



eISSN 2311-2468
Том 8, № 13. 2020
Vol. 8, no. 13. 2020

электронное периодическое издание
для студентов и аспирантов

Огарёв-онлайн Ogarev-online

<https://journal.mrsu.ru>



ПЛОДУХИН Д. М.

РЕАЛИЗАЦИЯ МОДЕЛИ АВТОМАТИЗИРОВАННОГО ТЕСТИРОВАНИЯ

Аннотация. В статье приведено описание концепции модели автоматизированного тестирования и ее реализации. Описаны технологии, необходимые в разработке, и продемонстрированы примеры работы системы.

Ключевые слова: автоматизированное тестирование, веб-приложение, Kotlin, Selenium, Selenide, REST Assured, JDBC, Allure.

PLODUKHIN D. M.

IMPLEMENTATION OF AUTOMATED TESTING MODEL

Abstract. The article presents the concept model of automated testing and its implementation. The technologies necessary for development are described and examples of the system are demonstrated.

Keywords: automated testing, web application, Kotlin, Selenium, Selenide, REST Assured, JDBC, Allure.

Основная концепция модели автоматизированного тестирования заключается в том, что за многие годы практики разработки в тестировании был накоплен определенный опыт и набор требований к тестовым фреймворкам, которые используются при написании автотестов для того или иного веб-приложения. Рассматривать каждое приложение по очевидным причинам не представляется возможным, поэтому в данной модели собраны общие тенденции и практики, которые универсальны и переносимы на любой процесс автоматизированного тестирования [1; 2].

Тестовая модель представляет собой систему для тестирования «идеального», моделируемого веб-приложения, разработанного по современным требованиям и стандартам. Следует рассмотреть веб-приложение по трем основным его составляющим:

1. Клиентская часть веб-приложения (Front-end) – это графический пользовательский интерфейс. То, что видит пользователи на странице браузера. Пользователь взаимодействует с веб-приложением именно через эту часть, нажимая на различные элементы управления. К этой части относится вся логика, отвечающая за визуализацию контента.

2. Серверная часть веб-приложения (Back-end – иногда этим термином называют всю совокупность программно-аппаратной части сервиса) – это скрипт или программное обеспечение, которые располагаются на сервере и обрабатывают запросы пользователя (корректнее, запросы браузера).

3. База данных (БД) – программное обеспечение на сервере, которое хранит данные и выдает их в нужный момент. Большинство современных веб-приложений работают реляционными СУБД, но постепенно занимают свое место на рынке и нереляционные БД (NoSQL).

В соответствии с этими требованиями был сформирован технологический стек, необходимый для разработки.

1. **Kotlin** (Кóтлин) – статически типизированный язык программирования, запускаемый поверх Java Virtual Machine и разрабатываемый компанией JetBrains. Это гибкий и лаконичный язык, заточенный под функциональное программирование и полностью совместимый с Java.

2. **Gradle** – система автоматической сборки, построенная на принципах Apache Ant и Apache Maven, но предоставляющая DSL на языках Groovy и Kotlin вместо традиционной XML-образной формы представления конфигурации проекта. Gradle идеально подходит для инкрементальных сборок и очень удобен.

3. **Selenium WebDriver** – программное обеспечение для автоматизации манипуляций с веб-браузером. Чаще всего используется для тестирования Web-приложений, но не ограничивается этим. Практически бескомпромиссное решение в работе с браузерами.

4. **Selenide** – библиотека для написания лаконичных и стабильных UI тестов с открытым исходным кодом. Selenide – это обёртка вокруг Selenium WebDriver, позволяющая быстро и просто его использовать при написании тестов. Для примера, не нужно акцентировать внимание на работе с ожиданиями элементов в процессе автоматизации тестирования динамических веб-приложений, а также на реализации высокоуровневых действий над элементами.

5. **TestNG/JUnit** – являются наиболее популярными средами модульного тестирования на текущий момент для Java/Kotlin. Выбор между ними не стоит так остро, поэтому имеет смысл поддержки разработки под обе среды.

6. **JDBC** (англ. Java DataBase Connectivity соединение с базами данных на Java) – платформенно независимый стандарт взаимодействия Java-приложений с различными СУБД, сделанный в виде пакета `java.sql`, входящего в состав Java SE. Данный стандарт – классическое решение для разработки на Java или Kotlin. Всё, что потребуется во время разработки, это написание удобной и понятной обертки для методов библиотек JDBC.

7. **REST Assured** – DSL для тестирования REST-сервисов, который встраивается в тесты на Java/Kotlin. Это решение существует уже более десяти лет и остается популярным из-за своей простоты и удобной функциональности.

8. **Allure** – инструмент, позволяющий создавать удобные и понятные отчеты, чтобы внести прозрачность в процесс создания и выполнения функциональных тестов. Allure помогает разработчикам и бизнесу более активно взаимодействовать, благодаря отображению результата тестирования в понятной форме для нетехнических специалистов.

Продукт, основанный на этой модели, уже применяется на некоторых коммерческих проектах, поэтому возможна лишь демонстрация существующей функциональности, с ограничением обзора исходного кода.

При тестировании веб-интерфейсов, в данном проекте поддерживается использование паттерна Page Object Model. Page Object моделирует страницы тестируемого приложения в виде объектов в коде. Основное преимущество Page Object заключается в том, что в случае изменения разработчиками приложения интерфейса, можно сделать исправление только в одном месте, а не править каждый тест, в котором этот интерфейс тестируется.

Реализован POM в данной работе в виде заведомо статических объектов, что допустимо благодаря “ленивой” инициализации элементов Selenide. Объявление Page Object и работа с ним выглядит следующим образом:

```
val pageObject by PageObject()
class PageObject : Element by css("css") {
    val element1 by className("class")
    val element2 by tag("tag")
    val subPageObject by SubPageObject()
}
class SubPageObject : Element by linkText("text") {
    val element3 by xpath("//xpath")
}
```

Ключевым моментом здесь является сохранение контекста найденного элемента. Любой последующий вложенный элемент будет найден от родительского.

Структура условий перешла из Selenide практически без изменений. Но благодаря ей было реализовано одно из самых частых применений данного объекта – поиск по условию. Скорее всего самый распространенный из них это поиск по тексту (использование условий в примере ниже равнозначно):

```
class SimpleTest {
    @Test
    fun simpleTest() {
        pageObject.element1[text("Some text 1")]
        pageObject.element1["Some text 1"]
    }
}
```

```
}
}
```

Определяя экземпляр объекта типа `Element`, будет находиться всегда первый элемент, удовлетворяющий заданному локатору и/или условию. Но иногда приходится работать и с коллекциями элементов.

В текущем случае был выбран вариант, объединяющий выражение `all` из чистого Kotlin и `ElementsCollection` из Selenide – `Elements`, поэтому с ним можно взаимодействовать как с обычным списком, так из коллекций из Selenide:

```
class SimpleTest {
    @Test
    fun simpleTest() {
        // ElementsCollection - проверка размера коллекции с помощью CollectionCondition
        pageObject.element1.all.shouldBe(CollectionCondition.size(5))
        // Аналогично, средствами Kotlin
        assertEquals(pageObject.element1.all.size, 5)
    }
}
```

Для обзора всей функциональности работы с веб-элементами нет необходимости выводить весь листинг доступных методов для работы с веб-элементами (многие из них являются просто оберткой для Selenide). Однако стоит выделить несколько изменений различной степени значимости:

1. **Определение методов через свойства.** Использование геттеров и сеттеров для лаконичности вызова.
2. **Часто используемые действия вынесены в отдельные методы.** Например: нажатие кнопки через Javascript, очистка поля клавиатурой.
3. **Встроенные проверки.** Например, при вызове любого метода, элемент проверяется на видимость без лишнего указания на это.

Касательно веб-навигации, разработчику придется работать с данными объектами, только если потребуется произвести какие-то особые проверки и действия с браузерами и вкладками. В противном случае, есть объект `window`, который является по сути текущей активной вкладкой в браузер (и создает ее, если она еще не открыта). Общий вывод по тестированию пользовательских интерфейсов с данными спецификациями легче оценить на примере лаконичного теста:

```
class SimpleTest {
    @Test
```

```

fun simpleTest() {
    "Открытие окна" {
        window.url = "http://mysite.com"
    }
    "Авторизация" {
        authorizationForm {
            login.value = "login"
            password.value = "password"
            submit.click
        }
    }
    "Проверка отображения формы профиля и корректного текста" {
        profileForm {
            shouldBe(text("Your profile"))
            list["List #1"].shouldBe(exactText("List #1: item text"))
        }
    }
}

```

Логика тестирования API состоит из обертки для хостов и запросов:

```

val localApplicationClient by lazy {
    ApiClient(
        url = "localhost:8080",
        desc = "Локальное приложение"
    )
}

internal fun postRequest(param1: String, param2: Int) = Request(
    desc = "[POST] POST-запрос с параметром",
    spec = {
        addHeaders(
            mutableMapOf(
                "header1" to "hd",
                "header2" to "dh"
            )
        )
    }
)

```

```

        setBody(
            """{
                "param1" : "$param1"
                "param2" : $param2
            }""".trimIndent()
        )
    },
    send = { post("/post") },
    verify = { HTTP_CREATED }
)

```

Сам процесс тестирования API выглядит следующим образом:

```

class SimpleTest {
    @Test
    fun simpleTest() {
        val result = localApplicationClient
            .send(postRequest("p1", 2))
            .jsonPath()
            .assertEqual(result.getString("body.res1"), "result")
    }
}

```

Тестирование и работа с БД проводится по аналогичному принципу:

```

val dbMSSQL by lazy {
    DbDriver.MsSql(
        "localhost:1234;databaseName=myBase;",
        "username",
        "password"
    )
}

fun dbPostgres(dataBaseName: String) = DbDriver.Postgres(
    "localhost:1234;databaseName=$dataBaseName;",
    "username",
    "password"
)

class SimpleTest {
    @Test

```

```

fun simpleTest() {
    val result = dbMSSQL
        .execute(
            DbQuery(
                "SELECT * FROM table",
                "Select к таблице"
            )
        )
    assertEquals(result.first().get("name"), "myname")
}
}

```

Естественно, вся указанная выше функциональность не является достаточной для построения полноценного жизненного цикла тестирования. Существует некоторый спектр улучшений, которые нельзя отнести к ранее описанным категориям во многом в рамках интеграции:

1. **Слушатель (listener) Allure и упрощение Allure Step.** Для автоматического формирования отчетов и удобного написания своих шагов без привязки к методу.
2. **Интеграция с системами отслеживания ошибок, системами управления проектами и тест-менеджерами.** Результаты тестирования автоматически будут отправляться в указанные системы.
3. **Запуск тестов.** Локальный и удаленный запуск будут конфигурироваться специальными параметрами запуска.
4. **Работа с прокси-фильтрами и почтовыми серверами.**

СПИСОК ЛИТЕРАТУРЫ

1. Русское руководство по языку Kotlin [Электронный ресурс]. – Режим доступа: <https://kotlinlang.ru> (дата обращения 18.05.2020).
2. Официальный сайт Selenide [Электронный ресурс]. – Режим доступа: <https://ru.selenium.org> (дата обращения 12.05.2020).

ШАМАНАЕВ П. А., КАРЧИГАНОВ А. Ф.

**ИССЛЕДОВАНИЕ ВЫНУЖДЕННЫХ КОЛЕБАНИЙ ОДНОЙ ЛИНЕЙНОЙ
СИСТЕМЫ ДВУХ СВЯЗАННЫХ ОСЦИЛЛЯТОРОВ С МАЛЫМ ПАРАМЕТРОМ**

Аннотация. В работе реализован алгоритм нахождения периодических решений одной линейной системы двух связанных осцилляторов с малым параметром на основе метода Ляпунова-Шмидта. Рассмотрены случаи, когда частота вынужденных колебаний совпадает с одной из частот собственных колебаний. Построены графики периодических решений и фазовых траекторий системы двух связанных осцилляторов.

Ключевые слова: система двух связанных осцилляторов, малый параметр, колебания, метод Ляпунова-Шмидта, резонанс.

SHAMANAEV P. A., KARCHIGANOV A. F.

**INVESTIGATION OF FORCED OSCILLATIONS OF A LINEAR SYSTEM
OF TWO COUPLED OSCILLATORS WITH A SMALL PARAMETER**

Abstract. The article is devoted to the description of the development of an algorithm for finding periodic solutions of one linear system of two coupled oscillators with a small parameter based on the Lyapunov-Schmidt method. The cases when the frequency of forced oscillations coincides with one of the natural oscillation frequencies are considered. Graphs of periodic solutions and phase trajectories of a system of two coupled oscillators have been constructed.

Keywords: system of two coupled oscillators, small parameter, oscillations, Lyapunov-Schmidt method, resonance.

Введение. Основные подходы метода Ляпунова-Шмидта нахождения решений нелинейных дифференциальных уравнений с малым параметром в банаховом пространстве изложены в работах [1; 2]. Для линейных дифференциальных уравнений с малым параметром в банаховом пространстве задача нахождения периодических решений методом Ляпунова-Шмидта рассматривалась в работах [3–5]. В работе [6] методом Ляпунова-Шмидта построено периодическое решение для одного класса линейных неоднородных систем обыкновенных дифференциальных уравнений с малым линейным возмущением. В настоящей работе на основании результатов работы [5] методом Ляпунова-Шмидта построено периодическое решение одной линейной системы двух связанных осцилляторов с малым параметром в резонансном случае.

Математическая модель системы двух связанных осцилляторов с малым параметром. Рассмотрим математическую модель вынужденных колебаний двух связанных осцилляторов [7] с малым параметром

$$\begin{aligned} m_1 \ddot{q}_1 + (c_{11} + \varepsilon d_{11})q_1 + (c_{12} + \varepsilon d_{12})q_2 &= F_1(t), \\ m_2 \ddot{q}_2 + (c_{12} + \varepsilon d_{12})q_1 + (c_{22} + \varepsilon d_{22})q_2 &= F_2(t), \end{aligned} \quad (1)$$

где q_1 и q_2 – обобщенные координаты системы двух связанных осцилляторов; ε – малый вещественный параметр; $F_1(t)$, $F_2(t)$ – внешние силы, изменяющиеся по закону

$$F_1(t) = r_1 \sin(\omega t + \theta_1), \quad F_2(t) = r_2 \sin(\omega t + \theta_2),$$

здесь $r_1 > 0$, $r_2 > 0$, $\theta_1, \theta_2, \omega \in R$.

Будем предполагать, что для параметров $m_i, c_{ij}, d_{ij} \in R, i \leq j, i, j = 1, 2$, системы (1) и достаточно малого ε справедливы неравенства

$$m_i > 0, c_{11} + \varepsilon d_{11} > 0, (c_{11} + \varepsilon d_{11})(c_{22} + \varepsilon d_{22}) - (c_{12} + \varepsilon d_{12})^2 > 0.$$

Выполняя замену координат

$$\begin{cases} x_1 = q_1 \\ x_3 = q_2 \end{cases} \Rightarrow \begin{cases} x_2 = \dot{x}_1 = \dot{q}_1 \\ x_4 = \dot{x}_3 = \dot{q}_2 \end{cases},$$

в системе (1), запишем ее в векторной форме

$$A \frac{dx}{dt} = (B_0 - \varepsilon B_1)x - f(t), \quad (2)$$

где $x = colon(x_1, x_2, x_3, x_4)$,

$$\begin{aligned} A &= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & m_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & m_2 \end{pmatrix}, & B_0 &= -\begin{pmatrix} 0 & -1 & 0 & 0 \\ c_{11} & 0 & c_{12} & 0 \\ 0 & 0 & 0 & -1 \\ c_{12} & 0 & c_{22} & 0 \end{pmatrix}, \\ B_1 &= \begin{pmatrix} 0 & 0 & 0 & 0 \\ d_{11} & 0 & d_{1,2} & 0 \\ 0 & 0 & 0 & 0 \\ d_{12} & 0 & d_{22} & 0 \end{pmatrix}, & f(t) &= -\begin{pmatrix} 0 \\ r_1 \sin(\omega t + \theta_1) \\ 0 \\ r_2 \sin(\omega t + \theta_2) \end{pmatrix}. \end{aligned}$$

На основании результатов работы [5] исследуем вынужденные колебания системы (2) в математическом пакете Maple при следующих значениях параметров

$$m_1 = m_2 = 1, \quad c_{11} = c_{22} = 5, \quad c_{12} = 4, \quad d_{11} = -2, \quad d_{22} = 2, \quad d_{12} = 0.$$

При выбранных значениях параметров матрица B_0 имеет следующие собственные значения

$$\lambda_{1,2} = \pm i, \quad \lambda_{3,4} = \pm 3i.$$

Тогда возможны два случая возникновения резонанса: 1) $\omega = 1$; 2) $\omega = 3$.
Следовательно, согласно [5], в случае 1) $T = 2\pi$, $\alpha_1 = 1$, $\alpha_2 = 3$, а в случае 2) $T = \frac{2\pi}{3}$, $\alpha_1 = 3$.

Ставится задача [1]: при достаточно малых вещественных ε найти T -периодические решения $x(t, \varepsilon)$ уравнения (2), удовлетворяющее условию $x(t, 0) = z(t)$, где $z(t)$ является T -периодическим решением уравнения

$$A \frac{dz}{dt} = B_0 z - f(t). \quad (3)$$

В дальнейшем систему (2) будем называть возмущенной системой, а систему (3) – невозмущенной системой.

Обобщенные жордановы наборы, удовлетворяющие условию биортогональности.

Элементы $\varphi_k^{(j)}$, $\psi_k^{(j)}$, $k = \overline{1, n}$, $j = \overline{1, p_k}$ обобщенных жордановых цепочек оператора

$$B_0 \equiv B_0 - A \frac{d}{dt}$$

и его сопряженного

$$B_0^* \equiv B_0^* + A^* \frac{d}{dt}$$

определяются по формулам [5]

$$\begin{aligned} B_0 \varphi_k^{(1)} &= 0, & B_0 \varphi_k^{(j)} &= B_1 \varphi_k^{(j-1)}, & j &= \overline{2, p_k}, \\ B_0^* \psi_s^{(1)} &= 0, & B_0^* \psi_s^{(l)} &= B_1^* \psi_s^{(l-1)}, & l &= \overline{2, p_s}, \\ & & k, s &= \overline{1, n}, \end{aligned} \quad (4)$$

где B_1^* – матрица, сопряженная к B_1 .

Полагая

$$\varphi_k^{(j)} = v_k^{(j)} e^{\lambda t}, \quad \psi_s^{(l)} = u_s^{(l)} e^{\lambda t},$$

где $v_k^{(j)}$, $u_s^{(l)} \in C^4$ из формул (4) вычислим

1) в случае $\omega = 1$: $n = 2$, $p_1 = p_2 = 2$,

$$\varphi_1^{(1)} = e^{3it} \begin{pmatrix} \frac{i}{3} \\ -1 \\ -\frac{i}{3} \\ 1 \end{pmatrix}, \quad \varphi_1^{(2)} = e^{3it} \begin{pmatrix} -\frac{i}{6} \\ -\frac{1}{2} \\ 0 \\ 0 \end{pmatrix}, \quad \varphi_2^{(1)} = e^{it} \begin{pmatrix} -i \\ 1 \\ -i \\ 1 \end{pmatrix}, \quad \varphi_2^{(2)} = e^{it} \begin{pmatrix} -\frac{i}{2} \\ \frac{1}{2} \\ 0 \\ 0 \end{pmatrix},$$

$$\psi_1^{(1)} = e^{3it} \begin{pmatrix} 9 \\ 3i \\ -9 \\ -3i \end{pmatrix}, \quad \psi_1^{(2)} = e^{3it} \begin{pmatrix} 0 \\ 0 \\ -\frac{9}{2} \\ 3i \\ -\frac{9}{2} \end{pmatrix}, \quad \psi_2^{(1)} = e^{it} \begin{pmatrix} 1 \\ i \\ 1 \\ i \end{pmatrix}, \quad \psi_2^{(2)} = e^{it} \begin{pmatrix} 0 \\ 0 \\ -\frac{1}{2} \\ i \\ -\frac{1}{2} \end{pmatrix}.$$

2) в случае $\omega = 3$: $n = 1, p_1 = 2$,

$$\varphi_1^{(1)} = e^{3it} \begin{pmatrix} \frac{i}{3} \\ -1 \\ -\frac{i}{3} \\ 1 \end{pmatrix}, \quad \varphi_1^{(2)} = e^{3it} \begin{pmatrix} -\frac{i}{6} \\ -\frac{1}{2} \\ 0 \\ 0 \end{pmatrix}, \quad \psi_1^{(1)} = e^{3it} \begin{pmatrix} 9 \\ 3i \\ -9 \\ -3i \end{pmatrix}, \quad \psi_1^{(2)} = e^{3it} \begin{pmatrix} 0 \\ 0 \\ -\frac{9}{2} \\ -\frac{3i}{2} \end{pmatrix}.$$

Непосредственной проверкой убеждаемся, что обобщенные жордановы наборы в случаях

1) и 2) удовлетворяют условиям биортогональности [5]

$$\begin{aligned} \langle\langle z_k^{(j)}, \psi_s^{(l)} \rangle\rangle &= \delta_{ks} \delta_{jl}, & \langle\langle \varphi_k^{(j)}, \gamma_s^{(l)} \rangle\rangle &= \delta_{ks} \delta_{jl}, \\ z_k^{(j)} &= B_1 \varphi_k^{(p_k+1-j)}, & \gamma_s^{(l)} &= B_1^* \psi_s^{(p_s+1-l)}, \\ j &= \overline{2, p_k}, & l &= \overline{2, p_s}, & k, s &= \overline{1, n}. \end{aligned}$$

где δ_{ks}, δ_{jl} – символы Кронекера,

$$\langle\langle z_k^{(j)}, \psi_s^{(l)} \rangle\rangle = \frac{1}{T} \int_0^T \langle z_k^{(j)}(t), \psi_s^{(l)}(t) \rangle dt, \quad \langle\langle \varphi_k^{(j)}, \gamma_s^{(l)} \rangle\rangle = \frac{1}{T} \int_0^T \langle \varphi_k^{(j)}(t), \gamma_s^{(l)}(t) \rangle dt,$$

где $\langle \cdot, \cdot \rangle$ – скалярное произведение векторов.

Нахождение коэффициентов в разложении искомого периодического решения по обобщенному жорданову набору. Согласно [5] вычислим коэффициенты в разложении периодического решения возмущенной системы по формуле

$$g_{kj} = \langle\langle f, \psi_k^{(j)} \rangle\rangle \equiv \frac{1}{T} \int_0^T \langle f(t), \psi_k^{(j)}(t) \rangle dt, \quad k = \overline{1, n}, j = \overline{1, p_k}.$$

Заметим, что для того, чтобы невозмущенная система (3) имела T -периодические решения необходимо и достаточно, чтобы $g_{k1} = 0, k = \overline{1, n}$.

Рассмотрим случай 1) $\omega = 1$. Имеем

$$g_{11} = 0, \quad g_{12} = 0, \quad g_{21} = i(-r_1 e^{i\theta_1} + r_2 e^{i\theta_2}), \quad g_{22} = \frac{1}{4}.$$

Если $-r_1 e^{i\theta_1} + r_2 e^{i\theta_2} = 0$, то $g_{21} = 0$ и система (3) имеет семейство 2π -периодических решений. В частности, это условие выполняется при $r_1 = r_2 = 1$ и $\theta_1 = \theta_2 = \frac{\pi}{2}$.

