляет сложность сборке.

Инструментом для упрощения и ускорения процесса разработки является генерация *Bazel BUILD* [2] файлов из исходных файлов. Он предоставляет возможность автоматического создания файлов сборки на основе исходных кодов проекта. В качестве языка программирования исходных файлов был выбран мультипарадигмальный язык *Scala* [4]. Иными словами, такой инструмент будет автоматически анализировать *scala* код и создавать или обновлять соответствующие правила сборки для системы сборки *Bazel* [5], тем самым обеспечивая консистентность построения проекта, уменьшение ручного труда и минимизацию ошибок. Это существенно экономит время и силы разработчиков, позволяя им концентрировать свое внимание на разработке функционала и улучшении качества кода. Также наличие всех исходных текстов в одном правиле сборки не позволяет *Bazel* распараллеливать сборки и кэшировать их. Чтобы в полной мере воспользоваться преимуществами *Bazel*, необходимо написать несколько правил сборки и соединить их.

Таким образом, целью данной работы является создание генератора *BUILD* файлов, который будет анализировать исходный код на *Scala*, автоматически определять зависимости между файлами и целевыми объектами и генерировать необходимые *BUILD* файлы для *Bazel*.

За основу будет взят генератор файлов сборки Gazelle [1] для проектов на *Bazel*. Изначально *Gazelle* создавалась как генератор файлов сборки для проектов на языке *Go* [6], но генератор можно расширить для поддержки других языков и пользовательских наборов правил.

Чтобы расширить *Gazelle,* необходимо:

1. Реализовать языковой интерфейс. Этот интерфейс должен предоставлять перехватчики для генерации правил, анализа конфигурационных директив и разрешения импорта в *Bazel*.
2. Подключить написанную библиотеку в *Gazelle*.
3. Реализовать *Bazel* правила *Gazelle*, которые будет подменять двоичный файл по умолчанию на двоичный файл, созданный раннее для расширяемого языка *Scala*.

Языковой интерфейс был реализован на языке *Golang,* он выполняет следующее:

1. Проходит по исходному коду проекта, строит дерева файлов и папок, идентифицирует элементы, которые должны быть включены в процесс сборки. Для этой задачи была использована *Scalameta* - библиотека, которая предоставляет общие функции для обработки исходного кода на *Scala*, включая разбор кода, получение его абстрактного синтаксического дерева (*AST*) и многое другое.

2. Определяет библиотеки и зависимости. Необходимый импорт рассчитывается по следующему алгоритму:

– Если у правила есть атрибут *main_class*, это имя добавляется к импорту.

– Остальная часть импорта правил собирается из импорта файлов для всех исходных *scala* файлов в правиле. Выполняется синтаксический анализ *scala* файлов, собираются инструкции импорта, включая вложенный импорт, набор имен собирается путем обхода тела *AST*. Некоторые из этих имен являются вызовами функций, некоторые из них являются типами и т.д.

– Разрешаются имена. Набор символов в области видимости для файла создается из: подстановочных знаков, импортируемых файловыми пакетами. "Имена" представляют множество объектов в файле *scala*. Это может быть создание класса, вызов статического метода и другие подобные имена.

3. Обновляет файлы сборки. В *Bazel* используются отдельные правила для *Scala*, которые лучше подходят для обработки ее особенностей. Также обновленные зависимости не должны иметь повторений и образовывать циклические зависимости.

Таким образом, было реализовано расширение для *Gazelle* для языка программирования *Scala*, которое генерирует *Bazel Build* файлы на основе исходного *Scala* кода.

ЛИТЕРАТУРА

1. Gazelle build file generator. [Электронный ресурс] Режим доступа: https://github.com/bazelbuild/bazel-gazelle/tree/master (дата обращения 10.03.2024).
2. Bazel BUILD files. [Электронный ресурс] Режим доступа: https://bazel.build/concepts/build-files (дата обращения 11.03.2024).
3. Golang programming language. [Электронный ресурс] Режим доступа: https://go.dev/ (дата обращения 11.03.2024).
4. Одерски Мартин, Спун Лекс, Веннерс Билл, Соммерс Фрэнк. Scala. Профессиональное программирование. – Издательский дом «Питер», 26 мая 2022 – С. 608 – ISBN 978-5-4461-1914-1.
5. Benjamin Muschko. Getting Started with Bazel. – O'Reilly Media, 2021 – P 46 – ISBN-13 (pbk): 978-1-4842-5193-5.
6. Adam Freeman. Pro Go // The Complete Guide to Programming Reliable and Efficient Software Using Golang. – Apress, 2022 – ISBN 978-1-4842-7354-8.
7. Алдан А. Введение в генерацию программного кода. – Национальный открытый университет ИНТУИТ, 2016 – ISBN intuit086.

Секция «Программная инженерия: методы и алгоритмы теории программирования»

УДК 004.822

В. А. Андрейченкова (2 курс магистратуры),
Т. С. Горавнева, к.т.н., доцент

ИССЛЕДОВАНИЕ И РАЗРАБОТКА МОДЕЛЕЙ ПРЕДСТАВЛЕНИЯ ЗНАНИЙ ПРИ ВЫПОЛНЕНИИ СМЫСЛОВОГО ПОИСКА

Целью работы является исследование и разработка методов смыслового поиска в текстовых документах, основанных на продвинутых моделях представления знаний. В современном мире, где объемы информации растут экспоненциально, задача эффективного поиска и обработки текстовых данных становится особенно актуальной. Семантический поиск, опирающийся на понимание смысла текста, а не только на ключевые слова, обещает значительное улучшение результатов поиска, предоставляя пользователю более релевантную информацию.

Для достижения поставленной цели были рассмотрены существующие подходы к обработке естественного языка, методы представления знаний, а также способы их интеграции в системы смыслового поиска. Основное внимание было уделено анализу таких моделей представления знаний, как продукционные системы, семантические сети и фреймы.

Анализ существующих моделей и подходов позволяет выявить их достоинства и недостатки, а также определить направления для разработки улучшенных методов смыслового поиска [1]. Особое внимание уделяется возможностям интеграции этих моделей для создания более эффективных систем поиска, способных учитывать семантические связи в тексте и обеспечивать высокую точность и релевантность результатов.

Продукционные системы, основанные на правилах "если-то", показывают себя как гибкий инструмент для моделирования знаний, но сталкиваются с проблемами масштабируемости и управления сложностью правил. Семантические сети, представляющие знания в виде графов связей между понятиями, обеспечивают наглядность и универсальность, однако могут быть сложны в построении и модификации. Фреймы, описывающие структурированные описания объектов и ситуаций, предлагают удобный способ представления комплексных знаний, но также имеют свои ограничения, включая сложность и отсутствие строгой формализации.

В результате сравнения и анализа этих моделей было предложено направление для дальнейшего развития и совершенствования методов смыслового поиска. Данным направлением является интеграция различных моделей представления знаний для создания гибридных систем, способных объединять преимущества каждого подхода и обеспечивать более высокую точность и релевантность результатов поиска.

В частности, была предложена улучшенная модель, сочетающая элементы семантических сетей и продукционных систем, для более точного и контекстно-ориентированного анализа документов. Эта модель предполагает создание базы знаний на основе семантических анализов и продукционных правил, позволяющую углубленно работать с текстовыми запросами и документами. Этот подход начинается с построения семантической сети, которая визуализирует связи между понятиями и облегчает идентификацию смысловых аспектов в запросах пользователей. Затем, на основе сформулированных продукционных правил [2], система принимает решения, основываясь на анализе семантических связей,

выявленных в семантической сети, что позволяет ей более точно и эффективно находить и представлять релевантную информацию.

Предложенные дополнения к модели, такие как использование лемматизации и учет омонимов, паронимов, фразеологизмов, а также правил русского языка, направлены на повышение ее точности и адаптацию к спецификам обработки естественного языка [3]. Так, лемматизация позволяет учитывать разные формы слова как одно и то же понятие, а учет омонимов и паронимов помогает избежать путаницы в значениях слов, что крайне важно для точного семантического анализа.

Реализация разработанной модели на практике, проиллюстрированная анализом рабочей программы Санкт-Петербургского государственного морского технического университета по дисциплине «Интернет-технологии», демонстрирует ее потенциал в выявлении и классификации знаний, умений и навыков, указанных в документе. Однако выявленные в процессе тестирования недостатки, такие как дублирование информации в итоговых строках, указывают на необходимость дальнейшего усовершенствования модели и поиска новых подходов к ее оптимизации.

В качестве направлений для улучшения предлагается разработка более совершенных механизмов распознавания синонимичных выражений и семантических аналогий, что позволит уменьшить дублирование и повысить качество анализа. Внедрение онтологического подхода может стать следующим шагом в структурировании и классификации знаний, обеспечивая более глубокое понимание смысловых связей и иерархий в предметной области.

Представленный анализ подчеркивает важность изучения и развития методов смыслового поиска и представления знаний. Улучшенные модели, интегрирующие разнообразные подходы и технологии, открывают новые перспективы для эффективной обработки и анализа растущих объемов текстовой информации, способствуя более точному и полному пониманию смысла текстовых данных в различных приложениях, от академических исследований до коммерческих систем поиска и аналитики.

Средства создания разработки:

1. Среда разработки Visual Studio 2022;
2. Язык программирования C#;
3. NET 8;
4. Библиотека Cyriller.

ЛИТЕРАТУРА

1. Modeling Techniques for Knowledge Representation of Expert System: A Survey // Journal of Applied Computer Science; Mathematics 2019.
2. Универсальные зависимости // URL: https://universaldependencies.org/#language-ru – (дата обращения: 18.03.2024).
3. Семантический анализ для автоматической обработки естественного языка //URL: https://rdc.grfc.ru/2021/09/semantic_analysis/ – (дата обращения: 18.03.2024).

УДК 004.428

Д. А. Дорошин (4 курс бакалавриата),
А. И. Тышкевич, к.т.н., доцент

ОПТИЧЕСКИЕ ИНТЕРФЕЙСЫ ДЛЯ ИНТЕРПРЕТАЦИИ И МАНИПУЛЯЦИИ БИТОВЫХ ДАННЫХ

Компонуемость и абстракция представляют собой два фундаментальных столпа современной программной инженерии [4, 7]. Они являются основополагающими качествами высокоуровневого программирования, обеспечивая модульность, удобство повторного использования кода и гибкость при разработке сложных систем. На протяжении последних десятилетий индустрия программного обеспечения стремительно развивалась, и этот процесс

значительно ускорился благодаря улучшению методов абстракции и техник компоновки модулей.

Профункторная оптика представляет собой абстракцию, позволяющую элегантно и единообразно манипулировать сложными структурами данных [2, 9]. Концепт оптики больше всего развит в функциональных языках программирования, так как там он решает важные аспекты работы с данными, а библиотеки реализаций основываются на глубоких возможностях абстракции, которые те языки предоставляют [3].

Неотъемлемое место в создании современных программных решений занимают битовые операции. Эффективность работы программ зависит от способности обрабатывать и хранить данные на самом низком уровне, используя примитивные структуры данных, такие как биты, и техники оптимизации непосредственного доступа к памяти и элементарные манипуляции с данными [1, 5, 6, 8]. Несмотря на то, что в контексте битовых операций часто приходится работать с высокоспециализированными и жёстко заданными структурами данных, использование профункторной оптики допускает интересные перспективы в улучшении гибкости и масштабируемости таких программ.

Целью данной работы является анализ изменений объективных и субъективных метрик программного кода от использования подходов функционального программирования и профункторной оптики.

Для достижения этой цели необходимо решение следующих задач:

– Поиск примеров задач, которые хорошо отражают проблематику имплементации алгоритмов с битовыми операциями;
– Предложение набора объективных и субъективных метрик, по которым можно будет сравнивать программные реализации этих алгоритмов;
– Написание программ контрольной группы, реализующих данные алгоритмы;
– Предложение и реализация библиотеки профункторной оптики для битовых операций;
– Написание программ тестовой группы, использующих эту библиотеку;
– Сравнение выбранных метрик между программами двух групп.

Основным результатом выполнения задач будет программная библиотека для языка Haskell, которая интегрирует профункторную оптику с операциями по работе и интерпретации битовых данных. Содержание библиотеки будет отражать современное понимание проблематики и конкретные сложности, выявленные в задачах-примерах.

Эта библиотека сравнивается с набором существующих решений на различных языках, как и с использованием специализированных библиотек, так и без. Сравнение происходит по критериям и метрикам, которые выделяются как важные для решения рассматриваемых проблем.

ЛИТЕРАТУРА

1. Han S., Mao H., Dally W. J. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding // 2016.
2. Clarke B. [и др.]. Profunctor Optics, a Categorical Update // 2022.
3. Boisseau G. Understanding profunctor optics: a representation theorem (extended abstract) Programming ’18 / New York, NY, USA: Association for Computing Machinery, 2018.C. 30–32.
4. Foster J. University of Pennsylvania. Bidirectional Programming Languages. Department of Computer & Information Science, 2010.
5. Karandikar S. [и др.]. A Hardware Accelerator for Protocol Buffers MICRO ’21 / New York, NY, USA: Association for Computing Machinery, 2021.C. 462–478.
6. McCann J., Murphy Vii T. The fluint8 Software Integer Library // A Record of the Proceedings of SIGBOVIK 2018. 2018. C. 129–135.
7. Pickering M., Gibbons J., Wu N. Profunctor Optics: Modular Data Accessors // The Art, Science, and Engineering of Programming. 2017. № 2 (1). C. 7.
8. Ríos J. O. [и др.]. Dynamically Adapting Floating-Point Precision to Accelerate Deep Neural Network Training 2021.C. 980–987.
9. Steckermeier A. Lenses in Functional Programming.

УДК 004.05

Е. Ю. Кириченко, А. М. Голощапова (3 курс бакалавриата),
А. Ю. Крылатов, д.ф.–м.н., профессор

СИСТЕМА ТЕСТИРОВАНИЯ НАДЕЖНОСТИ СМАРТ-КОНТРАКТОВ НА ЯЗЫКЕ SOLIDITY

В настоящий момент блокчейн-сеть - уже не что-то новое и уникальное. Большинство работников в сфере IT так или иначе имеют понимание, что это такое, и многие хотели бы себя попробовать в данной сфере.

Ethereum - блокчейн, входящий сегодня в тройку самых популярных, а связанный с ним язык разработки смарт-контрактов - Solidity - обзавелся немалым количеством последователей. Существуют множество англоязычных статей в интернете по теме, различные курсы, в том числе и бесплатные.

Несмотря на это, в мире наблюдается нехватка solidity-разработчиков, хотя и это направление привлекает людей высокой заработной платой, новизной и наличием рабочих мест. Почему?

Начинающие solidity-разработчики пишут код с логическими ошибками из-за того, что не знают, какие уязвимости существуют в смарт-контракте. Также Solidity - активно развивающийся язык: часто выходят обновления версий, которые влияют на работоспособность кода, появляются новые библиотеки и устаревают текущие, устаревают примеры кода. Разработчики не знают, к кому обратиться за советом, а если и находят, то это стоит больших денег. Много времени уходит на поиск актуальной информации. Все эти проблемы могут демотивировать разработчика заниматься развитием своих навыков в этой сфере.

Таким образом, целью нашего проекта является помощь начинающим разработчикам в поиске уязвимостей кода, написанного на языке Solidity. Для достижения этой цели необходимо:

1. Обзор существующих решений.
2. Проведение их сравнительного анализа.
3. Реализация собственной идеи, объединяющей в себе основные преимущества имеющихся.
4. Доработка продукта на основе опыта использования первых разработчиков.

В результате пристального рассмотрения мы выявили ряд недостатков существующих решений и, учтя их, создали собственную концепцию - сайт с алгоритмом по обработке текста кода смарт-контракта на языке Solidity, направленный на поиск уязвимостей в контракте пользователя.

Начинающий разработчик получает бесплатный и удобный продукт, внутри которого с помощью простой логики быстро анализируется написанный им код. Результат работы алгоритма - ряд подсказок, в каких местах пользователю нужно развиваться далее. Разработчику больше не нужно тратить время на поиск своих слабых мест и качественной информации по теме - продукт делает это за него. Приятным бонусом для него будет вхождение в сообщество людей, интересующихся данной темой за счет ссылок на существующие форумы по проблемам, которые есть в его коде.

Основным элементом в сервисе является автоматизированная проверка без развертывания тестового блокчейна - мы исключаем человеческий фактор при проверке и делаем ее быстрой и бесплатной для начинающего разработчика.

ЛИТЕРАТУРА

1. Andrea M. Rozario, Miklos A. Vasarhelyi Auditing with Smart Contracts, The International Journal of Digital Accounting Research Vol. 18, 2018, pp. 1-27 ISSN: 2340-5058

2. Жуков А. С. Статический анализ смарт-контрактов на Solidity, Санкт–Петербургский государственный университет, бакалаврская работа
3. Vitalik Buterin (2014). Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform
4. Ethereum. [Электронный ресурс] Режим доступа: https://ethereum.org/en/
5. Как устроен Ethereum и смарт-контракты [Электронный ресурс] Режим доступа: https://vas3k.blog/blog/ethereum/
6. Solidity. [Электронный ресурс] Режим доступа: https://soliditylang.org
7. GitHub. [Электронный ресурс] Режим доступа: https://github.com/

УДК 004.4

А. А. Лекомцев (2 курс бакалавриата),
К. К. Смирнов, ст. преподаватель

ИССЛЕДОВАНИЕ ОПТИМИЗАЦИИ УМНОЖЕНИЯ В АРХИТЕКТУРЕ RISC-V С ПОМОЩЬЮ ИНСТРУКЦИЙ РАСШИРЕНИЯ BITMANIP

RISC-V – открытая расширяемая ISA (Instruction Set Architecture) с модульной структурой. Спецификация архитектуры доступна для свободного и бесплатного использования [1], что способствует формированию активного сообщества и созданию пользовательских расширений. Помимо этого, отличительной чертой архитектуры является ее простота в сравнении с другими ISA. Стандартный набор содержит в себе всего 47 базовых инструкций [2], а новая функциональность появляется за счет расширений, часть из которых принимается в стандарт.

Примерами дополнительных операций, которых нет в базовом наборе, являются умножение и деление целых чисел, инструкции для работы с числами с плавающей точкой, атомарные инструкции и битовые манипуляции. Например, bitmanip [3] – стандартное расширение RISC-V, которое включает в себя инструкции для быстрого вычисления адресов, для операций с битами и другие. Это расширение позволяет заменить несколько базовых инструкций одной, что зачастую эффективнее как по времени исполнения, так и по энергопотреблению [3]. Расширение является относительно новым, из-за чего компиляторы не всегда умеют генерировать более оптимальные команды.

В некоторых архитектурах, например x86, есть инструкции для эффективной генерации адресов (lea), которая позже стала применяться и для умножения. Например, умножение на 5 можно записать следующим образом: «LEA EAX, [EAX * 4 + EAX]». Аналогом данной инструкции в архитектуре RISC-V являются команды *sh1add*, *sh2add* и *sh3add* (в дальнейшем будет использоваться обозначение *shNadd* для описания какой-либо инструкции из этих трёх) из расширения bitmanip. Например, *sh1add A, B, C* помещает в регистр A результат «(*B*<<*1)* + *C*». Последовательность нескольких таких инструкций позволяет заменить умножение. Таким образом, умножение a_0 на 45 (инструкции «li a_1, 45; mulw a_0, a_0, a_1») можно записать с помощью последовательности:

sh3add a_0, a_0, a_0

sh1add a_0, a_0, a_0

Цель данного исследования – изучить, умножение на какие константы можно заменить последовательностью инструкций вида shNadd и проверить, используются ли эти способы в компиляторах GCC и Clang.

Для достижения поставленных целей необходимо перебрать все возможные комбинации и сгенерировать тесты на языке C, в которых ожидается соответствующая генерация. Проверка результата компиляции будет осуществляться с помощью утилиты для анализа кодогенерации компиляторов – riscv-check [4].

Все возможные умножения, которые можно заменить последовательностью двух битовых инструкций, уже использовались в компиляторах GCC 13.2.0 и Clang 18.1.0 (а именно умножение на [11; 19; 13; 21; 37; 25; 41; 73; 27; 45; 81]). Для перебора последовательностей из трёх инструкций был использован параметризированный шаблон следующего вида:

shXadd ? a_0 a_0

shYadd ? ? ?

shZadd a_0 ? ?

Подставляя на место «?» регистры a_0 и a_1 (вспомогательный-буфер регистр для хранения промежуточных значений) и на место букв X, Y, Z цифры из набора [1,2,3], можно получить конкретные комбинации, способные заменить умножение. Например, подстановкой можно получить последовательность

sh3add a_1 a_0 a_0

sh3add a_0 a_1 a_0

sh3add a_0 a_1 a_0

которая будет эквивалентна умножению на 585.

В результате перебора 1728 значений (27 вариантов значений (X,Y,Z) и 64 варианта регистров на месте «?») было обнаружено, что уникальных последовательностей только 95. Результаты, полученные с помощью утилиты riscv-check:

Компилятор	Тип int	Количество оптимизаций (из 95 тестов)
Clang 17.0.4	int32_t и int64_t	19
GCC 13.1.0	int64_t	95
GCC 13.1.0	int32_t	56

Все тестирования проводились при уровне оптимизации O3.

Работа выполнена в Лаборатории технологий программирования инфраструктурных решений, СПбГУ.

ЛИТЕРАТУРА

1. Официальный сайт RISC-V [Электронный ресурс]. Режим доступа: https://riscv.org/about/. [Дата обращения 16.03.2024]
2. RISC-V ISA [Электронный ресурс]. Режим доступа: https://riscv.org/wp-content/uploads/2017/05/riscv-spec-v2.2.pdf. [Дата обращения 16.03.2024]
3. RISC-V Bit-manipulation A, B, C and S Extensions [Электронный ресурс]. Режим доступа: https://five-embeddev.com/riscv-bitmanip/draft/bitmanip.html. [Дата обращения 16.03.2024]
4. GitHub-репозиторий [Электронный ресурс]. Режим доступа: https://github.com/vacmannnn/riscv-check. [Дата обращения 16.03.2024]

УДК 004.4'274

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

РАЗРАБОТКА МУЛЬТИМЕДИА-РЕДАКТОРА НА ОСНОВЕ БИБЛИОТЕКИ FFMPEG

Со времён «информационной революции» создание и распространение мультимедиа файлов приобрело колоссальный спрос. Фото/аудио/видеофайлы стали неотъемлемой частью повседневной коммуникации и составляют основу для многочисленных видов цифрового (и не только) творчества [1]. Актуальность работы обусловлена такими факторами, как развитие социальных сетей и платформ, что повышает потребность в программных средствах обработки медиафайлов и предоставлении расширенных возможностей к творческому выражению и росте потребления мультимедийных файлов во всех сферах массовой

культуры [2].

Приобретение лицензированных копий профессиональных редакторов мультимедиа требует немалых затрат (от 199 долларов США), что является далеко не самым рентабельным или экономически выгодным решением для среднего пользователя. Важно добавить, что у большинства всех остальных популярных инструментов для редактирования медиафайлов закрыт исходный код, в связи с чем невозможно выявить, работает ли заявленная разработчиками политика конфиденциальности в программном продукте и как проходит сбор данных пользователей, если не применить алгоритмы так называемой «обратной разработки» (англ. reverse engineering) [3].

С другой стороны, решения с открытым исходным кодом либо не удовлетворяют заявленным требованиям, либо являются лишь составной частью текущей задачи. Например, редактор ShotCut на C++ обладает весьма обширным функционалом, но в то же время его графический интерфейс слишком перегружен. Ко второму случаю относятся, например, различные проекты с GitHub (ozmartian/vidcutter или Zulko/moviepy) [4], которые нацелены на выполнение отдельных задач – склейка двух или более видео, их обрезка и т.п.

Цель работы заключается в повышении эффективности обработки и редактирования медиафайлов (изображений, видео и аудио) путём разработки мультимедиа-редактора на языке программирования Python с использованием библиотеки FFMPEG.

Для достижения цели необходимо решение следующих задач:

- Обзор существующих программных open-source решений в области обработки мультимедийных файлов с целью выявления их преимуществ и недостатков.
- Анализ возможностей библиотеки FFMPEG, принципов работы её функций и инструментов.
- Разработка новой методики редактирования мультимедиа файлов для обеспечения большей эффективности и качества обработки.
- Реализация мультимедиа-редактора на языке Python с использованием FFMPEG на основе предложенной методики и сформулированных требований.
- Демонстрация возможностей разработанного редактора.

В ходе работы будет применяться следующий стек технологий:

- Python - основной набор функций для реализации основной логики приложения [5,6];
- PyQt - графический интерфейс со встроенными виджетами и обработчиками событий [7];
- PyAV - обёрточная библиотека для FFMPEG, предоставляющая все необходимые возможности для редактирования и обработки файлов [8];
- JetBrains PyCharm - интегрированная среда разработки от JetBrains, ориентированная на разработку на Python.

Выбранный стек технологий предоставляет полный и достаточный инструментарий для реализации поставленных задач.

ЛИТЕРАТУРА

1. Ситдиков, А. А. Использование видеоредакторов в учебном процессе / А. А. Ситдиков, Е. Н. Малышева // Современные наукоемкие технологии. – 2014. – № 5-1. – С. 205-207. – EDN SAJIEF.
2. Алексеева, А. А. Анализ современных программных средств видеомонтажа и перспективы их использования / А. А. Алексеева // Современные ТЕХНОЛОГИИ: АКТУАЛЬНЫЕ ВОПРОСЫ, ДОСТИЖЕНИЯ и ИННОВАЦИИ: сборник статей XXXIII Международной научно-практической конференции, Пенза, 20 декабря 2019 года. – Пенза: "Наука и Просвещение" (ИП Гуляев Г.Ю.), 2019. – С. 136-138. – EDN DVCDSG.
3. Пономаренко, Т. М. Сравнительный анализ и обзор программных средств видеомонтажа / Т. М. Пономаренко // Научно-практические исследования. – 2019. – № 8-3(23). – С. 172-174. – EDN ITCNQW.
4. Система обработки больших данных для анализа событий репозитория GitHub / Н. В. Воинов, К Родригес Гарсон, И. В. Никифоров, П. Д. Дробинцев // Международная конференция по мягким вычислениям и измерениям. – 2019. – Т. 1. – С. 283-286. – EDN TSGWMD.

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. – EDN GEJJDU.
6. Python Documentation. [Электронный ресурс] Режим доступа: https://docs.python.org/3/
7. Qt Documentation. [Электронный ресурс] Режим доступа: https://doc.qt.io/qtforpython-6/
8. PyAV Documentation. [Электронный ресурс] Режим доступа: https://pyav.org/docs/stable/

УДК 004.4'22

А. И. Поликарпова (3 курс аспирантуры),
А. В. Самочадин, к.т.н. доцент

СИСТЕМА РЕИНЖИНИРИНГА ДЛЯ ГЕНЕРАЦИИ UML-ДОКУМЕНТАЦИИ КОДА X++.

Документация является неотъемлемой частью любого проекта. В документации, как правило, надо указывать основную логику работы тех или иных модулей, классов и прочих частей программного обеспечения. Большой проект, особенно тот, который разрабатывался, долгое время всегда обладает так называемым унаследованным кодом (legacy-кодом). Почему важно своевременно актуализировать документацию legacy-кода [1]:

– Появляются новые бизнес-кейсы, которые не отображены в изначальной документации.

– Сопровождение legacy-кода усложняется уходом из проектов старших разработчиков, отвечающих за тот код.

– Legacy-код чаще всего нельзя расширить «безопасным» способом.

– Ручной анализ legacy-кода требует большого количество ресурсов, как минимум несколько человеко-часов.

Особенно сложно разбирать код, написанный на специфическом и редком языке. Таком, например, как X++, который используется только в рамках разработки в Microsoft Dynamix AX [2].

Решением этой проблемы являются системы реинжиниринга и рефакторинга, позволяющие проводить анализ систем, переводя их на более высокий уровень абстракции (например, в UML-модели). Подобные системы позволяют быстрее понять ключевые идеи алгоритмов как для того, чтобы найти уязвимые места, так и для того, чтобы впоследствии провести миграцию кода на новую платформу [3, 4] и, впоследствии, при помощи таких систем показать, что при изменениях в коде основная бизнес-логика не была нарушена [5].

Под документацией UML-документации имеется в виду создание UML-диаграмм (в основном это диаграммы классов и диаграммы действий). Язык X++ построен на базе C# и является объектно-ориентированным языком, поэтому в плане лучшего понимания генерация диаграмм классов также необходима. На рисунке 1 представлено изображение с общим алгоритмом всей системы, которая формирует из исходного кода UML-документацию.

Рисунок 1 – Общий алгоритм реинжиниринга кода в UML-документацию.

Как видно из представленного рисунка сам по себе процесс весьма простой, наибольшую проблему здесь представляет сама реализация каждого из шагов (обозначены стрелками).

Первый шаг, как видно из схемы на рисунке 1, предполагает использование либо уже готового решения для конвертации кода в XML-формат, либо написание собственного парсера. Говоря об использовании уже готового решения, например, можно использовать open-source проект NRefactory для рефакторинга C#-проектов [6], так как сама по себе структура X++ похожа на язык C#. Но X++ обладает уникальными особенностями, которые отличают его от C#:

– В Microsoft Dynamix AX можно создавать таблицы БД, и они в терминах X++ являются классами, со своими методами со сложной логикой.

– В коде X++ можно напрямую писать SQL-запросы, синтаксис которых сильно схож с синтаксисом Transact-SQL.

Особенности синтаксиса X++ должны учитываться при рефакторинге и система реинжиниринга должна уметь обрабатывать и их, поэтому появляется необходимость в создании своей системы перевода исходного кода в XML-дерево. Microsoft Dynamix AX обладает возможностью сгруппировать различные виды объектов в один проект и выгрузить их в текстовый XPO-файл с особой неизменной структурой, учитывая это на рисунке 2 представлен процесс работы XML-парсера исходного кода X++.

```
Входные данные: исходный код проекта в XPO-формате.
Выходные данные: XML-деревья на каждый объект проекта.

Разбить файл на отдельные объекты
Для каждого объекта:
    Генерация XML-дерева
        Если создается XML-дерево на объект, для которого была создана пустышка:
            То обновить связь между объектом и его XML-деревом и забыть про связь-пустышку
        Если в исходном коде встречается объект, на который уже было создано XML-дерево:
            То создать в строке связь между объектом и его XML-деревом
            Иначе создать связь-пустышку и запомнить её
Сохранить все полученные структуры в файл
```

Рисунок 2 – Алгоритм преобразования исходного кода в XPO-файле в XML-деревья.

Разработка данного парсера предполагается на C# языке ввиду наличия большого спектра NuGet-библиотек, которые упростят разработку парсера.

Генерация XML-деревьев позволит получить представление о различных объектах в едином формате, следовательно, образование диаграмм на их основе становится тривиальной задачей.

ЛИТЕРАТУРА

1. M. Habringer, M. Moser, J. Pichler. Reverse Engineering PL/SQL Legacy Code: An Experience Report // IEEE International Conference on Software Maintenance and Evolution. – 2014. – DOI: 10.1109/ICSME.2014.93.
2. X++ [Электронный ресурс] Режим доступа: https://learn.microsoft.com/en-us/dynamics365/fin-ops-core/dev-itpro/dev-ref/xpp-language-reference.
3. U. Sabir, F. Azam, S. U. Haq, M. W. Anwar, W. H. Butt and A. Amjad. A Model Driven Reverse Engineering Framework for Generating High Level UML Models From Java Source Code // IEEE Access. – 2019. – Vol. 7. – P. 158931 - 158950. – DOI: 10.1109/ACCESS.2019.2950884.
4. Hiba Muneer Yahya Al-shakarjy, Dujan Basheer Taha. Software Code Refactoring: A Comprehensive Review // Journal of Education and Science. – 2023. – Vol. 32. – Issue 1. – P. 71-80. – DOI: 10.33899/EDUSJ.2023.137163.1298.
5. C. Deknop, J. Fabry, K. Mens, V. Zaytsev. Generating Customised Control Flow Graphs for Legacy Languages with Semi-Parsing // 2022 IEEE International Conference on Software Maintenance and Evolution (ICSME) . – 2022. – DOI: 10.1109/ICSME55016.2022.00072.
6. NRefactory [Электронный ресурс] Режим доступа: https://github.com/icsharpcode/NRefactory.

УДК 004.4

О. А. Пилецкий (3 курс бакалавриата),
В. И. Гориховский, к.ф.-м.н., доцент

РАЗРАБОТКА ИНСТРУМЕНТА ДЛЯ СРАВНЕНИЯ АЛГОРИТМОВ CONTENT DEFINED CHUNKING НА RUST

На данный момент существует множество систем хранения и обработки данных, и объемы информации, которые через них проходят, огромны. Чтобы уменьшить объем хранимых данных и, соответственно, стоимость обслуживания этих систем, используются различные подходы. Одним из них является дедупликация. Метод, основанный на том, что в хранилищах содержится большой объем повторяющихся данных. Благодаря хэшированию непрерывных отрезков данных, называемых чанками, возможно их переиспользование для сокращения хранимого объема.

Наиболее затратный по времени и нагрузке на центральный процессор этап дедупликации – разбиение на чанки, поскольку требует побайтового чтения и обработки всего файла. Большинство современных алгоритмов чанкирования относится к семейству Content Defined Chunking, которые разделяют файл на чанки в зависимости от его содержимого, например, FastCDC [1] и SuperCDC [2]. Поскольку многие алгоритмы чанкирования были разработаны в последнем десятилетии, нет работ, в которых проводится полное сравнение основных алгоритмов. Более того, в существующих работах сравнивалась алгоритмическая эффективность некоторых чистых алгоритмов без привязки к работе файловой системы. Кроме того, сравнивались только реализации на языках C/C++.

В последние годы набирает популярность язык программирования Rust [3], используемый, как безопасная замена C/C++ с сопоставимой с ними скоростью работы. Он активно развивается, и для него существует множество различных библиотек, благодаря чему его можно применять для самых разных целей. Лучше всего он показывает себя в системном программировании, например, при разработке низкоуровневых библиотек и приложений, таких как файловые системы. Примерами файловых систем с открытым исходным кодом, написанных на Rust, являются ZboxFS [4] и sandboxfs [5]. ZboxFS встраивается в приложение и предоставляет зашифрованную виртуальную файловую систему. Ее целью является безопасное и надежное хранение данных, в том числе с помощью дедупликации.

На данный момент существует несколько библиотек на Rust, таких как fastcdc [6] и cdcchunking [7], реализующих отдельные алгоритмы чанкинга, однако их достаточно мало, и они не покрывают все изобилие существующих алгоритмов. Целью работы является разработки методологии и инструментария сравнения эффективности алгоритмов чанкинга на Rust. Для ее достижения были поставлены следующие задачи:

1. Выполнить обзор алгоритмов Content Defined Chunking.
2. Реализовать избранные алгоритмы на Rust.
3. Встроить в ZboxFS трейт для возможности менять используемый алгоритм чанкинга в ходе работы программы.
4. Провести первичное исследование эффективности алгоритмов, интегрированных в ZboxFS.

В ходе обзора были рассмотрены десять алгоритмов: RabinCDC, LeapCDC, FastCDC, SuperCDC, UltraCDC, GearCDC, RapidCDC, QuickCDC, Double Sliding Window, Speculative Jump. Из них были выбраны первые пять как имеющие наилучшую теоретическую эффективность. Они были реализованы и встроены в ZboxFS. После этого ZboxFS был модифицирован так, чтобы позволить пользователю менять алгоритм чанкинга при использовании. Для этого был создан трейт (аналог интерфейса) для алгоритмов и его реализация для каждого из них.

Для сравнения алгоритмов основными критериями являются скорость работы, коэффициент дедупликации (размер данных до / размер данных после), средний фактический

размер чанка. Были рассмотрены следующие сценарии: запись файла, чтение файла, копирование файла внутри файловой системы. Были написаны бенчмарки, для сравнения отдельных алгоритмов использовались тесты в отдельном модуле, содержащем чанкер.

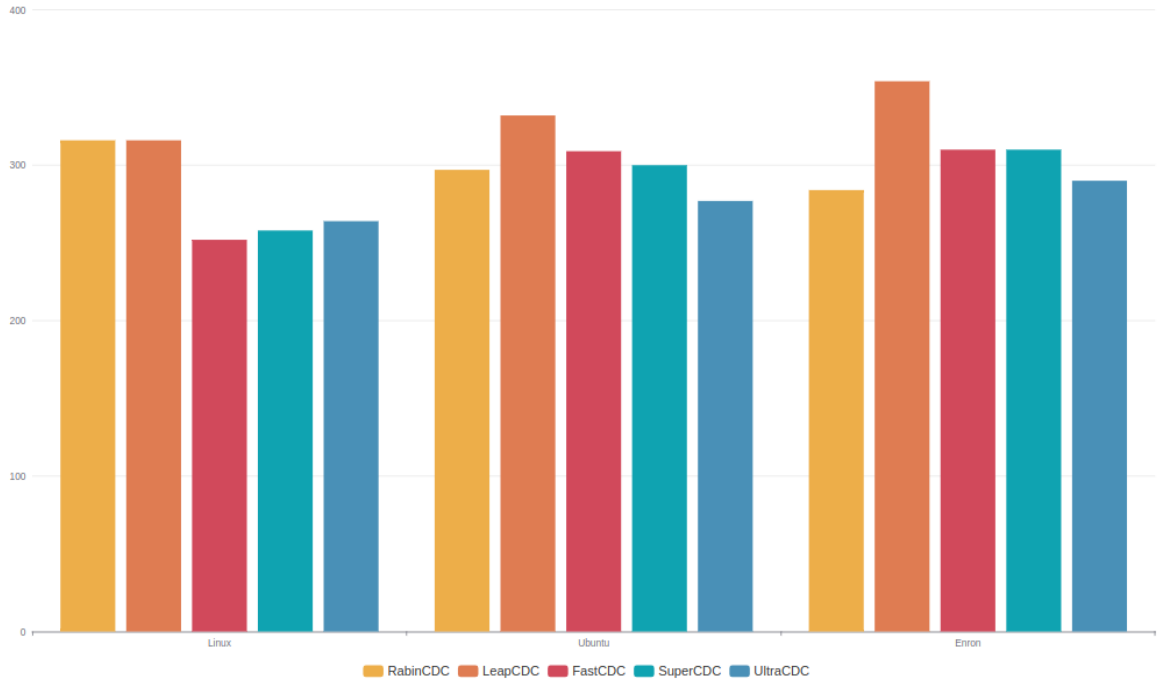


Рисунок 1 – скорость операции записи больших файлов в систему, МБ/с

Эксперименты показали высокую производительность реализаций алгоритмов на Rust. Также они показали, что оптимистичные теоретические оценки, представленные в статьях, не распространяются на реализации на Rust. Интеграционное тестирование, результаты которого представлены на рисунке 1, показало, что ZboxFS может быть минимально использован для сравнения алгоритмов, однако точность результатов сильно размывается из-за крайне больших накладных расходов на шифрование. В результате чего при смене алгоритмов производительность практически не меняется, хотя при рассмотрении их в отдельности некоторые алгоритмы достигали 20-кратного прироста в скорости. Это приводит к необходимости создания стенда файловой системы на Rust для проведения более точного сравнения алгоритмов чанкирования. Сейчас она находится в разработке.

ЛИТЕРАТУРА

1. W. Xia et al., "The Design of Fast Content-Defined Chunking for Data Deduplication Based Storage Systems," in IEEE Transactions on Parallel and Distributed Systems, vol. 31, no. 9, pp. 2017-2031, 1 Sept. 2020, doi: 10.1109/TPDS.2020.2984632
2. B. Wan, L. Pu, X. Zou, S. Li, P. Wang and W. Xia, "SuperCDC: A Hybrid Design of High-Performance Content-Defined Chunking for Fast Deduplication," 2022 IEEE 40th International Conference on Computer Design (ICCD), Olympic Valley, CA, USA, 2022, pp. 170-178, doi: 10.1109/ICCD56317.2022.00034
3. Rust Programming Language [Электронный ресурс] Режим доступа: https://www.rust-lang.org/
4. ZboxFS [Электронный ресурс] Режим доступа: https://github.com/zboxfs/zbox
5. sandboxfs [Электронный ресурс] Режим доступа: https://github.com/bazelbuild/sandboxfs
6. fastcdc Rust Package [Электронный ресурс] Режим доступа: https://crates.io/crates/fastcdc/
7. cdcchunking Rust Package [Электронный ресурс] Режим доступа: https://crates.io/crates/cdchunking

УДК 004.428.4

С. А. Селиванова (2 курс магистратуры),
А. Д. Ковалев, к.т.н., доцент

АВТОМАТИЗАЦИЯ ПРОГНОЗИРОВАНИЯ ПОЛИТИЧЕСКИХ КОНФЛИКТОВ НА ОСНОВЕ МЕТОДОВ АНАЛИЗА ВРЕМЕННЫХ РЯДОВ

Введение. В условиях современной политической обстановки создание эффективных инструментов для предсказания потенциальных политических конфликтов становится актуальной проблемой. Несмотря на существующие методы анализа и прогнозирования, имеется потребность в разработке более точных инструментов, способных адаптироваться к динамике современных геополитических событий.

Цель работы заключается в разработке приложения, способного предоставлять прогнозы политических конфликтов на основе актуальных данных о событиях в мире.

Наборы данных и известные методы решения. На данный момент существует несколько открытых и периодически обновляемых источников данных о мировых политических событиях: UCDP (Uppsala Conflict Data Program) [1] и ACLED (Armed Conflict Location and Event Data Project) [2]. UCDP ежемесячно предоставляет информацию о вооруженных конфликтах, включая данные о местоположении, характере событий, сторонах конфликта и уровне насилия. Этот набор данных охватывает различные аспекты конфликтов по всему миру с 1946 по 2024 год. ACLED фокусируется на детализированных событиях (бои, взрывы, беспорядки и т.д.), еженедельно предоставляя точные координаты и подробную информацию о каждом инциденте с 1997 по 2024 год. Этот набор данных обеспечивает более детальный взгляд на характер и распределение конфликтных событий [3].

Готовые инструменты позволяют оценить вероятность существования конфликта, либо количество случаев насилия определенных типов. Данные, предсказанные с помощью инструмента «ACLED CAST» публикуются в виде таблиц с количеством предсказанных событий для конкретного месяца и административной единицы страны на 6 месяцев вперед. В конце месяца публикуются актуальные данные. Таким образом можно отследить точность работы данного инструмента. [4]. Также существует инструмент «VIEWS», разработанный в Уппсальском университете в Швеции. В модели инструмента используются данные проекта «UCDP», регистрирующего случаи организованного насилия. [5]. VIEWS использует различные методы машинного обучения и делает предсказания в интервале от 1 до 36 месяцев.

Существующие исследования. Большой вклад в развитие данной темы вносит проект EC-LISTCO [6], который исследует, как сочетание различных факторов риска способствует возникновению переломных моментов, вызывающих насильственные конфликты и угрозы безопасности государства и его граждан.

Исследователи Х. Мюллер и К. Раух [7] предлагают методологию прогнозирования вооруженного конфликта на основе использования журналистских публикаций. С помощью методов машинного обучения производится интерпретация текста газетных статей и их классификация по темам, после чего применяется регрессия на панельных данных для прогнозирования начала конфликта.

В работе [8], сфокусированной на анализе политических конфликтах в Бангладеш, использован наивный байесовский алгоритм для создания прототипа набора данных, который затем был обучен с применением алгоритма random forest для построения модели прогнозирования.

Работа [9] вводит новый набор данных для прогнозирования интенсивности конфликтов, включающий семь социо-экономических и политических показателей. Авторы тестируют различные методы предсказания, такие как линейный дискриминантный анализ, деревья классификации и регрессии, метод k-ближайших соседей, случайный лес и несколько видов искусственных нейронных сетей.

Постановка задачи исследования. В рамках данного исследования ставится задача на основе изученных технологий и инструментов создать программное средство, способное с минимальными ошибками предсказывать количество конфликтных событий в стране за определенный период с использованием методов машинного обучения. Требуемая функциональность приложения заключается в возможности загрузки данных о политических событиях из внешних ресурсов, прогнозирования количества конфликтов на 6 месяцев вперед, представления пользователю интерфейса для визуализации результатов прогноза. Ожидается успешная реализация веб-приложения с удобным интерфейсом для загрузки данных и получения прогнозов по политическим событиям.

Результаты исследования. В данной работе рассмотрены основные методы и принципы анализа политических конфликтов с помощью машинного обучения. Изучены наиболее распространенные наборы данных о событиях, актуальные алгоритмы машинного обучения. Также произведен обзор существующих исследований на тему прогнозирования политических конфликтов.

В результате создано веб-приложение с бэкендом, написанным на языке Java (фреймворк Spring), и фронтендом на TypeScript (React). Для прогнозирования временного ряда событий выбраны алгоритмы ARIMA и LSTM [10] с реализациями на языке Java. Данные о событиях взяты из набора UCDP GED и преобразованы для использования алгоритмов временных рядов (вычислено количество событий для каждого месяца). Для подбора параметров алгоритма ARIMA использован метод grid search, а также кросс-валидация для обучения LSTM.

Полученный инструмент предоставляет возможность получать предсказания о количествах конфликтов во всех странах Африки на 6 месяцев вперед. Далее в рамках работы над проектом планируется исследовать точность обученных моделей и оценить полученное решение.

ЛИТЕРАТУРА

1. Uppsala Conflict Data Program (ucdp). [Электронный ресурс] Режим доступа: https://ucdp.uu.se/
2. The armed conflict location event data project (acled). [Электронный ресурс] Режим доступа: https://acleddata.com//dashboard
3. K. Eck, In data we trust? A comparison of UCDP GED and ACLED conflict events datasets // Cooperation and Conflict. – 2012. – Vol. 47. – No. 1. – P. 124–141.
4. ACLED. “ACLED Conflict Alert System.” Armed Conflict Location & Event Data Project (ACLED). [Электронный ресурс] Режим доступа: https://acleddata.com/early-warning-research-hub/conflict-alert-system/
5. Hegre, Håvard, Curtis Bell, Michael Colaresi, Mihai Croicu, Frederick Hoyles, Remco Jansen, Maxine Ria Leis, Angelica Lindqvist-McGowan, David Randahl, Espen Geelmuyden Rød, Paola Vesco. ViEWS 2020: Revising and evaluating the ViEWS political Violence Early-Warning System // Journal of Peace Research. – 2021. – Vol. 58. – No. 3. – P. 599–611.
6. Bressan, S., Nygård, H. M. & Seefeldt, D. Forecasting And Foresight: Methods for Anticipating Governance Breakdown and Violent Conflict // EU-LISTCO. – 2019. – No. 2.
7. Mueller H., Rauh C. Reading between the Lines: Prediction of Political Violence using Newspaper Text // American Political Science Review. – 2018. – No. 2. – P. 358–375.
8. H. M. M. Hasan, A. Ahnaf and N. Hossain. Prediction of Political and Local Conflicts in Bangladesh: An Event Analysis // International Conference on Science & Contemporary Technologies (ICSCT), Dhaka, Bangladesh. – 2021. – P. 1-6. – DOI 10.1109/ICSCT53883.2021.9642517.
9. Ettensperger, F. Comparing supervised learning algorithms and artificial neural networks for conflict prediction: performance and applicability of deep learning in the field // Qual Quant – 2020. – Vol. 54. – P. 567–601.
10. S. Chen, R. Lin and W. Zeng. Short-Term Load Forecasting Method Based on ARIMA and LSTM // 2022 IEEE 22nd International Conference on Communication Technology (ICCT), Nanjing, China. – 2022. – P. 1913-1917. – DOI 10.1109/ICCT56141.2022.10073051.

УДК 004.852

Е. А. Судаков (4 курс бакалавриата),
О. Г. Малеев., к.т.н, доцент

ПРИМЕНЕНИЕ БИНАРНЫХ НЕЙРОННЫХ СЕТЕЙ ДЛЯ КЛАССИФИКАЦИИ ИЗОБРАЖЕНИЙ МЕХАНИЧЕСКИХ ТРАНСПОРТНЫХ СРЕДСТВ

Глубокие сверточные нейронные сети являются одним из самых распространенных методов решения проблем компьютерного зрения, таких как классификация изображений, обнаружение объектов, сегментация изображений. Несмотря на то, что постоянно представлялись и развивались новые модели сверточных нейронных сетей, их архитектура не слишком сильно изменилась по сравнению с тем, что было до 2017 года. И поскольку модели сверточных нейронных сетей становятся крупнее, что требует большей вычислительной мощности и объема памяти, возникают трудности с их размещением на платформах с ограниченными ресурсами, таких как смартфоны и небольшие устройства с малой вычислительной мощностью и ограниченной памятью. Однако такая потребность только растёт с течением времени, так как всё стремится к цифровизации, использованию быстрых и высокоточных моделей искусственного интеллекта в различных сферах нашей жизни. Бинарные нейронные сети также могут применяться в области классификации изображений, детектирования объекта на изображении [1] и других задачах компьютерного зрения, так как являются бинаризованными вариантами сверточных полноточных нейронных сетей.

В рамках работы исследуются современные способы решения проблемы использования глубоких нейронных сетей на маломощных устройствах с помощью применения методов бинаризации.

В бинарных нейронных сетях [2] активации и веса приводятся к двоичным значениям (-1, 1), что теоретически позволяет в 32 раза сократить объем занимаемой моделью памяти и в 58 раз увеличить скорость работы обученной бинарной нейронной сети по сравнению с полноточной (32-битной) сверточной нейронной сетью.

Таким образом, целью данной работы является исследование подходов бинаризации глубоких нейронных сетей, реализация наиболее прогрессивных методов для задачи классификации изображений механических транспортных средств, анализ и сравнение характеристик данных решений с полноточной (32-битной) сверточной нейронной сетью.

Для достижения этой цели необходимо решение следующих задач:

1. Обзор существующих подходов реализации бинарных нейронных сетей.
2. Проведение сравнительного анализа подходов.
3. Выбор и подготовка датасета для изучения поставленной задачи.
4. Подготовка стенда для обучения и проведения экспериментов для выявления метрик рассматриваемых подходов.
5. Реализация методов.
6. Демонстрация результатов исследования.

В ходе сравнительного анализа были выбраны наиболее эффективные методы бинаризации: Bi-Real Net [3], MeliusNet [4], AdaBin (Adaptive Binary) [5].

Bi-Real Net улучшает репрезентативную способность сети, путем введения связи между полноточными (32-битными) активациями перед применением функции знака для бинаризации и выходом того же слоя после операции бинарной свертки и пакетной нормализации с помощью оператора сложения, вводит более близкую аппроксимацию функции знака, что увеличивает эффективность ее производной.