Рассмотрим случай 2) $\omega = 3$. Имеем

$$g_{11} = 3i(r_1 e^{i\theta_1} + r_2 e^{i\theta_2}), \quad g_{12} = -\frac{3i}{4}.$$

Если $(r_1 e^{i\theta_1} + r_2 e^{i\theta_2}) = 0$, то $g_{11} = 0$ и система (3) имеет семейство $\frac{2\pi}{3}$ -периодических решений. Это условие выполняется, например, при $r_1 = r_2 = 1$, $\theta_1 = 0$ и $\theta_2 = \pi$.

Нахождение дополнительного линейного слагаемого. Дополнительное линейное слагаемое, входящее в периодическое решение возмущенной системы и принадлежащее к дополнению корневого пространства, имеет вид

$$y(t, \varepsilon) = e^{i\omega t} b + e^{-i\omega t} \bar{b},$$

где $b \in C^4$ находится из системы линейных алгебраических уравнений

$$\begin{pmatrix} B_0 - \varepsilon B_1 + \operatorname{Re}(S) & \omega A - \operatorname{Im}(S) \\ -\omega A + \operatorname{Im}(S) & B_0 - \varepsilon B_1 + \operatorname{Re}(S) \end{pmatrix} \begin{pmatrix} \operatorname{Re}(b) \\ \operatorname{Im}(b) \end{pmatrix} = \begin{pmatrix} \operatorname{Re}(h) \\ \operatorname{Im}(h) \end{pmatrix}. \quad (5)$$

Здесь $h = \operatorname{colon}(0, r_1 e^{i\theta_1}, 0, r_2 e^{i\theta_2})$, $S - (2 \times 2)$ – матрица, элементы которой вычисляются по формулам

$$S = \sum_{k=1}^n S_k, \quad S_k = \frac{1}{T} \int_0^T z_k^{(1)}(t) \bar{\gamma}_k^{(1)}(t) dt, \\ z_k^{(1)} = B_1 \varphi_k^{(p_k)}, \quad \gamma_k^{(1)} = B_1^* \psi_k^{(p_k)}, \\ k = \overline{1, n}.$$

Решая систему (5) в случае $\omega = 1$, получим

$$y(t, \varepsilon) = \begin{pmatrix} -\frac{(2\varepsilon + 1) \sin t}{4(\varepsilon^2 - 1)} \\ \frac{(2\varepsilon + 1) \cos t}{4(\varepsilon^2 - 1)} \\ \frac{\varepsilon \sin t}{2(\varepsilon^2 - 1)} \\ -\frac{\varepsilon \cos t}{2(\varepsilon^2 - 1)} \end{pmatrix}, \quad y(t, 0) = \begin{pmatrix} \frac{\sin t}{4} \\ \frac{\cos t}{4} \\ 0 \\ 0 \end{pmatrix}.$$

Решая систему (5) в случае $\omega = 3$, получим

$$y(t, \varepsilon) = \begin{pmatrix} -\frac{(2\varepsilon - 1) \cos 3t}{4(\varepsilon^2 - 1)} \\ \frac{3(2\varepsilon - 1) \sin 2t}{4(\varepsilon^2 - 1)} \\ \frac{\varepsilon \cos 3t}{2(\varepsilon^2 - 1)} \\ -\frac{3\varepsilon \sin 3t}{2(\varepsilon^2 - 1)} \end{pmatrix}, \quad y(t, 0) = \begin{pmatrix} -\frac{\cos 3t}{4} \\ \frac{3 \sin 3t}{4} \\ 0 \\ 0 \end{pmatrix}.$$

Периодические решения возмущенной и невозмущенной систем. С использованием математического пакета Maple методом Ляпунова-Шмидта [5] построены следующие периодические решения возмущенной и невозмущенной систем.

Единственное T -периодическое решение системы (2) имеет вид [5]

$$x(t, \varepsilon) = \sum_{k=1}^n \sum_{j=1}^{p_k} \frac{\varepsilon^{j-1}}{1 - \varepsilon^{p_k}} \left(\xi_{k1} \varphi_k^{(j)} + \bar{\xi}_{k1} \bar{\varphi}_k^{(j)} \right) + y(t, \varepsilon), \quad (6)$$

где ξ_{k1} определяются из равенств

$$\xi_{k1} = -\frac{1}{\varepsilon^{p_k}} (g_{k1} + g_{k2}\varepsilon + g_{k3}\varepsilon^2 + \dots + g_{kp_k}\varepsilon^{p_k-1}), \quad k = \overline{1, n}.$$

Семейство T -периодических решений системы (3), имеет вид [5]

$$x(t, 0) \equiv z(t) = \sum_{k=1}^2 c_k \left[\varphi_k^{(1)} + \bar{\varphi}_k^{(1)} \right] + y(t, 0), \quad (7)$$

1) Случай $\omega = 1$. По формулам (6) и (7) найдем 2π -периодические решения систем (2) и (3), соответственно,

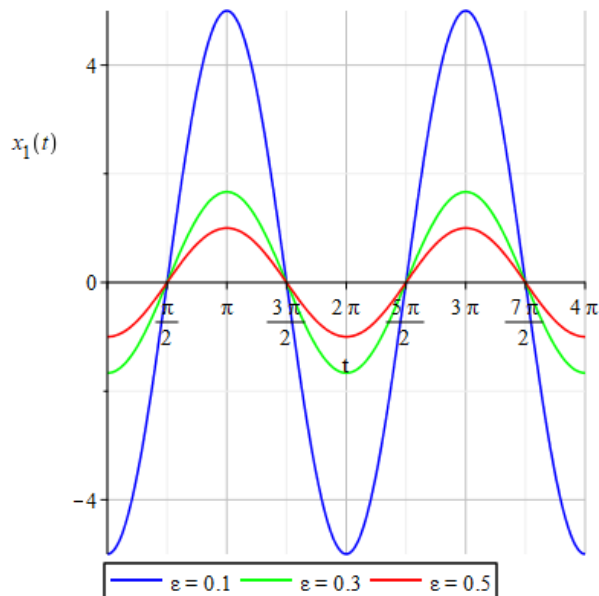
$$x(t, \varepsilon) = \begin{pmatrix} -\frac{\cos t}{2\varepsilon} \\ \frac{\sin t}{2\varepsilon} \\ \frac{\cos t}{2\varepsilon} \\ -\frac{\sin t}{2\varepsilon} \end{pmatrix}, \quad z(t) = \begin{pmatrix} \frac{2}{3}c_1 \sin 3t - 2c_2 \sin t + \frac{1}{4}\cos t \\ 2c_1 \cos 3t - 2c_2 \cos t - \frac{1}{4}\sin t \\ \frac{2}{3}c_1 \sin 3t + 2c_2 \sin t \\ 2c_1 \cos 3t + 2c_2 \cos t \end{pmatrix}.$$

2) Случай $\omega = 3$. Учитывая формулы (6) и (7), найдем $\frac{2\pi}{3}$ -периодические решения систем (2) и (3), соответственно,

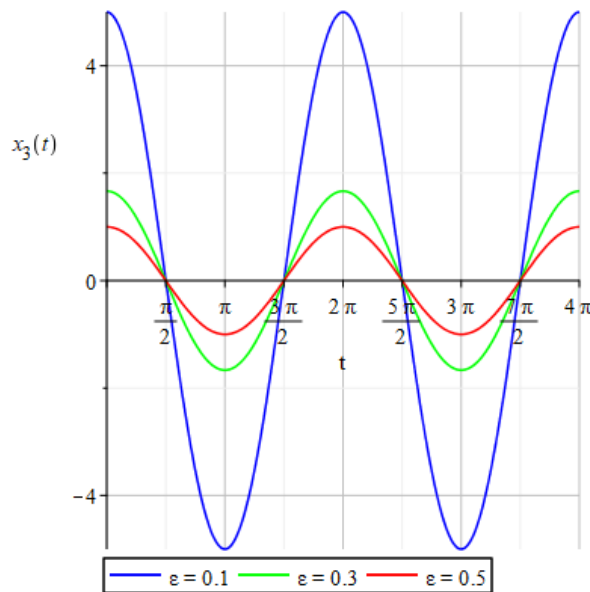
$$x(t, \varepsilon) = \begin{pmatrix} -\frac{\sin 3t}{2\varepsilon} \\ \frac{3 \cos 3t}{2\varepsilon} \\ -\frac{\sin 3t}{2\varepsilon} \\ \frac{3 \sin 3t}{2\varepsilon} \end{pmatrix}, \quad z(t) = \begin{pmatrix} \frac{2}{3}c_1 \sin 3t - \frac{1}{4}\sin 3t \\ 2c_1 \cos 3t - \frac{3}{4}\cos 3t \\ \frac{2}{3}c_1 \sin 3t \\ 2c_1 \cos 3t \end{pmatrix}.$$

Построение графиков компонент периодических решений возмущенной системы.

Учитывая, что обобщенным координатам q_1 и q_2 , соответствуют компоненты $x_1(t, \varepsilon)$ и $x_3(t, \varepsilon)$ периодического решения возмущенной системы (2), приведем графики этих компонент при различных ε в случаях 1) $\omega = 1$ и 2) $\omega = 3$.

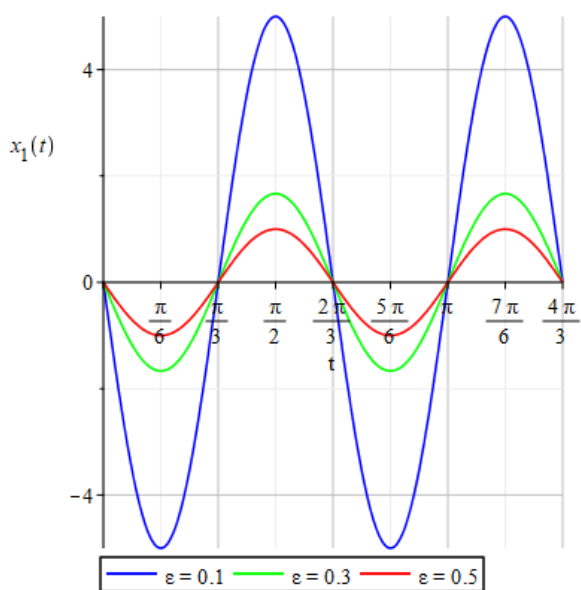


а) $x_1(t, \varepsilon)$;

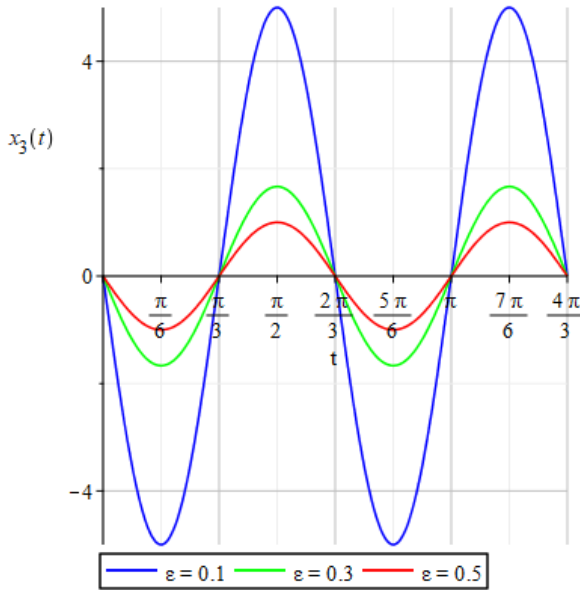


б) $x_3(t, \varepsilon)$;

Рис. 1. Графики компонент а) $x_1(t, \varepsilon)$ и б) $x_3(t, \varepsilon)$ 2π -периодических решений системы (2) при различных ε в случае 1) $\omega = 1$.



а) $x_1(t, \varepsilon)$;



б) $x_3(t, \varepsilon)$;

Рис. 2. Графики компонент а) $x_1(t, \varepsilon)$ и б) $x_3(t, \varepsilon)$ $\frac{2\pi}{3}$ -периодических решений системы (2) при различных ε в случае 2) $\omega = 3$.

Построение графиков проекций фазовых траекторий возмущенной и невозмущенной систем. Приведем графики проекций фазовых траекторий возмущенной и невозмущенной систем (2) и (3).

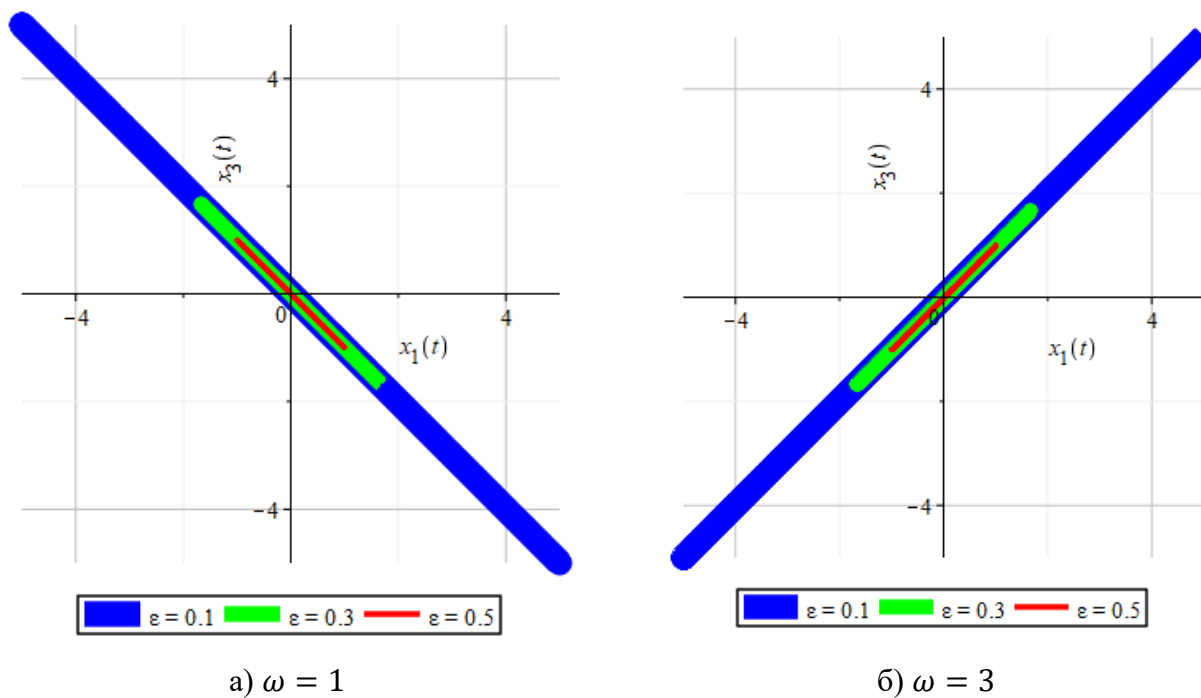


Рис. 3. Графики проекций фазовых траекторий на плоскость (x_1, x_3) возмущенной системы (2) при различных ε в случае а) $\omega = 1$ и б) $\omega = 3$.

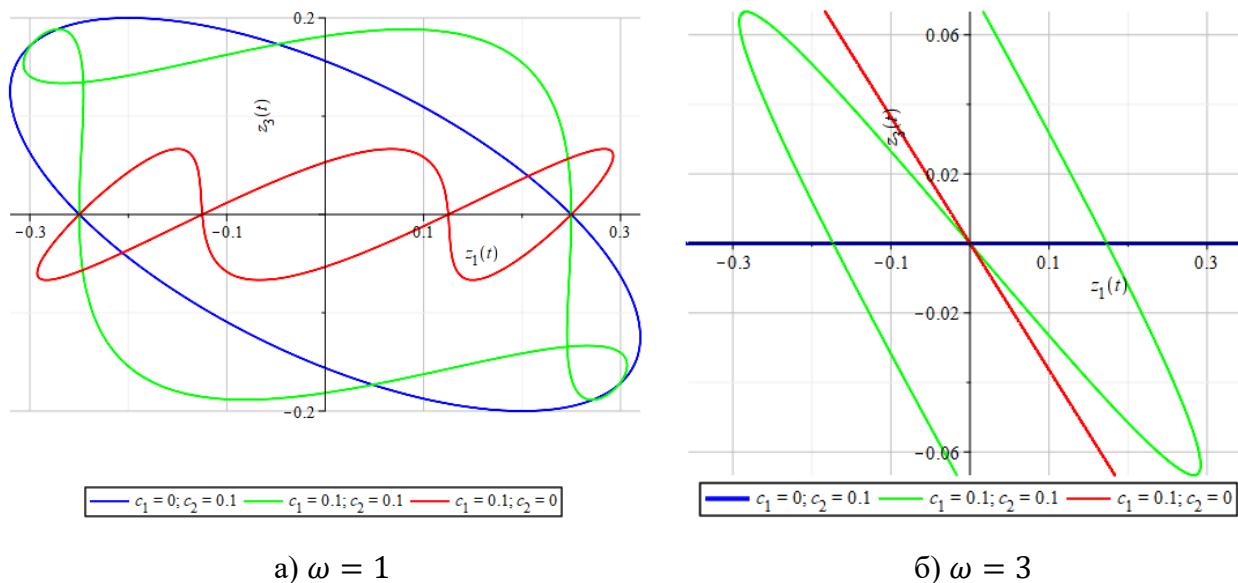


Рис. 4. Графики проекций фазовых траекторий на плоскость (z_1, z_3) невозмущенной системы (3) при различных ε в случае а) $\omega = 1$ и б) $\omega = 3$.

Заключение. Из формул (4) и (5) следует, что периодические решения $x(t, \varepsilon)$ возмущенной системы (2) в случаях 1) $\omega = 1$ и 2) $\omega = 3$ имеют полюс первого порядка в точке $\varepsilon = 0$, и, следовательно, при $\varepsilon \rightarrow 0$ не стремятся к соответствующим решениям $z(t)$ невозмущенной системы (3). Наличие полюса первого порядка обуславливается тем, что коэффициенты g_{22} в случае 1) $\omega = 1$ и g_{12} в случае 2) $\omega = 3$ отличны от нуля.

СПИСОК ЛИТЕРАТУРЫ

1. Вайнберг М. М., Треногин В. А. Теория ветвления решений нелинейных уравнений. – М.: Наука. – 1964. – 524 с.
2. Коноплева И. В., Логинов Б. В. Обобщенная жорданова структура и симметрия разрешающих систем ветвления // Вестн. Самарск. ун-та. – 2001. – №4. – С. 56–84.
3. Кяшкин А. А., Логинов Б. В., Шаманаев П. А. Комментарии к задачам о возмущениях линейного уравнения малым линейным слагаемым и спектральных характеристик фредгольмова оператора // Журнал Средневолжского математического общества. – 2013. – Т. 15, № 3. – С. 100–107.
4. Кяшкин А. А., Логинов Б. В., Шаманаев П. А. Комментарии к задаче о ветвлении периодических решений при бифуркации Андронова-Хопфа в дифференциальных уравнениях с вырожденным оператором при производной // Журнал Средневолжского математического общества. – 2014. – Т. 16, № 4. – С. 33–40.
5. Кяшкин А. А., Логинов Б. В., Шаманаев П. А. О ветвлении периодических решений линейных неоднородных дифференциальных уравнений с вырожденным или тождественным оператором при производной и возмущением в виде малого линейного слагаемого // Журнал Средневолжского математического общества. – 2016. – Т. 18, № 1. – С. 45–53.
6. Кадрякова М. Р., Логинов Б. В., Шаманаев П. А. О периодических решениях одного класса линейных неоднородных систем обыкновенных дифференциальных уравнений с малым параметром в резонансном случае [Электронный ресурс] // Ogarev-online. – 2017. – №13. – Режим доступа: <http://journal.mrsu.ru/arts/o-periodicheskix-resheniyax-odnogo-klassa-linejnyx-neodnorodnyx-sistem-obyknovennyx-differencialnyx-uravnenij-s-malym-parametrom-v-rezonansnom-sluchae> (дата обращения 12.09.2020).
7. Стрелков С. П. Введение в теорию колебаний: Учебник. 3-е изд., испр. – СПб.: Лань, 2005. – 440 с.

БУТКИНА А. А., КОМАРОВ Р. В.

**ПРОЕКТИРОВАНИЕ И РЕАЛИЗАЦИЯ МОБИЛЬНОГО ПРИЛОЖЕНИЯ
ДЛЯ ИНФОРМАЦИОННОГО ПОРТАЛА «ГОРОДСКИЕ РЕЙТИНГИ | САРАНСК»**

Аннотация. В статье описан процесс создания мобильного приложения для информационного портала «Городские рейтинги | Саранск». На примере процесса реализации данного приложения рассмотрены базовые этапы разработки мобильного приложения: формулирование требований, проектирование и программная реализация компонентов, тестирование.

Ключевые слова: проектирование, разработка, тестирование, мобильное приложение, iOS, унифицированный язык моделирования, Model-View-Controller.

BUTKINA A. A., KOMAROV R. V.

**DESIGN AND IMPLEMENTATION OF MOBILE APPLICATION
FOR THE INFORMATION PORTAL "CITY RATINGS | SARANSK"**

Abstract. The article considers the process of creating a mobile application for the information portal "City ratings | Saransk". Doing this project, the authors provide a description of the basic stages of the development of a mobile application: the formulation of requirements, design and implementing of the system components, testing.

Keywords: design, development, testing, mobile application, iOS, unified modeling language, Model-View-Controller.

Введение. В течение последнего десятилетия быстрыми темпами развивается рынок мобильных устройств. По данным отчета о состоянии цифровой сферы Digital 2020 [1], уже более 5 миллиардов людей используют мобильные телефоны. При этом на мобильные телефоны в настоящее время приходится больше половины времени, которое люди проводят в интернете — 50,1%, что естественно связано с появлением и бурным развитием мобильного Интернета. Изменилась в последний период и специфика использования «мобильного времени». Аналитическая платформа «App Annie» заявляет, что на мобильные приложения теперь приходится 10 из каждых 11 минут пользования мобильным устройством, а на просмотр веб-страниц уходит соответственно только 9% нашего «мобильного времени». Поэтому сложно переоценить все более возрастающую актуальность разработки мобильных приложений.

Целью данной работы является разработка мобильного приложения для операционной системы iOS, которое позволит пользователям получить доступ к информационному

контенту портала «Городские Рейтинги | Саранск». Для достижения поставленной цели в работе были поставлены и решены следующие задачи:

- изучить предметную область и сформулировать функциональные и нефункциональные требования к приложению;
- разработать архитектурную модель приложения;
- создать минималистичный и удобный дизайн мобильного приложения;
- реализовать приложение, отвечающее всем сформулированным требованиям;
- выполнить тестирование приложения.

Технологии реализации мобильного приложения. При разработке мобильного приложения для информационного портала «Городские Рейтинги | Саранск» был использован следующий стек технологий:

- кросс-платформенный онлайн-сервис Figma [2], применяемый при разработке дизайна экранов мобильного приложения;
- интегрированная среда разработки Xcode [3], выбранная для разработки приложения;
- язык программирования Swift [4], на котором создавалось приложение для iOS;
- библиотека Moya [5], выбранная для организации работы с сетью;
- библиотека Kingfisher [6], выбранная для загрузки и кэширования изображений из сети;
- распределённая система контроля версий Git.

Проектирование мобильного приложения. Перед проектированием мобильного приложения были разработаны функциональные требования, нашедшие отражение в диаграмме вариантов использования, построенной при разработке концептуальной модели приложения (рис. 1).