MeliusNet привносит улучшения в качество и емкость признаков путем архитектурных изменений сети, используя блоки Dense Block и Improvement Block.

AdaBin также направлен на увеличение репрезентативной способности, которая страдает после бинаризации весов и активаций, за счёт использования адаптивных двоичных наборов,

которые не ограничены фиксированными значениями для разных слоев, для получения наилучшего соответствия вещественному распределению.

Методы реализованы на языке Python в Google Colab с применением библиотеки PyTorch[6].

В виде базовой топологии сверточной нейронной сети выбрана ResNet (Residual Network) [7]. Данная архитектура показала широкие возможности для получения высокой точности с возможностями масштабирования числа слоев

На основе выбранной топологии реализованы методы Bi-Real Net, Melius Net, AdaBin Net и получены характеристики.

Таблица 1 – Характеристики изучаемых подходов на сконструированном датасете

Название	Топ-1 точность	Топ-5 точность
Bi-Real Net	82.5%	96.1%
Melius Net	84.0%	97.2%
AdaBin Net	81.6%	97.1%

При проведении исследования эффективности использования бинарных нейронных сетей для задачи классификации изображений было решено взять 15 классов из набора данных ImageNet, чтобы сконструировать свой собственный набор данных с механическими транспортными средствами. Выбранные классы изображений: авиалайнер, дирижабль, машина скорой помощи, фура, скоростной поезд, такси, электровоз, пожарная машина, мусоровоз, гольф-кар, пикап, гоночный болид, дом на колесах, спортивный автомобиль, троллейбус. В результате чего был получен датасет из 15 классов механических транспортных средств, включающий в себя 19484 тренировочных изображений, и 750 валидационных. Суммарно 20234 изображений.

ЛИТЕРАТУРА

1. Исследование эффективности применения бинарных нейронных сетей при детектировании объекта на изображении / Королев Д. О, Малеев О. Г. URL: https://elibrary.ru/item.asp?id=54646981.
2. Binary Neural Networks: A Survey / Haotong Qina, Ruihao Gonga, Xianglong Liu, Xiao Baie, Jingkuan Songc, Nicu Sebed. Текст электронный. URL: https://arxiv.org/pdf/2004.03333.pdf
3. Bi-Real Net: Enhancing the Performance of 1-bit CNNs With Improved Representational Capability and Advanced Training Algorithm / Zechun Liu, Baoyuan Wu, Wenhan Luo, Xin Yang, Wei Liu, Kwang-Ting Cheng. Текст электронный. URL: https://arxiv.org/pdf/1808.00278.pdf.
4. MeliusNet: An Improved Network Architecture for Binary Neural Networks / Joseph Bethge, Christian Bartz, Haojin Yang, Ying Chen, Christoph Meinel. Текст электронный. URL: https://openaccess.thecvf.com/content/WACV2021/papers/Bethge_MeliusNet_An_Improved_Network_Architecture_for_Binary_Neural_Networks_WACV_2021_paper.pdf.
5. AdaBin: Improving Binary Neural Networks with Adaptive Binary Sets / Zhijun Tu, Xinghao Chen, Pengju Ren , Yunhe Wang. Текст электронный. URL: https://arxiv.org/pdf/2208.08084.pdf.
6. PyTorch Documentation. URL: https://pytorch.org/docs/stable/index.html.
7. Deep Residual Learning for Image Recognition/ Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun. Текст электронный. URL: https://arxiv.org/pdf/1512.03385.pdf.

УДК 004.932

Д. О. Слепов (4 курс бакалавриата),
С. В. Хлопин, к.т.н., доцент

МОДИФИКАЦИЯ АЛГОРИТМА НАХОЖДЕНИЯ КОНТУРОВ НА ИЗОБРАЖЕНИИ

Актуальность алгоритмов нахождения контуров заключается в их ключевой роли в обработке изображений и анализе данных в области компьютерного зрения. Они используются для выделения структурной информации, такой как края объектов или границы различных областей на изображении, что является важным этапом для множества приложений, включая распознавание образов, сегментацию изображений, трекинг объектов, автоматическое определение и классификацию объектов, и многие другие. В современных

задачах обработки изображений возникает потребность в высокой точности, скорости и универсальности в различных условиях и сценариях применения.

Проведя анализ наиболее популярных алгоритмов нахождения контуров, таких как - Детектор границ Кэнни, Оператор Собеля, Векторный оператор Лапласа, Оператор Прюитт, Перекрестный оператор Робертса и Алгоритм Сузуки85, я пришел к выводу, что каждый из рассмотренных методов имеет свои ограничения и недостатки. Например, алгоритмы, такие как Кэнни и Собеля, обладают высокой точностью, однако их скорость работы может быть недостаточной для обработки больших объемов данных в реальном времени. В то время как операторы Лапласа, Прюитта и Сузуки85 демонстрируют более высокую скорость обработки, их точность и способность к шумоподавлению оказываются менее эффективными по сравнению с другими методами [3]. На основе этих данных мной было принято решение модифицировать один из уже существующих алгоритмов для обеспечения оптимального баланса между скоростью и точностью работы алгоритма.

Для обеспечения оптимального баланса между скоростью и точностью был выбран вариант модификации алгоритма Сузуки85. Алгоритм основан на пиксельном анализе бинарного изображения. С помощью данного метода можно получать список контуров, а также их иерархию относительно друг друга [2]. Оригинальный способ нахождения границ состоит из 3 шагов: добавление границы к изображению в виде рамки, состоящей из нулей, в результате чего размеры изображения станут $((h+2)*(w+2))$, где h и w высота и ширина; сканирование изображения слева направо по каждой строке в поисках точки, принадлежащей контуру. Принадлежность определяется условием: $D(x,y)=1$ или же если точка является внутренней частью: $(D(x-1,y)\neq 0$ и $D(x,y)\neq 0$ и $D(x+1,y)\neq 0)$; определение принадлежности найденного контура к существующей иерархии путем обхода каждой из восьми соседних точек против часовой стрелки и нахождение уже существующих контуров [1]. На выходе метода мы получаем массив контуров и исходную картинку, где каждый пиксель контура представлен номером этого контура.

В ходе подробного анализа данного метода были выявлены следующие ошибки: алгоритм не распознает горизонтальные линии шириной в 1 или 2 пикселя; строится неправильная иерархия контуров.

В качестве решения для исправления первой ошибки была предложена модификация, которая заключается в последовательном обходе изображения: первый проход осуществляется горизонтально, от левого края к правому, а затем второй проход выполняется вертикально, от верхнего края к нижнему. Данное нововведение позволило обойти ограничение метода, связанное с поиском точки начала контура, не требуя при этом значительных дополнительных вычислительных ресурсов или времени (рисунок 1).

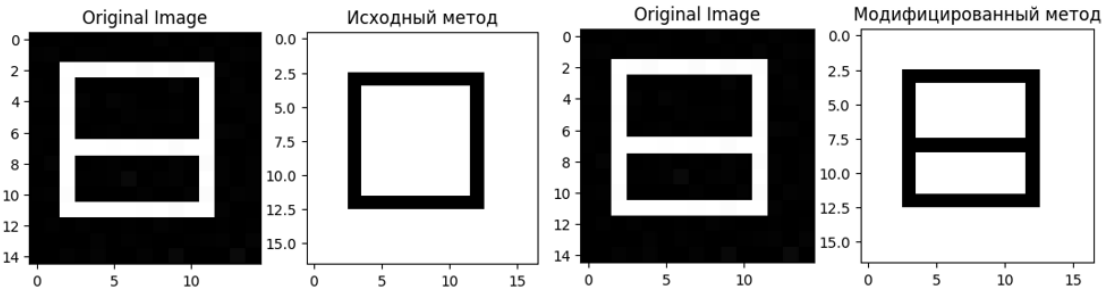


Рисунок 1 – Пример работы исходного и модифицированного метода

Для решения проблемы недостаточной точности в определении иерархии контуров был представлен новый подход, основанный на поиске кольца вокруг любой точки контура. После завершения обработки изображения и получения в пикселях вместо единиц номера контуров, мы можем проанализировать, имеют ли пиксели в смежных направлениях одинаковые метки. Если так, это указывает на то, что эти пиксели принадлежат одному и тому же контуру (рисунок 2).

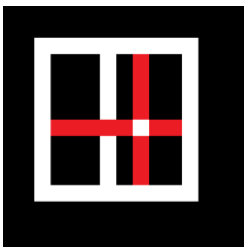


Рисунок 2 – Алгоритм нахождения родительских контуров

В проведенном исследовании алгоритмов нахождения контуров на изображениях выявлены их ключевая роль и актуальность в области компьютерного зрения. Несмотря на широкое применение и высокую точность некоторых методов, они часто сталкиваются с ограничениями в скорости работы или точности определения контуров. В связи с этим, модификация алгоритма Сузуки85 была проведена с целью улучшения баланса между скоростью и точностью работы. Предложенные изменения позволили устранить ошибки и улучшить точность определения контуров, при этом, не увеличивая существенно вычислительную нагрузку. Полученные результаты свидетельствуют о перспективности предложенного метода для практических задач обработки изображений и расширения его применения в области компьютерного зрения.

ЛИТЕРАТУРА

1. Салех Л.О.А., Хлопин С.В., Черненькая Л.В., Тарасевский Ф.Г., Царев М. М., Алгоритм определения концентрации примесей в жидкости по оптическим данным. Известия Тульского государственного университета. Технические науки. 2023. № 1. С. 247-256
2. Satoshi S. Computer Vision, Graphics, and Image processing N30, 32-46 (1985) Topological Structural Analysis of Digitized Binary Images by Border Following
3. А.А. Ханова, М.И. Озерова, ОБЗОР МЕТОДОВ ВЫДЕЛЕНИЯ КОНТУРОВ НА ИЗОБРАЖЕНИЯХ. Владимирский государственный университет. 2020
4. Белим Сергей Викторович, Кутлунин Павел Евгеньевич Выделение контуров на изображениях с помощью алгоритма кластеризации // КО. 2015. №1. URL: https://cyberleninka.ru/article/n/vydelenie-konturov-na-izobrazheniyah-s-pomoschyu-algoritma-klasterizatsii (дата обращения: 18.03.2024).

УДК 004.457

А. Р. Тесленко, С. Н. Ролецкая (3 курс бакалавриата),
Т. В. Коликова, ст. преподаватель

ИССЛЕДОВАНИЕ ПОДХОДОВ И РЕАЛИЗАЦИЯ ИНСТРУМЕНТА ДИНАМИЧЕСКОГО МОНИТОРИНГА КОМПОНЕНТОВ LINUX ПРИЛОЖЕНИЙ: ВЫСОКОУРОВНЕВЫЙ ЯЗЫК ТРАССИРОВКИ BPFTRACE

Современные программные продукты сложны и имеют разные архитектуры, которые устанавливают собственное поведение и зависимости между компонентами, а поддержка их бесперебойной работы требует постоянного анализа всей системы. В процессе работы продуктов существует необходимость сравнивать их между собой и с «базовой» платформой по составу и взаимодействиям внутри операционной системы, находить «дельту» между релизами, выявлять динамические «паразитные» зависимости, которые не отследить при статическом анализе. Помимо этого, необходимо анализировать и выявлять статические и динамические проблемы в самой ОС: утечки памяти, ошибки системных вызовов, ошибки доступа, и другие.

Наиболее полным по функциональности и удобным инструментом для мониторинга различных процессов является трассировка системных вызовов [1] - обращения программы к ядру ОС для выполнения каких-либо операций. Они позволяют изучать поведение

пользовательских программ и операционной системы, понимать, как работает система в целом, отслеживать проблемы с производительностью и находить причины отклоняющегося поведения. Инструментом для работы с ними в ОС Linux является высокоуровневый язык трассировки bpftrace – улучшенная технология Berkley Packet Filter (eBPF) [2].

Утилита bpftrace - обеспечивает динамическую трассировку, которая представляет собой возможность инструментирования запущенного ядра операционной системы. Она связывает действия, такие как сбор или печать трассировок стека, аргументов функций, временных меток и статистических агрегатов, с зондами, которые могут быть событиями времени выполнения или местоположениями исходного кода. Экосистема трассировки (bpftrace Probe Types) представлена на рисунке 1.

Благодаря собственному скриптовому языку bpftrace позволяет выяснить, что происходит внутри системного или библиотечного вызова, имеет возможность не просто составить список вызовов, но и, например, собрать статистику по определённому поведению, а также трассировать несколько процессов и сопоставить данные из нескольких источников в формате, удобном разработчику [3]. Имеется доступ к различной контекстной информации, такой как текущий PID, трассировка стека, время, аргументы вызовов, возвращаемые значения, и т.д.

Таким образом целью данного исследования является анализ преимуществ bpftrace в сравнении с другими инструментами мониторинга системы и реализация bpftrace-сценариев для наглядной демонстрации возможностей и использования языка в реальных проектах.

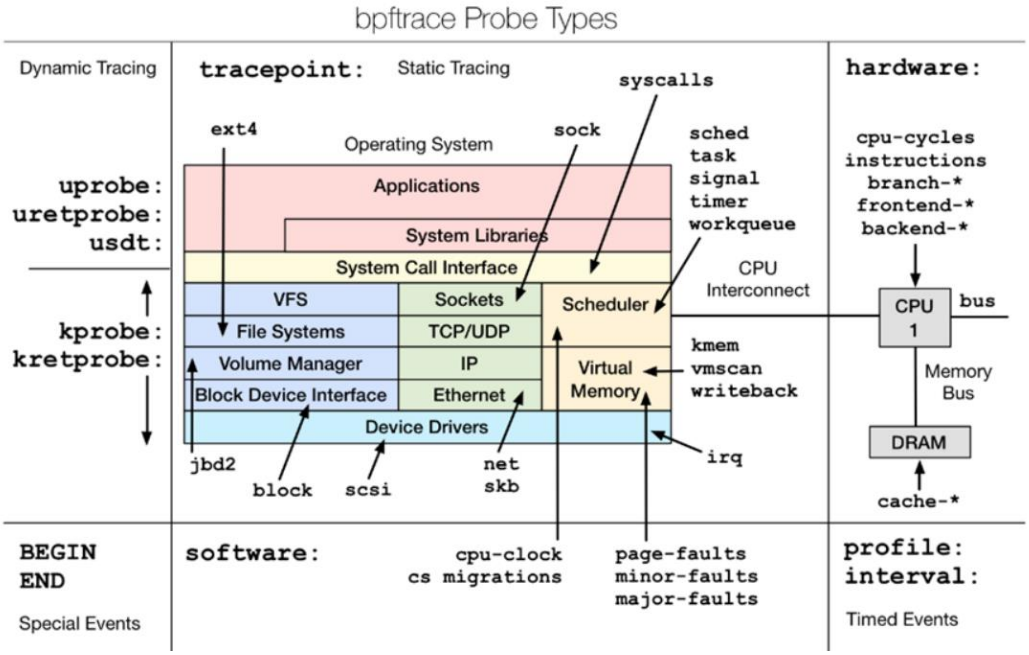


Рисунок 1 – Экосистема трассировки (bpftrace Probe Types).

Для демонстрации возможностей были реализованы различные bpftrace-скрипты, работающие с компонентами ОС: файловой системой, сетевыми трафиком и соединениями, межпроцессорными взаимодействиями ipc. Дальнейшие возможности интегрирования bpftrace-скриптов в реальные проекты продемонстрированы на примере реализации программного продукта для отслеживания неиспользуемых пакетов на системе средствами языка программирования Golang [4].

ЛИТЕРАТУРА

1. Brendan Gregg; Systems Performance (Addison-Wesley Professional Computing Series): Enterprise and the Cloud, 2nd Edition (2020). [Электронный ресурс] Режим доступа: https://www.brendangregg.com/systems-performance-2nd-edition-book.html

2. Brendan Gregg; BPF Performance Tools: Linux System and Application Observability, published by Addison Wesley (2019). [Электронный ресурс] Режим доступа: https://www.brendangregg.com/bpf-performance-tools-book.html
3. Bpftrace. [Электронный ресурс] Режим доступа: https://github.com/bpftrace/bpftrace?tab=readme-ov-file
4. Golang programming language. [Электронный ресурс] Режим доступа: https://go.dev/

УДК 004.031.2

А. Ю. Шестакова (2 курс магистратуры),
И. Г. Черноруцкий, д.т.н., профессор

РАЗРАБОТКА АНАЛИЗАТОРА СТОЙКОСТИ ПАРОЛЯ НА ОСНОВЕ СОВРЕМЕННЫХ АЛГОРИТМОВ ОЦЕНКИ ПАРОЛЕЙ

Подсистемы идентификации и аутентификации пользователя играют важную роль в системах защиты информации. Одним из основных и наиболее часто используемых методов пользовательской аутентификации являются парольные системы идентификации и аутентификации. В данном случае информацией, аутентифицирующей пользователя, является некоторый секретный пароль, известный только легальному пользователю [1].

Методы парольной аутентификации пользователя наиболее просты и при несоблюдении определенных требований к выбору пароля являются достаточно уязвимыми.

Целью работы является разработка анализатора стойкости паролей на основании современных алгоритмов оценки паролей.

Для достижения цели были поставлены следующие задачи:

- анализ существующих алгоритмов оценки паролей;
- реализация алгоритмов проверки стойкости введенного пользователем пароля;
- создание веб-приложения, позволяющего пользователю ввести пароль, запросить его анализ и получить результаты;
- реализация вывода пользователю результатов анализа;
- реализация предоставления пользователю критериев стойкого пароля, присущих алгоритму, давшему отрицательный результат при анализе.

Для решения вопроса защиты информации парольных систем существует широкий набор средств, основной целью которых является затруднение работы парольного взломщика — программы подбора пароля по тем или иным алгоритмам.

В ходе работы были рассмотрены наиболее популярные анализаторы проверки стойкости паролей, позволяющие пользователям узнать надежность выбранного пароля: анализатор SeaMonkey [2], Password Strength Meter (jQuery plugin) [3], Cornell University Password Strength Checker [4], Password Strength Tester [5], основной и усложненный алгоритмы Microsoft [6].

Все вышеперечисленные анализаторы в своих алгоритмах при проверке пароля учитывают его длину, многие из них также учитывают наличие символов из разных алфавитов (строчные буквы, заглавные буквы, цифры, другие символы), некоторые анализаторы также проверяют пароль на его наличие в словарях наиболее часто встречающихся паролей.

Кроме того, часть рассмотренных анализаторов используют более сложные алгоритмы проверки паролей.

Так, Password Strength Meter [3] при оценке выявляет пароли, построенные путём повторения групп символов. Если в пароле встречается подстрока вида SS, где S – строка длины 1, 2, 3 или 4 символов, то повторяющаяся часть этой подстроки удаляется и сжатие продолжается с позиции начала второй части этой подстроки.

Password Strength Tester [5] использует математическую модель, построенную на основе положений теории информации. В качестве основной оценки пароля используется его энтропия — мера случайности выбора последовательности символов, составляющих пароль,

оцененная методами теории информации. Информационная ёмкость измеряется в битах и характеризует стойкость к подбору пароля методом полного перебора при условии отсутствия априорной информации о характере пароля и применении злоумышленником оптимальной стратегии перебора.

Для проверки стойкости пароля в усложненном алгоритме от Microsoft [6] помимо проверок длины пароля, наличия символов из разных алфавитов используется проверка не только на наличие искомого пароля в словаре паролей, но и на близость искомого пароля к словарному слову с определенной вероятностью.

Для реализации были выбрны пять алгоритмов анализаторов: анализаторы SeaMonkey, Password Strength Meter, Cornell University - Password Strength Checker, Password Strength Tester и усложненный алгоритм от Microsoft.

Для программной реализации выбранных алгоритмов были использованы следующие технологии: Javascript, HTML, CSS, среда разработки IntelliJ IDEA.

Помимо реализации алгоритмов проверки стойкости пароля был также реализован механизм защиты веб-приложения от вредоносных инъекций [7]: добавлены проверки, выявляющие попытки злоумышленника внести вредоносный код в приложение через поле ввода пароля.

Было разработано веб-приложение, позволяющее пользователю проверить надежность своего пароля при помощи пяти наиболее распространенных алгоритмов проверки. Приложение предоставляет пользователю поле ввода пароля, кнопку запроса проверки введенного пароля, а также пять шкал, демонстрирующих результаты работы выбранных алгоритмов для введенного пароля.

ЛИТЕРАТУРА

1. Chaluvadi N. S. S., Chitteti L., Challa L., Srithar S. Improved Arbitrary Graphical Password Authentication for Web Application Safety // 5th International Conference on Smart Systems and Inventive Technology (ICSSIT), Tirunelveli, India, 2023, pp. 714-720, DOI:10.1109/ICSSIT55814.2023.10060964.
2. Документация SeaMonkey [Электронный ресурс] — URL: https://mozilla-russia.org/products/seamonkey // (Дата обращения: 26.02.2024).
3. Документация Password Strength Meter (jQuery plugin) [Электронный ресурс] — URL: https://www.jqueryscript.net/tags.php?/Password%20Strength // (Дата обращения: 26.02.2024).
4. Документация Cornell University Password Strength Checker [Электронный ресурс] — URL: https://it.cornell.edu/security-and-policy/manage-passwords-safely // (Дата обращения: 26.02.2024).
5. Документация Password Strength Tester [Электронный ресурс] — URL: https://rumkin.com/tools/password // (Дата обращения: 26.02.2024).
6. Документация Microsoft Security [Электронный ресурс] — URL: https://www.microsoft.com/ru-ru/security // (Дата обращения: 26.02.2024).
7. Kerschbaumer C., Ritter T. , Braun F. Hardening Firefox against Injection Attacks // IEEE European Symposium on Security and Privacy Workshops (EuroS&PW), Genoa, Italy, 2020, pp. 653-663, DOI:10.1109/EuroSPW51379.2020.00094.

УДК 004.457

А. Ю. Шестакова (2 курс магистратуры),
И. Г. Черноруцкий, д.т.н., профессор

ПРИМЕНЕНИЕ ПРОФИЛИРОВАНИЯ ДЛЯ ПОВЫШЕНИЯ ПРОИЗВОДИТЕЛЬНОСТИ HTTP-СЕРВЕРА

В современном мире, где масштаб разрабатываемых программных продуктов стремительно растет, возникает необходимость написания не просто работающего, но и высокопроизводительного и не ресурсоемкого кода. Проведение анализа кода с целью выявления дефектов и уязвимостей, а также обнаружения кода, неэффективно использующего ресурсы, является необходимой частью процесса разработки качественного программного продукта.

Профилирование [1] — это процесс анализа поведения приложения во время выполнения. Он включает в себя измерение использования таких ресурсов, как процессорное время, память и длительность блокировок. При проведении профилирования обычно собираются следующие характеристики: время выполнения отдельных строк кода, количество вызовов и время выполнения отдельных функций, дерево вызовов функций, загрузку CPU и потребление памяти, обращение к другим ресурсам.

Целью работы является проведение профилирования кода HTTP-сервера, реализованного на языке программирования Java.

В качестве инструмента был выбран профилировщик Async Profiler, разработанный компанией «Одноклассники». В отличие от других профилировщиков, таких как AProf [2] и AQtime [3], использующих JVM Tool Interface (JVMTI) [4], выбранный инструмент может не только отслеживать потребление ресурсов, вызовы методов из разных потоков программы, при этом оказывая минимальное влияние на производительность, собирать образцы трассировки стека, включающие методы Java, собственные вызовы, код JVM и функции ядра, но и решает проблему safepoint bias [5] — необходимость полной остановки потоков приложения для снятия stacktrace.

С целью охватить не только основной режим работы сервера, но и режим высокой нагрузки, при котором появляется использование таких операций, как запись данных из памяти на диск, чтение с диска, фоновый compaction (объединение лежащих на диске файлов), профилирование проводится одновременно с нагрузочным тестированием. В качестве инструмента нагрузочного тестирования был выбран WRK2 [6]. Выбранный инструмент позволяет запускать переданный в аргументах скрипт в течение указанного времени с указанными количеством потоков, количеством соединений и значением пропускной способности.

Пример запускаемых запросов представлен ниже в таблице 1.

Таблица 1 — Примеры сриптов, формирующих PUT-запросы и GET-запросы

`counter = 0`
 	
`request = function()`
`  path = "/v0/entity?id=" .. counter`
`  wrk.method = "PUT"`
`  wrk.body = "value" .. counter`
`  counter = counter + 1`
`  return wrk.format(nil, path)`
`end`
	`counter = 0`
	
	`request = function()`
	`  path = "/v0/entity?id=" .. counter`
	`  wrk.method = "GET"`
	`  counter = counter + 1`
	`  return wrk.format(nil, path)`
	`end`

PUT-запрос («put.lua»)	GET-запрос («get.lua»)

Опытным путем были выявлены оптимальные параметры запуска нагрузочного тестирования: нагрузка 15000 запросов в секунду для PUT-запросов и 100 запросов в секунду для GET-запросов.

На рисунке 1 представлена часть результатов работы профилировщика для запуска нагрузочного тестирования на PUT-запросах, относящаяся к потоку «SelectorThread», при наведении курсором на интересующий метод появляется информация о проценте аллокаций (выделений памяти), приходящемся на метод. Рассматриваемый на рисунке 1 метод «getNodesSortedByRendezvousHashing» использует структуру данных «TreeMap», а также методы «stream», «limit», «toList». Для уменьшения выделения памяти стоит рассмотреть более подходящую структуру данных, а также пересмотреть необходимость вызова методов «stream» и «toList», в результате работы которых выделяется половина используемой методом «getNodesSortedByRendezvousHashing» памяти.

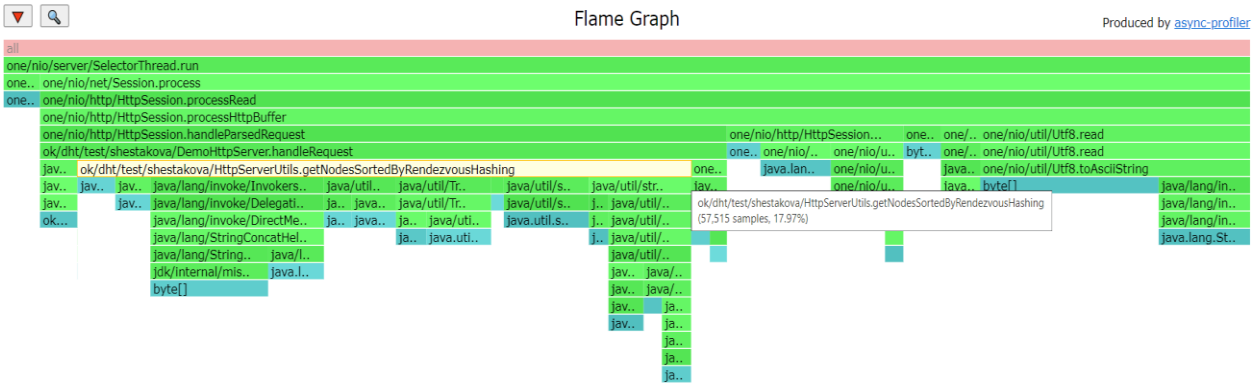


Рисунок 1 — Выделение памяти на потоке «SelectorThread» для PUT-запросов

В результате проведения профилирования под указанной выше нагрузкой были выявлены работающие потоки, выполняемые в них операции, процент выделяемых для операций ресурсов — памяти, процессорного времени и блокировок, — а также обнаружены участки кода, использующие ресурсы не оптимально.

ЛИТЕРАТУРА

1. Tiganourias E., Mavropoulos M., Keramidas G., Kelefouras V., Antonopoulos C. P., Voros N. A Hierarchical Profiler of Intermediate Representation Code based on LLVM // 10th Mediterranean Conference on Embedded Computing (MECO), Budva, Montenegro, 2021, pp. 1-5, DOI:10.1109/MECO52532.2021.9460203.
2. Rodrigues M., Guimarães B., Pereira F. M. Q. Generation of In-Bounds Inputs for Arrays in Memory-Unsafe Languages // IEEE/ACM International Symposium on Code Generation and Optimization (CGO), Washington, DC, USA, 2019, pp. 136-148, DOI:10.1109/CGO.2019.8661178.
3. Базарова А. М. Профайлеры и механизмы оптимизации в программировании // Информатика, вычислительная техника и управление. 2021. №5(2). DOI:10.37882/2223–2966.2021.05–2.06.
4. Howarth J., Altas I., Dalgarno B. Information Flow Control Using the Java Virtual Machine Tool Interface (JVMTI) // International Conference on Availability, Reliability and Security, Krakow, Poland, 2010, pp. 689-695, DOI:10.1109/ARES.2010.75.
5. Filatov A. Evaluation of thread-local garbage collection // Proceedings - 2020 Ivannikov Memorial Workshop, IVMEM 2020, Orel, 25–26 сентября 2020 года. – Orel, 2020. – P. 15-21. – DOI 10.1109/IVMEM51402.2020.00009.
6. Документация WRK2 [Электронный ресурс] — URL: https://github.com/giltene/wrk2 // (Дата обращения: 26.02.2024).

УДК 004.421

Я. С. Щурихин (4 курс, бакалавриата),
А. С. Герасимов, к.ф.-м.н., доцент

ГЕНЕРАЦИЯ УЧЕБНЫХ ЗАДАНИЙ НА ПОИСК В ГЛУБИНУ И ШИРИНУ В ГРАФАХ

В образовательном процессе существует множество различных способов проверки знаний, одним из которых является проведение тестирования. Однако для обеспечения эффективности и достоверности этого процесса необходимо уделить внимание разработке уникальных учебных заданий. Для заданий на поиск в глубину и ширину в графах (по учебному курсу «Алгоритмы и анализ сложности» [1]) необходимо придумывать много различных графов, в которых учащиеся применяли бы алгоритмы поиска. Более того, преподавателю также будет необходимо самостоятельно выполнять построение деревьев поиска для придуманных графов, чтобы получить правильный ответ на задание, что занимает много времени. Поэтому встаёт задача автоматизации данного процесса с помощью

компьютерной программы-генератора учебных заданий на поиск в глубину и ширину в графах по некоторому заданному шаблону задания.

Таким образом, *целью* работы является разработка программы, которая генерирует учебное задание на поиск в глубину или ширину в графах по шаблону:

Задание. Дан ориентированный/неориентированный граф $G = (V, E)$, где
$V = \{1, 2, \ldots, \text{ЧислоВершин}\}$ и $E = \text{МножествоРёбер}$.
Сколько всего деревьев поиска в глубину/ширину из вершины 1 имеет G?
(1) более 5; (2) 5; (3) 4; (4) 3; (5) 2; (6) 1; (7) 0

Также программа должна иметь возможность предоставить пользователю правильный ответ и его обоснование.

Для достижения этой цели необходимо решение следующих *задач*:

1. Поиск и анализ подобных программных решений.
2. Разработка высокоуровневой спецификации генератора.
3. Разработка алгоритма генерации подходящего для задания случайного графа и алгоритмов построения всех деревьев поиска в глубину и ширину из заданной вершины графа.
4. Реализация программы-генератора учебных заданий в соответствии со спецификацией.

Был произведён поиск существующих программных решений, которые могут частично или полностью решить задачу генерации учебных заданий на поиск в глубину и ширину. На данный момент создано достаточно много библиотек для различных языков программирования, которые реализуют алгоритмы для работы с графами. Например, JGraphT [2] для Java, Boost.Graph [3] для C++, QuickGraph [4] для C#, NetworkX [5] для Python. Описанные библиотеки дают возможность создавать случайные графы, но не имеют инструментов для получения правильных ответов для данного выше шаблона учебного задания: стандартные алгоритмы поиска в них строят лишь одно дерево из заданной вершины.

Удалось найти онлайн-ресурс [6], который позволяет решать задания на поиск в глубину и ширину и даёт графическое пояснение к ответу. Данный сайт предоставляет возможность создать граф по матрице смежности, списку рёбер или графически (добавляя вершины и рёбра на поле для рисования). С помощью этого ресурса можно решить некоторые задания на поиск в глубину и ширину, но он, как и библиотеки, описанные выше, не может найти более одного дерева поиска из заданной начальной вершины. Также этот веб-сайт не дает возможности генерировать случайные графы.

Найти полноценные программные продукты, которые имеют функционал для генерации заданий на поиск в ширину или глубину с вариантами ответов и пояснениями к правильному из них, не удалось.

Одним из основных компонентов разрабатываемой программы-генератора является создание случайных графов, которые будут достаточно интересны для выполнения поиска в глубину и ширину из вершины 1. Для этого был разработан алгоритм, работающий следующим образом:

– Случайным образом генерируется код Прюфера [7], длина которого будет на 2 меньше, чем желаемое число вершин графа в условии задания.

– Из кода Прюфера восстанавливается дерево с множеством вершин, равным $\{1, 2, \ldots, \text{ЧислоВершин}\}$. Если требуется ориентированный граф, то придаётся ориентация рёбрам дерева: вдоль простых путей из вершины 1 во все остальные вершины.

– В дерево добавляется некоторое количество (может быть указано пользователем) ранее не существовавших в нём рёбер.

Также важным аспектом являлась реализация алгоритмов построения всех деревьев поиска в глубину и ширину из заданной вершины графа. На данный момент разработаны алгоритмы для построения всех деревьев поиска в глубину и ширину из заданной вершины, которые строят все деревья в том случае, если из заданной вершины достижимы все остальные

вершины графа. Их общая идея заключается в обходе смежных с данной вершин (начиная с заданной начальной) всеми возможными способами.

Программный продукт реализуется на языке Java версии 20. Для разработки графического интерфейса пользователя используется Swing API [8]. Программа имеет как графический интерфейс, так и интерфейс командной строки.

ЛИТЕРАТУРА

1. 1. Герасимов А. С. Материалы учебного курса «Алгоритмы и анализ сложности», 2023.
2. 2. JGraphT Java library of graph theory data structures and algorithms. [Электронный ресурс] Режим доступа: https://jgrapht.org/
3. 3. The Boost Graph Library (BGL). [Электронный ресурс] Режим доступа: https://www.boost.org/doc/libs/1_82_0/libs/graph/doc/index.html
4. 4. QuickGraph .NET library that provide mainly graphs structures and algorithms for C#. [Электронный ресурс] Режим доступа: https://www.nuget.org/packages/QuikGraph
5. 5. NetworkX Python package for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks. [Электронный ресурс] Режим доступа: https://networkx.org/
6. 6. Graph Online. [Электронный ресурс] Режим доступа: https://graphonline.ru
7. Caminiti S., Finocchi I., Petreschi R. On coding labeled trees // Theoretical Computer Science, Vol. 382, No. 2, 2007, pp. 97-108. – DOI 10.1016/j.tcs.2007.03.009
8. Java Swing API Documentation. [Электронный ресурс] Режим доступа: https://docs.oracle.com/javase/7/docs/api/javax/swing/package-summary.html.

УДК 004.4

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

РАЗРАБОТКА ПРОГРАММНЫХ СРЕДСТВ КЛАССИФИКАЦИИ СИГНАЛОВ БЕСПРОВОДНЫХ СИСТЕМ ПЕРЕДАЧИ ДАННЫХ

Классификации сигналов беспроводных систем передачи данных является актуальной задачей систем радиомониторинга, обеспечивающих контроль использования радиочастот, а также систем радиотехнической разведки и противодействия, которые применяются для защиты гражданских и военных объектов.

Одним из перспективных и быстроразвивающихся направлений в классификации типов модуляций радиосигналов является глубокое обучение и искусственный интеллект. Из основных преимуществ данного подхода можно выделить способность распознавать и классифицировать сигналы с вероятностью правильной классификации, достигающей более 80% для сигналов с цифровой и аналоговой модуляциями при отношении сигнал/шум более 0 дБ. Однако применение глубокого обучения для классификации типа модуляции радиосигналов сталкивается с несколькими проблемами. Сложность обучения модели связана с большим числом используемых форм сигналов, скоростей передачи данных и длительностей кадров, а также с не стационарностью характеристик радиолиний, приводящей к различным искажениям радиосигналов. Для многих приложений важно иметь способность классифицировать радиосигналы в реальном времени, что требует как аппаратной поддержки обработки, так и оптимизации программных средств.

Целью работы является создание кроссплатформенных программных средств классификации типа модуляции радиосигналов с использованием глубокого обучения.

Для достижения поставленной цели необходимо решение следующих задач:

– Обзор существующих подходов классификации типов модуляций сигналов с использованием глубокого обучения и нейронных сетей.

– Выбор данных для глубокого обучения, подбор источника данных, в котором представлен исчерпывающий набор типов модуляций при различных отношениях сигнал/шум.
– Определение типа структуры нейронной сети, способной классифицировать сигналы и ее обучение. Определение зависимости точности классификации от значений сигнал/шум.
– Разработка кроссплатформенных программных средств, обеспечивающих анализ радио сигналов без дополнительной предварительной обработки со стороны пользователя.

В качестве набора данных, был выбран датасет RADIOML 2018A [1] – это один из широко используемых наборов данных для задач классификации типов модуляций радиосигналов. В нем присутствуют 24 типа модуляций при значениях сигнал/шум в диапазоне [-20; 30] дБ. Обучающая выборка содержит квадратурные сигналы с искажениями, соответствующими условиям передачи данных в беспроводных коммуникационных системах.

Для решения задачи классификации радиосигналов, которая включает в себя анализ и распознавание шаблонов в данных, связанных с временными и частотными характеристиками сигналов, можно использовать сверточные нейронные сети (CNN) и глубокие сверточные сети с остаточными блоками (ResNet) [2, 3]. CNN и ResNet хорошо подходят для анализа пространственных шаблонов в данных. Это свойство может быть использовано для классификации радиосигналов, которые могут подвергаться различным искажениям и изменениям во времени и частоте [4].

Для задачи классификации радиосигналов ResNet обладает преимуществами по сравнению с CNN: ResNet менее чувствительна к проблеме затухания градиента, обладает меньшими вычислительных затратами и требованиями к памяти.

Разработанная архитектура сети содержит:
– Входной слой: принимает данные размером (None, 1024, 2).
– Сверточные блоки: каждый блок содержит два сверточных слоя (Conv1D) с активацией ReLU, за которыми следует пропускное соединение (Add), добавляющее исходный ввод к выходу сверток.
– Остаточные блоки: модель использует остаточные блоков, каждый из которых содержит несколько остаточных блоков с 40 фильтрами. После каждого блока применяются слои макс-пулинга (MaxPooling1D).
– Полностью соединенные слои: после сверточных слоев сеть сплющивает данные (Flatten) и использует три полностью соединенных слоя (Dense) с активацией SELU и с регуляризацией 0.5 (Dropout).
– Выходной слой: Представляет собой слой softmax, который выдаёт вероятности принадлежности к 24 классам.

Разработанная модель нейронной сети Resnet была обучена выборкой из 1277952 сигналов с 24 типами модуляции при отношении сигнал/шум в диапазоне [0; 30] дБ. При тестировании обученной сети тестовой выборкой, содержащей 144180 сигналов, была получена средняя вероятность правильной классификации не менее 82% при отношении сигнал/шум большему или равному 0 дБ.

Разработанные программные средства состоят из нейронной сети ResNet в виде экспортированной версии нейронной сети TensorFlow Lite и приложения на языке Rust [5], которое используется для подготовки входных данных, содержащих 1024 отчета квадратурных компонент (IQ) сигнала и соответствующих формату представления сигналов в датасете RADIOML 2018A.

Разработанные программные средства могут быть использованы для классификации сигналов в системах радиоконтроля [6] как этап необходимый для демодуляции и декодирования передаваемой информации [7].

ЛИТЕРАТУРА

1. Deepsig [Электронный ресурс] Режим доступа: https://www.deepsig.ai/datasets/

2. Over-the-Air Deep Learning Based Radio Signal Classification [Электронный ресурс] Режим доступа: https://ieeexplore.ieee.org/document/8267032
3. Classification of Radio Signals and HF Transmission Modes with Deep Learning [Электронный ресурс] Режим доступа: https://arxiv.org/abs/1906.04459
4. RF Signal Classification with Synthetic Training Data and its Real-World Performance [Электронный ресурс] Режим доступа: https://arxiv.org/abs/2206.12967
5. Rust Programming Language [Электронный ресурс] Режим доступа: https://www.rust-lang.org/
6. Трошев Д. М., Медведев Б. М. Программные средства радиоконтроля. Современные технологии в теории и практике программирования : сборник материалов научно-практической конференции студентов, аспирантов и молодых ученых, 26–27 апреля 2022 г. – СПб. : ПОЛИТЕХ-ПРЕСС, 2022. – С. 11-12
7. Соколов Н. А., Медведев Б. М. Разработка программных компонентов для отображения и анализа кадровой структуры битового потока. Современные технологии в теории и практике программирования : сборник материалов научно-практической конференции студентов, аспирантов и молодых ученых, 26–27 апреля 2023 г. – СПб. : ПОЛИТЕХ-ПРЕСС, 2023. – С. 11-12

Секция «Подходы к разработке программного обеспечения на основе технологий компании YADRO»

УДК 004.4`42

А. Р. Габдрахманов (3 курс бакалавриата),
Д. С. Косарев, ассистент

ИНТРИНСИКИ ДЛЯ ИСПОЛЬЗОВАНИЯ ВЕКТОРНЫХ РАСШИРЕНИЙ RISC-V В КОМПИЛЯТОРЕ НАТИВНОГО КОДА OCAML

Для ускорения работы программ в современных процессорах используется ряд технологий, в том числе векторные и SIMD инструкции, которые позволяют процессору обрабатывать несколько элементов данных одновременно, улучшая производительность при выполнении операций с векторами данных. Основным отличием векторных расширений от SIMD (Single Instruction, Multiple Data) является возможность работы с векторами данных различных размеров и типов, в то время как SIMD чаще ориентирован на выполнение операций над векторами данных фиксированного размера. Многие современные процессорные архитектуры имеют поддержку векторных расширений. Примером может служить ARM с технологией SVE (Scalable Vector Extension) [1] или Intel с набором инструкций AVX-512 (Advanced Vector Extension) [2].

На данный момент в высокоуровневом языке программирования OCaml отсутствует поддержка векторной обработки массивов и автовекторизация в целом.

Например, в стандартной библиотеке для массивов нет операции поэлементного сложения массивов одинаковой размерности. Поэтому является нормальной практикой писать самостоятельную реализацию функций для работы с векторами, например, переписывать “горячий”, то есть часто используемый, код на языке C. Чтобы оставаться в рамках OCaml нужны интринсики. Добавлением интринсиков в OCaml для архитектур ARM и x86_64 занимается компания Jane Street [3]. В рамках данной работы предлагается реализовать интринсики для векторных инструкций RISC-V в компиляторе нативного кода OCaml.

Использование векторных интринсиков расширения RISCV-V “V” [4] для языка Си приводит к существенному ускорению поэлементного сложения массивов небольшой длины (см. Рисунок 1). Для замеров использовался миникомпьютер RISC-V Sipeed LicheePi 4A Risc-V TH1520.

OCaml использует 63-битное представление целого числа [5], что, на первый взгляд, может усложнить процесс работы с векторами. Однако, накладные расходы возникают как в скалярном, так и в векторном случае, поэтому ожидается схожий прирост производительности при использовании векторов, как и в реализации на C.

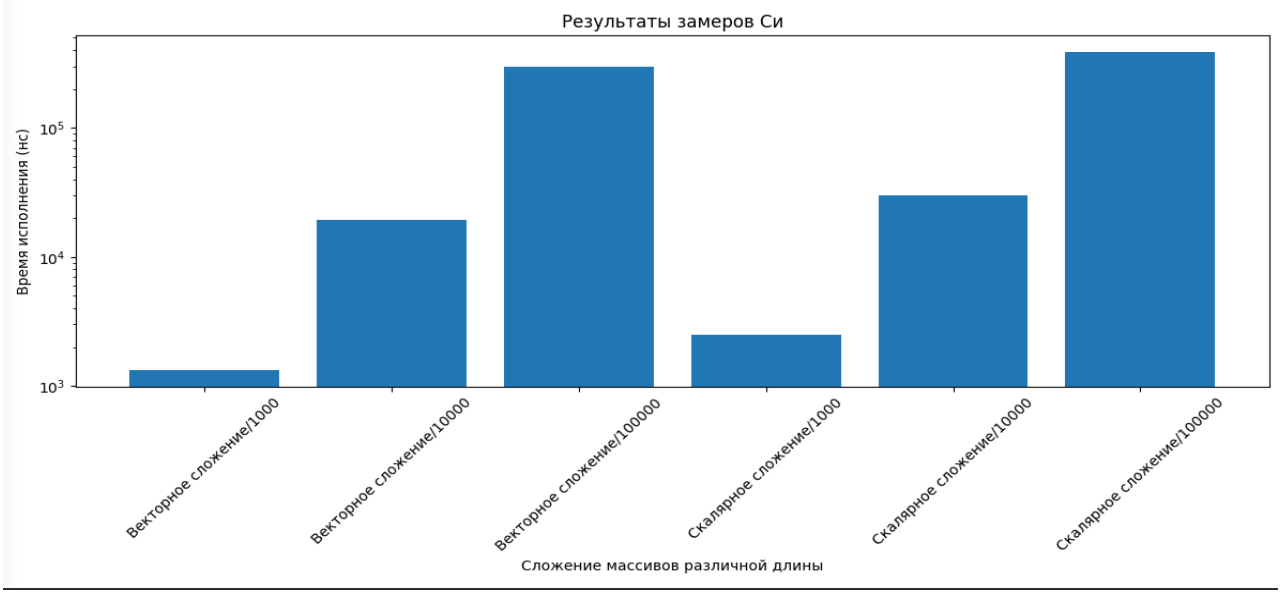


Рисунок 1 – Результаты замеров сложения массивов

ЛИТЕРАТУРА

1. Nigel Stephens, Stuart Biles, Matthias Boettcher, Jacob Eapen, Mbou Eyole, Giacomo Gabrielli, Matt Horsnell, Grigorios Magklis, Nathanael Premillieu, Alastair Reid, Alejandro Rico, Paul Walker. The ARM Scalable Vector Extension [Электронный ресурс]. Режим доступа:https://ieeexplore.ieee.org/abstract/document/7924233 (Дата обращения: 19.03.2024)
2. Chris Lomont. Introduction to Intel® Advanced Vector Extensions [Электронный ресурс]. Режим доступа: https://hpc.llnl.gov/sites/default/files/intelAVXintro.pdf (Дата обращения: 19.03.2024)
3. Flambda-backend [Электронный ресурс]. Режим доступа: https://github.com/ocaml-flambda/flambda-backend/blob/430bb7c8b1222698a92fbd41eb8164df8d71dc96/backend/amd64/simd_selection.ml (Дата обращения: 19.03.2024)
4. Riscv-v-spec [Электронный ресурс]. Режим доступа: https://inst.eecs.berkeley.edu/~cs152/sp20/handouts/sp20/riscv-v-spec.pdf (Дата обращения: 19.03.2024)
5. Vladimir Brankov. What is gained and lost with 63-bit integers? [Электронный ресурс]. Режим доступа: https://blog.janestreet.com/what-is-gained-and-lost-with-63-bit-integers/ (Дата обращения: 19.03.2024)

УДК 004.42

Д. Е. Голубев (4 курс бакалавриата),
П. А. Шагалова, к.т.н., доцент

РАЗРАБОТКА ПРОГРАММНОГО РЕШЕНИЯ ДЛЯ АВТОМАТИЧЕСКОГО РЕФАКТОРИНГА КОДА

В современном мире разработки программного обеспечения автоматизация процессов является одним из ключевых факторов успеха. Одним из таких процессов является рефакторинг кода. Рефакторинг — это процесс изменения внутренней структуры программы без изменения ее внешнего поведения. Он необходим для поддержания качества кода, его читаемости и удобства сопровождения. В частности, автоматический рефакторинг позволяет избежать ошибок, связанных с человеческим фактором, и значительно ускорить процесс разработки.

В данной работе будет рассмотрена разработка плагина для инструмента статического анализатора с открытым исходным кодом, разрабатываемая как часть LLVM для анализа

языков C/C++/Objective-C - clang-tidy [1], для преобразования векторных оптимизаций библиотеки OpenCV.

Clang-Tidy делает анализ на уровне абстрактного синтаксического дерева (AST) [2] программы и на уровне препроцессора, используя для этого инфраструктуру компилятора Clang.

Векторная оптимизация — это метод оптимизации кода, который позволяет увеличить производительность программы за счет использования векторных операций. Такие операции выполняются над массивами данных, что позволяет уменьшить количество операций и улучшить быстродействие. Применение специализированных векторных инструкций позволяет значительно поднять скорость обработки блоков данных [3].

Автоматизация процессов рефакторинга позволяет улучшить качество и читаемость кода, избежать ошибок и ускорить разработку. Распространенным является подход, при котором в уже написанный работоспособный код программы вносятся изменения, направленные на повышение читаемости кода, его гибкости, облегчение возможностей сопровождения и т.д [4].

Выбор clang-tidy обусловлен его широкой популярностью и функциональностью, включая статический анализ кода и исправление ошибок.

Цель данной работы - разработка программного решения для автоматизации преобразования векторных оптимизаций библиотеки OpenCV.

Для достижения этой цели были решены следующие задачи:

1. Знакомство с исходным кодом простейшей программы написанной на языке программирования C++, представленной в виде абстрактного синтаксического дерева (AST).
2. Знакомство с инфраструктурой LLVM/Clang-Tidy. Конфигурация и генерация сборки LLVM/Clang с помощью утилиты CMake [5].
3. Создание простого скрипта-плагина для clang-tidy с помощью внутреннего python-скрипта.
4. Добавление элементарного образца исправления кода и тестирование его функциональности.
5. Реализация и применение предупреждения и исправления на одном определенном файле.
6. Разработка дополнительной проверки и стандарта на основе потребностей библиотеки.

При выполнении работы использована система компиляции LLVM/Clang, которая обладает возможностью экспортировать AST и предлагает готовый инструмент Clang-Tidy для обработки дерева. Этот инструмент собирает дерево, анализирует и проверяет его, а также вносит автоматические исправления в код. Важно отметить, что Clang-Tidy является открытым проектом с доступным исходным кодом. Это позволяет легко добавлять новые функции при необходимости.