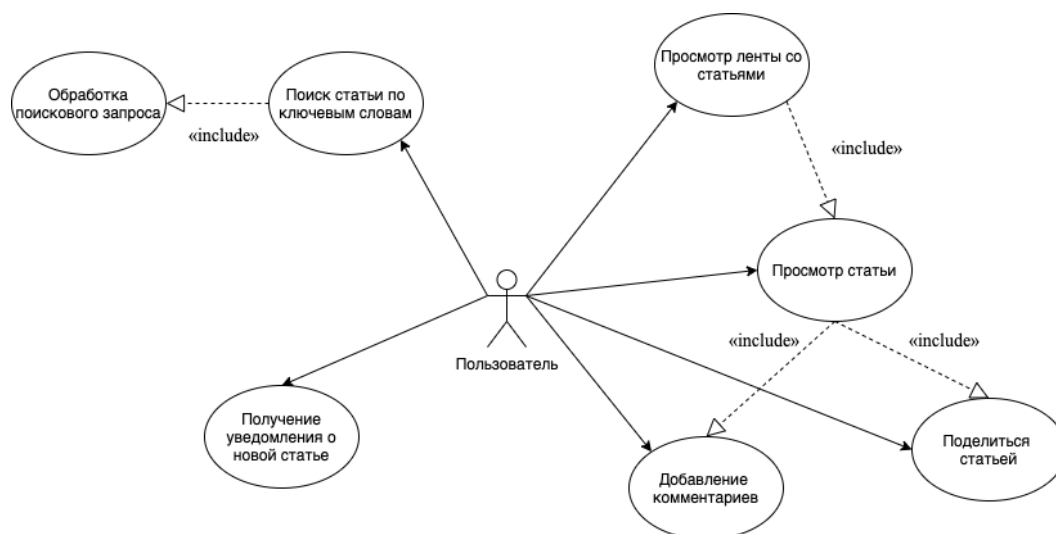


Рис. 1. Диаграмма вариантов использования.

После описания приложения на концептуальном уровне была разработана его логическая модель, включающая описание основных классов системы и используемых в них полей и методы, а также диаграмму наиболее важных модулей приложения, описания значимых классов, показанную на рис. 2.

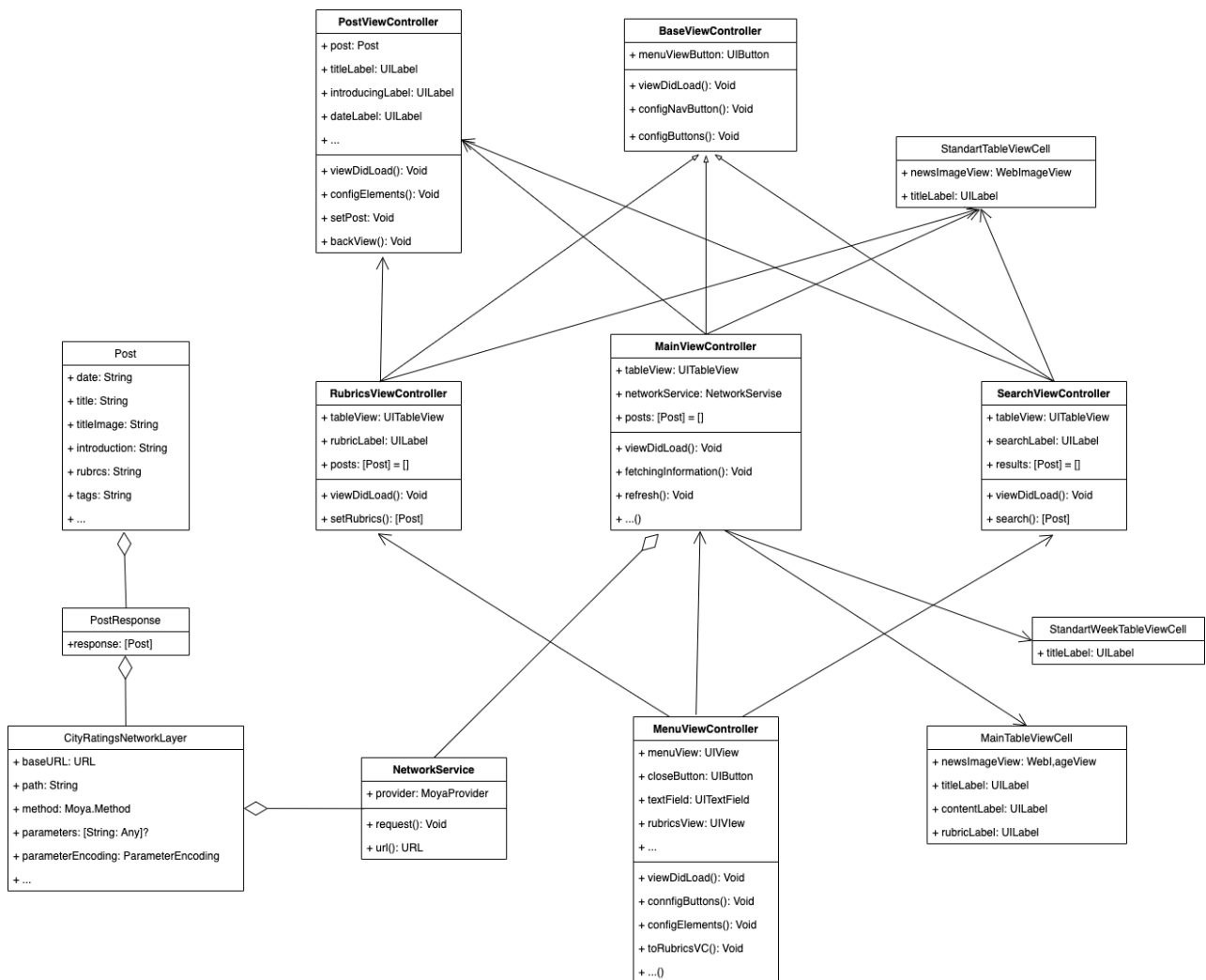


Рис. 2. Диаграмма классов.

Среди продемонстрированных на рисунке 2 классов наиболее важными являются:

- класс «MainViewController», который отвечает за отображение информации на главном экране. В этом классе происходит запрос к API для получения данных, а также выполняется конвертация полученных данных в вид, понятный пользователю;
- класс «PostViewController», который отвечает за отображение выбранной статьи при нажатии на соответствующую ячейку в таблице;
- класс «MenuViewController», который отвечает за управление основными экранами приложения, также в данном классе содержатся строка поиска и кнопки меню для перехода к нужной рубрике;

– структура «PostResponse», которая включает атрибут, содержащий в себе массив данных типа Post, в котором хранится вся информация, получаемая от сервера.

На следующем этапе проектирования архитектуры приложения была разработана модель реализации, учитывающая применяемый архитектурный шаблон Model-View-Controller (MVC). Для представления модели реализации используется диаграмма компонентов, представленная на рисунке 3.

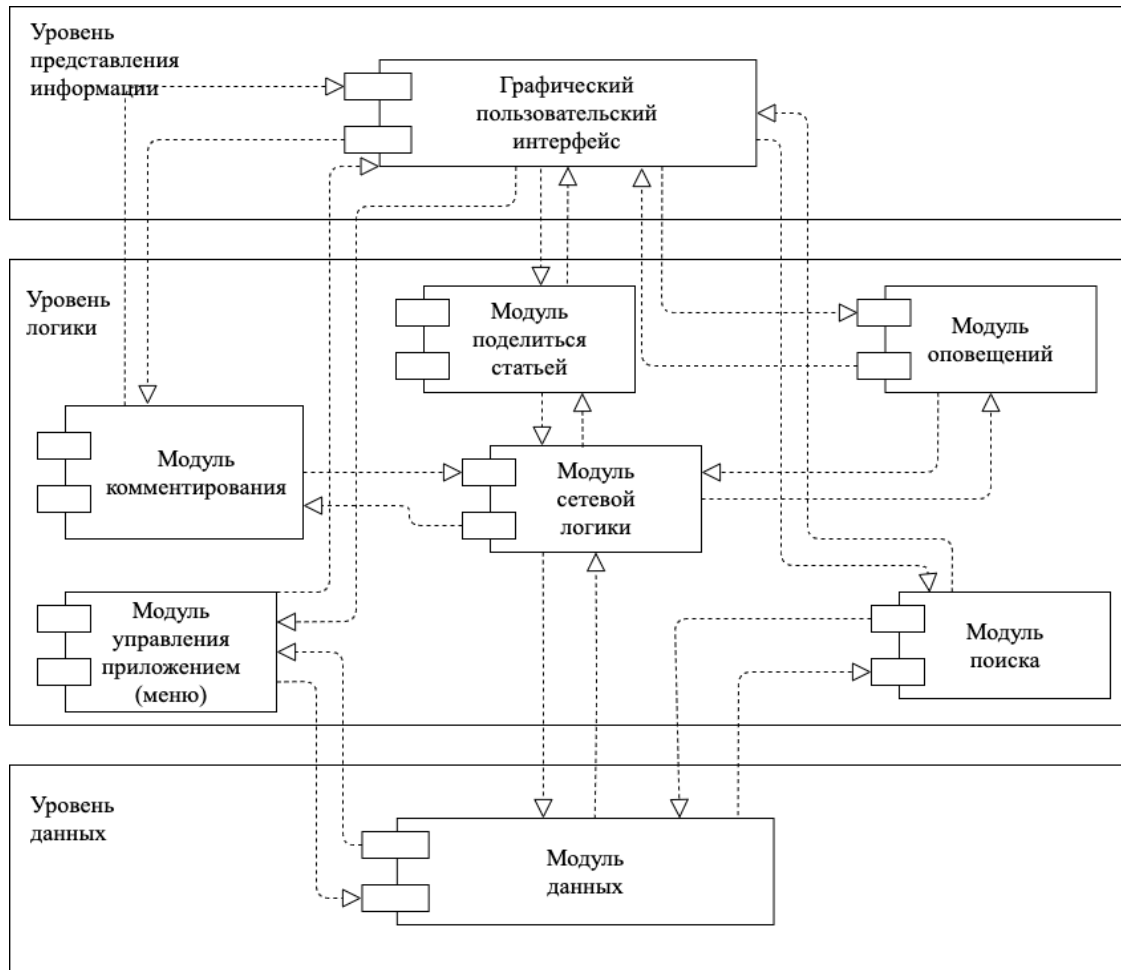


Рис. 3. Диаграмма компонентов.

В соответствии с разработанной моделью архитектура мобильного приложения делится на три уровня:

1. Уровень представления информации, на котором осуществляется взаимодействие приложения с пользователем посредством графического пользовательского интерфейса.
2. Уровень логики, на котором выполняется всё внутреннее взаимодействие между модулями приложения – происходят сетевые запросы, осуществляется передача данных между контроллерами, выполняется заполнение данными элементов отображения на экране.
3. Уровень данных, который содержит массив структур данных, который формируется в результате сетевого запроса к API из полученного в нем JSON ответа.

После завершения этапа разработки архитектурной модели приложения был разработан дизайн основных экранов мобильного приложения с помощью сервиса Figma – экран запуска приложения, главный экран, экран рубрик, экран статьи, экран поиска и меню.

Обзор основных модулей реализованного мобильного приложения. Перейдем к рассмотрению основных модулей, реализованных в мобильном приложении, которые были продемонстрированы на диаграмме компонентов (рис. 3):

1. Модуль «Статьи». Основной функцией мобильного приложения «Городские рейтинги | Саранск» является возможность просматривать статьи с одноименного портала. На главном экране представлены статьи, которые отображаются, начиная с последней опубликованной на данном портале (рис. 4).

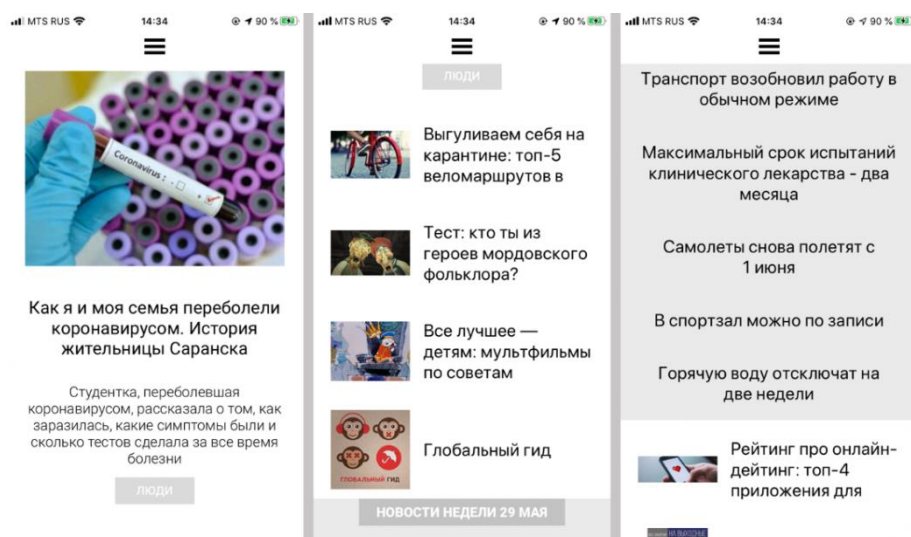


Рис. 4. Главный экран приложения.

2. Модуль «Рубрики», который отображает статьи, соответствующие определенной рубрике с информационного портала, что является для той категории пользователей, которая следит только за новостями определенной тематики (рис. 5).

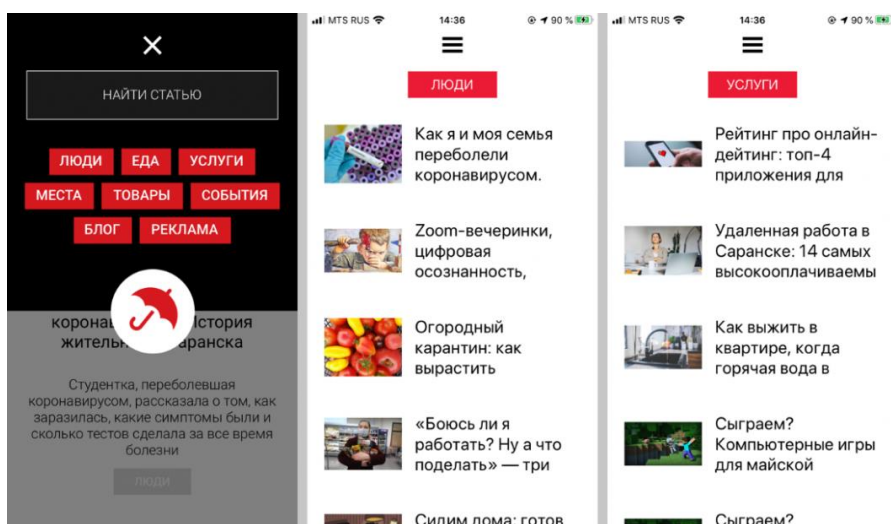


Рис. 5. Раздел рубрики.

3. Модуль «Просмотр статьи», который отвечает за отображение информации, содержащейся в статье, включая не только название и вступление, а также вложения, которые содержатся внутри статьи (рис. 6).

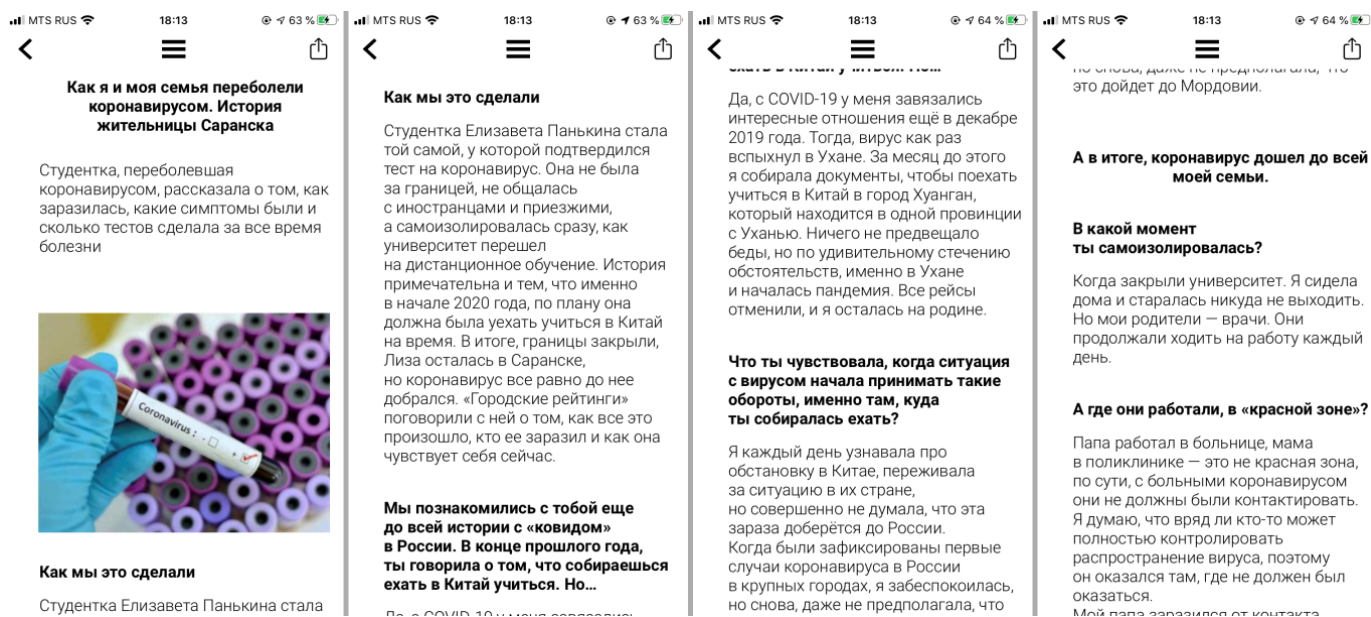


Рис. 6. Просмотр статьи.

4. Модуль «Поиск», отвечающий за поиск статей по ключевым словам (рис. 7).

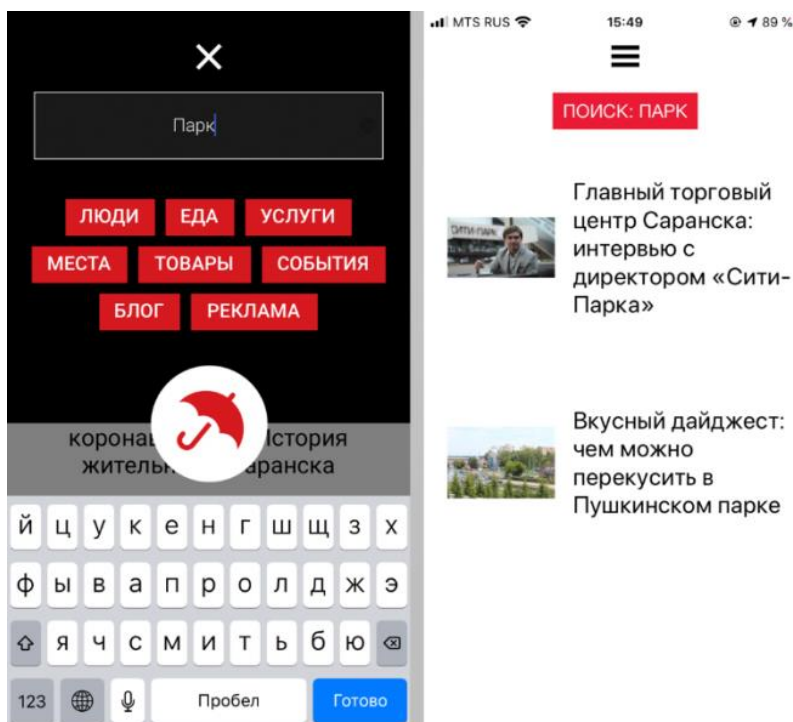


Рис. 7. Поиск статей по словам.

5. Модуль «Добавление комментариев», который позволяет пользователям просматривать существующие и оставлять новые комментарии под статьями (рис. 8).

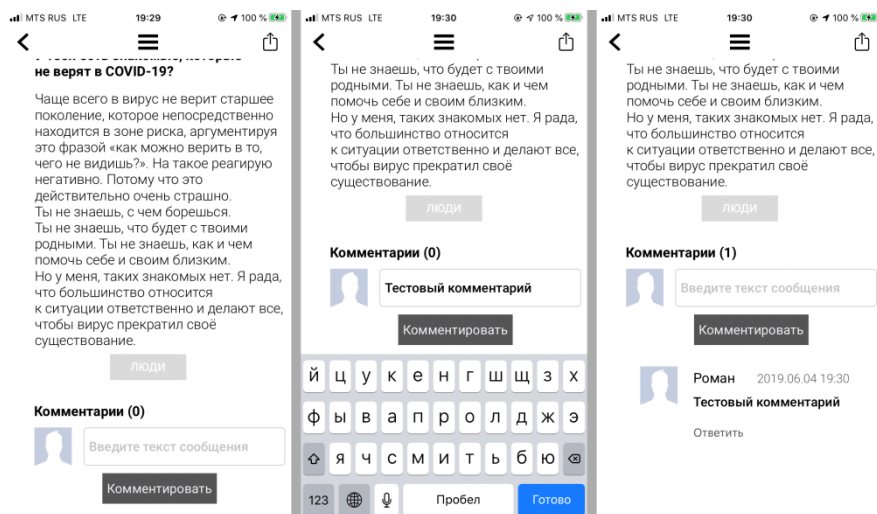


Рис. 8. Добавление комментариев.

6. Модуль «Поделиться статьей», с помощью которого можно поделиться ссылкой на статью – отправить в различные социальные сети, в виде сообщения или на электронную почту, сохранить в облачное хранилище данных Dropbox или же просто скопировать ссылку.



Рис. 9. Поделиться статьей.

7. Модуль «Оповещения», который позволяет уведомлять пользователей о публикации новых статей на портале.

Тестирование пользовательского интерфейса мобильного приложения. В современном мире используется огромное количество устройств с различными характеристиками. Даже в актуальной линейке смартфонов Apple находится более 10 моделей с различными размерами экранов. Зачастую проблемой современных приложений является то, что не все интерфейсы хорошо адаптированы под это большое

количество диагоналей и разрешений. Если же интерфейс настроен только под одно устройство, то возникают большие проблемы при открытии приложения на других устройствах, поскольку все графические элементы могут «плавать по экрану» или же вообще пропадут из зоны видимости. Для проверки корректного отображения разработанного мной интерфейса я решил запустить приложение на 3 устройствах, на мой взгляд, с самыми распространенными размерами экранов:

- 1) iPhone 5S – экран 4 дюйма с разрешением 1136x640 пикселей;
- 2) iPhone 8 – экран 4,7 дюйма с разрешением 1334x750 пикселей;
- 3) iPhone 11 – экран 6,1 дюйм с разрешением 1792x828 пикселей.

В результате проведенного анализа работоспособности были получены следующие результаты, представленные на рис. 10. Таким образом, было продемонстрировано обеспечение соответствия стандартам пользовательского интерфейса разработанного мобильного приложения.