ЛИТЕРАТУРА

1. Бучацкий Р.А., Чуркин Я.А., Чибисов К.А. Проверка программ на соответствие стандарту MISRA C с использованием инфраструктуры CLANG // Труды института системного программирования РАН, 2020.
2. Introduction to the Clang AST. [Электронный ресурс] Режим доступа: https://clang.llvm.org/docs/IntroductionToTheClangAST.html
3. Барлит, А. В. Увеличение быстродействия вычислений в полях Галуа с помощью векторных инструкций / А. В. Барлит // Информационные технологии, системный анализ и управление (ИТСАУ-2020) : Сборник трудов XVIII Всероссийской научной конференции молодых ученых, аспирантов и студентов, 2020. – С. 166-169.
4. Дерюгина, О. А. Проектирование интерпретатора языка QVT Operational Mappings для программного средства UML Refactoring в рамках модельно-ориентированного подхода / О. А. Дерюгина, Е. В. Крючкова // Программные продукты и системы. – 2019. – № 3. – С. 389-397.
5. Дубров, Д. В. Система построения проектов CMake : Учебник / Д. В. Дубров. – Ростов-на-Дону : Южный федеральный университет, 2015. – 419 с. – ISBN 978-5-9275-1852-4. – EDN WYRJJN.

УДК 004.051

А. А. Ефремов, Д. А. Слинчук (2 курс бакалавриата),
В. А. Кутуев, инженер-исследователь

РАЗРАБОТКА ИНСТРУМЕНТА ДЛЯ АНАЛИЗА ЭФФЕКТИВНОСТИ BPU

С момента возникновения вычислительной техники всегда существовало стремление к увеличению скорости выполнения программы. Одним из важных способов ускорения процессора является метод конвейеризации инструкций, позволяющий одновременно обрабатывать несколько инструкций, разделяя их исполнение на несколько этапов. Однако его эффективность не всегда постоянна, например, инструкции переходов (такие как условные, безусловные, вызовы функций, возвраты из них) могут вызвать сброс конвейера, что приводит к простою компонентов процессора. Для решения этой проблемы используется специальный модуль — branch prediction unit (BPU). BPU старается предсказать, какая часть кода будет выполнена дальше, чтобы уменьшить время простоя процессора во время переходов.

Однако не все реализации BPU одинаково эффективны [1, 2, 3]. Например, процессоры, выпускаемые компанией, которая не занималась их разработкой раньше, будут уступать в этом аспекте флагманским процессорам таких компаний, как Intel и AMD, эти компании имеют богатый опыт и специализируются на повышении производительности процессоров на протяжении длительного времени [4, 5]. Выявление сценариев, при которых BPU некого процессора показывает результаты существенно хуже, чем BPU массовых современных процессоров, может способствовать улучшению архитектуры этого самого BPU путем детального изучения таких сценариев.

Для нахождения сценариев, выявляющих разницу в эффективности различных BPU, используется фаззинг. Суть метода заключается в многократном исполнении случайно сгенерированного кода, чтобы найти блоки инструкций, выявляющие слабые места BPU. Для реализации данного метода была начата работа по созданию инструмента, включающего в себя:

– *generate*: случайная генерацию тестов на языке C;

– *analyze*: сборка, запуск тестов и сбор данных со счетчиков процессора, содержащих информацию о работе BPU;

– *summarize*: подведение и обработка статистики по запущенным тестам с визуализацией наглядных результатов на графике.

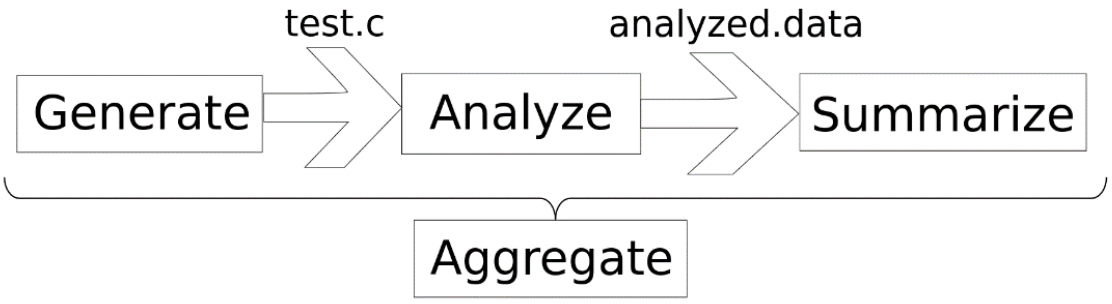


Рисунок 1 – Архитектура инструмента

Для четкого разделения зон ответственности каждого компонента в основе архитектуры, представленной на рисунке 1, используется паттерн цепочка обязанностей с драйвером (*aggregate*) для выполнения всех шагов последовательно. Такая архитектура позволяет предоставить пользователю доступ к каждому компоненту отдельно для достижения необходимых целей, например, запуск собственных тестов, а также позволяет легко расширять инструмент.

На шаге *generate* создаются тесты, представляющие из себя программы на языке программирования C, чтобы их можно было скомпилировать под широкий спектр архитектур. Генерация начинается с рандомизированного построения абстрактного синтаксического дерева (AST), включающего в себя конструкции, создающие нагрузку на BPU (циклы и ветвления), а также различные вычисления для изменения условий ветвления. Далее построенное AST преобразуется в код на языке C и записывается в файл теста.

На шаге *analyze* инструмент запускает тесты и получает данные о работе BPU на них. При анализе используются различные методы для улучшения точности и повторяемости результатов тестирования: изоляция тестового процесса на одном ядре и повышения его приоритета, сбор данных только с тестируемой части кода, исключая код, связанный инициализацией процесса, множественные запуски теста, вычитание из итоговых результатов теста значений, потраченных на настройку тестового окружения. Это позволяет пользователю избежать необходимости самому настраивать окружающую систему и проводить множество измерений для получения надежных данных. Кроме получения данных с реального процессора инструмент также обеспечивает поддержку сбора информации с симулятора процессоров Gem5. В нём пользователь может смоделировать свой процессор, и даже его отдельные компоненты, в том числе BPU. Таким образом, разработчик процессора может не создавать новый физический процессор, а имеет возможность анализировать и тестировать новую модель BPU для своего центрального процессора с помощью симуляционной модели в Gem5. Это предоставляет разработчикам удобный и эффективный способ проверки и проработки моделей новых компонентов процессора перед их реализацией.

На шаге *summarize* строится график, отображающий отношение количества неугаданных переходов к общему количеству переходов для каждого теста, что позволяет пользователю быстро найти интересные и аномальные результаты, а также инструмент сохраняет все полученные данные с каждого теста для возможной дальнейшей обработки пользователем.

Код инструмента открытый и выложен в репозитории на GitHub [6].

В дальнейшем развитии проекта планируется реализовать направленную генерацию тестов, а также автоматизацию работы в распределённом режиме, при котором запуск тестов выполняется на другом узле.

Работа выполнена в Лаборатории технологий программирования инфраструктурных решений СПбГУ.

ЛИТЕРАТУРА

1. Marek Majkowski. Branch predictor: How many ”if”s are too many? Including x86 and M1 benchmarks! CloudFlare. [Электронный ресурс]. Режим доступа: https://blog.cloudflare.com/branch-predictor [Дата обращения: 16.03.2024].
2. Youssif A, Ismail N, Torkey F. Comparison of branch prediction schemes for superscalar processors iceec 2004 // Electrical, Electronic and Computer Engineering, 2004. ICEEC’04. 2004 International Conference on. — 2004. — P. 257–260.
3. Quadri Syed Ali Imran, Jahangir Mohd Ziauddin. Design, Implementation and Performance Comparison of Different Branch Predictors on 16 Pipelined-CPU // 2017 International Conference on Computer, Electrical & Communication Engineering (ICCECE) / IEEE. — 2017. — P. 1–7
4. Roberts-Hoffman Katie, Hegde Pawankumar. ARM cortex-a8 vs. Intel atom: Architectural and benchmark comparisons // Dallas: University of Texas at Dallas. — 2009. — Vol. 5.
5. Fast sort on cpus, gpus and intel mic architectures / Nadathur Satish, Changkyu Kim, Jatin Chhugani et al. // Intel Labs. — 2010. — P. 77–80.
6. GitHub-репозиторий [Электронный ресурс]. Режим доступа: https://github.com/osogi/control-hazard-analyzer [Дата обращения: 16.03.2024].

УДК 004.4

Б. А. Денисенко, М. В. Тянутов (1 курс магистратуры),
И. В. Никифоров, к.т.н., доцент

СИСТЕМА ДЕКЛАРАТИВНОГО УПРАВЛЕНИЯ СЕТЕВОЙ КОНФИГУРАЦИЕЙ LINUX

В современных сетевых инфраструктурах все больше используется декларативный подход к управлению сетевой конфигурацией. Это позволяет администраторам описывать желаемое состояние сети в виде декларативных файлов или кода, а затем автоматически применять эти настройки [1, 2]. Актуальность этой темы обусловлена ростом сложности сетевых инфраструктур [3], а также требованиями к быстрому развертыванию сетевых сервисов.

Целью работы является анализ различий между декларативным и императивным управлением сетью, а также изучение основных инструментов декларативного управления сетью и разработка собственного инструмента.

Для императивной настройки сети в Linux существует множество утилит, которые позволяют управлять сетевыми параметрами через командную строку: iproute2, ifconfig [4], ethtool и т.д. Однако, несмотря на широкое распространение императивных методов настройки сети, декларативное управление становится всё более предпочтительным в современных сценариях администрирования сетей. Одной из главных причин этого является уменьшение трудоёмкости и повышение уровня абстракции, что приводит к улучшению управляемости и предсказуемости сетевых конфигураций. В декларативном подходе администратору необходимо лишь описать желаемое состояние сети, вместо того чтобы явно указывать последовательность команд для достижения этого состояния. Это позволяет избежать ошибок, связанных с опечатками или неправильными последовательностями команд, и облегчает сопровождение конфигураций на протяжении времени.

Существует несколько программных средств в Linux, которые предлагают декларативный подход к настройке сетевой конфигурации, и одной из самых распространенных среди них является Netplan [5]. Однако, одним из его недостатков является необходимость перезагрузки системы для применения изменений конфигурации. Это замедляет процесс обновления сетевых настроек и может привести к простою системы. Важно иметь возможность применять изменения в сетевой конфигурации без перезагрузки системы, чтобы обеспечить бесперебойную работу сетевых сервисов и минимизировать время простоя системы.

Идея разрабатываемой системы заключается в предоставлении возможности динамического изменения сетевой конфигурации Linux "на лету". Она работает путем преобразования декларативно описанных "сетевых единиц" в команды и в дальнейшем их применении на системе. Для обеспечения защиты от невалидных сетевых конфигураций предусмотрено 2 уровня валидации входящей сетевой конфигурации, а также откат к текущей конфигурации, если конфигурацию установить не удалось. Алгоритм применения конфигурации представлен на рисунке 1. Для начала выполняется простая валидация входящих данных – проверка корректности полей “сетевых единиц”, поиск дубликатов и т.д. Далее описанные пользователем “сетевые единицы” преобразуются в команды, необходимые для установки данной конфигурации. Эти команды сначала выполняются на внутренней модели Linux для определения возможных конфликтов, препятствующих установки данной конфигурации. После успешного применения на внутренней модели эти команды пытаются примениться к реальной системе. Если в процессе применения на реальную систему произойдет ошибка, тогда сервис попытается откатиться к предыдущей сетевой конфигурации. Такой подход обеспечивает безопасное и надежное внесение изменений в сетевую конфигурацию без прерывания работы системы.

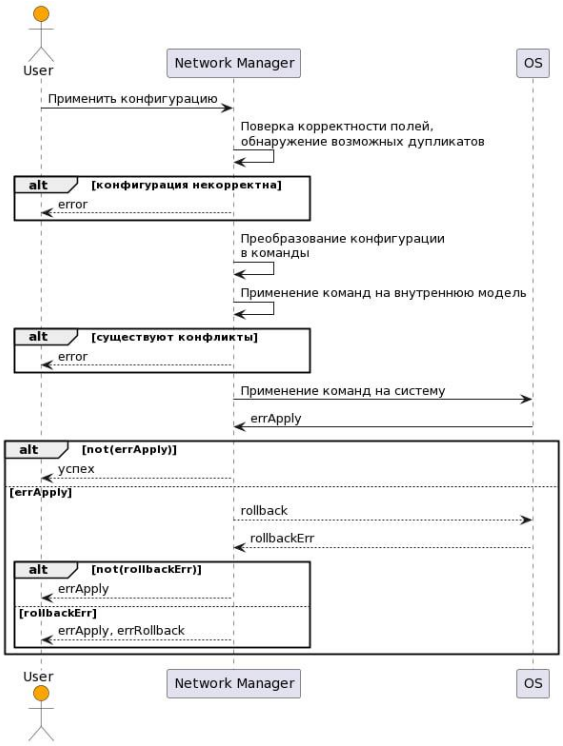


Рисунок 1 – Принцип работы разрабатываемого инструмента

Ожидаемые результаты от разрабатываемой системы декларативного управления сетью в Linux включают не только возможность взаимодействия с программой через конфигурационные файлы [6], но и через gRPC API [7], что значительно расширит спектр возможностей для автоматизации и интеграции с другими инструментами и системами. Помимо этого, ожидается значительное улучшение эффективности управления сетью, что приведет к сокращению времени простоя машин и повышению производительности.

В ходе данного исследования были выявлены преимущества декларативного подхода к управлению сетевой конфигурацией по сравнению с императивным. Также были рассмотрены существующие аналоги, такие как Netplan, которые предоставляют возможности для декларативного управления сетью. Кроме того, был описан принцип работы собственного разрабатываемого сервиса, который позволяет эффективно осуществлять декларативное управление сетью в Linux и выделяется среди аналогов благодаря предоставлению gRPC API и способности изменять сетевую конфигурацию без необходимости перезагрузки системы.

ЛИТЕРАТУРА

1. Исследование систем автоматизации настройки инфраструктуры ИТ-проекта / В. А. Ивлев, Ф. Ю. Чемашкин, И. В. Никифоров, А. Д. Ковалев // Современные технологии в теории и практике программирования : Сборник материалов научно-практической конференции студентов, аспирантов и молодых ученых, Санкт-Петербург, 26–27 апреля 2023 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургс

кий политехнический университет Петра Великого", 2023. – С. 204-205. – EDN HYVLSK.

2. Ивлев, В. А. Актуальность автоматизированной настройки инфраструктуры ит-проекта / В. А. Ивлев, И. В. Никифоров // Современные технологии в теории и практике программирования : Сборник материалов конференции, Санкт-Петербург, 26 апреля 2022 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2022. – С. 79-80. – EDN QQAVXT.
3. Weijnen M., Bouwmans I. Innovation in networked infrastructures: Coping with complexity. International Journal of Critical Infrastructures 2(2/3):121-132, January 2006.
4. Charles J. Brooks, Christopher Grow, Philip Craig, Donald Short. Tools and Utilities. Cybersecurity (pp.565-589), September 2018.

5. Netplan documentation [Электронный ресурс] Режим доступа: https://netplan.readthedocs.io/en/stable/
6. Сысоев, И. М. Создание подхода автоматизации обновления конфигурационных файлов программного обеспечения / И. М. Сысоев, И. В. Никифоров, Е. В. Каплан // 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. – EDN YARVRM.
7. Akshata Sangwai, Shreya Sapale, Shweta Ghodake, Rahul Jadhav. Barricading System - System Communication using gRPC and Protocol Buffers. 2023 5th Biennial International Conference on Nascent Technologies in Engineering (ICNTE), 20-21 January 2023.

УДК 004.383

М. В. Кардынов, К. Е. Сидягин (1 курс магистратуры),
О. Н. Корелин, к.т.н., доцент

ОДНОПЛАТНЫЕ КОМПЬЮТЕРЫ В ЗАДАЧЕ СПЕКТРАЛЬНОГО АНАЛИЗА

Большое количество современных технологий, таких как сжатие информации или распознавание, неизменно включают спектральный анализ. Переход из временной области в частотную позволяет выявить дополнительные особенности и отличия разных сигналов или изображений. Стандартные методы для спектрального анализа дискретизированного сигнала это ДПФ (дискретное преобразование Фурье) или его разновидность БПФ (быстрое преобразование Фурье). Как правило эти алгоритмы широко известны и не вызывают вопросов.

Одноплатный компьютер (SBC Single Board Computer) Mango Pi популярное цифровое устройство. Основой его является процессор RISC-V архитектуры D1 (Allwinner). Это недорогой 64-х битный, одноядерный процессор. Mango Pi имеет “на борту” 1Гб памяти, различные интерфейсы (включая HDMI) и Wi-Fi.

Несмотря на малые размеры, Mango Pi может выполнять достаточно широкий круг задач.

Он работает под управлением ОС Armbian (Linux), что позволяет применять богатый графический интерфейс и современные языки программирования.

Для сравнения был выбран SBC BBB (BeagleBone Black). Это давно себя зарекомендовавший одноплатный компьютер на основе 32-х разрядного процессора ARM A7 (AM3358 Texas Instruments). BBB работает под управлением ОС Debian и продолжает активно поддерживаться производителем. “На борту“ он имеет 512Мб памяти и большой набор интерфейсов с наибольшим количеством контактов для подключения внешних устройств (2*46 PINS), что выделяет его в общей линейке SBC.

Так как образы ОС на обоих SBC позволяют устанавливать Python3, то было принято решение реализовать задачу спектрального анализа на этом языке. Неоспоримыми преимуществами этого языка является большое количество программных модулей и возможности использования GUI интерфейса. За основу был взят проект, описанный в [1].

Для оцифровки сигнала использовалась USB звуковая карта (“свисток”), которая вставлялась в USB HUB, подключаемый к SBC. Звуковая карта имеет аудио вход и аудио выход. На вход (через кабель) подаётся исследуемый сигнал звуковой частоты с выхода звуковой карты ПК. Программа CoolEdit, запускаемая на ПК и генерирует сигнал заданной формы и частоты. Естественно, что для отладки алгоритма ДПФ необходим тестовый синусоидальный сигнал. Был выбран сигнал с частотой 1кГц. Из теории известно, что ожидаемым результатом должна быть одна спектральная составляющая.

Как показал дальнейший анализ разрядность АЦП в “свистке”, 16 бит и максимальная частота дискретизации 44100Гц.

При превышении амплитуды исследуемого сигнала опорного напряжения АЦП наблюдалось срезание верхушек синуса, что мгновенно отображалось в виде расширения спектра.

При выполнении задачи были преодолены проблемы с установкой необходимых библиотек и модулей Python3 (pip3, PyAudio, numpy, matplotlib, wave). Это заняло естественно большой промежуток времени. Было потрачено много сил. Как правило, это было связано с несовместивостью различных версий.

Для исследования возможностей использовались разные оконные менеджеры, для Mango Pi –xfce4 и icewm, для BBB - lxqt.

В результате работы программы на экран HDMI монитора, подключенного к Mango Pi, выводится изображение оцифрованного сигнала и результирующий спектр. Для такого же результата на BBB использовался механизм X11 Windows операционной системы Linux, позволяющий отображать на экране ПК графику с удалённого рабочего стола SBC.

Сравнение обоих SBC показало сходное быстродействие. Т.о. при выборе конкретного одноплатного компьютера для спектрального анализа необходимо руководствоваться соображениями различия функционала Mango Pi и BBB и приемлемых размеров.

ЛИТЕРАТУРА

1. Audio Processing in Python Part II: Exploring Windowing, Sound Pressure Levels, and A-Weighting Using an iPhone X. [Электронный ресурс] Режим доступа: https://makersportal.com/blog/2018/9/17/audio-processing-in-python-part-ii-exploring-windowing-sound-pressure-levels-and-a-weighting-using-an-iphone-x
2. Audio Processing in Python Part I: Sampling, Nyquist, and the Fast Fourier Transform

[Электронный ресурс] Режим доступа:https://makersportal.com/blog/2018/9/13/audio-processing-in-python-part-i-sampling-and-the-fast-fourier-transform

УДК 004.4

Л. И. Клочкова, В. А. Ромашов (4 курс бакалавриата),
И. В. Никифоров, к.т.н., доцент

ИНТЕГРАЦИЯ ИНСТРУМЕНТА GRAFANA В ИНЖЕНЕРНЫЙ ПОРТАЛ КОМПАНИИ

Для поддержания высокого уровня разработки продуктов внутри компании весь потенциал сотрудников принято направлять на создание уникальных решений для поставленных задач. Неэффективное использование рабочего времени на поиск технической информации должно быть сокращено до минимума. Для выполнения данной цели многие организации используют различные рабочие пространства для оптимизации обмена информацией между работниками – базы знаний [1].

В инженерном портале компании участникам проектов необходимо на постоянной основе анализировать большое число разнообразных метрик: разработчикам - работоспособность и производительность сервисов для оценки их отказоустойчивости [2], менеджерам - статистику по выполненным за спринт, или любой другой промежуток времени, задачам с разными уровнями важности в рамках конкретной версии релиза, инженерам по тестированию - диагностику панелей с результатами тестирования [3], инженерам технической поддержки – исправность серверов, рабочих станций и офисного оборудования [4].

Существует набор инструментов и фреймворков мониторинга и визуализации данных. Примером мощной, популярной и развивающейся программной системы является Grafana [5], позволяющая пользователем отслеживать и анализировать данные, благодаря насыщенной визуализации с помощью различных типов графиков, тепловых карт, гистограмм, диаграмм и таблиц.

Целью данной работы является сокращение трудоемкости работы менеджеров и инженеров по анализу работоспособности сервисов и результатов экспериментов за счет интеграции фреймворка Grafana в инженерный портал компании.

Ввиду невозможности на стороне инженерного портала встроить iFrame с предоставленными платформой по сбору данных и построению отчетов на чтение графиками по причине работы веб-приложения на протоколе HTTP [6], было принято решение реализовать собственную библиотеку для отрисовки графиков по полученным с сервиса данным.

Разработанная библиотека должна являться оберткой над любым из существующих графических проектов с открытым исходным кодом для обеспечения обратной совместимости в случае смены графической библиотеки. Более того, созданное решение должно предоставлять возможность отображения нескольких видов графиков разного масштаба на одном холсте с определением оптимального шага на осях абсцисс и ординат.

Для разработки универсального средства визуализации данных требуется выбрать React-библиотеку, на основе которой будет происходить отрисовка графиков. На данный момент в мире существует большое количество готовых решений. В ходе работы был сформулирован ряд наиболее значимых критериев, на основе которых был проведен сравнительный анализ популярных на рынке React-библиотек для отображения данных. Итоги исследования приведены в таблице 1.

Результаты проведенного анализа существующих графических библиотек показали, что среди высокорейтинговых проектов наиболее мощным, перспективным и удобным в использовании инструментом является Visx [7], предоставляющий не готовые для отображения графики, а набор компонентов-конструкторов, используя которые, можно строить гибкие решения для поставленной задачи.

Таблица 1 – Сравнение React-библиотек для визуализации данных

		Rechart	**React-Vis**	**Victory**	**Visx**	**React-chartjs-2**		**Nivo**
Основные характеристики	**Размер**	618.6 kB	391.9 kB	665.7 kB	828.1 kB	**React-chartjs-2** 2.9 kB	**Chart.js** 199.4 kB	389.5 kB
	Tree shaking	+	+	+	+	+	+	+
	Кол-во подключаемых модулей	9	17	27	12	0	1	17
Популярность	**Кол-во звезд на Github**	21.1k	8.6k	10.5k	17.8k	6.1k	62.3k	11.9k
	Кол-во скачиваний за неделю через npm	1 319 620	72 187	196 298	17 975	997 358	2 244 493	5 403
	Кол-во открытых задач	477	308	279	125	64	308	93
Технология отрисовки		SVG	SVG, Canvas	SVG	SVG	Canvas		SVG, Canvas
Кастомизация	**Корпоративная цветовая палитра**	+	+	+	+	+		+
	Отметки данных	+	-	+	+	+		+
	Всплывающая подсказка	-	-	-	+	+		+
Лицензия		MIT	MIT	MIT	MIT	MIT	MIT	MIT

Таким образом, в работе была актуализирована задача интеграции инструмента Grafana [8] в инженерный портал компании по причине частого столкновения организаций с необходимостью внедрения базы знаний, служащей единым информационным ресурсом для всех сотрудников с целью повышения продуктивности команд путем снижения трудоемкости персонала по мониторингу ряда метрик по работающим сервисам, тестам и задачам проектов.

Результаты проведенного анализа существующих графических библиотек по таким критериям, как размер, популярность, технология отрисовки, возможность кастомизации и лицензия, показали, что Visx является наиболее подходящим инструментом.

Работа выполняется при постановке задачи от компании YADRO.

ЛИТЕРАТУРА

1. Сеченова В. В. Анализ необходимости создания базы знаний в ИТ-компаниях // Информационные технологии в строительных, социальных и экономических системах. - 2018. - № 3 (13). - С. 73-77.
2. Агеев Д. Ю., Воинов Н. В. Повышение отказоустойчивости для надежного соединения между клиентом и сервером // Современные технологии в теории и практике программирования: сборник материалов конференции, 26 апреля 2022 года / Санкт-Петербургский политехнический университет Петра Великого. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", - 2022. - С. 187-188.
3. П. Д. Дробинцев, В. П. Котляров, И. В. Никифоров, А. А. Летичевский Инкрементальный подход к технологии создания тестов для индустриальных проектов // Моделирование и анализ информационных систем. – 2014. – Т. 21, № 6. – С. 144-154. – EDN TCCAKB.
4. 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. – EDN CFRNSJ.
5. Яремчук С. Ставим мониторинг Prometheus + Grafana // Системный администратор. - 2017. - №5 (174). - С. 36-44.
6. Гамидов Ш. С. Использование iframe в веб-разработке // Научный аспект. - 2023. - №6. - С. 2214-2222.
7. Visx: сайт. – URL: https://airbnb.io/visx (дата обращения: 03.03.2024).
8. Alexandrov E. I., Alexandrov I. N., Mineev M. A. Grafana and Splunk as an example of the data visualization solution for the modern data taking systems // Selected Papers of the 7th International Conference Distributed Computing and Grid-technologies in Science and Education, Dubna, 04–09 July 2016 . - 2016. - P. 91-96.

УДК 004.453

А. А. Кремлев (2 курс магистратуры),
Д. В. Дмитриев., к.т.н., доцент

ИССЛЕДОВАНИЕ БЫСТРОДЕЙСТВИЯ АЛГОРИТМОВ УПРАВЛЕНИЯ ОДНОПЛАТНЫМИ КОМПЬЮТЕРАМИ MANGO PI И RASPBERRY PI С ИСПОЛЬЗОВАНИЕМ MQTT ПРОТОКОЛА ЧЕРЕЗ МОБИЛЬНОЕ УСТРОЙСТВО НА БАЗЕ ANDROID

Сегодня требования к эффективности и удобству управления устройствами постоянно растут. Одноплатные компьютеры Mango Pi и Raspberry Pi предоставляют уникальные возможности для создания гибких и недорогих систем управления. Актуальность данного исследования обусловлена ключевыми факторами: встраиваемость и импортозамещение.

Проведенное исследование показало, что разработанные алгоритмы управления для архитектуры RISC-V [1], позволили создать компактные и мощные системы автоматизированного управления с применением MQTT (Message Queuing Telemetry Transport) протокола, который представляет собой легковесный, надежный и эффективный

механизм обмена сообщениями [2-4]. MQTT демонстрирует значительные преимущества в контексте промышленной автоматизации. Одним из главных достоинств данного протокола является его минимальное потребление ресурсов сети и устройств, что делает его идеальным выбором для массового использования в распределенных системах, где ограничены пропускная способность и энергопотребление.

Научная новизна данной исследовательской работы заключается в создании полноценной системы управления одноплатными компьютерами, с использованием мобильного устройства Android в качестве интерфейса управления и удаленного сервера в качестве посредника передачи команд и сравнении реализации под архитектуру RISC-V устройства Mango Pi с реализацией под ARM архитектуру устройства Raspberry Pi. При этом обеспечивается интеграция разработанной системы в целостное решение, предоставляющее пользователям удобный и интуитивно понятный интерфейс для управления GPIO портами. Внедрение данной системы в сферу умного дома или промышленной автоматизации уменьшает время отклика и улучшает процесс управления и контроля, предоставляя гибкие возможности дистанционного мониторинга и управления.

Таким образом, создается целостная система, объединяющая мобильное устройство, удаленный сервер и одноплатный компьютер в единое управляемое пространство.

Для решения поставленных в исследовательской работе задач использовались методы разработки и тестирования эффективных алгоритмов управления одноплатными компьютерами в том числе под архитектуру RISC-V и ARM. Методы объектно-ориентированного проектирования и программирования. Проведен анализ возможностей архитектуры RISC-V для задач удаленного управления устройствами посредством GPIO портов. Экспериментальные испытания алгоритмов проведены с использованием устройства Mango Pi и Raspberry Pi для оценки их производительности и эффективности.

Архитектура, обеспечивающая эффективную передачу команд по протоколу MQTT между мобильным устройством Android и одноплатным компьютером, представлена на рисунке 1. Особое внимание уделено разработке программного обеспечения для мобильного устройства Android, предоставляющего удобный интерфейс пользователя для взаимодействия с устройством на базе архитектуры RISC-V и ARM. Представленная система состоит из трех основных компонент: SBC – одноплатный микрокомпьютер (Mango Pi, Raspberry Pi), Backend [5] – удаленный сервер, AndroidApp – приложение под OS Android для управления микрокомпьютером.

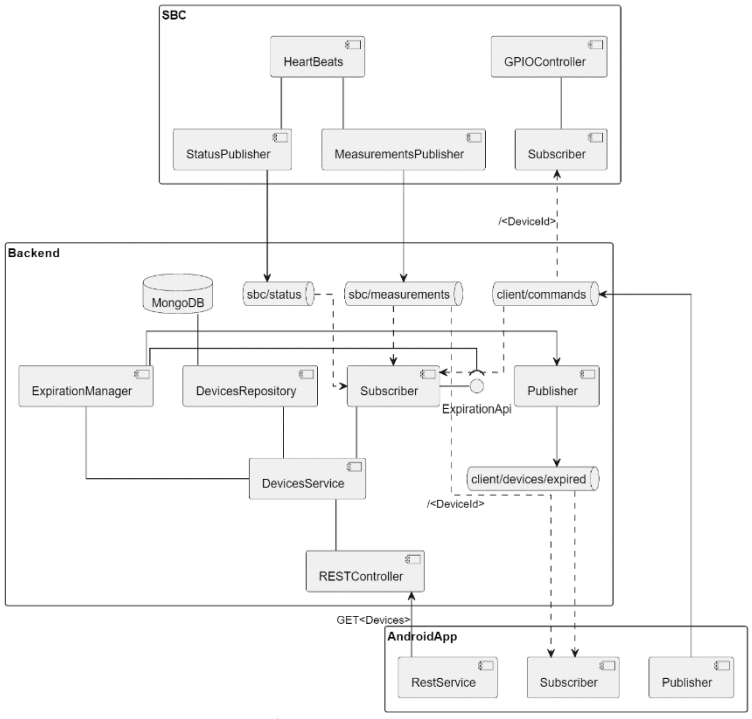


Рисунок 1 – Архитектура системы.

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.

В результате проведенного моделирования получены средние значения отклика системы на команду (таблица 1).

Таблица 1 – Средние значения отклика системы на команду

	MangoPi	RaspberryPi
AVG Response Time,	0,3425	0,3595

Полученные результаты являются приемлемыми для применения в системах умного дома и большинстве систем промышленной автоматизации (время отклика ~350 мс.), но недостаточно для систем реального времени.

ЛИТЕРАТУРА

1. Сара Л. Харрис, Дэвид Харрис // Цифровая схемотехника и архитектура компьютера: RISC-V / пер. с англ. В. С. Яценкова, А. Ю. Романова; под ред. А. Ю. Романова. – М.: ДМК Пресс, 2021. – 810 с.: ил
2. CloudMQTT CloudMQTT Documentation [Электронный ресурс] // Режим доступа: https://www.cloudmqtt.com/docs/index.html //(Дата обращения Март 2024).
3. Протокол MQTT [Электронный ресурс] // Режим доступа: https://cqr.company/ru/wiki/protocols/mqtt-protocol/ (Дата обращения Март 2024)
4. Dipa Soni A SURVEY ON MQTT: A PROTOCOL OF INTERNET OF THINGS(IOT) // Chennai India International Conference on Telecommunication, Power Analysis and Computing Techniques (ICTPACT). 2017. С.1-5
5. Документация бекенд сервиса Micronaut [Электронный ресурс] // Режим доступа: https://micronaut.io/ (Дата обращения Март 2024).

УДК 004.421

Д. О. Королев (2 курс магистратуры),
И. В. Никифоров, к.т.н., доцент,
С. М. Устинов, д.т.н., профессор

ПОДХОД ФОРМИРОВАНИЯ ОТВЕТОВ ОТ МНОЖЕСТВА МИКРОСЕРВИСОВ С ИСПОЛЬЗОВАНИЕМ ПАТТЕРНА BFF

Для разработки качественных программных продуктов используется микросервисный подход, который обеспечивает независимость разработки, отказоустойчивость и масштабируемость системы [1]. Однако при большом количестве микросервисов проблемой становится формирование ответов для пользователей на основе данных от множества микросервисов. Поэтому актуальной задачей является повышение эффективности формирования таких ответов. Причем под эффективностью в данном случае понимается как скорость ответа, так и возможность динамического формирования ответа.

Одним из самых распространённых и простых с точки зрения реализации подходов формирования ответов от множества микросервисов является подход без применения паттернов. При таком подходе логика по формированию ответа расположена на стороне пользовательского интерфейса. Такой подход имеет множество минусов, таких как небольшая производительность, сложность горизонтальной масштабируемости и отказоустойчивости и сложность реализации логики формирования гибкого ответа. Однако реализация данного подхода требует наименьшее количество ресурсов и хорошо подходит для создания минимально жизнеспособного продукта (англ. minimum viable product, MVP).

Вторым рассматриваемым подходом является подход с применением паттерна API Gateway [2]. API Gateway - промежуточный слой между клиентскими приложениями и микросервисами. Его основной функцией является предоставление единой точки входа для запросов от клиентов. Обычно он используется для аутентификации, авторизации, маршрутизации запросов и обработки ошибок. API Gateway также может выполнять функции кеширования, логирования и мониторинга [3]. Но подход с добавлением логики формирования пользовательского ответа излишне усложняет API Gateway и является анти-паттерном.

Последним рассматриваемым подходом является применение паттерна BFF (Backend for Frontend), который заключается в создании отдельных микросервисов серверной части для каждого клиента или типа клиентов, абстрагирующих пользовательский интерфейс от сложной серверной архитектуры и предоставляющих упрощенный, однородный и специализированный интерфейс [4]. Это позволяет разрабатывать и поддерживать логику формирования гибкого ответа от множества микросервисов в отдельном масштабируемом микросервисе, а не на стороне пользовательского интерфейса, что упрощает разработку и развертывание новой логики. Поскольку в основе данного подхода лежит микросервисная архитектура, для развертывания системы могут быть применены инструменты Docker Swarm и Kubernetes, обеспечивающие автоматическую балансировку нагрузки, горизонтальную масштабируемость и отказоустойчивость [5]. Однако не стоит забывать, что внедрение данного подхода приводит к увеличению числа микросервисов и усложнению архитектуры системы.

Подход с применением паттерна BFF является лучшим выбором для повышения эффективности формирования ответов от множества микросервисов.

Таким образом, целью работы является повышение эффективности формирования ответа от множества микросервисов за счет применения паттерна BFF в подходе, реализованном в масштабируемом микросервисе с использованием языка программирования Golang.

Реализованный микросервис представляет собой REST API, реализованный на языке Golang с применением пакета Echo. Он осуществляет асинхронное получение данных из указанных в конфигурационном файле по REST API или gRPC и динамически формирует ответ на основе структуры, полученной из микросервиса управления структурой ответа. Сформированный гибкий ответ запрашивается пользовательским интерфейсом и отображается пользователю. Для обеспечения масштабируемости и отказоустойчивости используется Docker Swarm. В микросервисе также реализовано логирование на нескольких уровнях и функциональность Circuit Breaker и Retry. На рисунке 1 представлена схема подхода с применением паттерна BFF.

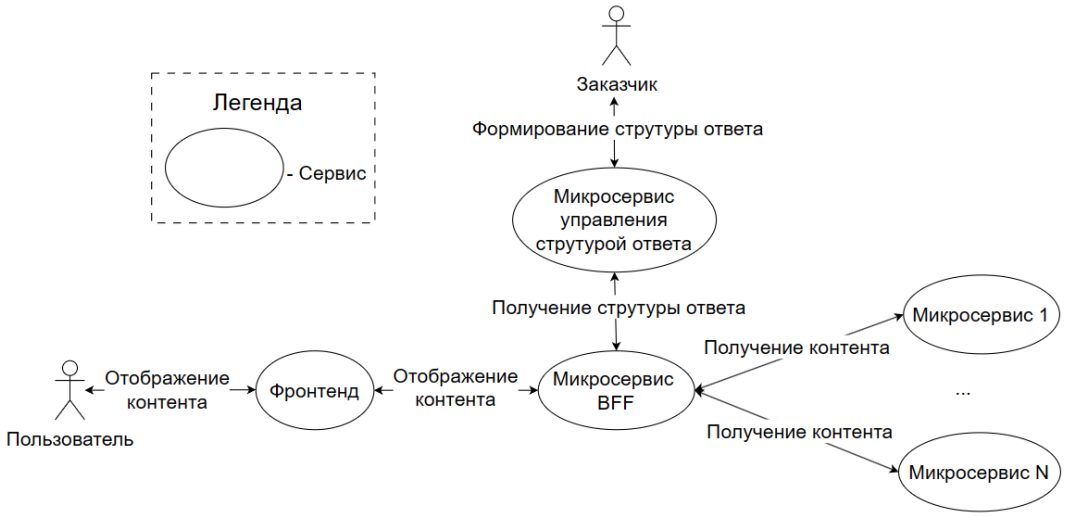


Рисунок 1 – Подход формирования ответа с применением паттерна BFF

Вычисленное по времени ответов от пользовательского интерфейса математическое ожидание по пяти экспериментам составило 313.8 мс для системы с реализованным микросервисом и 325.28 мс для системы без применения паттернов.

В работе было проведено исследование существующих подходов формирования ответа от множества микросервисов и реализован микросервис с использованием паттерна BFF, увеличивающий эффективность формирования ответа от множества микросервисов и удовлетворяющий требованиям по независимости разработки, улучшению производительности, возможности горизонтальной и вертикальной масштабируемости, наличию отказоустойчивости и другим критериям.

ЛИТЕРАТУРА

1. Ивлев, В. А. Обработка данных в геоинформационных системах для выбора местоположения рекламы / В. А. Ивлев, И. В. Никифоров, Т. В. Леонтьева // Современные технологии в теории и практике программирования : сборник материалов конференции, Санкт-Петербург, 19 апреля 2019 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2019. – С. 27-30. – EDN DNQPMC.
2. Нуркаев Р., Пивоваров В.В. Управление API-шлюзом на основе архитектуры микросервиса // Инновации и инвестиции. – 2023. – №5. – С. 193-197.
3. Analysis of student activity on the e-learning course based on "OpenEdu" platform logs / N. D. Barsukov, I. M. Sysoev, A. A. Pereskokova [et al.] // Proceedings of the Institute for System Programming of the RAS. – 2020. – Vol. 32, No. 3. – P. 91-100. – DOI 10.15514/ISPRAS-2020-32(3)-8. – EDN QYVJBQ.
4. Florian Auer, Valentina Lenarduzzi, Michael Felderer, Davide Taibi. From monolithic systems to Microservices: An assessment framework // Information and Software Technology. – 2021. – №137.
5. Сафронов, Д. Автоматическая балансировка нагрузки между потоковой обработкой данных и внутренними задачами кластера с использованием Kubernetes / Д. Сафронов, К. М. Стоноженко, И. В. Никифоров // Современные технологии в теории и практике программирования : сборник материалов конференции, Санкт-Петербург, 23 апреля 2020 года / Санкт-Петербургский политехнический университет Петра Великого; Dell Technologies; EPAM Systems. – Санкт-Петербург: ПОЛИТЕХ-ПРЕСС, 2020. – С. 165-167. – EDN VOXGIM.

УДК 004.428.4

А. В. Лысенко (4 курс бакалавриата),
И. А. Шемякин, ст. преподаватель,
А. В. Касилов, ассистент

ГЕНЕРАТОР SQL ЗАПРОСОВ НА ОСНОВЕ ПОЛЬЗОВАТЕЛЬСКИХ ТИПОВ В РАМКАХ АСИНХРОННОГО КЛИЕНТА СИСТЕМЫ УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ POSTGRESQL

Взаимодействие с системами управления базами данных является неотъемлемой составляющей большого числа программных продуктов [1]. Рынок систем управления базами данных растёт высокими темпами, удовлетворяя широкий круг потребностей пользователей. По данным рейтинга издания DB-Engines[2] самой популярной системой управления данными в 2023 году стал PostgreSQL. Из 417 отслеживаемых систем PostgreSQL продемонстрировал наибольший рост популярности, что закономерно, так как PostgreSQL является самой широко используемой свободно распространяемой системой управления реляционными базами данных.

Одним из ограничений при работе с системами управления базами данных, реализующих реляционную модель, является необходимость знания определенного диалекта языка запросов SQL, а также избежания антипаттернов в построении запросов [3]. Так как

существует достаточно много диалектов, имеющих значительные различия, появляется потребность в инструменте, реализующем нативное построение SQL запросов.

Высокая потребность в качественной работе с системами управления базами данных диктует потребность в программном обеспечении, реализующем взаимодействие с ними. Таким образом, существует необходимость в появлении клиентов систем управления базами данных, отражающих ключевые тенденции современной разработки программного обеспечения. Одной из тенденций является разработка программ, работающих асинхронно, то есть поддерживающих многопоточную работу с ресурсами. Создание асинхронных программных продуктов позволяет эффективно работать с высоконагруженными системами.

Таким образом, *целью* данной работы является повышение скорости и качества взаимодействия с системой управления базами данных PostgreSQL за счёт разработки асинхронного клиента с конструктором SQL запросов.

Для достижения этой цели необходим решение нескольких *задач:*

1. Обзор существующих клиенты системы управления базами данных PostgreSQL и конструкторов SQL запросов
2. Исследование протокола взаимодействия с системой управления базами данных PostgreSQL
3. Разработка конструктора SQL запросов на базе асинхронного клиента системы управления базами данных PostgreSQL
4. Проведение тестирования приложения

В рамках исследования нас интересуют клиенты, которые применимы на C++ или поддерживают асинхронность.

Таблица 1 – Сравнительная таблица существующих решений-клиентов PostgreSQL [4]

Решение	Клиент на языке C++	Зависимость от libpq	Поддержка асинхронности	Библиотека примитивов синхронизации	Наличие конструктора SQL запросов
libpqxx	+	+	-	-	-
Pgfe	+	+	-	-	-
PgCC	+	+	-	-	простейшие
ozo	+	+	+	Boost.Asio	-
rust-postgres	-	-	+	tokio	-
pgmoon	-	-	+	LuaSocket,cqueries	-
sqlpp11	+_	+	-	-	+

Таким образом, ни одно существующее решение не предоставляет независимый от libpqxx клиент с возможностью ORM-like конструирования запросов.

В качестве языка разработке был выбран язык высокого уровня - C++. Выбор продиктован высокой производительностью и возможностью полного контроля над используемыми ресурсами. Также язык C++ поддерживает асинхронное программирование, что является одним из критериев выбора языка программирования для разработки.

Так как разработка клиент-серверного протокола [5] является задачей асинхронного программирования, было принято решение использовать библиотеку Seastar[6]. Seastar представляет собой высоко производительный фрэймворк языка C++ для серверов, поддерживающих многопоточность. Библиотека предоставляет широкий набор примитивов синхронизации.

Future и Promise являются основными инструментами асинхронного программирования в Seastar. Future представляет собой результат, который, возможно, еще не был вычислен, например, буфер, считываемый с диска, или результат функции, выполняемой на другом процессоре. Promise позволяет в конечном итоге вычислить Future, присвоив ему значение. Обычный способ работы с Future — привязывать к ним континюаторы. Континюатор — это блок кода (обычно лямбда), который вызывается, когда Future присваивается значение (будущее разрешено); континюатор может затем получить доступ к фактическому значению.

Так как целью является разработка конструктора SQL запросов, программа будет часто работать со строковыми данными. Во избежание чрезмерного копирования строк используется библиотека {fmt}[7]. {fmt} — это библиотека форматирования с открытым исходным кодом, предоставляющая быструю и безопасную альтернативу потокам C stdio и C++ iostreams.

Для упрощения сборки проекта было принято использовать набор инструкций Makefile.

ЛИТЕРАТУРА

1. Новиков Б. А. Основы технологий баз данных: учебное пособие / Б. А. Новиков, Е. А. Горшкова, Н. Г. Графеева; под ред. Е. В. Рогова. — 2-е изд. — М.: ДМК Пресс, 2020. — 582 с.
2. Рейтинг систем управления базами данных DB-engines [Электронный ресурс] Режим доступа; https://db-engines.com/en/ranking
3. Bill Karwin “SQL Antipatterns” Version: 2010-6-9 -2010 – 334pp.
4. Клиенты СУБД PostgreSQL [Электронный ресурс] Режим доступа: https://wiki.postgresql.org/wiki/PostgreSQL_Clients#Other_Resources
5. Рогов Е. В. PostgreSQL изнутри. — М.: ДМК Пресс, 2022. — 660 с.
6. Документация Seastar [Электронный ресурс] Режим доступа: https://docs.seastar.io/master/index.html
7. Документация {fmt} [Электронный ресурс] Режим доступа: https://fmt.dev/latest/index.html

УДК 004.415.2

А. Ш. Муллагалиева (2 курс магистратуры),
И. В. Никифоров, к. т. н., доцент

СЕРВИС ДИНАМИЧЕСКОГО ПОДКЛЮЧЕНИЯ ПРОГРАММНЫХ МОДУЛЕЙ ДЛЯ СИСТЕМЫ УПРАВЛЕНИЯ ЦОД

Большинство современных организаций (предприятий) так или иначе используют совокупность объектов и сервисов, относящихся к области ИТ, которая называется «ИТ-инфраструктурой» [1,2]. Она включает в себя аппаратные и программные ресурсы, сети передачи данных и инженерные системы.

Для сложной ИТ-инфраструктуры нужна система управления, общее название DCIM – Data Center Infrastructure Management [3]. Управление инфраструктурой центра обработки данных (DCIM) — это интеграция дисциплин информационных технологий (ИТ) для централизации мониторинга, управления и интеллектуального планирования пропускной способности критически важных систем центра обработки данных.

Имеется рабочий программный продукт в виде приложения для управления и мониторинга центра обработки данных, написанный на языке Go с использованием микросервисной архитектуры на базе Hashicorp стека (Vault-Nomad-Consul [4]).

Целью данной работы является расширение в этом приложении device manager - сервиса действий над устройствами, динамическим подключением программных модулей.

Таким образом, можно выделить следующие задачи:

- рассмотреть существующие системы управления ЦОД [5];
- реализовать плагин-сервис динамического подключения программных модулей;
- продемонстрировать работоспособность системы.

Для реализации описанной функциональности планируется использовать плагин систему go-plugin [6] от Hashicorp для управления плагинами. Предполагаемая модель взаимодействия внутри device manager представлена на рисунке 1.

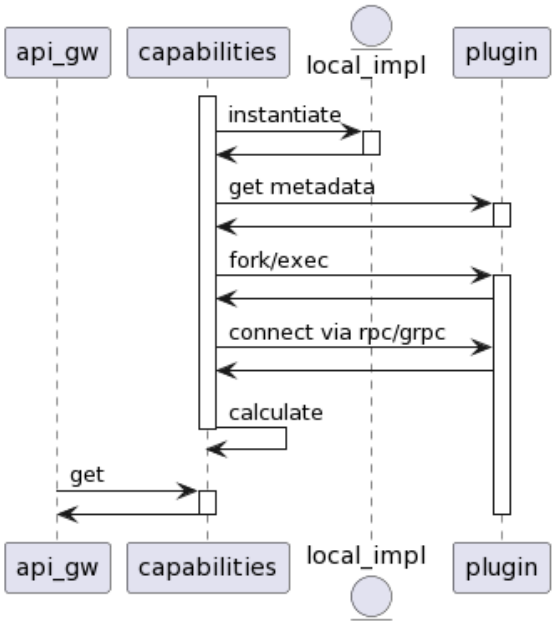


Рисунок 1 – Модель взаимодействия с плагинами

При запуске device manager проинициализирует локальные имплементации интерфейсов к устройствам, запустит плагин-сервис, соберет информацию о возможностях плагинов, после чего рассчитает карту возможных операций (capabilities) над устройствами.

На входе у плагин-сервиса будут модули, которые необходимо проинициализировать и запустить, также он будет предоставлять имплементации для интерфейсов device manager и выполнять запрашиваемые задачи.

Преимущество данного подхода в том, что сервис позволит определять имплементации на любом языке, поддерживаемом RPC/gRPC, и просто интегрировать их в основное приложение. Так же разработку самих плагинов можно передать другим командам для ускорения и эффективности работы.

ЛИТЕРАТУРА

1. Ивлев, В. А. Актуальность автоматизированной настройки инфраструктуры ит-проекта / В. А. Ивлев, И. В. Никифоров // Современные технологии в теории и практике программирования : Сборник материалов конференции, Санкт-Петербург, 26 апреля 2022 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2022. – С. 79-80. – EDN QQAVXT.
2. Генерация информационно-технологической инфраструктуры проекта на основе неформализованных требований / В. А. Ивлев, Г. В. Мироненков, И. В. Никифоров, А. Д. Ковалев // Современные технологии в теории и практике программирования : Сборник материалов научно-практической конференции студентов, аспирантов и молодых ученых, Санкт-Петербург, 26–27 апреля 2023 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2023. – С. 242-244. – EDN CENYIP.
3. "Data Center Management". The Data Center Journal. Retrieved October 28, 2018.
4. HashiCorp. [Электронный ресурс] Режим доступа: https://github.com/hashicorp.
5. Algorithm for calculating TCO and SCE metrics to assess the efficiency of using a data center / B. Denisenko, M. Tyanutov, I. Nikiforov, S. Ustinov // 2nd International Conference on Computer Applications for Management and Sustainable Development of Production and Industry (CMSD-II-2022), Dushanbe, 21–23 декабря 2022 года. – Washington: SPIE-SOC PHOTO-OPTICAL INSTRUMENTATION ENGINEERS, 2023. – P. 1256403. – DOI 10.1117/12.2669285. – EDN TSQXHS.
6. Go-plugin. [Электронный ресурс] Режим доступа: https://github.com/hashicorp/go-plugin.