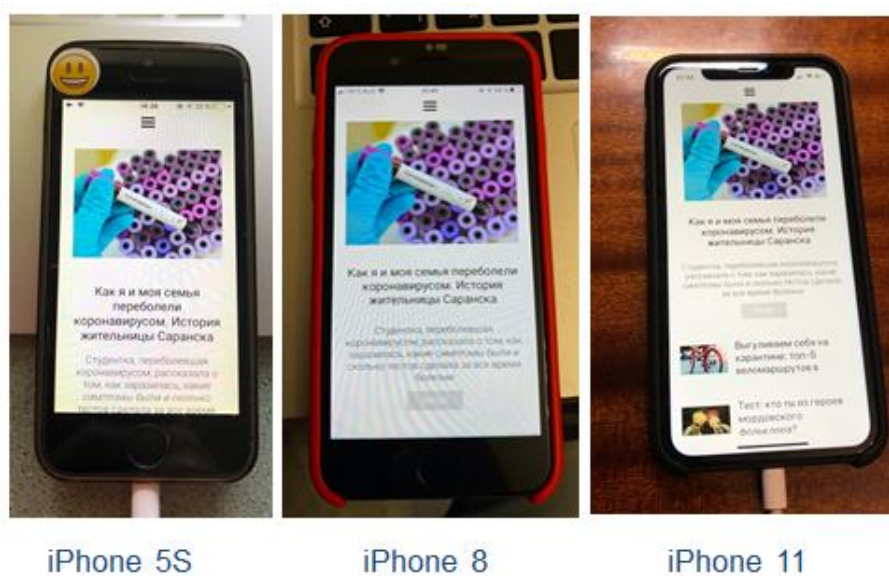


Рис. 10. Тестирование пользовательского интерфейса приложения.

Закключение. В ходе достижения поставленной цели работы были изучены принципы разработки мобильного приложения на платформе iOS, включая все основные стадии: формулирование функциональных и нефункциональных требований к приложению; проектирование архитектурой модели приложения, содержащей концептуальную и логическую модели, а также модель реализации; разработка дизайна 6 основных экранов приложения в соответствии с современными тенденциями; программная реализация всех модулей мобильного приложения, отвечающего всем сформулированным требованиям; тестирование приложения на мобильных устройствах с различными характеристиками.

СПИСОК ЛИТЕРАТУРЫ

1. Вся статистика интернета на 2020 год – цифры и тренды в мире и в России [Электронный ресурс]. – Режим доступа: <https://www.web-canape.ru/business/internet-2020-globalnaya-statistika-i-trendy/> (дата обращения 25.09.2020).
2. Что такое Figma: возможности и принципы работы [Электронный ресурс]. – Режим доступа: https://skillbox.ru/media/design/chto_takoe_figma/ (дата обращения 25.09.2020).
3. Что такое Xcode? [Электронный ресурс]. – Режим доступа: <https://apps.apple.com/ru/story/что-такое-xcode/id1439636456> (дата обращения 25.09.2020).
4. Swift (язык программирования) [Электронный ресурс]. – Режим доступа: [https://ru.wikipedia.org/wiki/Swift_\(язык_программирования\)](https://ru.wikipedia.org/wiki/Swift_(язык_программирования)) (дата обращения 25.09.2020).
5. Мoya – как перестать беспокоиться о сетевой части и начать жить [Электронный ресурс]. – Режим доступа: <https://habr.com/ru/post/332570/> (дата обращения 25.09.2020).
6. Kingfisher. [Электронный ресурс]. – Режим доступа: <https://github.com/onevc/Kingfisher> (дата обращения 25.09.2020).

ШАМАЕВ А. В., КАДАНЦЕВ А. А.

**РАЗРАБОТКА ДОПОЛНИТЕЛЬНОГО ПРОГРАММНОГО МОДУЛЯ
ДЛЯ PLM-СИСТЕМЫ TEAMCENTER (SIEMENS)**

Аннотация. В статье описан процесс создания дополнительного программного модуля для системы управления жизненным циклом изделия промышленного производства Teamcenter, автоматизирующего процесс создания структуры документа «Ведомость конфигурации». Рассмотрены этапы моделирования, проектирования и реализации данного модуля.

Ключевые слова: жизненный цикл, ведомость конфигурации, изделие, кастомизация программы, Teamcenter, Eclipse, код.

SHAMAEV A. V., KADANTSEV A. A.

**DEVELOPMENT OF ADDITIONAL SOFTWARE MODULE
FOR TEAMCENTER PLM SYSTEM (SIEMENS)**

Abstract. This article considers the process of creating an add-on software module for the Teamcenter industrial product lifecycle management system, which automates the process of creating the structure of the «Configuration Sheet» document. The authors describe the stages of modeling, design and implementation of this module.

Keywords: product lifecycle, configuration sheet, machine-building product, program customization, Teamcenter, Eclipse, code.

Введение. Аббревиатура PLM (Product Lifecycle Management) – это общепринятое сокращенное название совокупности технологий, применяемых для организации управления жизненным циклом продукта (изделия). PLM-системой называют организационно-техническую систему, которая обеспечивает управление всей информацией о продукте и связанных с ним процессах на протяжении всего жизненного цикла этого продукта, начиная с его проектирования и производства и завершая прекращением его эксплуатации.

PLM-системы доказали свою эффективность, предлагая эффективные способы хранения и обработки данных, а также комплексный набор бизнес-решений, предназначенных для поддержки процессов создания, управления, распределения и использования интеллектуальных активов предприятия. Описывая процессы проектирования и подготовки производства, производитель PLM-системы (которая представляет собой специализированный комплекс программ) предоставляет максимально детализированные шаблоны, формы и предлагает инструменты для их настройки. Однако всегда существует ряд задач, к которым невозможно применить готовые шаблоны. Для их решения

производители таких систем предоставляют пользователям возможность интеграции стороннего программного обеспечения, благодаря чему предприятие, применяющее такую систему, может вести разработку дополнительных программных модулей, которые позволят увеличить скорость работы, минимизировать количество ошибок при выполнении операций, что в итоге приведёт к снижению затрат на выполнение проекта.

Описание PLM-системы Teamcenter. Teamcenter представляет собой многофункциональную платформу, которая, помимо набора стандартных приложений Siemens PLM Software, предназначенных для проектирования и анализа, может включать другие специализированные программы для проектирования продуктов машиностроения, приборостроения, строительства и других областей. Она организует управление инженерными данными и производственными процессами на протяжении всего жизненного цикла продукта. Прежде всего данная платформа предназначена для создания единой корпоративной информационной системы, организующей управление компанией или группой компаний, филиалы которой географически удалены друг от друга [1, с. 32; 2, с. 8].

Ниже перечислены основные возможности Teamcenter, как единой системой управления данными о продукте [1, с. 32]:

- оценка влияния каждого изменения на общую конфигурацию всего продукта;
- создание специально согласованной структуры продукта, которая может послужить основой для анализа и учета изменений;
- проведение экспертизы поставленных и выполненных задач;
- уведомление пользователей, о том, когда должны быть реализованы задачи по внедрению изменений;
- ведение истории изменений продукта на всем протяжении его жизненного цикла;
- проверка сохранности функциональных характеристик после внесения изменений в конфигурацию продукта;
- сравнение ожидаемых и фактически полученных влияний от изменений, внесенных в продукт.

Архитектура Teamcenter. Многоуровневая архитектура системы Teamcenter изображена на рисунке 1.

Синим цветом на рисунке выделены самые низкие (серверные) уровни. Они расположены ближе к базе данных и томам с файлами. Зеленым цветом на рисунке выделены самые высокие (клиентские) уровни. Они расположены ближе к пользователю [3, с. 13].

Существует два варианта развертывания системы Teamcenter, основанные на четырёхуровневой и двухуровневой архитектуре, соответственно.

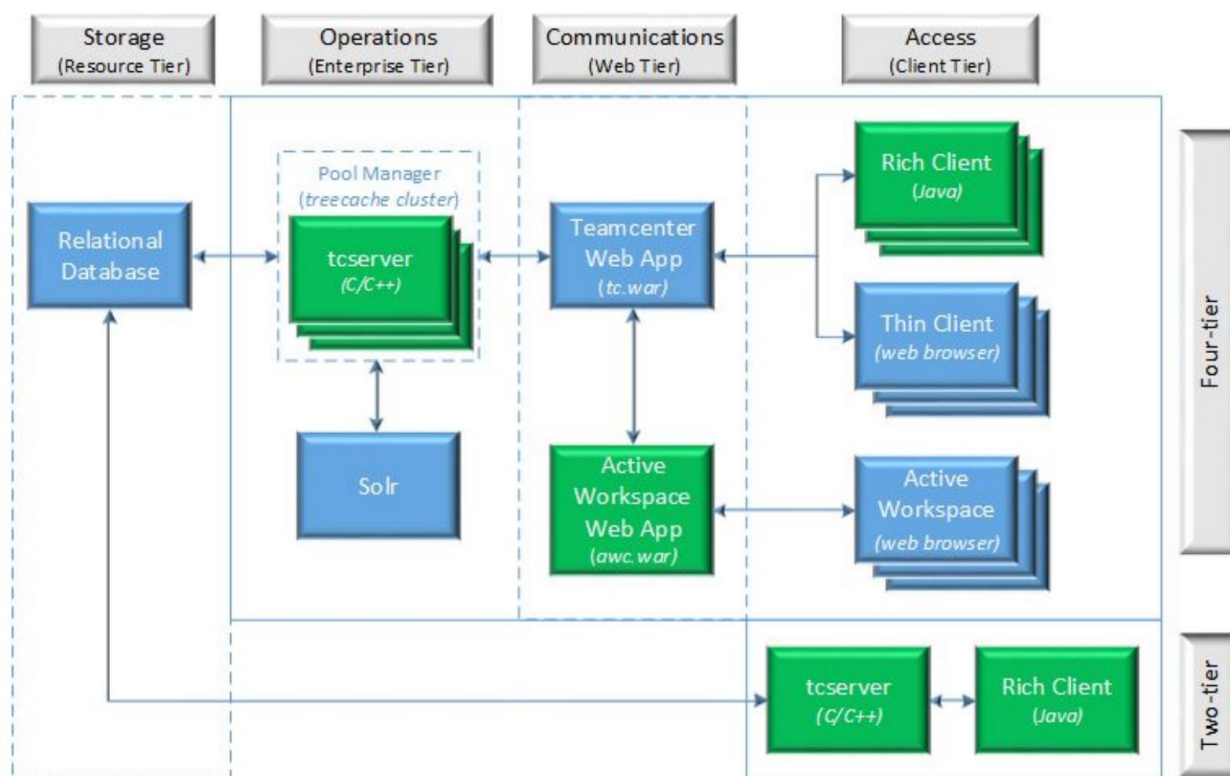


Рис. 1. Многоуровневая архитектура Teamcenter.

Двухуровневая архитектура в большей степени подходит для исследования задачи интеграции сторонних модулей в систему Teamcenter, поэтому в данной работе использовалась именно она.

При применении такой архитектуры серверная часть (бизнес-логика) системы и «толстый» клиент (далее просто клиент) запускаются непосредственно на компьютере клиента. Клиент и сервер могут быть подвержены кастомизации, под которой понимается расширение функционала программы.

Определение ключевых прецедентов программного модуля. Среди совокупности задач, которые можно решить с помощью стандартных методов Teamcenter, есть задачи, требующие создания дополнительных программных модулей. Одной из таких задач является задача формирования ведомости конфигурации.

Каждый сложный продукт подвергается декомпозиции и должен быть разбит на системы, подсистемы и объекты конфигурации.

Ведомость конфигурации (ВК) – проектный документ, который полностью и однозначно описывает в требуемом формате архитектуру и состав конечного продукта в его конкретной конфигурации в соответствии с вариантным правилом и правилом модификации.

В системе Teamcenter, а именно в приложении «Менеджер структуры», отсутствуют готовые инструменты, автоматизирующие процесс формирования данного документа и потому он формируется соответствующими специалистами вручную. Ручное создание

ведомости конфигурации представляет собой длительный и трудоемкий процесс обработки тысячи компонентов, часто приводящий к ошибкам и получению неверного результата. Поэтому задачей данной работы являлась частичная автоматизация этого процесса.

Инструменты для разработки модуля. Перед началом разработки дополнительного программного модуля в рамках Teamcenter версии 11.2 были установлены следующие компоненты [4, с. 13]:

1. Программа Teamcenter с двухуровневым вариантом развертывания.
2. Язык программирования Java и его комплект разработчика приложений (Java Development Kit).
3. Интегрированная среда разработки Eclipse.

Рассмотрим реализацию второго и третьего пункта пошагово:

а) Необходимо установить JDK той же версии, которая используется вместе с клиентом Teamcenter, чтобы обеспечить его полноценное взаимодействие с разрабатываемым модулем. Выберем версию, указанную в документации по настройке клиента на основе версии Teamcenter. В нашем случае это JDK 1.8.0_181.

б) Для работы полнофункционального клиента должна быть установлена и настроена среда разработки Eclipse 3.8 Software Development Kit с версией, указанной в документации по настройке клиента Teamcenter версии 11.2. Настройка клиента включает в себя настройку определенных параметров проекта и запуск клиента Teamcenter через Eclipse.

Подготовка и настройка программного обеспечения. После загрузки и установки вышеуказанных инструментов разработки в каталоге, где размещена среда разработки Eclipse, необходимо создать bat-файл, с помощью которого через параметры командной строки JDK будут устанавливаться значения переменных среды, запуск сервера Teamcenter и среды разработки Eclipse. Пример содержания этого файла приведен ниже [4, с.149].

```
set TPR="TC_ROOTportal"
if not defined FMS_HOME set FMS_HOME=" TC_ROOT \tccs"
set JAVA_HOME="jre-install-directory"
set JRE_HOME="jre-install-directory"
set ORIGINAL_PATH="%PATH%"
set PATH="%SYSTEMROOT%\system32; %FMS_HOME%\bin; %FMS_HOME%\lib; %TPR%; %AUX_PATH%"
start "TAO ImR" /min cmd /c "TC_ROOT\iiopservers\start_imr.bat"
Eclipse-install-directory\eclipse.exe -vm "jdk-install-directory\bin\javaw.exe"
```

При выполнении созданного bat-файла среда разработки Eclipse потребует задать путь к рабочему каталогу, в котором будут храниться создаваемые проекты и связанная с ними

информация. В результате выполнения описанных выше шагов среда Eclipse успешно запустится и будет готова к написанию кода программы.

Настройки проекта. При настройке проекта были указаны пути к виртуальной машине Java – Java Runtime Environment (JRE), а также целевая платформа, для которой разрабатывался дополнительный модуль. В нашем случае целевой платформой является клиент Teamcenter, поэтому также нужно указать каталог, в котором находятся все java-пакеты, используемые при работе клиента. Порядок дальнейших действий по настройке проекта подробно описан в [4, с.150].

Результатом выполненных действий является проект Eclipse, настроенный для работы с целевой платформой Teamcenter. Запуск клиента Teamcenter из Eclipse необходим для проверки работы разработанного модуля без необходимости обновления существующих клиентских модулей системы Teamcenter.

Разработанный модуль загружается динамически и отображается в списке установленных на данный момент модулей. Очевидными плюсами такого запуска является возможность быстрой проверки разработанного приложения и возможность его отладки во время работы. Процедура запуска разработанного модуля совместно с модулями Teamcenter подробно описана в [4 с. 151].

Таким образом, после завершения настройки конфигурации проекта можно использовать разработанный модуль совместно с установленными модулями клиента Teamcenter как в обычном режиме, так и в режиме отладки.

Разработка диаграммы последовательности. Структура и логика работы программного модуля определяется его архитектурой. Для её описания, руководствуясь методологией RUP, будем использовать диаграммы языка UML. Начнем с диаграммы последовательности, которая представлена на рисунке 2. Она поэтапно описывает взаимодействие пользователя системы Teamcenter с данным модулем.

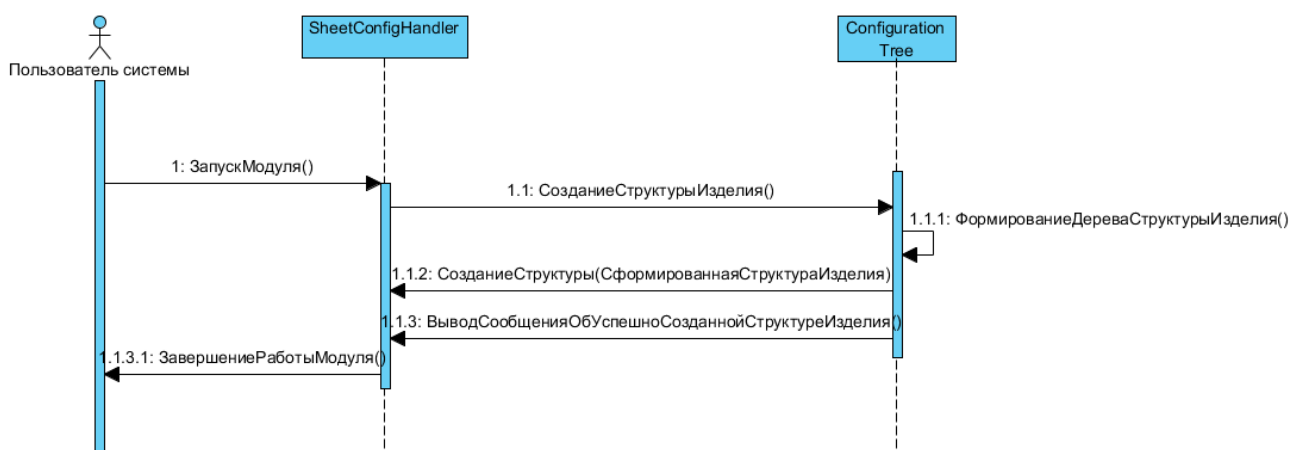


Рис. 2. Диаграмма последовательности.

Разработка диаграммы классов. На рисунке 3 приведена диаграмма классов разработанного программного модуля. Рассмотрим её с точки зрения реализации, то есть дадим краткое описание созданных классов:

- Activator – это класс, управляющий жизненным циклом модуля. Он должен реализовывать интерфейс. Среда разработки Eclipse вызывает метод start() этого интерфейса сразу после загрузки модуля и метод stop() непосредственно перед его выгрузкой;
- SheetConfigHandler – это класс-обработчик, который запускает программу при выборе соответствующего пункта меню «Создать Ведомость конфигурации...» из раздела «Отчеты» главного окна приложения «Менеджер структуры» системы Teamcenter;
- ConfigurationTree – это класс, который формирует структуру выбранного объекта и включает набор методов, каждый из которых отвечает за определенный этап создания конечного документа;
- Node – это класс, который содержит полный набор параметров объекта электронной структуры продукта и методы, применяемые для мониторинга ошибок в нем. Этот класс также реализует стандартный интерфейс языка Java Comparable, который позволяет классифицировать объекты по их идентификаторам. Объект класса Node может ссылаться как на своего родителя, так и на потомков данного класса.

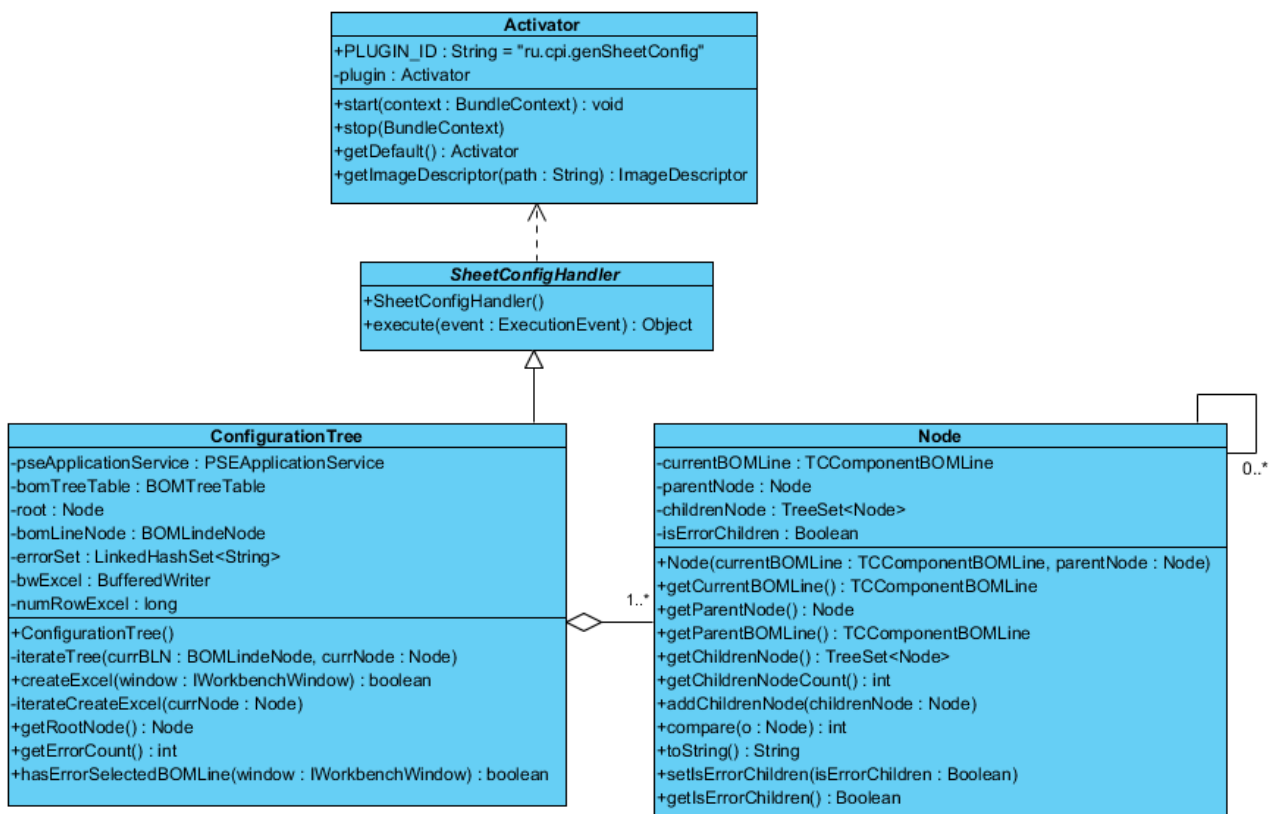


Рис. 3. Диаграмма классов.

Автоматизированное создание структуры документа. Формирование структуры документа «Ведомость конфигурации», выполняемое в PLM-системе Teamcenter с помощью разработанного дополнительного программного модуля, автоматизирующего данный процесс, включает в себя следующие этапы:

1. Запуск PLM-системы Teamcenter и авторизация в ней;
2. Запуск приложения «Менеджер структуры»;
3. Выбор пункта «Добавить структуру...» из раздела «Создание структуры» панели меню главного окна приложения «Менеджер структуры» системы Teamcenter;
4. Ввод необходимого количества уровней и элементов на каждом уровне;
5. Вывод информационного сообщения об успешном создании структуры.

Заключение. Результатом разработки является созданный дополнительный программный модуль, автоматизирующий процесс формирования структуры документа в PLM-системе Teamcenter. Он расширяет функциональность системы Teamcenter и позволяет:

- исключить выполнение рутинных операций;
- раскрыть структуру выбранного продукта до необходимого уровня;
- сократить время на создание структуры ведомости конфигурации (при работе с большими по объему структурами);
- уменьшить количество ошибок в структуре ведомости конфигурации при ее создании до нуля.