УДК 004.453

П. А. Панкова (4 курс бакалавриата),
И. В. Никифоров, к. т. н., доцент

МЕТОДЫ ОЦЕНКИ ПРОИЗВОДИТЕЛЬНОСТИ LINUX-ЯДРА НА ПРИМЕРЕ ПРИМИТИВА СИНХРОНИЗАЦИИ FUTEX

Исследование и разработка методов оценки производительности ядра операционной системы Linux является ключевым шагом в оптимизации работы вычислительных систем [1]. Такие методы не только обеспечат более глубокое понимание эффективности ядра, но и позволят создавать новые стратегии управления ресурсами, которые повысят общую производительность. Поэтому разработка новых подходов к оценке производительности становится неотъемлемой частью адаптации ядра к различным сценариям использования.

В контексте данной работы объектом исследования был выбран примитив синхронизации futex, на примере которого и будут разрабатываться методы оценки производительности Linux-ядра [2]. Linux futex представляет собой эффективный механизм синхронизации пользовательских потоков, обеспечивая минимальное взаимодействие с ядром операционной системы. Необходимость замера производительности системного вызова futex вытекает из стремления создать более точные и адаптивные методики оценки ядра Linux [3], которые учитывают особенности его взаимодействия с futex.

Итак, *целью* работы является составление универсальной методики для сравнения различных системных вызовов в операционной системе Linux, а также получение сводной таблицы и выбор подходящей метрики для анализа системного вызова futex на платформах ARM64, RISC-V, x86_64.

Для достижения цели необходимо решить следующие задачи:

1. Провести обзор существующих решений и литературы, выделить основные методы исследования производительности ядра Linux.
2. Определение архитектурно независимой метрики для сравнения системного вызова futex.
3. Выбрать или разработать инструменты для нагруженного тестирования системного вызова futex.
4. Провести инструментом тестирования замеры по выбранной метрике системного вызова futex на всех таргетированных платформах.
5. Замер по выбранной метрике системного вызова futex на всех таргетированных платформах.
6. Провести анализ первопричин полученных данных с выявлением возможных проблем на каждой из платформ.
7. Подвести итог и продемонстрировать архитектуру, на которой системный вызов отработал быстрее всего.

Выбор метрики, инструмента тестирования и отладки

Stress-ng [4] выбран в качестве инструмента для нагрузочного тестирования системного вызова futex, поскольку он обеспечивает гибкость в создании разнообразных нагрузочных сценариев. Это позволяет оценить производительность ядра Linux в различных условиях, что соответствует основной цели исследования. Stress-ng был изменен в тесте stress-futex.c: удалён тайм-аут для futex_wait syscall, а также был добавлен подсчет и вывод количества wake и wait системных вызовов.

Strace [5] – это отладочный инструмент, который может быть использован для измерения времени выполнения системных вызовов. Это делает его идеальным инструментом для измерения производительности системного вызова futex.

Strace записывает время, затрачиваемое на каждый системный вызов. Впоследствии записанная трассировка обрабатывается написанным мною парсером Python, который

вычисляет среднее, минимальное, максимальное и медианное время, затрачиваемое на каждый системный вызов. Время нормализуется до 125 МГц.

Результаты

Таблица 1 – Сравнительный анализ времени системного вызова futex.

Syscall	Metrics	x86	Lichee Pi	ARM	x86 vs Lichee Pi	x86 vs ARM	Lichee Pi vs ARM
Successful wake, milliseconds	minimum	0,096	0,282	0,247	2,9	2,6	1,1
	maximum	42,272	34,304	148,978	0,8	3,5	0,2
	average	0,228	1,153	0,704	5,1	3,1	1,6
	median	0,224	1,126	0,671	5,0	3,0	1,7
Unsuccesful wake, milliseconds	minimum	0,064	0,243	0,194	3,8	3,0	1,3
	maximum	5,264	25,088	6,871	4,8	1,3	3,7
	average	0,124	0,622	0,383	5,0	3,1	1,6
	median	0,120	0,614	0,318	5,1	2,6	1,9
Wait, milliseconds	minimum	0,168	0,678	0,601	4,0	3,6	1,1
	maximum	42,224	52,800	149,420	1,3	3,5	0,4
	average	0,241	1,260	0,696	5,2	2,9	1,8
	median	0,232	1,216	0,654	5,2	2,8	1,9

В таблице 1 показаны четыре различных типа центральных тенденций, которые могут характеризовать полученную выборку из трасс strace.

Таким образом, после анализа различных подходов к оценке эффективности syscall *были выбраны методы для оценки системного вызва* с определенной *метрикой минимума*. Это оправдано тем фактом, что минимум обеспечивает самый справедливый способ сравнения идентичных операций при разных условиях. Он игнорирует потенциальные замедления, вызванные различными операционными системами, и показывает "чистое" время, необходимое для выполнения системного вызова futex. По метрике минимального времени работы системного вызова futex платформа на x86 обгоняет остальные, а значит на ней наиболее оптимизированная реализация futex.

В дальнейшем возможно провести более детальную оценку полученных результатов, используя такие инструменты профилирования, как perf, ebpf или ftrace. Инструмент построения Flame Graphs [6] позволит провести визуальный анализ данных по стеку функций ядра.

ЛИТЕРАТУРА

1. Brendan Gregg, Linux Performance Analysis and Tools, 2021 [Электронный ресурс] Режим доступа: https://www.brendangregg.com/linuxperf.html
2. Michael Kerrisk, The Linux Programming Interface, published in October 2010, No Starch Press, ISBN 978-1-59327-220-3.
3. Гребенников Н. А., Щербань А. Б. Системные вызовы в ядре Linux // Современные информационные технологии, Пенза, 2018 год – С. 70-73.
4. Нагрузочное тестирование сервера на Linux [Электронный ресурс] Режим доступа: https://redos.red-soft.ru/base/manual/admin-manual/utilites/hardware-stress-test/stress-ng/
5. Linux manual page [Электронный ресурс] Режим доступа: https://man7.org/linux/man-pages/man1/strace.1.html
6. Flame Graphs [Электронный ресурс] Режим доступа: https://www.brendangregg.com/flamegraphs.html

УДК 004.453

А. А. Пономарев (2 курс бакалавриата),
И. В. Никифоров, к.т.н., доцент,
О. В. Александрова, ст. преподаватель

МЕТОДИКА РАЗРАБОТКИ ANSIBLE-МОДУЛЕЙ ДЛЯ СПЕЦИФИКАЦИИ SWORDFISH

Введение: Большинство современных систем хранения данных (СХД) предоставляют REST API для управления ими, включая настройку, конфигурирование, выполнение различных команд, получение данных логирования, работу с пользователями и т. д. В зависимости от производителя СХД REST API могут отличаться друг от друга. Если пользователь использует несколько разных систем, их одновременная поддержка становится проблемой: нужно учитывать многообразие протоколов, интерфейсов, команд и способов взаимодействия с API [1].

Одним из способов сделать управление разными СХД менее трудоемким является использование систем, которые удовлетворяют спецификации Swordfish [3], в связке с инструментом Ansible [4].

Спецификация Swordfish предоставляет богатый функционал для управления ресурсами хранилищ данных, конфигурации и мониторинга систем, поддерживает механизмы аутентификации и авторизации для обеспечения безопасности. Swordfish призвана решить большое количество проблем с универсальностью управления СХД в современной инфраструктуре.

Инструмент автоматизации Ansible имеет много преимуществ для использования с СХД. Одно из ключевых для задачи управления СХД – интегрируемость с другими инструментами, например, с системами мониторинга. Это обеспечивает согласованность и единый интерфейс для управления всей инфраструктурой, включая хранилища данных.

В итоге комбинация спецификации Swordfish и инструмента Ansible дает неплохое техническое решение для ввода в эксплуатацию и автоматического управления СХД в среде с большим количеством комплексов.

Цель работы: К сожалению, готовой коллекции Ansible-модулей [5], поддерживающих Swordfish, для взаимодействия с СХД не было, поэтому мы, командой от лаборатории при высшей школе программной инженерии, решили разработать эти модули совместно с инженерами YADRO.

При разработке таких комплексных продуктов как Ansible-коллекция [5] довольно сложно понять каким образом оптимально выстроить производственный процесс, чтобы все члены команды понимали свою задачу на данном этапе разработки и как ее выполнять [2]. Для этого была предложена методика разработки Ansible-модулей для спецификации Swordfish, которой мы следовали во время работы над коллекцией модулей.

Описание методики решения: Методика, изображенная на рисунке 1, включает в себя 6 основных шагов, состоящих из нескольких подзадач. На первом шаге необходимо ознакомиться с основными разделами спецификации Swordfish, которые являются приоритетными для успешного написания Ansible-модулей. Следующий этап включает в себя выбор ресурса таким образом, чтобы процесс разработки был последовательным и на этапе кодирования возникало минимум проблем. Шаг 3, в свою очередь, заключается в выборе сценария использования ресурса в порядке важности для пользователя. На четвертом шаге начинается непосредственно написание кода. На этом этапе важно правильно продумать архитектуру вспомогательных классов для моделей и утилит, чтобы код был легко расширяем и не трудоемок в поддержке. Последние два шага включают в себя написание интеграционных и unit-тестов [7], а также публикацию коллекции Ansible-модулей.

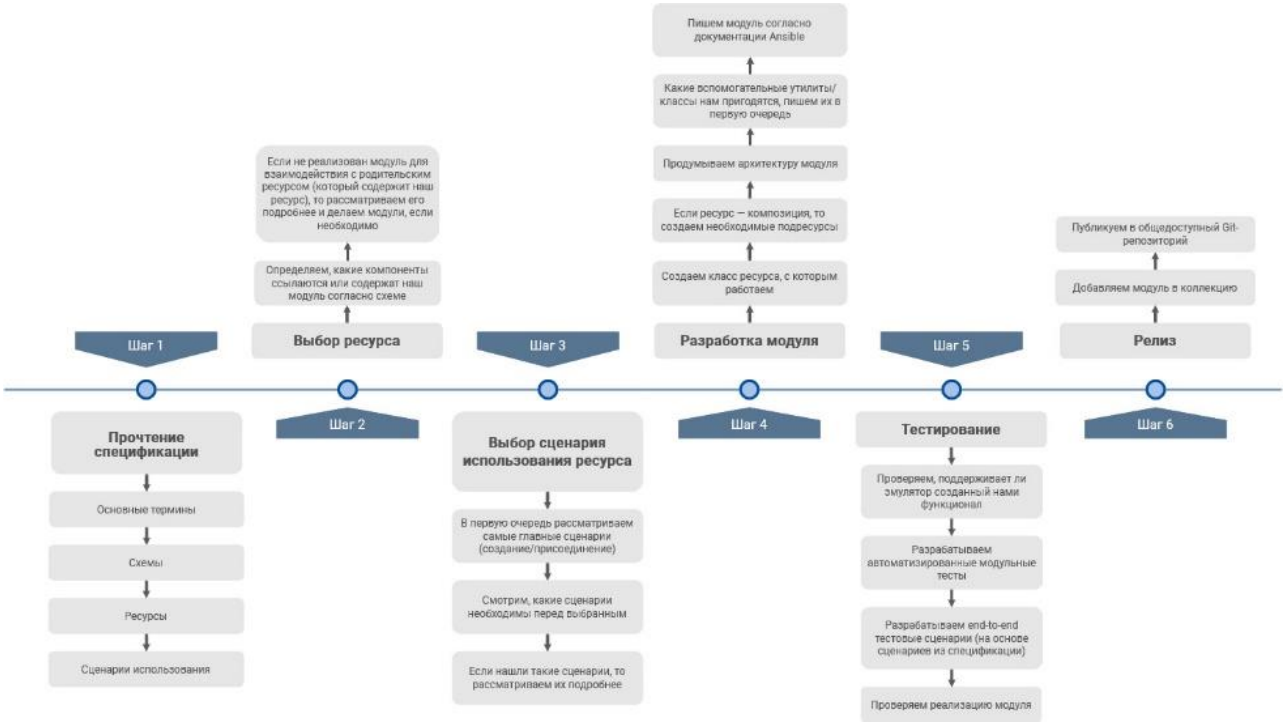


Рисунок 1 – Схема методики разработки Ansible-модулей для спецификации Swrodfish

Созданная в работе Ansible-коллеция [8], была разработана по представленной методике. Пример использования готового модуля представлен на Рисунке 2. Данный модуль необходим для подтверждения того, что новое пространство имен (Namespace) может быть создано.

```
TASK [namespace_can_be_created : Test if namespace can be created | Get info] ***
ok: [testhost]

TASK [namespace_can_be_created : set_fact] *************************************
ok: [testhost]

TASK [namespace_can_be_created : Test if namespace can be created | Show fact info] ***
ok: [testhost] => {
    "msg": [
        "Fact if namespace can be created: True"
    ]
}

TASK [namespace_can_be_created : Test if namespace can be created | Check received info] ***
ok: [testhost] => {
    "changed": false,
    "msg": "All assertions passed"
}

PLAY RECAP *********************************************************************
testhost                   : ok=7    changed=0    unreachable=0    failed=0    skipped=0    rescued=0
   ignored=0
```

Рисунок 2 – Пример модуля, написанного по предложенной методике

Выводы: В итоге, благодаря предложенной методике разработанная коллекция Ansible-модулей является легко интегрируемой с существующими Ansible-сценариями, предоставляет прозрачные интерфейсы для основных операций, таких как создание, чтение, обновление и удаление ресурсов. Это делает автоматизацию управления хранилищем данных более доступной и понятной для пользователя. Также, методика позволяет эффективно создавать новые модули для спецификации Swordfish и помогает новым членам команды в короткие сроки включаться в проект.

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. – EDN XXPLRR.

2. 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. – EDN VDVCYX.
3. Ahlvers R. Swordfish Overview: Extending Data Center Control to the World of Storage //Companion Proceedings of the10th International Conference on Utility and Cloud Computing. – 2017. – С. 239-239.
4. Hochstein L., Moser R. Ansible: Up and Running: Automating configuration management and deployment the easy way. – " O'Reilly Media, Inc.", 2017.
5. Ansible Documentation. [Электронный ресурс] Режим доступа: https://docs.ansible.com/
6. SNIA Swordfish v1.2.5a. [Электронный ресурс] Режим доступа: https://www.snia.org/sites/default/files/technical-work/swordfish/release/v1.2.5a/html/Specification/Swordfish_v1.2.5a_Specification.html#volume1.10.0
7. Инкрементальный подход к технологии создания тестов для индустриальных проектов / П. Д. Дробинцев, В. П. Котляров, И. В. Никифоров, А. А. Летичевский // Моделирование и анализ информационных систем. – 2014. – Т. 21, № 6. – С. 144-154. – EDN TCCAKB.
8. Разработанная Ansible-коллекция для спецификации Swordfish. [Электронный ресурс] Режим доступа: https://gitlab.com/IgorNikiforov/swordfish_ansible_plugin

УДК 004.4

В. А. Ромашов, Л. И. Клочкова (4 курс бакалавриата),
И. В. Никифоров, к.т.н., доцент

ПРОЕКТИРОВАНИЕ И РЕАЛИЗАЦИЯ ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА ДЛЯ ИНЖЕНЕРНОГО ПОРТАЛА

Объем информации в интернете растет с каждым годом, и, по прогнозам компании IDC, к 2025-ому году он составит 175 зеттабайт [1]. Такой рост обусловлен тем, что данные становятся самым ценным активом для любой организации [2], и потребность в них возникает во многих сферах: бизнес-аналитика, маркетинг, продажи, разработка инновационных продуктов и решений. Информация может храниться в Confluence, электронных письмах, чатах, каналах общения, форумах [3][4]. И это несет проблему быстрого доступа к определенным знаниям. Соответственно, тратится время на поиск ресурса, поиск внутри ресурса и фильтрацию найденной информации – на это может уходить не только часы, но и дни поиска. Эту проблему призвана решить база знаний.

База знаний – это онлайн хранилище информации о продукте, командах, услугах, проектах, документации [5]. Крупные организации используют БЗ по многочисленным причинам. Например, общий доступ к знаниям упрощает: устранение ошибок/неполадок, обучение новых специалистов, решение организационных вопросов.

Целью работы является повышение производительности работы продуктовой команды за счет подходов обмена информацией, которые реализованы в инженерном портале, отличительной особенностью которого является автоматическая интеграция и обновление данных.

Портал должен стать заменой используемых баз знаний и единой точкой входа для получения информации инженерами компании.

Веб интерфейс портала является простым, понятным и удобным инструментом для поиска и навигации по предоставленной информации. Информация должна предоставляться продуктовыми командами, которые хотят структурировать ее и предоставить внешним инженерам компании.

Для портала можно выделить следующие основные функциональные требования:

- портал должен предоставлять результаты обработки аналитических задач в графическом виде;

- предоставление актуальных статусов внедрения программного обеспечения в продукты компании;

- отображение результатов ночных тестирований продуктов команд;

- отображение документации по продукту.

Предлагаемая архитектура портала изображена на рисунке 1.

Веб интерфейс представляет из себя SPA [6] с применением паттерна разработки MVVM [7]. Такое разделение позволяет ускорить разработку и поддерживаемость интерфейса — можно менять один компонент, не затрагивая код другого.

Данные получаются посредством REST [8] запросов к различным серверам, приложение должно уметь получать данные из различных источников, за это отвечает слой с API. Хранение данных реализовано на хранилище Zustand и кэшировании запросов с помощью библиотеки TanStack Query. Это помогает порталу работать быстрее и не отправлять лишние запросы на сервера.

Далее идет слой реализации логики, который в себе содержит основные компоненты. Данные компоненты должны реализовывать всю логику отображения информации в интерфейсе. Бизнес логика реализуется посредством библиотеки react-router-dom.

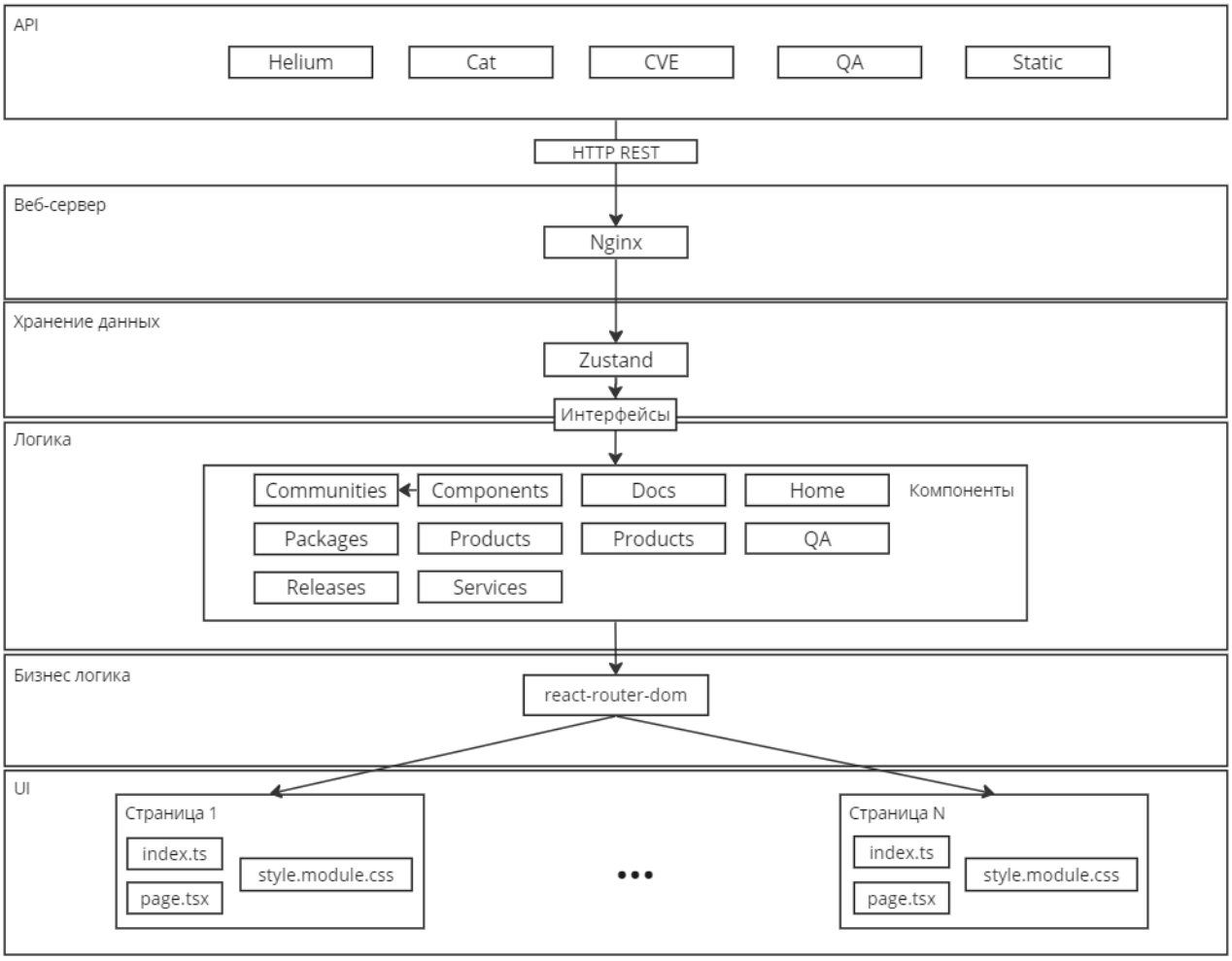


Рисунок 1 – Архитектура интерфейса инженерного портала

Уровень отображения данных - на нем собираются страницы в единую сущность и отображают информацию. Содержат в себе файлы стилей и кода, написанного на языке TypeScript, используемого для статической типизации и проверки типов.

Благодаря такой архитектуре приложение легко масштабируется под новые требования и задачи со стороны инженеров компании.

Таким образом, в работе был разработан и спроектирован пользовательский интерфейс для инженерного портала. Предложенное решение инженерного портала должно стать единой

точкой доступа для получения актуальных, точных сведений. Должно ускорить поиск и фильтрацию найденной информации.

Разработка портала выполняется при поддержке компании YADRO.

ЛИТЕРАТУРА

1. Reinsel D., Gantz J., Rydning J. The Digitization of the World. From Edge to Core, An IDC White Paper, #US44413318, November 2018.
2. Nielsen O.B. A comprehensive review of data governance literature", Selected Papers of the IRIS, Issue Nr 8, 2017, pp. 120-133.
3. Ковалев А. Д. Автоматизированный подход к семантическому поиску по программной документации на основе алгоритма Doc2Vec / А. Д. Ковалев, И. В. Никифоров, П. Д. Дробинцев // Информационно-управляющие системы. – 2021. – № 1(110). – С. 17-27. – DOI 10.31799/1684-8853-2021-1-17-27. – EDN SBRUMH.
4. 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. – EDN CFRNSJ.
5. База знаний: ваше решение для улучшения совместной работы // Atlassian. Режим доступа: https://www.atlassian.com/ru/itsm/knowledge-management/what-is-a-knowledge-base (дата обращения: 03.03.24).
6. Чернова С. В., Рассеева Е.В. Принцип работы Single Page Application // Актуальные проблемы информатики, радиотехники и связи. Материалы XXX Российской научно-технической конференции, Самара, 28 февраля – 03 марта 2023 года. – Самара: Поволжский государственный университет телекоммуникаций и информатики, 2023. – С. 191-192.
7. Гатин Р.Р. Применение паттерна проектирования MVVM в разработке современных программных решений // Опыт и проблемы реформирования системы менеджмента на современном предприятии: тактика и стратегия. Сборник статей XXII Международной научно-практической конференции, Пенза, 20-21 марта, 2023 года. – Пенза: Пензенский государственный аграрный университет, 2023 – С. 95-99.
8. Иванов Г. В., Власов Д. В., Околович Л. Д., Никифоров И. В. Проектирование rest-интерфейса системы транскодирования видеопотоков в режиме реального времени // Современные технологии в теории и практике программирования: Сборник материалов научно-практической конференции, Санкт-Петербург, 22 апреля 2021 года. – Санкт-Петербург: Федеральное государственное автономное образовательное учреждение высшего образования "Санкт-Петербургский политехнический университет Петра Великого", 2021. – С. 209-211.

УДК 004.453

С. Ф. Солодовников (4 курс бакалавриата),
Г. А. Худяков (3 курс бакалавриата),
И. В. Никифоров, к.т.н., доцент,
О. В. Александрова, ст. преподаватель

РАЗРАБОТКА ANSIBLE-МОДУЛЕЙ ПО СПЕЦИФИКАЦИИ SWORDFISH ДЛЯ УПРАВЛЕНИЯ КОНФИГУРАЦИЕЙ СХД

Введение: Существующие системы хранения данных в большинстве случаев предоставляют REST API для того, чтобы можно было ими управлять [1]. В зависимости от производителя предоставляемый REST API может отличаться, что влечет за собой многообразие различных протоколов, команд и способов взаимодействия. Очевидно, что для потребителей, у которых установлены различные СХД, возникает потребность одновременной поддержки различных протоколов, что является трудоемким как с точки зрения разработки, так и поддержки (рисунок 1).

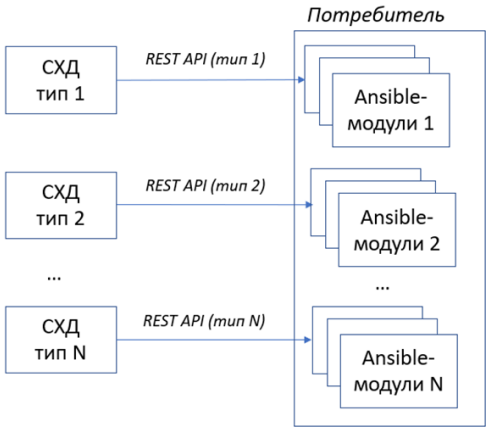


Рисунок 1 – Многообразие поддерживаемых модулей на стороне потребителя.

Для минимизации количества разнообразных способов взаимодействия некоммерческой организацией SNIA (Storage Networking Industry Association) была разработана унифицированная спецификация Swordfish [2,3] на REST API, которой должны удовлетворять различные системы хранения данных (СХД). При этом решение о том, что конкретное СХД будет удовлетворять спецификации принимается производителем данной СХД (рисунок 2).

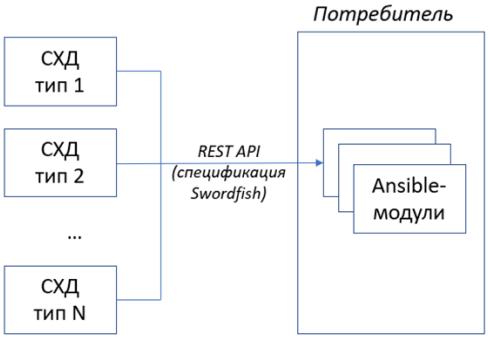


Рисунок 2 – Унифицированный REST API для СХД на основе Swordfish спецификации.

Для спецификации Swordfish нет playbook [4] и также нет Ansible-модулей [5]. Этот недостаток влечет за собой невозможность быстрого и эффективного управления СХД, удовлетворяющего спецификации Swordfish, и как следствие каждому производителю СХД требуется самостоятельно реализовывать необходимые команды.

Целью работы является разработка Ansible-модулей, которые могут использоваться в playbook, чтобы осуществлять управление СХД, которые удовлетворяют спецификации Swordfish.

Ход разработки

1. Изучить спецификацию Swordfish (Storage Managment API - v1.2.5).
2. Изучить инструмент Ansible [6].
3. Изучить подходы разработки Ansible palybook.
4. Изучить подходы разработки Ansible-модулей [7].
5. Разработать Ansbile-модули для спецификации Swordfish.
6. Разработанные модули необходимо проверить с использованием модульных тестов, автоматизация запуска которых подключена к CI.
7. Разработать интеграционные тесты, которые могут использовать Mock-сервер [8]. Запуск и автоматизация интеграционных текстов должна быть подключена к CI.
8. Разработанные модули необходимо разместить в opensource в виде библиотеки (открытого репозитория), который может использоваться другими (внешними) разработчиками СХД [9].
9. Разработать документацию.

На текущий момент силами команды студентов СПбПУ были реализованы восемь основных модулей для управления ресурсами СХД в соответствии с требованиями

спецификации Swordfish: хранилище и системы управления (storage_pools_info, storage_pool_volume), управление томами (volume), управление дисками (sp_drives_info), конфигурация сетевых протоколов (ntp), управление доступом пользователей (sp_user), управление логическим пространством имен (manage_attached_namespaces, namespace_capacity_info). Также реализованы модули для управления состоянием СХД (ping, restart). Эти модули позволяют осуществлять комплексное взаимодействие с СХД, начиная от инициализации и заканчивая более сложными операциями управления.

Дополнительно были разработаны два модуля для сценариев использования (storage_pool, namespace_can_be_created), которые демонстрируют практическое применение реализованных модулей в типичных задачах управления СХД. Эти модули служат примерами для разработчиков и администраторов систем, облегчая интеграцию и использование новых инструментов в существующих рабочих процессах.

Особое внимание было уделено фазе тестирования [10]. Разработан набор юнит и интеграционных тестов для обеспечения высокого качества, и надежности модулей, что позволяет гарантировать корректную работу модулей в заданных условиях. Также проведена настройка системы CI/CD для автоматизации тестирования при каждом обновлении кода, обеспечивая постоянный контроль за качеством разработки и упрощая процесс внедрения изменений.

Выводы: Демонстрационная версия разработанных Ansible-модулей для спецификации Swordfish представляет собой мощный инструмент для автоматизации и унификации управления конфигурацией СХД. Она демонстрирует впечатляющие результаты в ускорении процессов настройки и обслуживания, обеспечении надежности и стабильности работы систем, а также в гибкости и удобстве использования. Открытый исходный код и поддержка сообщества обеспечивают постоянное развитие и совершенствование данного решения.

ЛИТЕРАТУРА

1. Ковалев, А. Д. Разработка распределенной системы архивации данных на основе стандарта OAIS с использованием технологии Apache Hadoop / А. Д. Ковалев, И. В. Никифоров, В. П. Котляров // 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. – EDN WIBCLF.
2. Спецификация Swordfish. Электронный ресурс [Режим доступа]: https://www.snia.org/sites/default/files/technical-work/swordfish/release/v1.2.5a/html/Specification/Swordfish_v1.2.5a_Specification.html (Дата обращения: 23.06.2023).
3. Документация Swordfish. Электронный ресурс [Режим доступа]: https://www.snia.org/forums/smi/swordfish/ (Дата обращения: 23.06.2023).
4. Ansible плейбук. Электронный ресурс [Режим доступа]: https://docs.ansible.com/ansible/latest/playbook_guide/playbooks_intro.html (Дата обращения: 30.11.2023).
5. Ansible модуль. Электронный ресурс [Режим доступа]: https://docs.ansible.com/ansible/latest/module_plugin_guide/modules_intro.html (Дата обращения: 30.11.2023).
6. Документация на Ansible. Электронный ресурс [Режим доступа]: https://docs.ansible.com/ (Дата обращения: 23.06.2023).
7. Подходы к разработке Ansible модулей. Электронный ресурс [Режим доступа]: https://docs.ansible.com/ansible/latest/dev_guide/developing_modules_checklist.html#developing-modules-checklist (Дата обращения: 30.11.2023).
8. Описание Mock-интерфейса. Электронный ресурс [Режим доступа]: http://swordfishmockups.com/ (Дата обращения: 23.06.2023).
9. Репозиторий с Ansible-модулями для спецификации Swordfish. Электронный ресурс [Режим доступа]: https://gitlab.com/IgorNikiforov/swordfish_ansible_plugin

10. Инкрементальный подход к технологии создания тестов для индустриальных проектов / П. Д. Дробинцев, В. П. Котляров, И. В. Никифоров, А. А. Летичевский // Моделирование и анализ информационных систем. – 2014. – Т. 21, № 6. – С. 144-154. – EDN TCCAKB.

УДК 004.9

М. В. Тянутов, Б. А. Денисенко (1 курс магистратуры),
И. В. Никифоров, к.т.н., доцент

АВТОМАТИЗАЦИЯ НАСТРОЙКИ ОТЕЧЕСТВЕННЫХ СХД С ИСПОЛЬЗОВАНИЕМ ANSIBLE-МОДУЛЕЙ

В мире современных информационных технологий хранение данных играет ключевую роль. Сложные системы хранения данных (СХД) представляют собой множество компонентов, среди которых особое место занимает дисковый носитель информации. Постановка таких систем заказчику включает в себя монтаж и первичную настройку [1,2].

Ручная первичная настройка СХД является сложным и время затратным процессом, который подвержен человеческим ошибкам. Повторение этой настройки для разных заказчиков также требует значительных усилий. Поэтому автоматизация настройки СХД становится актуальной задачей.

Цель данной работы заключается в снижении трудоемкости настройки подключения хостов к системам хранения данных с помощью использования Ansible-модулей, включенных в Ansible-playbook[3, 4].

Предлагаемый подход к автоматизации настройки СХД включает в себя следующие шаги, которые необходимо реализовать в Ansible-playbook:

1. *Подготовка окружения.* Этот этап играет решающую роль в обеспечении гладкого процесса настройки. Здесь происходит установка необходимых зависимостей, создание конфигурационных файлов и установка прав доступа. Ansible-playbook позволяет автоматизировать эти шаги, что значительно сокращает риск ошибок и время, затрачиваемое на подготовку.
2. *Настройка драйвера Cinder.* В данном сценарии СХД работает в среде openstack, а потому, необходимо отдельные настройки для модуля Cinder. Добавление драйвера Cinder требует специальной конфигурации для обеспечения правильной работы системы хранения данных.
3. *Конфигурация дисковых массивов.* Определение объемов дисков и их связывание в логические массивы — это важный этап, который обеспечивает эффективное использование пространства и повышает отказоустойчивость СХД. Ansible-playbook позволяет автоматизировать этот процесс, что делает его более надежным и масштабируемым [5].
4. *Настройка доступа и безопасности.* Этот шаг включает в себя настройку прав доступа к данным, аутентификацию пользователей и шифрование данных. Использование Ansible-playbook обеспечивает консистентность в настройке доступа и безопасности, что является критически важным для обеспечения безопасности и конфиденциальности информации.
5. *Тестирование и проверка работоспособности*. В завершающем этапе производится тестирование настроек и функциональности СХД для гарантированного обеспечения их правильной работы. Автоматизация этого процесса с помощью Ansible-playbook позволяет быстро и эффективно выявить любые проблемы и устранить их [6].

Важным аспектом при автоматизации настройки систем хранения данных (СХД) является обеспечение идемпотентности - возможности многократного запуска Ansible-playbook без изменения состояния системы. Это позволяет избежать нежелательных побочных эффектов и гарантирует стабильность работы инфраструктуры [7].

Специально разработанный модуль Ansible для данного типа СХД играет ключевую роль в обеспечении идемпотентности. Он позволяет устанавливать необходимые настройки и проводить изменения в конфигурационных файлах системы с учетом текущего состояния, избегая дублирования или некорректных изменений.

Однако, когда речь идет о настройке драйвера Cinder, приходится самостоятельно достигать идемпотентности, адаптируя и настраивая конфигурационные файлы Cinder под требования системы хранения данных [8].

Таким образом, применение указанного подхода и соответствующих Ansible-модулей в Ansible-playbook позволяет существенно сократить время и усилия, затрачиваемые на настройку СХД, и уменьшить вероятность ошибок, что в свою очередь способствует повышению эффективности и надежности систем хранения данных.

ЛИТЕРАТУРА:

1. Ковалев, А. Д. Разработка распределенной системы архивации данных на основе стандарта OAIS с использованием технологии Apache Hadoop / А. Д. Ковалев, И. В. Никифоров, В. П. Котляров // 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. – EDN WIBCLF.
2. 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. – EDN XXPLRR.
3. Sesto, V. (2022). Configuration Management with Ansible. In: Practical Ansible. Apress, Berkeley, CA. https://doi.org/10.1007/978-1-4842-8643-2_1
4. Ansible documentation [Электронный ресурс] Режим доступа: https://docs.ansible.com/ansible/latest/
5. Mallett, A. (2021). Configuring Storage with Ansible. In: Red Hat Certified Engineer (RHCE) Study Guide. Apress, Berkeley, CA. https://doi.org/10.1007/978-1-4842-6861-2_14
6. 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. – EDN VDVCYX.
7. Инкрементальный подход к технологии создания тестов для индустриальных проектов / П. Д. Дробинцев, В. П. Котляров, И. В. Никифоров, А. А. Летичевский // Моделирование и анализ информационных систем. – 2014. – Т. 21, № 6. – С. 144-154. – EDN TCCAKB.
8. Automation of Server Configuration Using Ansible" in “International Journal for Research”, Applied Science & Engineering Technology (IJRASET), 2022.

УДК 004.453

В. М. Юлдашев (3 курс бакалавриата),
И. В. Никифоров, к.т.н., доцент,
О. В. Александрова, ст. преподаватель

ОСОБЕННОСТИ И ПРОБЛЕМЫ MOCK-СЕРВЕРА SWORDFISH API EMULATOR

Введение: Swordfish API Emulator — это веб-приложение для эмуляции системы, которая использует RESTful-операции, удовлетворяющие спецификации Swordfish. Он необходим для разработки программного обеспечения, которое взаимодействует с серверами СХД в соответствии со спецификацией. Swordfish API Emulator расширяет функционал Redfish

Interface Emulator, добавляя возможности взаимодействия со всеми схемами данных из Swordfish API.

Разберем особенности и проблемы mock-сервера Swordfish API Emulator, с которыми мы столкнулись, а также приведем пути, которые позволили разрешить ситуацию

Запутанная иерархия файлов эмулятора

В главной директории исходных кодов эмулятора лежат всевозможные файлы и папки: как необходимые для работы эмулятора, так и, в большинстве случаев, ненужные. Например, Swordfish API Emulator использует только один файл конфигурации — emulator-config.json. Остальные конфигурационные файлы остались от Redfish Interface Emulator. Одной из причин неструктурированности кода является автоматическая генерация кода для API ресурсов по XML-схеме.

Как исправить проблему: Необходима реструктуризация проекта. Для улучшения иерархии эмулятора необходимо изменить настройки генератора, а также требуется измененить архитектуру кода. Реструктуризация кода в будущем даст большую гибкость при обновлении функциональности эмулятора Swordfish.

Отсутствие некоторых API

Самый простой пример отсутствующих API — это Actions. У ресурсов есть отдельное поле Actions, в котором перечислены действия, которые можно совершить с помощью них. Например, перезапустить эмулятор. При попытке вызвать необходимый Action мы получим ошибку, так как необходимый URL не назначен в resource_manager.py — необходимой функциональности просто не существует.

Как исправить проблему: В качестве решения проблемы можно провести масштабное тестирование функциональности эмулятора в соответствии со спецификацией. Найти и добавить отсутствующий API.

Зависимость от Redfish Interface Emulator

Проблема зависимости Swordfish API Emulator от Redfish Interface Emulator проявляется на этапе установки эмулятора. Во время установки происходит клонирование репозитория эмулятора от DMTF. После удаляются ненужные файлы и добавляются схемы Swordfish. Таким образом Swordfish API Emulator зависит от команды разработчиков, которые не имеют отношение к SNIA. Если исходный код эмулятора Redfish будет подвержен некоторым серьезным изменениям, то Swordfish API Emulator может перестать работать корректно.

Как исправить проблему: Тут потенциальным решением может стать форк репозитория Swordfish API Emulator для дальнейшего редактирования эмулятора с целью убрать зависимость от Redfish Interface Emulator. В нашем случае было решено разработать свой эмулятор на языке Go. Этот язык мы выбрали не случайно: он прост в использовании и идеально подходит для написания клиент-серверных приложений, которым и является эмулятор СХД. Это уберет зависимость от стороннего кода и даст больше возможностей для реализации недостающей функциональности.

Неудобная архитектура кода и неработающая функциональность

Эмулятор имеет неудобную архитектуру кода, из-за чего его тяжело модифицировать. Например, при попытке добавить Actions на перезапуск СХД, мы столкнулись с проблемой циклических зависимостей. Проблема была в том, что в emulator.py происходит импорт ResourceManager, в котором происходит импорт нашего Action.Reset, в котором происходит импорт функции перезапуска из emulator.py.

Как исправить проблему: Необходимо произвести переработку архитектуры эмулятора, исправить неработающие функции и добавить новые. Например, изменение состояний работы дисков и эмуляцию их поломки.

Нереализованная система ролей

Подразумевается, что для работы с разными типами данных и ресурсами на уровне СХД — например, со Storage Pools или Volumes — важно определить права пользователей, в

пределах которых они могли бы взаимодействовать с системой. Ключом к решению задачи служит распределение пользователей по определенным ролям.

В спецификации Swordfish выделяется три основные роли: DevOps, StorageAdmin и CloudAdmin. У каждой есть спектр выполняемых задач:

- Роль DevOps необходима, чтобы использовать конфигурации систем хранения данных для автоматизации и масштабирования бизнес-операций.
- Роль StorageAdmin предназначена для управления и решения оперативных задач хранения, таких как планирование жизненного цикла хранилища и повседневное администрирование хранилища.
- Роль CloudAdmin позволяет отвечать за управление всеми аспектами облачной инфраструктуры, включая хранилище.

Также некоторые операции могут выполнять пользователи с разными привилегиями. Например, присоединить существующее пространство имен могут пользователи с ролями CloudAdmin и StorageAdmin. Стоит отметить, что для управления пользователями и распределением ролей уже существует особый вид роли из Redfish — Administrator. В итоге у нас четыре роли, права которых нужно реализовать.

Теперь посмотрим, как это сделать на практике:

Реализация многоролевой модели в сервер-эмуляторе Swordfish

Для упрощения решения задачи ее можно разбить на три этапа:

1. Тестовая проверка ролей и пользователей на уровне эмулятора, то есть все роли, пользователи и пароли берутся из кода Swordfish-эмулятора.
2. Реализация более адекватной версии со считыванием данных из ресурсов эмулятора и их непосредственное использование.
3. Реализация соответствия разделения ролей спецификации Swordfish.

Также можно интегрировать новые специфические роли для решения задач в рамках определенной СХД, а также ввести более совершенные методы верификации пользователей. Также рекомендуется обезопасить аккаунт главного администратора для минимизации риска взломов аккаунтов и последующих несанкционированных действий. Это можно реализовать введением двухфакторной аутентификации или с помощью YubiKey.

Выводы: В статье были разобраны особенности и проблемы Swordfish API Emulator, а также были приведены методы решения проблем. В будущем планируем исправить проблемы, выявленные во время эксплуатации эмулятора в полном объеме, и перенести его код на язык программирования Go, что позволит избавиться от зависимостей от Redfish Interface Emulator и даст больше возможностей для реализации недостающей функциональности.

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. – EDN XXPLRR.
2. 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. – EDN VDVCYX.
3. Ahlvers R. Swordfish Overview: Extending Data Center Control to the World of Storage //Companion Proceedings of the10th International Conference on Utility and Cloud Computing. – 2017. – С. 239-239.
4. Hochstein L., Moser R. Ansible: Up and Running: Automating configuration management and deployment the easy way. – " O'Reilly Media, Inc.", 2017.
5. Ansible Documentation. [Электронный ресурс] Режим доступа: https://docs.ansible.com/
6. SNIA Swordfish v1.2.5a. [Электронный ресурс] Режим доступа: https://www.snia.org/sites/default/files/technical-

work/swordfish/release/v1.2.5a/html/Specification/Swordfish_v1.2.5a_Specification.html#volume1.10.0
7. Особенности и проблемы mock-сервера Swordfish API Emulator / SPbPU Software Engineering Lab @polytech_se_lab / https://habr.com/ru/companies/yadro/articles/795211/
8. Пишем Ansible-модули для управления разными системами хранения данных через Swordfish / SPbPU Software Engineering Lab @polytech_se_lab / https://habr.com/ru/companies/yadro/articles/784070/

УДК 004.4`2

Д. А. Фоломкин (3 курс бакалавриата),
И. В. Никифоров, к.т.н., доцент,
О. В. Александрова, ст. преподаватель,
Л. П. Котлярова, ст. преподаватель

РЕАЛИЗАЦИЯ МОДУЛЯ УПРАВЛЕНИЯ ДОСТУПОМ НА ОСНОВЕ РОЛЕЙ ДЛЯ СЕРВЕР-ЭМУЛЯТОРА SWORDFISH

С развитием систем хранения данных (СХД) неизбежно увеличивается и их архитектурная сложность, что, в свою очередь, стимулирует появление новых стандартов для обеспечения эффективного управления и масштабируемости этих систем. Один из таких стандартов был разработан компанией SNIA (Storage Networking Industry Association) и получил название Swordfish [1]. Идея данной унифицированной спецификации - минимизировать количество разнообразных способов взаимодействия потребителей с СХД через REST API. К сожалению, для спецификации Swordfish отсутствуют Ansible-модули, которые позволяют автоматизировать различные задачи в системах управления конфигурацией. По этой причине лаборатория СПбПУ совместно с компанией Yadro взялась за разработку Ansible-модулей [2] по спецификации Swordfish [3].

В процессе разработки наша команда встретила целый ряд трудностей на уровне сервера-эмулятора, который должен был стать испытательным полигоном для создаваемых модулей. Дело в том, что спецификация Swordfish значительно расширяет количество схем взаимодействия с элементами хранилищ данных в сравнении со своим предком - спецификацией Redfish [4]. По этой причине Swordfish User Guide [5] определил три широкие функциональные роли и области их применения: DevOps, StorageAdmin и CloudAdmin. Помимо них, из спецификации Redfish интегрируются роли Administrator и ReadOnly [6]. Отсюда возникает следующая проблема: отсутствие разбиения доступа к ресурсам и функциям сервера-эмулятора с помощью вышеописанных ролей. Поэтому командой было принято решение реализовать RBAC в качестве оптимального и удачного подхода к решению проблемы. RBAC (Role-Based Access Control) — это метод управления доступом к ресурсам в системе, который определяет доступ к ресурсам на основе ролей, а не отдельных пользователей. Такой подход позволяет более гибко и эффективно управлять доступом к ресурсам, особенно в больших и сложных системах, например, СХД [7].

Для упрощения решения поставленной задачи реализация была разбита на 3 этапа:

– в ходе первого этапа происходит тестовая проверка ролей и пользователей на уровне эмулятора, то есть все роли, пользователи и пароли берутся из кода Swordfish-эмулятора;

– во втором этапе реализуется более адекватная версия со считыванием данных из ресурсов эмулятора и их непосредственное использование;

– на третьем этапе реализуется соответствие разделения ролей спецификации Swordfish.

В конечном итоге в качестве результата ожидается полностью функционирующая система пользователь-роль, где обязанности каждого участника будут распределены. По сути, должна получиться тот самый метод RBAC, о которой говорилось ранее.

В качестве дальнейшего развития возможна интеграция новых специфических ролей для решения задач в рамках определенной СХД, а также введение более совершенных методов

верификации пользователей. Также требуется обезопасить аккаунт главного администратора для минимизации риска взломов аккаунтов и последующих несанкционированных действий со стороны взломщиков [8]. Этого можно добиться внедрением двухфакторной аутентификации или путем использования YubiKey.

Подводя итог, можно сказать следующее: ролевая модель - важная и необходимая структура для сервера-эмулятора Swordfish и будущих СХД, где эта модель будет применяться [9]. Можно выделить следующие плюсы такого подхода:

– каждый пользователь четко знает область своей работы;

– удобный менеджмент аккаунтами для администратора;

– возможность изменить пользователю роль, тем самым изменив спектр выполняемых им задач.

Задача по прототипированию и интеграции ролевой модели была успешно выполнена на уровне сервера-эмулятора Swordfish. В силу несовершенства архитектуры было принято решение полностью отказаться от этого эмулятора. По этой причине в полной мере RBAC-метод будет реализован в разрабатываемом лабораторией СПбПУ сервере-эмуляторе, написанном на языке программирования Go.

Проект выполняется при поддержке компании Yadro.

ЛИТЕРАТУРА

1. Документация Swordfish. Электронный ресурс [Режим доступа]: https://www.snia.org/forums/smi/swordfish (Дата обращения: 23.06.2023).
2. Репозиторий с Ansible-модулями для спецификации Swordfish. Электронный ресурс [Режим доступа]: https://gitlab.com/IgorNikiforov/swordfish_ansible_plugin
3. Спецификация Swordfish. Электронный ресурс [Режим доступа]: https://www.snia.org/sites/default/files/technical-work/swordfish/release/v1.2.5a/html/Specification/Swordfish_v1.2.5a_Specification.html (Дата обращения: 23.06.2023).
4. Документация Redfish. Электронный ресурс [Режим доступа]: https://www.dmtf.org/standards/redfish (Дата обращения: 30.11.2023).
5. Swordfish User Guide. Электронный ресурс [Режим доступа]: https://www.snia.org/sites/default/files/technical-work/swordfish/release/v1.2.5a/html/UserGuide/Swordfish_v1.2.5a_UserGuide.html (Дата обращения: 23.06.2023).
6. Спецификация Redfish. Электронный ресурс [Режим доступа]: https://www.dmtf.org/sites/default/files/standards/documents/DSP0266_1.20.0.html (Дата обращения: 30.11.2023).
7. 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. – EDN XXPLRR.
8. A system prototype for real time automatic fraud detection in text data / N. Voinov, I. Nikiforov, P. Drobintsev, V. Kotlyarov // Исследования по геоинформатике: труды Геофизического центра РАН. – 2017. – Vol. 5, No. 1. – P. 84. – DOI 10.2205/CODATA2017. – EDN ZXAUEH.
9. Особенности и проблемы mock-сервера Swordfish API Emulator. Электронный ресурс [Режим доступа]: https://habr.com/ru/companies/yadro/articles/795211/ (Дата обращения: 28.02.2024).

СОДЕРЖАНИЕ

СОВРЕМЕННЫЕ ТЕХНОЛОГИИ В ТЕОРИИ И ПРАКТИКЕ ПРОГРАММИРОВАНИЯ

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

24–25 апреля 2024 года

Налоговая льгота – Общероссийский классификатор продукции
ОК 005-93, т. 2; 95 3004 – научная и производственная литература

Подписано в печать 24.04.2024. Формат 60×84/16. Печать цифровая.
Усл. печ. л. 19,5. Тираж 200. Заказ 1978.

Отпечатано с готового оригинал-макета,
предоставленного редакционной коллегией,
в Издательско-полиграфическом центре Политехнического университета.
195251, Санкт-Петербург, Политехническая ул., 29.
Тел.: (812) 552-77-17; 550-40-14.