СПИСОК ЛИТЕРАТУРЫ

1. Стародубов В. А. Управление конфигурацией: задачи, стандарты и реализация // CAD/CAM/CAE Observer. – 2006. – № 28. – С. 30–33.
2. Тороп Д. Н., Терликов В. В. Teamcenter. Начало работы. – М.: ДМК Пресс, 2011. – 280 с.
3. Teamcenter Getting Started with Customization Publication Number PLM00003 – Global Technical Access Center (GTAC) [Электронный ресурс]. – Режим доступа: https://docs.plm.automation.siemens.com/data_services/resources/tc/11.2/PDF/ru_RU/tdocExt/pdf/getting_started_with_customization.pdf (дата обращения 12.09.2020).
4. Teamcenter 11.2 Rich Client Customization Programmer's Guide Publication Number PLM00075 [Электронный ресурс]. – Режим доступа: https://docs.plm.automation.siemens.com/data_services/resources/tc/11.2/PDF/ru_RU/tdocExt/pdf/client_customization_programmers_guide.pdf (дата обращения 12.09.2020).

ФИРСОВА С. А., КАЗАНЦЕВ В. О.

**РАЗРАБОТКА ПРОГРАММНОГО МОДУЛЯ ДЛЯ АВТОМАТИЗАЦИИ
РАБОТЫ ОТДЕЛА КОНТРОЛЯ КАЧЕСТВА CALL-ЦЕНТРА**

Аннотация. В статье описывается разработанный авторами программный модуль, предназначенный для автоматизации работы отдела контроля качества call-центра, деятельность которого связана с проведением социологических исследований. Процесс автоматизации строится на использовании сервиса транскрибации – перевода аудиофайла в текст.

Ключевые слова: транскрибация, call-центр, программное обеспечение call-центров, социологическое исследование, приложение для Windows на платформе WPF, система распознавания речи с закрытым исходным кодом.

FIRSOVA S. A., KAZANTSEV V. O.

**DEVELOPMENT OF SOFTWARE MODULE FOR AUTOMATION
OF CALL CENTER QUALITY CONTROL DEPARTMENT**

Abstract. The article describes a software system developed by the authors and designed to automate the quality control department of the call center, specializing in conducting sociological research. The automation process is based on the transcribing service, i.e. translation of an audio file into text.

Keywords: transcribing, call center, call center software, sociological research, Windows application on WPF platform, closed-source speech recognition system.

Введение. В процессе развития технологий происходит автоматизация и компьютеризация различных областей деятельности. Это влечет за собой изменение подхода к выполнению задач, в частности отказ от труда человека в пользу автоматизированных систем. Такой переход позволяет экономить ресурсы и увеличивает скорость выполнения работы.

В данной работе мы будем говорить об автоматизации работы отдела контроля качества call-центра, занимающегося проведением социологических исследований. Социологические исследования уже давно пользуются большой популярностью как среди государственных органов, так и среди коммерческих организаций. Одним они позволяют понять, по различным критериям и индикаторам, отношение народа к ситуации в стране; другим помогают получить наибольшую выгоду путем улучшения качества оказываемых услуг или реализуемых товаров.

Как и в других сферах, с развитием технологий, в call-центрах появилась возможность автоматизации части выполняемых задач, таких как: учет рабочего времени или расчет эффективности работы сотрудников, автоматическое составление отчета о проведенном исследовании, да и вообще само понятие call-центра в том виде, в котором оно существует сейчас, стало возможным благодаря технологическому прогрессу: интервьюеры больше не сидят около проводных телефонов и не записывают в блокноты ответы респондентов. Напротив, у многих есть возможность работать удаленно на собственном ПК и все действия выполнять внутри автоматизированных систем.

Но не все возможные типы задач в call-центрах подверглись автоматизации. Так, например, качество проведенных анкет до сих пор контролируется людьми. Существует специальный отдел, который решает эту задачу путем прослушивания проведенных анкет и последующей проверкой соответствия между текстом анкеты ответами респондента в аудиозаписи разговора.

Ознакомление с работой call-центра. Постановка задачи автоматизации. При проведении социологического опроса человека, который проводит интервью называют интервьюером, а человека, которого опрашивают – респондентом.

Респонденты могут различаться по своему финансовому положению и социальному статусу, поэтому к каждому нужен свой подход и свое направление в разговоре. Таким образом, грамотная работа сотрудника позволяет задействовать в социологическом исследовании разные слои населения.

При проведении социологического исследования сохраняется аудиозапись разговора интервьюера и респондента. После завершения опроса интервьюер отправляет на сервер заполненную анкету, в которой отражено мнение респондента по всем вопросам, которые включает в себя опрос. Затем сотрудник отдела контроля качества начинает проверять соответствие между ответами респондента в аудиозаписи разговора и текстом анкеты, составленным интервьюером.

Существует много разных типов вопросов в социологических анкетах. Из-за сложностей программной интерпретации речи человека в качестве объекта автоматизации мы будем рассматривать контроль качества анкет, целиком состоящих из закрытых вопросов. Закрытый вопрос подразумевает выбор ответа(ов) респондентом из заранее заданного фиксированного списка.

Ключевой идеей автоматизации работы отдела контроля качества является применение процесса транскрибации – перевода аудиофайла в текст. Транскрибируя полученный в процессе интервью аудиофайл, мы перейдем к работе с текстовым файлом. Путем синтаксического анализа (парсинга) текстового файла мы определим, верно ли сотрудник

call-центра отразил мнение респондента в социологической анкете, проведя таким образом контроль качества работы интервьюера. Подмодуль, который отвечает за парсинг, мы разработаем самостоятельно. А для транскрибации записи интервью (аудиофайла) воспользуемся существующей системой распознавания речи с закрытым исходным кодом.

Рассмотрев самые распространенные системы распознавания речи с закрытым исходным кодом стоит заметить, что по своей библиотеке данных наиболее точным, в контексте процесса транскрибации, следует считать продукт на базе Dragon Mobile SDK. Однако необходимо заметить, что этот инструмент имеет очень сложную систему лицензирования и строгие правила использования данной технологии. Поэтому возникает сложность интеграции Dragon Mobile SDK с пользовательским продуктом.

Систему Microsoft Speech API удобно использовать там, где происходит непосредственная запись голоса и последующая транскрибация. В нашей задаче записывать голос не требуется, поэтому подключение стороннего движка или дополнительных пакетов к создаваемому приложению может уменьшить скорость его работы.

Более правильным для наших целей и задач следует считать использование речевого инструментария Google, который является быстрым за счет больших вычислительных мощностей и легче встраиваемым. От разработчика требуется только сформировать запрос, а затем получить от сервера ответ с искомым текстовым представлением. Также преимуществом распознавания речи от Google стало отсутствие ограничений по количеству запросов в сутки (у многих систем распознавания речи с закрытым исходным кодом есть такое ограничение).

Проектирование программного модуля (ПМ). Определим отношения между типами пользователей ПМ и опишем его на концептуальном уровне. Для этого построим диаграмму вариантов использования (см. рис. 1).

На диаграмме отражены четыре действующих лица: незарегистрированный пользователь, проверяющий, интервьюер и администратор. При этом видно, что при получении авторизационных данных, незарегистрированный пользователь может выступать как в роли проверяющего, так и в роли интервьюера. Роль незарегистрированного пользователя определяет администратор и выдает ему соответствующие авторизационные данные. Некоторые сценарии, а именно: «Осуществление проверки качества», «Просмотр результатов проверки» и «Сохранение отчета о результатах проверки» объединены в блок, так как являются зависимыми.

Самый широкий функционал предоставляется администратору системы. Помимо прецедентов, доступных проверяющему, ему так же доступен сценарий «Администрирование пользователей».

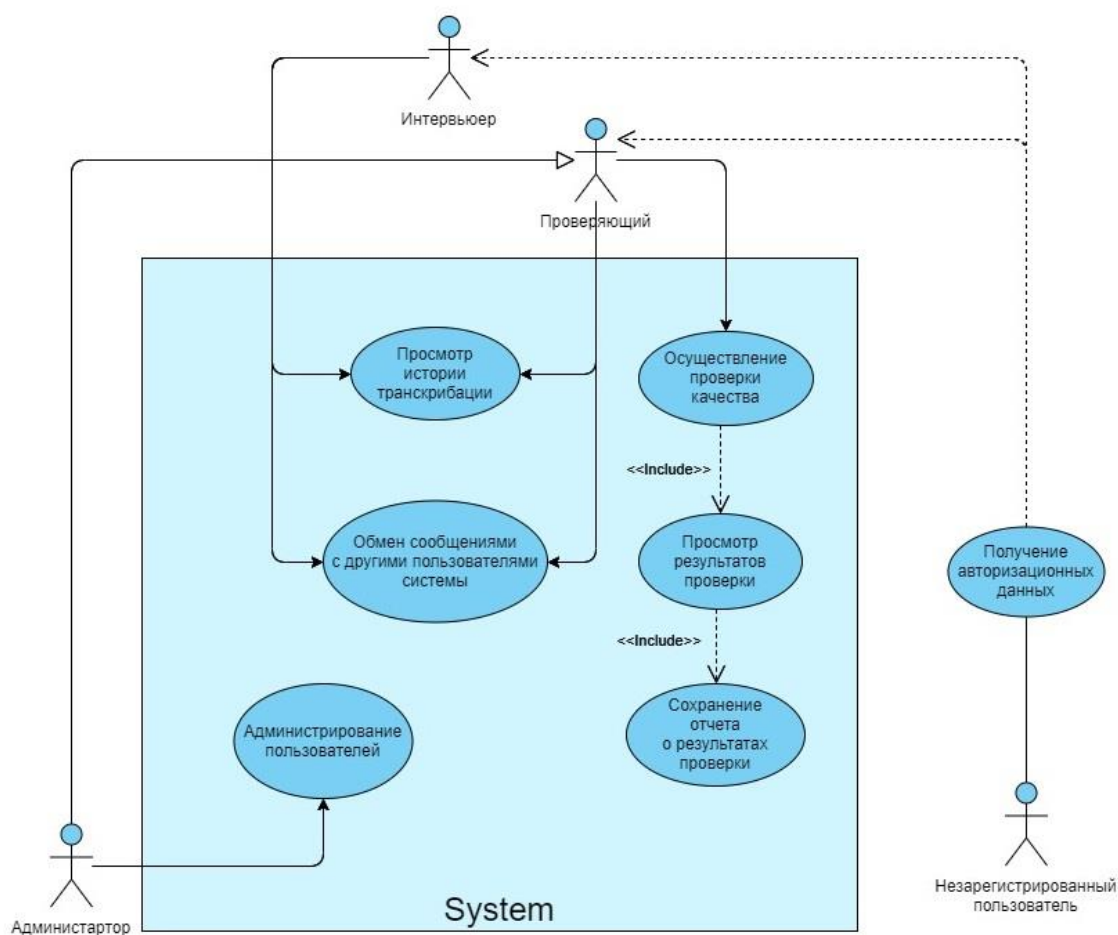


Рис. 1. Диаграмма вариантов использования.

Проверяющий в свою очередь может осуществлять: «Просмотр истории транскрибации», «Обмен сообщениями с другими пользователями системы»; а также имеет доступ к блоку сценариев, в котором проводится проверка качества анкет.

Интервьюеру доступен узкий круг сценариев: «Обмен сообщениями с другими пользователями системы» и «Просмотр истории проверок качества». Это связано с тем, что основной функцией разрабатываемого модуля является проверка качества работы интервьюеров, соответственно, самим интервьюерам в системе доступны исключительно вспомогательные прецеденты.

На следующем этапе проектирования для формирования пользовательского восприятия данных построим диаграмму базы данных (см. рис. 2).

Спроектированная база данных состоит из 5 таблиц:

- таблица Users хранит данные обо всех пользователях системы: об администраторах, о проверяющих и об интервьюерах; данные пользователя содержат: идентификатор сотрудника, логин, пароль, адрес электронной почты, номер телефона, идентификатор региона, ФИО и должность;

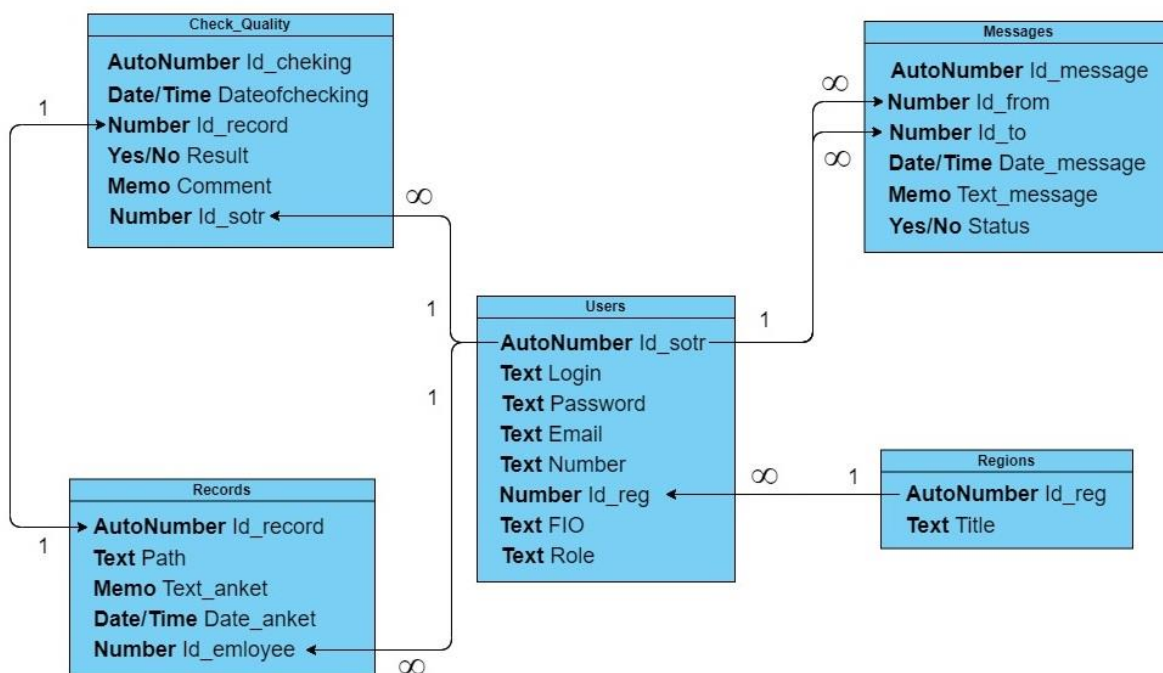


Рис. 2. Диаграмма базы данных.

- таблица Regions содержит идентификаторы и названия регионов;
- таблица Records хранит информацию о проведенных анкетах: идентификатор записи, путь до файла аудиозаписи анкеты, текст анкеты, дата проведения анкеты и идентификатор интервьюера;
- таблица Check_Quality содержит информацию о проверках качества анкет: идентификатор проверки, дата проверки, идентификатор аудиозаписи, результат проверки, комментарий к результату и идентификатор проверяющего;
- таблица Messages хранит информацию о сообщениях между пользователями: идентификатор сообщения, идентификатор адресанта, идентификатор адресата, дату отправки сообщения, текст сообщения и статус о прочтении.

Рассмотрим взаимосвязь модуля с существующими системами автоматизации (см. рис. 3). Dialer Express – это программное обеспечение для учета рабочего времени интервьюеров. X-Lite – это эмулятор телефона с возможностью переадресации и подключения виртуальных номеров. Survey Studio – это интернет сайт, на web – страницах которого заполняют анкеты при социологическом опросе. Все эти системы работают совокупно. Так, например, Dialer Express дает команду X-Lite осуществлять следующий звонок, когда текущий уже обработан, в свою очередь, X-Lite пересылает аудиозапись Dialer Express. Если по окончании разговора интервьюер проставляет в интерфейсе Dialer Express

статус «Успешно», то текст анкеты вместе с аудиозаписью разговора отправляются на сервер.

Разрабатываемый программный модуль должен дополнять уже имеющийся функционал, не нарушая совокупную работу существующих систем.

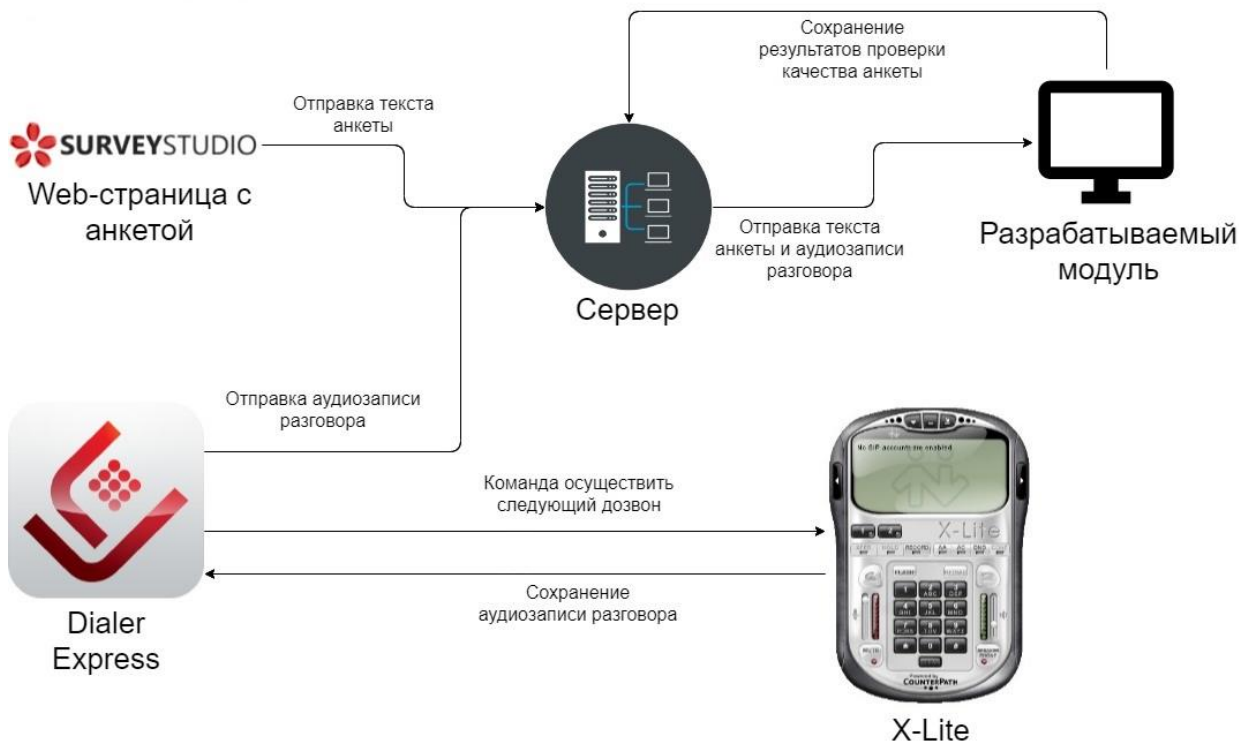


Рис. 3. Взаимодействие модуля с существующими автоматизированными системами.

Как видно из рисунка 3, модуль обменивается данными только с сервером: получает аудиозапись разговора и текст анкеты, а отправляет результаты проверки качества, то есть не изменяет информацию, а добавляет новую, тем самым минимизируя влияние на существующее программное обеспечение.

Опишем взаимосвязь компонентов модуля (см. рис. 4). При проектировании было выделено три компонента: WPF-приложение, сервер системы и сервера Google Voice. Понятно, что при рассмотрении внутреннего устройства приложения, можно выделить больше компонентов, но так или иначе, они будут являться средствами платформы WPF, а при составлении данной схемы учитывались внешние, по отношению друг к другу, сервисы.

Процесс транскрибации происходит на серверах Google Voice. Приложение посылает запрос в виде аудиофайла, переведенного в массив байтов, а в качестве ответа получает текстовую интерпретацию отправленной аудиозаписи. Абстрагироваться от процесса транскрибации позволяет интерфейс встраиванию Google Voice, который реализуется при помощи POST запроса.

WPF-приложение обменивается с сервером данными, а именно: загружает результаты проверок качества, причем как с сервера, при загрузке окна истории проверок, так и на сервер после проведения проверки качества анкет; принимает от сервера и отправляет на сервер сообщения пользователей; сохраняет на сервере информацию о новом пользователе при регистрации и проверяет авторизационные данные при аутентификации; принимает от сервера аудиозапись разговора и текст анкеты.

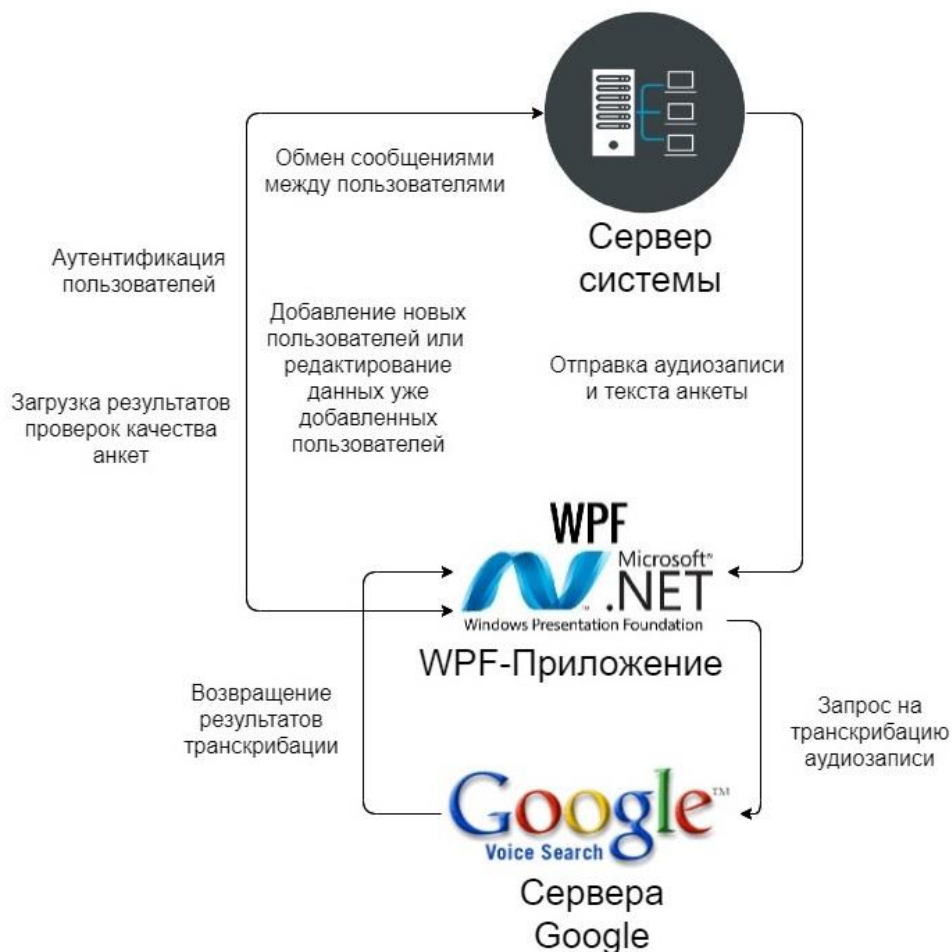


Рис. 4. Взаимосвязь компонентов модуля.

Реализация программной системы. Разрабатываемый программный модуль представляет собой приложение для Windows, созданное на платформе WPF с использованием средств языка C#. В качестве системы управления базами данных используется MS Access.

Рассмотрим основные моменты работы с приложением.

После авторизации пользователь попадает в главное окно программы, где доступный ему функционал зависит от типа учетной записи. Вид главного окна для администратора приведен на рисунке 5. При нажатии на кнопку «Транскрибация» открывается окно проверки

качества анкет, в котором администратор или проверяющий может проверить качество анкет, проведенных любым интервьюером (см. рис. 6). В таблице, содержащей данные об аудиозаписях анкет реализована сортировка записей по любому из столбцов, так же реализовано два механизма проверки: проверка выделенных в таблице анкет или проверка анкет, проведенных за выбранную дату.

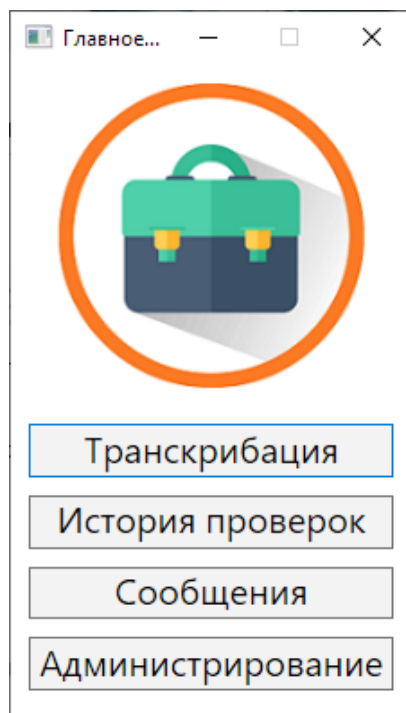


Рис. 5. Главное окно программы.

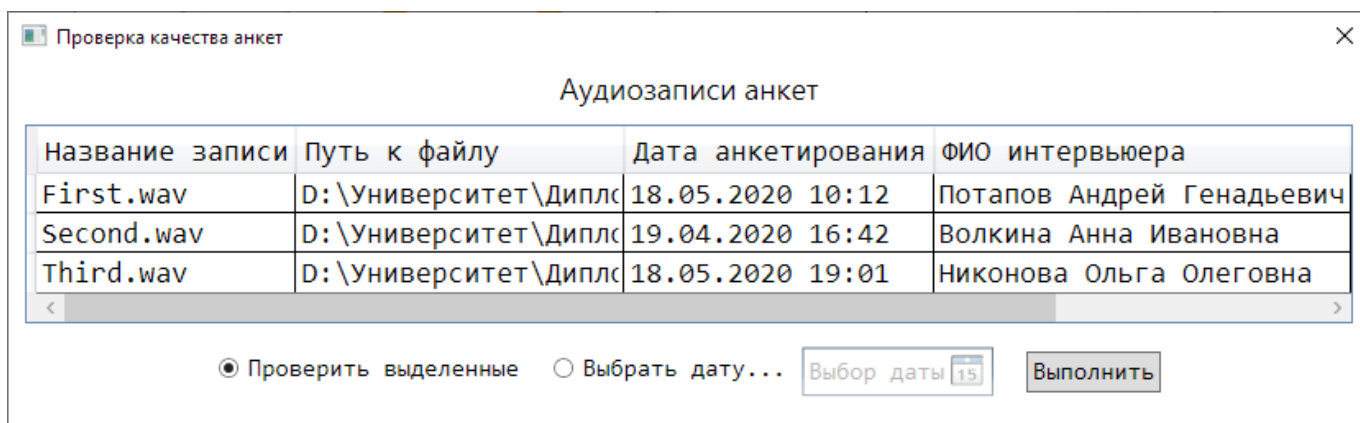
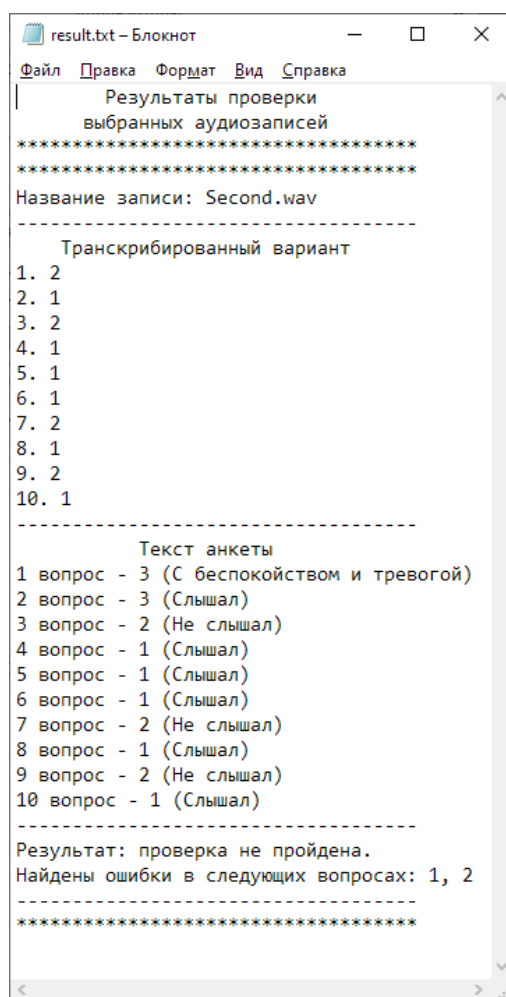


Рис. 6. Окно проверки качества анкет.

Результатом проверки является текстовый файл, содержащий текст анкеты, составленный интервьюером, транскрибированный вариант аудиозаписи разговора и информацию о том, пройдена проверка качества или нет (см. рис. 7).



```
result.txt - Блокнот
Файл  Правка  Формат  Вид  Справка
|
|      Результаты проверки
|      выбранных аудиозаписей
|      *****
|      *****
|      Название записи: Second.wav
|      -----
|      Транскрибированный вариант
|      1. 2
|      2. 1
|      3. 2
|      4. 1
|      5. 1
|      6. 1
|      7. 2
|      8. 1
|      9. 2
|      10. 1
|      -----
|      Текст анкеты
|      1 вопрос - 3 (С беспокойством и тревогой)
|      2 вопрос - 3 (Слышал)
|      3 вопрос - 2 (Не слышал)
|      4 вопрос - 1 (Слышал)
|      5 вопрос - 1 (Слышал)
|      6 вопрос - 1 (Слышал)
|      7 вопрос - 2 (Не слышал)
|      8 вопрос - 1 (Слышал)
|      9 вопрос - 2 (Не слышал)
|      10 вопрос - 1 (Слышал)
|      -----
|      Результат: проверка не пройдена.
|      Найдены ошибки в следующих вопросах: 1, 2
|      -----
|      *****
```

Рис. 7. Результат проверки качества.

Понятно, что в реальных условиях количество проверяемых анкет может составлять десятки и сотни. Поэтому для упрощения анализа и систематизации информации о качестве работы интервьюеров реализовано хранение истории проверок качества анкет. В окне истории проверок качества представлена схожая с окном транскрибации структура хранения данных с возможностью сортировки по всем полям. Помимо этого, файл выгрузки истории включает не только информацию о правильности проведения каждой анкеты, но и общую статистику выбранных проверок качества, которая включает общее кол-во выгруженных проверок и информацию об успешных и неуспешных проверках качества в количественном и процентом соотношении (см. рис. 8).

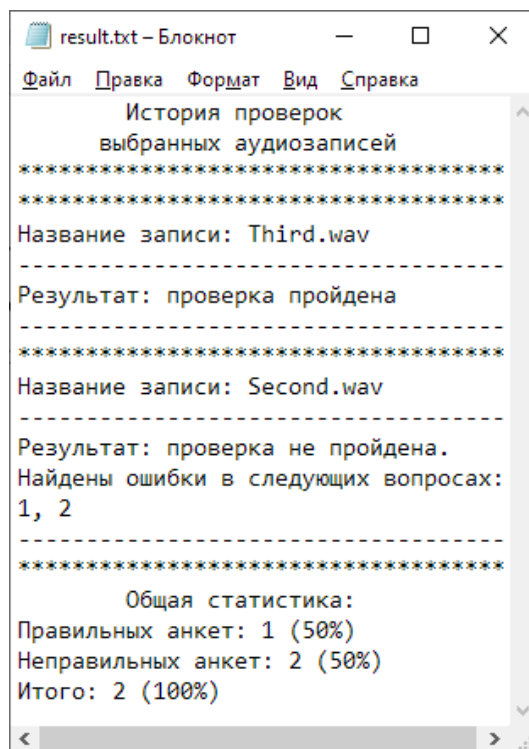


Рис. 8. Результат выгрузки истории.

В окне обмена сообщениями пользователи модуля могут обмениваться сообщениями друг с другом (см. рис. 9). Причем каждый пользователь может написать каждому, выбрав в элементе ListBox интересующего его адресата. Так, например, менеджеры офисов могут обсуждать рабочие моменты с интервьюерами, а также на основе результатов проверки делать замечания по поводу: превышения нормы паузы или обработки, прослушивания автоответчиков, нарушений при заполнении анкеты.

В данном окне реализован механизм уведомления о прочтении сообщений адресатом. За это отвечает последний столбец таблицы с названием «Статус». Удаление сообщений является двухсторонним: сообщения удаляются как у адресанта, так и у адресата.

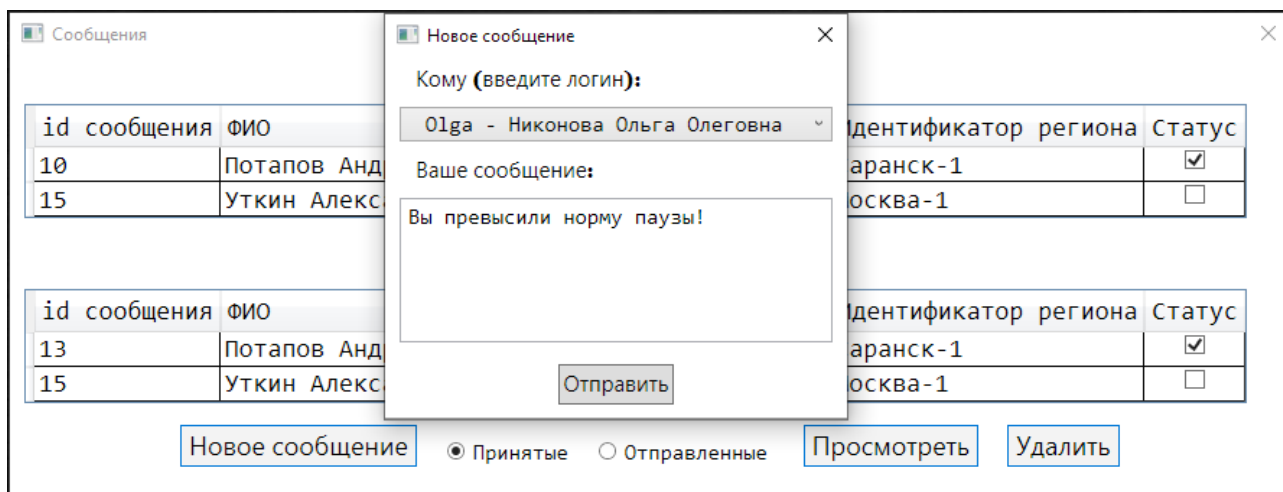


Рис. 9. Окно обмена сообщениями.

Заключение. Внедрение новых решений по автоматизации работы отдела контроля качества call-центра облегчает и ускоряет процесс проверки качества проведенных при социологическом исследовании анкет, а также уменьшает затраты организации на оплату работы сотрудников данного отдела.

Разработанный программный модуль имеет перспективы интеграции при соответствующей доработке под деятельность конкретной организации.

Наличие опций администрирования сотрудников и обмена сообщениями между пользователями приложения позволяет сузить круг используемого организацией программного обеспечения и отказаться от использования локальных мессенджеров.

СПИСОК ЛИТЕРАТУРЫ

1. Ляковский Н. А. О перспективах использования технологий распознавания речи // Глобальные вызовы развития естественных и технических наук: сборник научных трудов по материалам Международной научно-практической конференции. – Белгород, 2018. – С. 194–197.
2. Латышева М. П., Механцева К. Ф. Статистический анализ основного процесса call-центра для повышения качества его услуг // Качество продукции: контроль, управление, повышение, планирование: сборник научных трудов 5-й Международной молодежной научно-практической конференции. – Курск, 2018. – С. 11–14.
3. Золотарёв С. П. Разработка и внедрение программно-информационной системы для call-центра // Информационные технологии в науке и производстве: материалы V Всероссийской молодежной научно-технической конференции. – Омск, 2018. – С. 14–17.
4. Гахова Н. Н., Кизилова О. Г. Задача оптимизации функционирования call-центра // Новая наука: Современное состояние и пути развития. – 2016. – № 6-2. – С. 142–144.
5. Давыдова А. А., Бабенко А. В., Новикова Т. Б. Постановка задачи деятельности call-центра // Современная техника и технологии. – 2016. – № 11-2 (63). – С. 247–252.
6. Accessing Google Speech API / Chrome 11 – mike pultz [Электронный ресурс]. – Режим доступа: <https://mikepultz.com/2011/03/accessing-google-speech-api-chrome-11/> (дата обращения 10.11.2020).

КУЗНЕЦОВА И. С., МАМЕДОВА Т. Ф.

ЗАДАЧА ОБ ОПТИМАЛЬНОМ ЭКОНОМИЧЕСКОМ РОСТЕ

Аннотация. В статье рассматривается вопрос оптимизации уровня потребления для экономического объекта на примере конкретного предприятия. На основе принципа максимума решена задача оптимизации сбережений. Получены оптимальные траектории потребления и капиталовооруженности.

Ключевые слова: экономический рост, оптимизация, потребление, производственная функция, капиталовооруженность.

KUZNETSOVA I. S., MAMEDOVA T. F.

THE PROBLEM OF OPTIMAL ECONOMIC GROWTH

Abstract. The problem of optimizing the level of consumption for an economic object on the example of a specific enterprise is considered in the article. On the basis of the maximum principle, the problem of savings optimization is solved. The optimal trajectories of consumption and capital-labor ratio are obtained.

Keywords: economic growth, optimization, consumption, production function, capital-labor ratio.

Актуальность проблемы. Математики совместно с экономистами занимаются изучением факторов, которые влияют на устойчивый экономический рост, способствующий улучшению благосостояния населения. В период переломных моментов поднимаются следующие вопросы: возможно ли моделировать направления экономического роста, насколько истинны построенные на основе таких моделей прогнозы, и в какой степени оправданы эти прогнозы в современных условиях.

Инновационные технологии, научно-технический прогресс, автоматизация производства являются значительным фундаментом высоких темпов роста экономики в длительном периоде. Таким образом, становится актуальным использование моделей экономического роста.

Актуальность проблемы обусловлена тем, что в современной экономике часто используются оптимизационные методы, составляющие основу математического программирования. Грамотное потребление ресурсов имеет огромное значение, поскольку их оптимизация может увеличить прибыль [1].

Постановка задачи. Задача об оптимальном экономическом росте представляет собой задачу о выборе оптимальных траекторий $c(t)$, $k(t)$, для которых благосостояние максимально. Она имеет следующий вид [2]:

$$J(c) = \int_{t_1}^{t_2} e^{-\delta(t-t_1)} U(c(t)) dt \rightarrow \max,$$

$$\frac{dk}{dt} = f(k) - \lambda k - c,$$

$$k(t_1) = k_1,$$

$$0 \leq c(t) \leq f(k),$$

где k_1, λ, δ – заданные числа, а $f(k), U(c)$ – заданные функции.

Решением задачи оптимизации является траектория потребления на единицу эффективного труда $\{c(t)\}$ и траектория капиталовооруженности эффективного труда $\{k(t)\}$, вдоль которых функционал $J(c)$ достигает максимума. Решение зависит от двух функций: функции полезности U и производственной функции $k(t)$, от трех параметров: нормы дисконтирования, нормы амортизации, темпа рабочей силы и от начального значения капиталовооруженности рабочего. Задача оптимизации может решаться на основе принципа максимума Понтрягина.

Требуется оптимизировать уровень потребления сельскохозяйственного предприятия «Продовольствие», которое занимается выработкой молочной продукции. Имеются данные за 2004 – 2018 гг. о выручке от продажи сельскохозяйственной продукции, о среднегодовой стоимости основных средств и о численности персонала.

Описание алгоритма решения задачи.

- 1) Построение производственной функции $f(k)$ на основе данных о динамике выпуска продукции и производственных факторах (капитал, труд);
- 2) Выбор функции полезности $U(c)$;
- 3) Вычисление точки равновесия (k^*, c^*) , соответствующей сбалансированному росту;
- 4) Решение системы двух дифференциальных уравнений;
- 5) Построение траекторий потребления и капиталовооруженности;
- 6) Вычисление значения функционала для нахождения оптимального уровня потребления.

Решение задачи. На основе исходных данных таблицы 1 построим производственную функцию, которая характеризует объем выпуска продукции, величина которого зависит от затрат факторов производства.

Экономические показатели предприятия

Год	Выручка от продажи сельскохозяйственной продукции, тыс. руб., Y	Среднегодовая стоимость основных средств, тыс. руб., K	Численность персонала, L
2004	843750	195000	533
2005	860625	210000	540
2006	904500	240000	530
2007	968625	270000	597
2008	995625	285000	543
2009	1046250	288000	535
2010	1113750	300000	570
2011	1181250	330000	600
2012	1215000	360000	615
2013	12048750	375000	630
2014	1282500	382500	628
2015	1333125	390000	637
2016	1390601	409590	676
2017	1380500	410000	650
2018	1367218	405967	670

Производственная функция $y = f(k)$ должна удовлетворять условиям [3; 4]:

$$f(k) > 0, f'(k) > 0, f''(k) < 0 \forall k > 0,$$

$$f(0) = 0, f'(0) = \infty.$$

В результате обработки данных, используя программу Microsoft Excel, был построен график производственной функции за данный промежуток времени. С помощью аппроксимации найдена математическую модель, которая наилучшим образом описывает наблюдаемые значения. Найденная функция имеет вид:

$$y = 1506,1k^{0,123}. \quad (1)$$

Эта функция удовлетворяет условиям:

$$\begin{aligned}
f(k) &= 1506,1k^{0,123} > 0, \\
f'(k) &= 185,2503k^{-0,877} > 0, \\
f''(k) &= -162,4645k^{-1,877} < 0, \\
f(0) &= 1506,1 * 0^{0,123} = 0, \\
f'(0) &= 185,2503 * 0^{-0,877} = \infty.
\end{aligned}$$

Примем, что функция полезности имеет вид:

$$U(c) = \sqrt[3]{c}, \quad (2)$$

где $c(t)$ – потребление, которое приходится на одну рабочую единицу.

Данная функция так же удовлетворяет требуемым условиям:

$$\begin{aligned}
U(c) &= \sqrt[3]{c} > 0, \\
U'(c) &= \frac{1}{3\sqrt[3]{c^2}} > 0, \\
U''(c) &= -\frac{2}{9\sqrt[3]{c^5}} < 0, \\
U'(0) &= \frac{1}{3\sqrt[3]{0}} = \infty.
\end{aligned}$$

Начальные условия: $t_1 = 0, t_2 = 14$. Коэффициент амортизации капитала $\mu = 0,06$.

Годовой темп прироста числа занятых n составляет 0,016. Норма дисконтирования $\delta = 0,5$.

Найдем стационарные траектории $c^* = \text{const}, k^* = \text{const}$.

k^* получим из уравнения: $f'(k^*) = \mu + n + \delta$:

$$\begin{aligned}
185,2503k^{*-0,877} &= 0,06 + 0,016 + 0,5, \\
k^* &= 722,76.
\end{aligned}$$

Вычислим $c^* = f(k^*) - (\mu + n)k^*$.

$$\begin{aligned}
c^* &= 1506,1 * 722,76^{0,123} - (0,06 + 0,016) * 722,76 \\
c^* &= 3329,69.
\end{aligned}$$

Величины k^*, c^* удовлетворяют неравенству: $0 < c^* < f(k^*)$. Точка равновесия (k^*, c^*) является траекторией сбалансированного роста.

Вычислим значение функционала $J(c)$:

$$\begin{aligned}
J(c) &= \int_0^{14} e^{-0,5(t-0)} * \sqrt[3]{3329,69} dt = 14,9 \int_0^{14} e^{-0,5t} dt = \\
&= 14,9 \left(-\frac{1}{0,5} e^{-7} + \frac{1}{0,5} e^0 \right) = 29,84.
\end{aligned}$$

Полагая

$$\mu = 0,06, n = 0,016, \delta = 0,5,$$

методом Рунге-Кутты 4 порядка вычислим решение системы дифференциальных уравнений

$$\begin{cases} \dot{c} = \frac{1}{\sigma(c)}(f'(k) - \mu - n - \delta)c, \\ \dot{k} = f(k) - (\mu + n)k - c. \end{cases} \quad (3)$$

Показатели потребления и капиталовооруженности имеют тенденцию роста, но потребление растет значительно быстрее, чем капиталовооруженность. Значение функционала $J(c)$ при этом равно 29,84.

График траектории потребления на рисунке 1 изображен линией зеленого цвета, график капиталовооруженности – линией синего цвета.

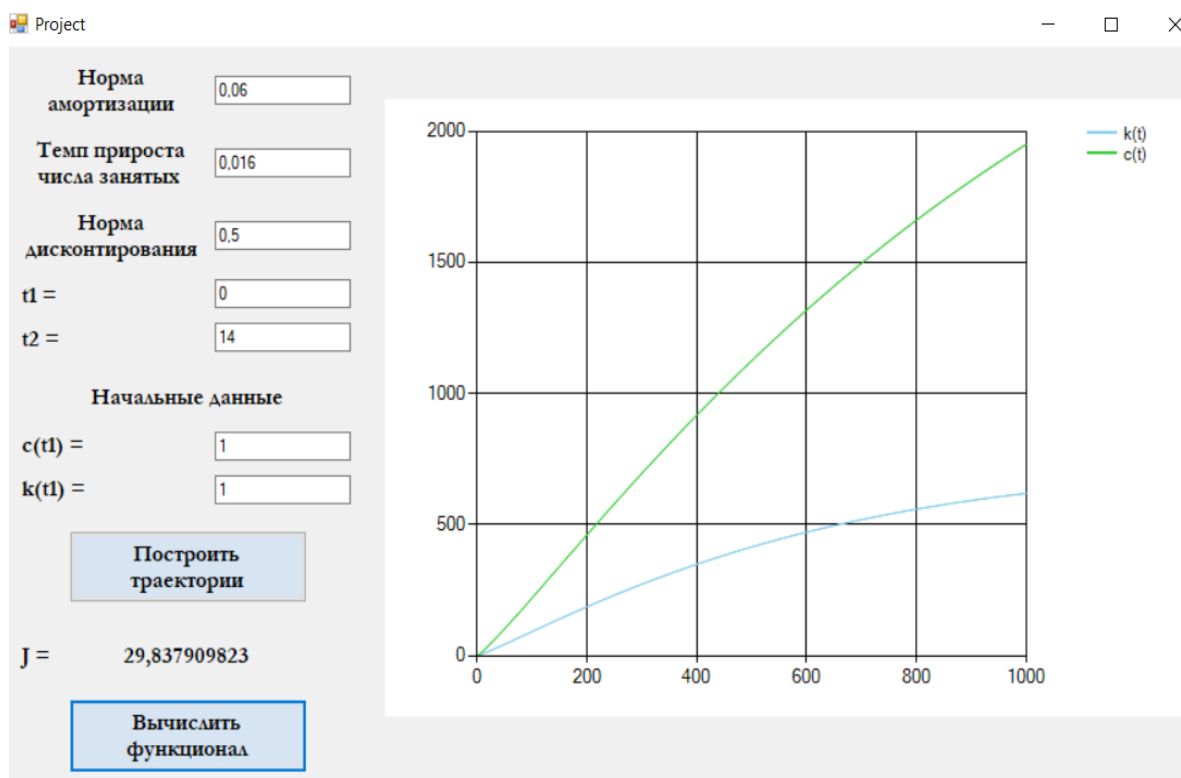


Рис. 1. Траектории потребления и капиталовооружённости.

Увеличим коэффициент амортизации μ до 0,3 и посмотрим на поведение модели. Решим систему уравнений с параметрами:

$$\mu = 0,03, n = 0,016, \delta = 0,5.$$

Полученные результаты показывают, увеличение коэффициента амортизации капитала незначительно замедляет рост потребления и капиталовооруженности. Только в случае уменьшения уровня инфляции можно добиться роста капиталовооруженности. Значение функционала в данном случае равно 29,04.

Теперь рассмотрим модель при уменьшении нормы дисконтирования. Начальные условия: $t_1 = 0, t_2 = 14, \mu = 0,06, n = 0,016, \delta = 0,03$.

Полученные результаты показывают, что в данной ситуации наблюдается рост как траектории потребления на единицу эффективного труда, так и траектории капиталовооруженности эффективного труда. Следовательно, уменьшение нормы амортизации увеличивает скорость роста обеих величин, что говорит о важности долгосрочного планирования. При данных параметрах значение функционала равно 180,14.

Вывод. Таким образом, в результате проделанной работы поставленная цель была достигнута, а именно: разработана экономико-математическая модель и выполнена оптимизация уровня потребления для экономического объекта. На основе принципа максимума решена задача оптимизации сбережений. Получены оптимальные траектории потребления и капиталовооруженности. Анализ полученных результатов позволяет сделать вывод о том, что максимизация потребления происходит в том случае, когда норма дисконтирования будет минимальной.

СПИСОК ЛИТЕРАТУРЫ

1. Андреева Е. А., Цирулева В. М. Вариационное исчисление и методы оптимизации: учеб. пособие. – Тверь: Твер. гос. ун-т, 2001. – 576 с.
2. Ашманов С. А. Введение в математическую экономику: учеб. пособие. – М.: Наука, гл. ред. физ.-мат. лит. 1984. – 296 с.
3. Мамедова Т. Ф., Каледин О. Е., Шабанова В. Г., Кирейчева Е. Ю. Математическая модель оптимизации управления хозяйственной деятельностью одного производственного предприятия // Аналитические и численные методы моделирования естественно-научных и социальных проблем : сб. ст. X Междунар. науч.-техн. конф. (г. Пенза 28–30 октября 2015 г.) / под ред. И. В. Бойкова. – Пенза: ПГУ, 2016. – С. 125–130.
4. Шабанова В. Г., Василькин Н. В., Поверинов А. И. О методике прогнозирования роста капитала предприятия // Математические методы и информационные технологии управления в науке, образовании и правоохранительной сфере: Сборник материалов Всероссийской научно-технической конференции / Московский государственный технический университет имени Н. Э. Баумана, Академия ФСИН России. – Рязань: Рязанский государственный университет имени С.А. Есенина. – 2017. – С. 51–55.

ТИТКОВ М. Д., МАМЕДОВА Т. Ф.
МАТЕМАТИЧЕСКАЯ МОДЕЛЬ РАСПРОСТРАНЕНИЯ
ИНФЕКЦИОННЫХ ЗАБОЛЕВАНИЙ

Аннотация. В статье представлена математическая модель распространения инфекционных заболеваний. Получена картина динамики распространения заболеваний для Китая и Италии. Сделано прогнозирование эпидемии с учетом индивидуальных данных для каждой из стран.

Ключевые слова: математическая модель, общество, эпидемии, sir-модель, инфекционное заболевание, численное моделирование, дифференциальное уравнение.

TITKOV M. D., MAMEDOVA T. F.
MATHEMATICAL MODEL FOR SPREAD OF INFECTIOUS DISEASES

Abstract. The article presents a mathematical model for the spread of infectious diseases. The pictures of the dynamics of the spread of diseases for China and Italy were obtained. Forecasting of the epidemic was made, taking into account the data for each country.

Keywords: mathematical model, society, epidemics, sir-model, infectious disease, numerical modeling, differential equation.

Актуальность проблемы. COVID-19 – потенциально тяжелая острая респираторная инфекция. Методики профилактики и лечения пока отсутствуют, как и противовирусные лекарственные средства. В связи с эпидемией, в области здравоохранения, ВОЗ объявила чрезвычайную ситуацию международного значения. В связи с этим, необходимы исследования и прогнозы, не без помощи математического аппарата. Поскольку эпидемия происходит в данный момент, необходимо понять общую динамику ее поведения, что показывает актуальность этой работы [1; 2; 3].

С точки зрения математики, эпидемический процесс и его динамика могут быть рассмотрены несколькими способами, например, системой дифференциальных уравнений. Решения этой системы будут характеризовать динамику изменения численности в рассматриваемых подгруппах популяции.

Постановка задачи. Необходимо построить математическую модель наиболее подходящую для рассмотрения случая с COVID-19. Также появляется возможность оценить влияние дополнительно рассмотренных аспектов течения болезни. Для этого необходимо подробнее рассмотреть эпидемиологию данного заболевания.

От тел умерших людей заразиться COVID-19 практически невозможно – для этого нужно иметь дело с биологическими жидкостями умершего, так как болезнь передается

воздушно-капельным путем. Также стоит брать во внимание тот факт, что органы власти и религиозные организации корректируют условия захоронения для того, чтобы максимально ограничить распространения вируса. Поэтому можно не брать во внимание группу людей, заразившихся через не похороненные тела, поскольку их процент будет очень мал [4-6].

Также мал процент людей, которые заразились COVID-19 повторно после выздоровления. В настоящей математической модели не будет учитываться повторное заражение, поскольку этот вопрос недостаточно исследован.

Вопрос о естественном иммунитете на данный момент также исследован недостаточно. ВОЗ не выявила зависимость между наличием антител к COVID-19 и наличием естественного иммунитета. Однако в данной математической модели будет предполагаться, что небольшой процент людей в исследуемой области имеет естественный иммунитет к заболеванию, поскольку разные новостные источники сообщают о его наличии.

Разделим популяцию на шесть групп:

1. Группа S – восприимчивые индивидуумы;
2. Группа E – индивидуумы, имеющие вирус в инкубационном периоде;
3. Группа I – инфицированные индивидуумы;
4. Группа H – госпитализированные индивидуумы;
5. Группа D – умершие индивидуумы;
6. Группа R – иммунные (выздоровевшие) индивидуумы.

Рассматриваемая модель будет учитывать рождаемость и смертность. Как в случае модели SIR, $1/\mu$ можно интерпретировать как коэффициент средней продолжительности жизни. Рассмотрим:

1. $\mu_1 = k\mu$;
2. $\mu_2 = (1 - k)\mu$,

где k – коэффициент доли индивидуумов с естественным иммунитетом. За счет коэффициентов μ_1 и μ_2 , группы S и R могут пополняться за счет рождаемости, тогда как группа D не может на это влиять. Смертность рассматривается в группах S, E, I, H и R . В начальный момент времени, небольшая часть индивидуумов будет находиться в классе R , как имеющие естественный иммунитет, в дальнейшем эти kN человек так и будут оставаться в группе иммунных индивидуумов.

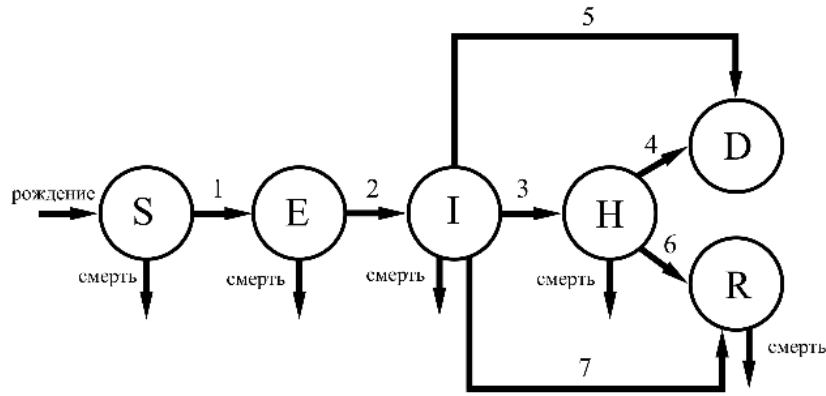


Рис. 1. Схема возможных переходов индивидуумов из группы в группу модели SEIHDR, где S – группа восприимчивых, E – группа латентных I – инфицированных, H – группа госпитализированных, D – группа умерших, R – группа невосприимчивых индивидуумов.

Рассмотрим скорости перехода подробнее:

1. $\beta_I SI/N$ описывает изменение размера популяции группы S за счет заражения индивидуумами из группы I , $\beta_H SH/N$ описывает изменения размера популяции группы S за счет заражения индивидуумами из группы;
2. αE описывает изменение размера популяции группы E за счет окончания инкубационного периода вируса у индивидуума и перехода заболевания в инфекционную стадию;
3. $\gamma_H \theta_1 I$ описывает изменение размера популяции группы I за счет госпитализации заразившихся индивидуумов;
4. $\gamma_{DH} \delta_2 H$ описывает изменение размера популяции группы H за счет смерти индивидуумов в медицинских учреждениях;
5. $\gamma_D (1 - \theta_1) \delta_1 I$ описывает изменение размера популяции группы I за счет смерти индивидуумов вследствие болезни вне медицинских учреждений;
6. $\gamma_{IH} (1 - \delta_2) H$ описывает изменение размера популяции группы H за счет выздоровления индивидуумов в медицинских учреждениях;
7. $\gamma_I (1 - \theta_1) (1 - \delta_1) I$ описывает изменение размера популяции группы I за счет выздоровления индивидуумов вне медицинских учреждений.

Рождаемость в группах S и R описывается как $\mu_2(N - D(t))$ и $\mu_1(N - D(t))$ соответственно. Смертность описывается как $\mu E(t)$, $\mu I(t)$, $\mu H(t)$, $\mu R(t)$ в группах E , I , H и R соответственно.

Общий размер популяции N должен остаться неизменным, т.е должно соблюдаться правило: $S(t) + E(t) + I(t) + H(t) + D(t) + R(t) = \text{const} = N$.

Тогда, учитывая все вышеописанное, SEIHDR-модель описывается следующим набором дифференциальных уравнений:

$$\begin{aligned}\frac{dS(t)}{dt} &= -\frac{1}{N} (\beta_I S(t)I(t) + \beta_H S(t)H(t)) + \mu_2(N - D(t)) - \mu S(t), \\ \frac{dE(t)}{dt} &= \frac{1}{N} (\beta_I S(t)I(t) + \beta_H S(t)H(t)) - \alpha E(t) - \mu E(t), \\ \frac{dI(t)}{dt} &= \alpha E(t) - (\gamma_H \theta_1 + \gamma_I(1 - \theta_1)(1 - \delta_1) + \gamma_D(1 - \theta_1)\delta_1)I(t) - \mu I(t), \\ \frac{dH(t)}{dt} &= \gamma_H \theta_1 I(t) - (\gamma_{DH}\delta_2 + \gamma_{IH}(1 - \delta_2))H(t) - \mu H(t), \\ \frac{dD(t)}{dt} &= \gamma_D(1 - \theta_1)\delta_1 I(t) + \gamma_{DH}\delta_2 H(t), \\ \frac{dR(t)}{dt} &= \gamma_I(1 - \theta_1)(1 - \delta_1)I(t) + \gamma_{IH}(1 - \delta_2)H(t) + \mu_1(N - D(t)) - \mu R(t),\end{aligned}$$

где $\beta_I, \beta_H, \alpha, \gamma_H, \gamma_{DH}, \gamma_I, \gamma_D, \gamma_{IH}, \theta_1, \delta_1, \delta_2$ определяются аналогично вышеописанным моделям, а начальные условия следующие:

$$\begin{aligned}S(0) &= S_0 \geq 0, \\ E(0) &= E_0 \geq 0, \\ I(0) &= I_0 \geq 0, \\ H(0) &= H_0 \geq 0, \\ D(0) &= D_0 \geq 0, \\ R(0) &= kN + R_0 \geq 0, \\ t &= 0.\end{aligned}$$

Численное моделирование. Для параметризации вышеописанных математических моделей распространения COVID-19, использованы данные по двум странам – Китай и Италия (см. табл. 1). Результаты получены с помощью пакета прикладных программ MATLAB, где будет применяться функция ode45, позволяющая численное интегрирование систем обыкновенных дифференциальных уравнений, используя при этом, формулы Рунге-Кутты пятого и четвертого порядка.

Исходные данные по эпидемиям

Параметр	Китай	Италия
Коэффициент контакта в обществе, β_I	0.328	0.393
Коэффициент контакта в госпитале, β_H	0.00008	0.05699
Инкубационный период, α	8.5 дней	8.5 дней
Время до госпитализации, γ_H	1.25 дней	2.5 дней
Время от госпитализации до смерти, γ_{DH}	9.4 дней	9.4 дней
Инфекционный период, для выздоровевших, γ_I	18.75 дней	18.75 дней
Инфекционный период, для умерших, γ_D	10.65 дней	11.90 дней
Время от госпитализации до выздоровления, γ_{IH}	20 дней	21.25 дней
Доля летальных случаев	3.44%	7.27 %
δ_1	0.0198	0.0464
δ_2	0.0164	0.0335
Доля случаев госпитализации	65%	51.73%
θ_1	0.111	0.126
Средняя продолжительность жизни, μ	74.7 лет	81.8 лет
Население, N	1.409.342.854	60.015.664

Для Китая использовалась статистика трехнедельного временного промежутка – с 02.02.2020 по 23.02.2020 – именно в это время был зарегистрирован пик заболевания [7; 8]. Аналогично для Италии – с 16.03.2020 по 05.04.2020 – зарегистрирован пик заболевания [9; 10].

Китай. Параметры модели следующие: $\beta_I = 0.328$, $\beta_H = 0.00008$, $\alpha = 0.1176$, $\gamma_I = 0.0533$, $\gamma_D = 0.0939$, $\gamma_H = 0.8$, $\gamma_{DH} = 0.1063$, $\gamma_{IH} = 0.05$, $t = [0,183]$, $N = 1409342854$, $\delta_1 = 0.0198$, $\delta_2 = 0.0164$, $\theta_1 = 0.111$, $\mu = 0.000036$. Так как вопрос о естественном иммунитете еще недостаточно хорошо изучен, то возьмем параметр $k = 0.05$.

Далее, на графиках: синий цвет обозначает группу S , красный – группу E , желтый – группу I , фиолетовый – группу H , зеленый – группу D , голубой – группу R . Ось ординат означает размер популяции, а ось абсцисс – время в днях.

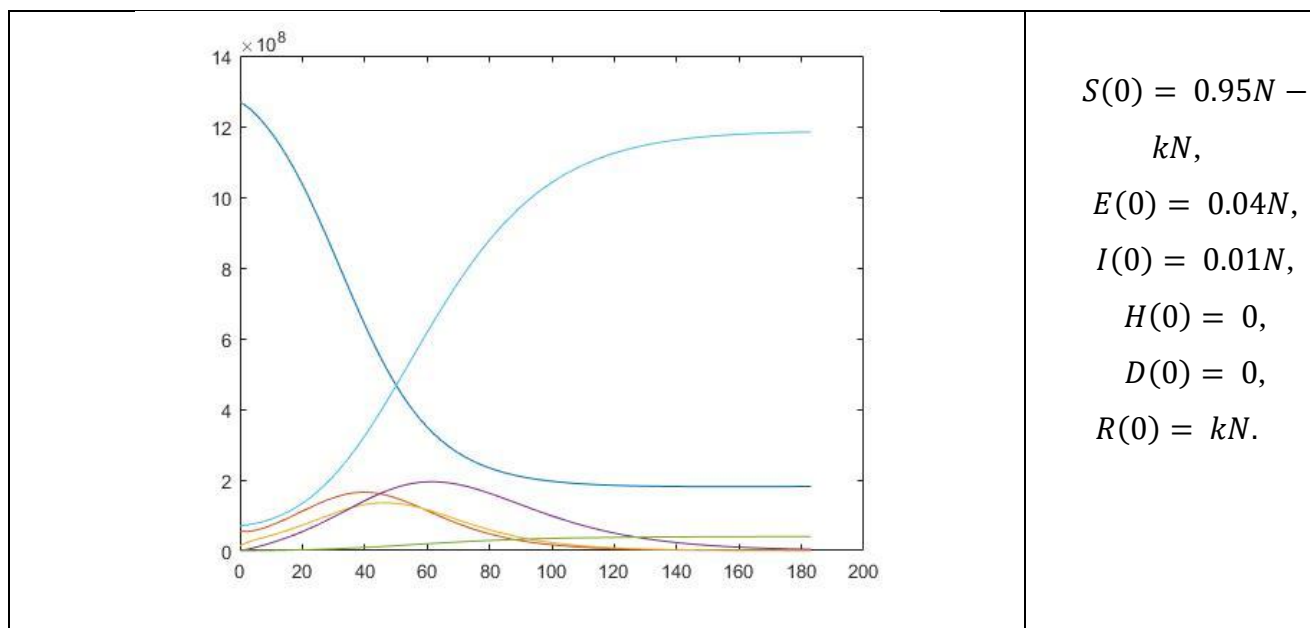


Рис. 2. График распространения COVID-19 для Китая. Синий цвет обозначает группу S , красный – группу E , желтый – группу I , фиолетовый – группу H , зеленый – группу D , голубой – группу R .
Ось ординат означает размер популяции, а ось абсцисс – время в днях.

Италия. Параметры модели следующие: $\beta_I = 0.393$, $\beta_H = 0.0570$, $\alpha = 0.1176$, $\gamma_I = 0.0533$, $\gamma_D = 0.0840$, $\gamma_H = 0.4$, $\gamma_{DH} = 0.1063$, $\gamma_{IH} = 0.0471$, $t = [0, 183]$, $N = 60015664$, $\delta_1 = 0.0464$, $\delta_2 = 0.0335$, $\theta_1 = 0.126$, $\mu = 0.000033$. Так как вопрос о естественном иммунитете еще недостаточно хорошо изучен, то возьмем параметр $k = 0.05$.

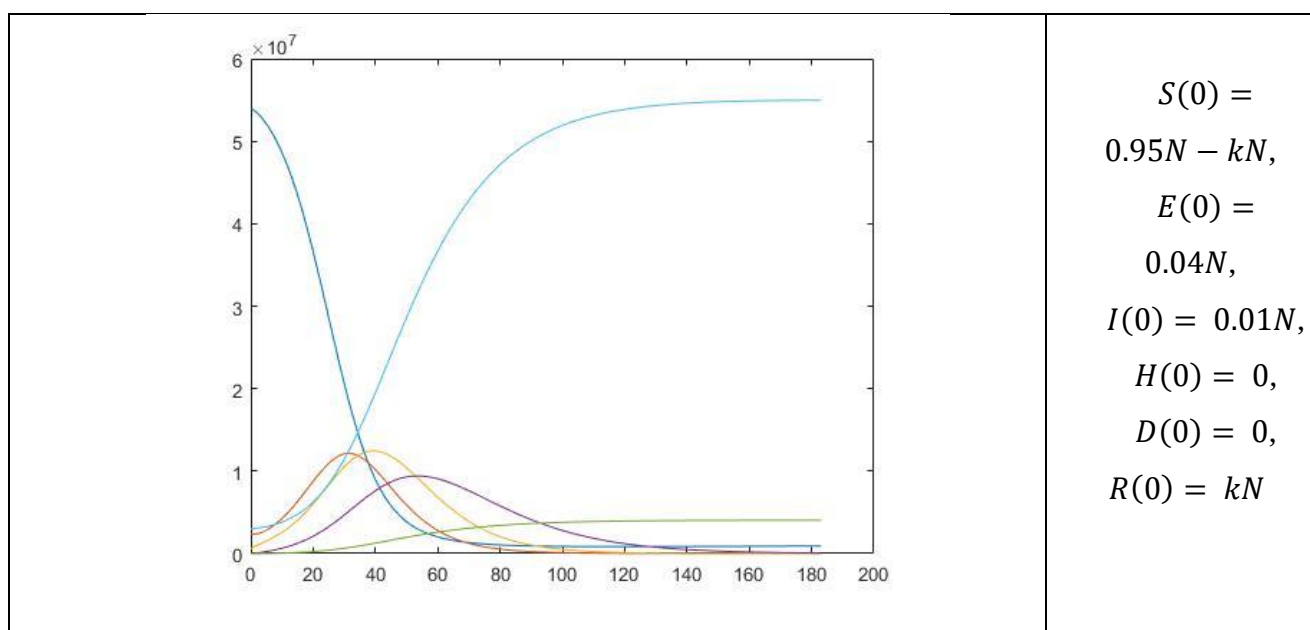


Рис. 3. График распространения COVID-19 для Италии. Синий цвет обозначает группу S , красный – группу E , желтый – группу I , фиолетовый – группу H , зеленый – группу D , голубой – группу R .

На основе полученных результатов и тех знаний, что мы имеем относительно Китая и Италии на данный момент, можно сделать вывод о том, что модель, как инструмент прогноза, довольно точно описывает динамику болезни в том или ином регионе.

Данные, взятые с официальных источников, в каком-то смысле учитывают социальную структуру, поэтому можно заметить, что болезнь, при соблюдении карантинного режима и большого количества госпитализаций, быстро заканчивается при относительно небольших потерях среди населения.

Также можно заметить, что индивидуальные данные для каждой из стран отражают индивидуальную динамику изменения численности групп индивидуумов, при этом точность прогнозирования, в обоих случаях, примерно одинаковая.

СПИСОК ЛИТЕРАТУРЫ

1. Kermack W. O., McKendrick A. G. A Contribution to the Mathematical Theory of Epidemics // Proceedings of the Royal Society. – 1927. – Vol. 115, No. A771. – P. 700-721.
2. Edelstein-Keshet L. Mathematical Models in Biology // Society for Industrial and Applied Mathematics. – 2005. – P. 242–254.
3. Hethcote H. W. The Mathematics of Infectious Diseases // SIAM Review. – 2000. – Vol. 42, Iss. 4. – P. 599–653.
4. Legrand J., Grais P. Y., Boelle A. J., Valleron A. Understanding the dynamics of Ebola epidemics // Epidemiology and Infection. – 2007. – Vol. 135, Iss. 04. – P. 610–621.
5. WHO Coronavirus Disease (COVID-19) Dashboard [Электронный ресурс]. – Режим доступа: <https://covid19.who.int/> (дата обращения 25.04.2020).
6. Coronavirus disease 2019 [Электронный ресурс]. – Режим доступа: <https://www.who.int/emergencies/diseases/novel-coronavirus-2019> (дата обращения 25.04.2020).
7. National Health Commission of the People's Republic of China [Электронный ресурс] – Режим доступа: http://www.nhc.gov.cn/xcs/yqtb/list_gzbd.shtml (дата обращения 25.04.2020).
8. Ministero della Salute [Электронный ресурс]. – Режим доступа: <http://www.salute.gov.it/portale/home.html> (дата обращения 25.04.2020).
9. Девяткин А. В., Девяткин А. А. Новая коронавирусная инфекция – covid-19. Вопросы происхождения, тропности возбудителя, путей передачи инфекции, лабораторной диагностики и специфической терапии // Кремлевская медицина. Клинический вестник. - 2020. – № 2. – С. 5–13.

11. Coronavirus COVID-19 (2019-nCov) [Электронный ресурс]. – Режим доступа: <https://gisanddata.maps.arcgis.com/apps/opsdashboard/index.html#/bda7594740fd40299423467b48e9ecf6> (дата обращения 25.04.2020).

ГОЛИКОВА И. В., ПОЧИНКА О. В.

НАДСТРОЙКИ НАД ГРУБЫМИ ПРЕОБРАЗОВАНИЯМИ ОКРУЖНОСТИ¹

Аннотация. В первом разделе данной статьи даны основополагающие определения рассмотренной темы, второй раздел посвящен классификации грубых преобразований окружности, последний – надстройкам над модельными преобразованиями окружности. Основным результатом исследования является теорема 3.1 об эквивалентности надстроек над модельными диффеоморфизмами, сохраняющими и меняющими ориентацию, а также утверждение о том, что из эквивалентности надстроек над меняющими ориентацию диффеоморфизмами следует сопряженность этих диффеоморфизмов.

Ключевые слова: градиентно-подобные диффеоморфизмы, топологическая сопряженность, топологическая эквивалентность, надстройки, грубые преобразования окружности.

GOLIKOVA I. V., POCHINKA O. V.

SUSPENSION OVER ROUGH CIRCLE TRANSFORMATIONS

Abstract. The first section of this article presents the basic definitions of the topic, the second chapter focuses on structurally stable circle transformations, and the last chapter is dedicated to suspensions on model transformation of $\mathbb{S}^1$. The basic results of the research are theorem 3.1 about equivalence of suspensions on model diffeomorphisms and statement 4.1 that equivalence of suspensions on orientation-reversing diffeomorphisms implies the diffeomorphisms conjugacy.

Keywords: gradient-like diffeomorphisms, topological conjugacy and equivalence, suspensions, structurally stable circle transformations.

1. Необходимые определения и факты.

Определение 1.1 Диффеоморфизм $f : M^n \rightarrow M^n$ называется диффеоморфизмом Морса-Смейла, если:

- 1) неблуждающее множество Ω_f состоит из конечного числа гиперболических орбит;
- 2) многообразия W_p^s, W_q^u пересекаются трансверсально для любых неблуждающих точек p, q .

Диффеоморфизм Морса-Смейла называется градиентно-подобным, если из условия $W_{\sigma_1}^s \cap W_{\sigma_2}^u \neq \emptyset$ для различных точек $\sigma_1, \sigma_2 \in \Omega_f$ следует, что $\dim W_{\sigma_1}^u < \dim W_{\sigma_2}^u$.

Аналогичным образом определяется поток Морса-Смейла на многообразии M^n ,

¹ Исследование поддержано Лабораторией ДСП, НИУ ВШЭ, грант Правительства РФ, договор 075-15-2019-1931.

который называется градиентно-подобным в случае отсутствия периодических траекторий и называется неособым, в случае отсутствия неподвижных точек.

Неособыми, например, являются потоки, полученные с помощью следующей конструкции надстройки над диффеоморфизмом Морса-Смейла.

Определение 1.2. Пусть $f: M^n \rightarrow M^n$ и φ^t — поток на многообразии $M^n \times \mathbb{R}$, порожденный векторным полем, состоящим из единичных векторов, параллельных $\mathbb{R}$ и направленных в $+\infty$ такой, что $\varphi^t(x, r) = (x, r + t)$. Определим диффеоморфизм $g: M^n \times \mathbb{R} \rightarrow M^n \times \mathbb{R}$ формулой $g(x, r) = (f(x), r - 1)$. Положим $G = \{g^k, k \in \mathbb{Z}\}$ и $W = (M^n \times \mathbb{R})/G$. Обозначим через $p_W: M^n \times \mathbb{R} \rightarrow W$ естественную проекцию и через f^t поток на многообразии W , заданный формулой $f^t(x) = p_W(\varphi^t(p_W^{-1}(x)))$. Поток f^t называется надстройкой над диффеоморфизмом f .

2. Классификация грубых преобразований окружности.

Динамика грубых преобразований окружности. В данном разделе будут изучаться грубые (структурно устойчивые) диффеоморфизмы окружности, то есть те, для которых C^1 -близкие (близкие по координатам и по производной) к ним диффеоморфизмы принадлежат одному и тому же классу эквивалентности.

Результат А. Г. Майера [3] говорит о том, что грубые диффеоморфизмы (обозначим их множество через G) имеют очень простую динамику, которая с современной точки зрения может быть описана следующим предложением (см., например, [1]).

Предложение 2.1. Пусть $f \in G$. Тогда

- 1) f имеет конечное (не меньшее двух) число периодических точек p , то есть точек, для которых существует натуральное число m_p такое, что $f^{m_p}(p) = p$ и $f^j(p) \neq p$ для любого целого $0 < j < m_p$ (обозначим через $Per(f)$ множество периодических точек диффеоморфизма f);
- 2) в окрестности U_p точки p , являющейся компонентой связности множества $\mathbb{S}^1 \setminus (Per(f) \setminus p)$, диффеоморфизм f^{m_p} топологически сопряжен либо с растяжением $\alpha_{\pm}(x) = \pm 2x: \mathbb{R} \rightarrow \mathbb{R}$ (в этом случае точка p называется источником, а окрестность U_p — его неустойчивым многообразием W_p^u), либо со сжатием $\alpha_{\pm}^{-1} = \pm \frac{x}{2}: \mathbb{R} \rightarrow \mathbb{R}$ (в этом случае точка p называется стоком, а окрестность U_p — его устойчивым многообразием W_p^s).

Непосредственным следствием такого описания грубого преобразования окружности f является тот факт, что множество $Per(f)$ содержит одинаковое число (обозначим его через r) источников и стоков, которые чередуются на окружности.

Будем считать окружность ориентированной по ходу часовой стрелки и называть ориентацию на дуге $\gamma \subset \mathbb{S}^1$ положительной (отрицательной), если она совпадает (не совпадает) с ориентацией окружности.

Разобьем множество G на два подкласса G_+ и G_- , состоящих из сохраняющих ориентацию и меняющих ориентацию диффеоморфизмов соответственно.

Непосредственным следствием определения ориентируемости отображения и предложения 2.1 являются следующие факты.

Следствие 2.1.

1. Любой диффеоморфизм $f \in G_-$ имеет неподвижную точку.
2. Любой диффеоморфизм $f \in G_+$ ($f \in G_-$) на многообразии $W_{\omega_i}^S$ топологически сопряжен посредством некоторого гомеоморфизма $h_{\alpha_i} : W_{\alpha_i}^u \rightarrow \mathbb{R}$ диффеоморфизму α_+^{-1} (α_-^{-1}) и на многообразии $W_{\alpha_i}^u$ топологически сопряжен посредством некоторого гомеоморфизма $h_{\alpha_i} : W_{\alpha_i}^u \rightarrow \mathbb{R}$ диффеоморфизму α_+ (α_-).

В следующей теореме уточняется структура периодических данных диффеоморфизмов класса G .

Теорема 2.1. ([3]).

1. Для каждого диффеоморфизма $f \in G_+$ множество $Per(f)$ состоит из $2n$, $n \in \mathbb{N}$ периодических орбит, каждая из которых имеет период k .
2. Для каждого диффеоморфизма $f \in G_-$ множество $Per(f)$ состоит из $2q$, $q \in \mathbb{N}$ периодических точек, две из которых являются неподвижными, а другие имеют период 2.

Необходимые и достаточные условия топологической сопряженности грубых преобразований окружности. Пусть $f \in G_+$. Перенумеруем периодические точки множества $Per(f)$:

$p_0, p_1, \dots, p_{2nk-1}, p_{2nk} = p_0$ начиная с произвольной периодической точки p_0 по часовой стрелке, тогда существует целое число l такое, что $f(p_0) = p_{2nl}$, причем $l = 0$ для $k = 1$, $l \in 1, \dots, k-1$ для $k > 1$ и числа (k, l) являются взаимнопростыми. Заметим, что l не зависит от выбора точки p_0 .

Для $f \in G_-$ положим $v = -1$; $v = 0$; $v = +1$, если его неподвижные точки являются источниками; стоком и источником; стоками, соответственно. Заметим, что $v = 0$, если q нечетное и $v = \pm 1$, если q четное.

Следующий результат доставляет необходимые и достаточные условия топологической сопряженности диффеоморфизмов класса G .

Теорема 2.2 ([3]).

1. Дiffeоморфизмы $f; f' \in G_+$ с параметрами $n, k, l; n', k', l'$ топологически сопряжены тогда и только тогда, когда $n = n', k = k'$ и верно одно из следующих утверждений:
 - $l = l'$ (при этом, если $l \neq 0$, то сопрягающий гомеоморфизм сохраняет ориентацию),
 - $l = k' - l'$ (при этом сопрягающий гомеоморфизм меняет ориентацию).
2. Дiffeоморфизмы $f; f' \in G_-$ с параметрами $q, v; q', v'$ топологически сопряжены тогда и только тогда, когда $q = q'$ и $v = v'$.

Реализация грубых преобразований окружности. Следующая теорема реализации является заключительным этапом топологической классификации и гарантирует существование грубого диффеоморфизма окружности с любым допустимым набором параметров.

Теорема 2.3 ([3]).

1. Для любой тройки целых чисел n, k, l такой, что $n, k \in \mathbb{N}$, $l = 0$ для $k = 1$, $l \in \{1, \dots, k-1\}$ и является взаимно простым с k для $k > 1$, существует грубый сохраняющий ориентацию диффеоморфизм окружности с заданными параметрами.
2. Для любой пары целых чисел q, v такой, что $v = 0$, если q нечетное и $v = \pm 1$, если q четное, существует грубый меняющий ориентацию диффеоморфизм окружности с заданными параметрами.

3. Надстройки над грубыми преобразованиями окружности.

В настоящем разделе мы рассмотрим надстройки над модельными грубыми преобразованиями окружности. Именно:

1. Сохраняющие ориентацию: $\phi_{n,k,l} = \pi F_{n,k,l} \pi^{-1}: \mathbb{S}^1 \rightarrow \mathbb{S}^1$, где

$F_{n,k,l}(x) = x + \frac{1}{4\pi nk} \sin(2\pi nkx) + \frac{l}{k}$ для любой тройки целых чисел n, k, l такой, что $n, k \in \mathbb{N}$, $l = 0$ для $k = 1$, $l \in \{1, \dots, k-1\}$ и является взаимно простым с k для $k > 1$.

2. Меняющие ориентацию: $\phi_{q,v} = \pi F_{q,v} \pi^{-1}: \mathbb{S}^1 \rightarrow \mathbb{S}^1$, где

$$F_{2m+1,0}(x) = -x - \frac{1}{4\pi(2m+1)} \sin(2\pi(2m+1)x); F_{q,+1} = -x + \frac{1}{4\pi q} \sin(2\pi qx);$$

$F_{q,-1} = -x - \frac{1}{4\pi q} \sin(2\pi qx)$ для любой пары целых чисел q, v такой, что $v = 0$, если q нечетное и $v = \pm 1$, если q четное, существует грубый меняющий ориентацию диффеоморфизм окружности с заданными параметрами.

Обозначим через $\phi_{n,k,l}^t, \phi_{q,v}^t$ надстройки над модельными диффеоморфизмами. По построению поток $\phi_{n,k,l}^t$ задан на двумерном торе, тогда как поток $\phi_{q,v}^t$ задан на бутылке

Клейна.

Основным результатом этой главы является доказательство следующей теоремы.

Теорема 3.1.

1. Потоки $\phi_{n,k,l}^t$; $\phi_{n',k',l'}^t$ топологически эквивалентны тогда и только тогда, когда $n = n'$.
2. Потоки $\phi_{q,v}^t$; $\phi_{q',v'}^t$ топологически эквивалентны тогда и только тогда, когда $q = q'$ и $v = v'$.

Доказательство. Для доказательства теоремы воспользуемся результатом работы [2], в которой классификация омега-устойчивых потоков на поверхностях сведена к изоморфности их графов. Напомним понятие графа Υ_{ϕ^t} для потока ϕ^t на замкнутой поверхности S , который совпадает либо с $\phi_{n,k,l}^t$ на торе, либо с $\phi_{q,v}^t$ на бутылке Клейна. Любая замкнутая траектория с потока ϕ обладает трубчатой окрестностью U_c , которая не пересекается с окрестностями других замкнутых траекторий и чья граница R_c трансверсальна траекториям потока ϕ^t .

Пусть $\mathcal{R} = \bigcup_{c \in \Omega_{\phi^t}} R_c$. Компоненты связности множества $\mathcal{R}$ назовём разрезающими окрестностями. Элементарной ячейкой называется компонента связности множества $\hat{S} = S \setminus \mathcal{R}$. Таким образом, элементарные ячейки потока ϕ^t могут быть одного из двух типов:

- а) ячейка типа $\mathcal{L}$ – содержит ровно одну замкнутую траекторию потока ϕ^t ,
- б) ячейка типа $\mathcal{E}$ – не пересекается с неблуждающим множеством потока ϕ^t .

Ориентируемый граф Υ_{ϕ^t} называют графом потока ϕ^t , если вершины Υ_{ϕ^t} взаимно однозначно соответствуют элементарным ячейкам потока ϕ^t . Мы будем называть $\mathcal{L}$ –, $\mathcal{E}$ –вершиной вершину, соответствующую ячейке типа $\mathcal{L}$, $\mathcal{E}$.

Трубчатая окрестность U_c окружности c на поверхности может быть либо кольцом с границей из двух окружностей, либо пленкой Мебиуса с границей из одной окружности (см. рис. 1).

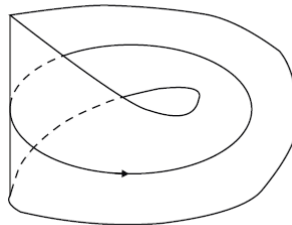


Рис. 1. Случай, когда U_c гомеоморфна пленке Мебиуса.

Тогда соответствующая такой окрестности $\mathcal{L}$ -вершина графа инцидентна либо двум

ребрам одинаковой направленности по отношению к этой вершине, либо одному ребру, соответственно. Поскольку плёнки Мебиуса существуют только на неориентируемых поверхностях, то $\mathcal{L}$ -вершина с одним инцидентным ей ребром может быть только на бутылке Клейна.

В общем случае у потока на поверхности $\mathcal{E}$ -ячейки могут быть двух типов: с согласованной и несогласованной ориентацией компонент связности границы области $\mathcal{E}$ (см. рис. 2).

Поскольку рассматриваемый поток ϕ^t является надстройкой над диффеоморфизмом окружности, то он имеет глобальную секущую и, следовательно, все его $\mathcal{E}$ -ячейки имеют согласованную ориентацию компонент связности границы области $\mathcal{E}$.

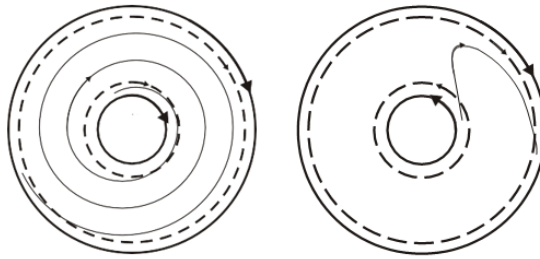


Рис. 2. Два типа $\mathcal{E}$ -ячеек.

Кроме того, любое ребро графа Υ_{ϕ^t} соединяет $\mathcal{L}$ -вершину с $\mathcal{E}$ -вершиной.

Согласно результатам работы [2], потоки ϕ^t, ϕ'^t топологически эквивалентны тогда и только тогда, когда существует изоморфизм графов $\Upsilon_{\phi^t}, \Upsilon_{\phi'^t}$, сохраняющий тип вершин и ориентацию ребер.

Таким образом, доказательство теоремы сводится к классификации описанных графов с точностью до изоморфизма с описанными свойствами. Приведем эту классификацию отдельно для тора и бутылки Клейна.

Для потока $\phi_{n,k,l}^t$ на торе граф $\Upsilon_{\phi_{n,k,l}^t}$ представляет из себя цикл из $4n$ чередующихся $\mathcal{L}$ - и $\mathcal{E}$ -вершин (см. рис. 3). Таким образом графы $\Upsilon_{\phi_{n,k,l}^t}; \Upsilon_{\phi_{n',k',l'}^t}$ изоморфны тогда и только тогда, когда $n = n'$.

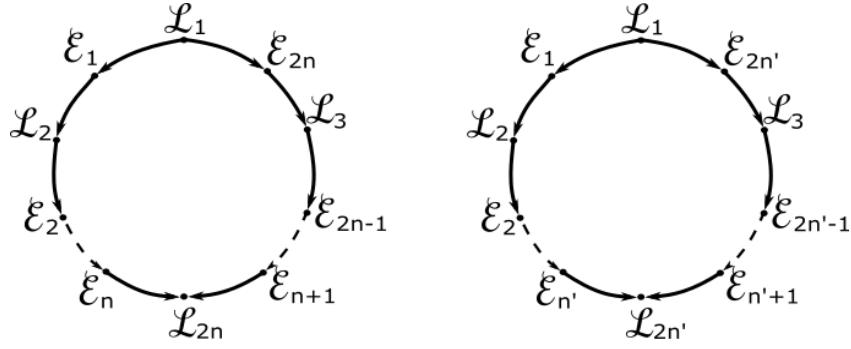


Рис. 3. Граф $Y_{\phi_{n,k,l}^t}$

Рассмотрим теперь две надстройки $\phi_{q,v}^t$ и $\phi_{q',v'}^t$ над меняющими ориентацию диффеоморфизмами с заданными различными q , q' , v и v' . Начнём с примера, который покажет выполнение условия (2) данной теоремы, для доказательства которого так же будем пользоваться изоморфизмом графов надстроек.

Пусть есть два диффеоморфизма, меняющих ориентацию, определенных следующими формулами: $\phi_{3,0} = \pi F_{3,0} \pi^{-1}: \mathbb{S}^1 \rightarrow \mathbb{S}^1$, где $F_{3,0}(x) = -x - \frac{1}{12\pi} \sin(6\pi x)$, и $\phi_{2,v'} = \pi F_{2,v'} \pi^{-1}: \mathbb{S}^1 \rightarrow \mathbb{S}^1$, где v' принимает значения ± 1 в силу четности $q' = 2$ и $F_{2,+1} = -x + \frac{1}{8\pi} \sin(4\pi x)$, $F_{2,-1} = -x - \frac{1}{8\pi} \sin(4\pi x)$.

У диффеоморфизма $\phi_{3,0}$ точка $x = 0$ – источник, а $x = \frac{1}{2}$ – сток, точки $x = \frac{1}{6}$ и $x = \frac{5}{6}$ – притягивающие периода 2, а точки $x = \frac{1}{3}$ и $x = \frac{2}{3}$ – отталкивающие периода 2. Для диффеоморфизма $\phi_{2,v'}$ имеем при $v = 1$ два стока $x = 0$, $x = \frac{1}{2}$ и две отталкивающие точки периода 2: $x = \frac{1}{4}$, $x = \frac{3}{4}$. При $v = -1$ имеем 2 источника $x = 0$ и $x = \frac{1}{2}$, две притягивающие точки: $x = \frac{1}{4}$ и $x = \frac{3}{4}$.

На рисунке 4 изображены ячейки типа $\mathcal{L}$ и $\mathcal{E}$ для потока $\phi_{3,0}^t$ и соответствующий ему граф $Y_{\phi_{3,0}^t}$.

Так как при склейке неподвижным точкам $x = 0$ и $x = \frac{1}{2}$ диффеоморфизма $\phi_{3,0}$ соответствуют периодические орбиты на поверхности бутылки Клейна, окрестностью которых будут плёнки Мёбиуса, то исходя из того, что граница такой окрестности – это одна окружность, получим, что каждая из вершин $\mathcal{L}_1$ и $\mathcal{L}_4$ графа надстройки для данного диффеоморфизма инцидентна одному ребру. Направление ребра, соединяющего вершины $\mathcal{L}_i$ и $\mathcal{E}_{i-1}$ зависит от характера орбиты. На рисунке 4 внизу изображен граф, где направление

ребер задано следующим образом: так как $x = \frac{1}{2}$ – отталкивающая неподвижная точка, то ребро, инцидентное вершине $\mathcal{L}_4$, направлено к вершине $\mathcal{E}_3$. Окрестностью каждой из периодических орбит, отмеченных красным цветом на рисунке 4, является кольцо, следовательно, вершинам $\mathcal{L}_2$ и $\mathcal{L}_3$ инциденты два ребра.

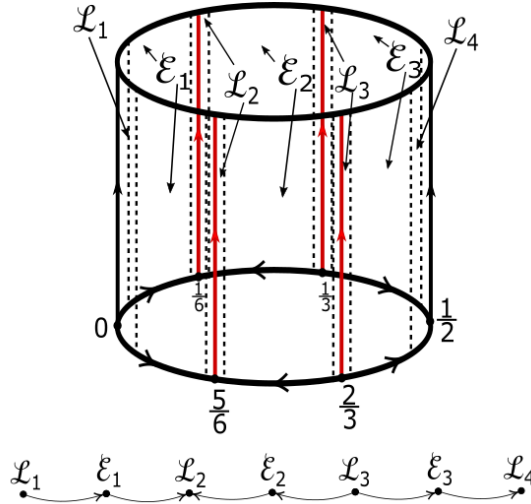


Рис. 4. Надстройка над $\phi_{3,0}$ и соответствующий ей граф $\mathcal{Y}_{\phi_{3,0}^t}$.

Направление рёбер от одной вершины к другой обусловлено поведением траекторий на поверхности бутылки Клейна, полученной склейкой точек на окружностях нижнего и верхнего уровней так, как это показано на рисунке 5 стрелками, изображенными черным пунктиром на видимой части цилиндра и серым – на задней части.

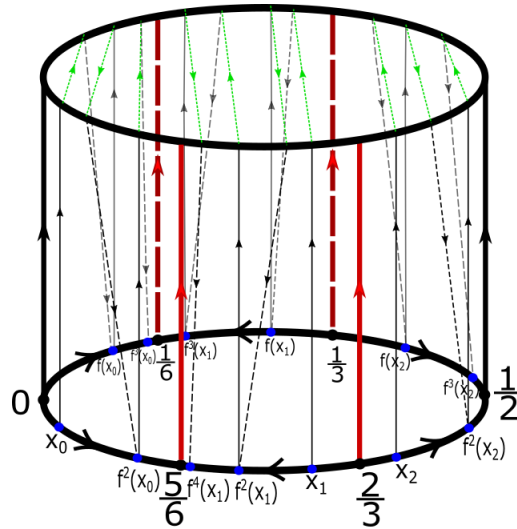


Рис. 5. Поведение траекторий на $\phi_{3,0}^t$ и направления склейки (пунктиром черного и серого цвета).

Аналогичным рассуждениям поддается и пример с грубым преобразованием окружности $\phi_{2,1}$. Так как орбиты, отмеченные красным на рисунке 3.6, отталкивающие, в силу того что $x = \frac{1}{4}$ и $x = \frac{3}{4}$ – источники (см. поведение орбит на рис. 6), то ребра графа $Y_{\phi_{2,v'}^t}$, инцидентные вершине L_2 выходят из неё к вершинам ε_2 . Следовательно, получим граф, изображенный ниже.

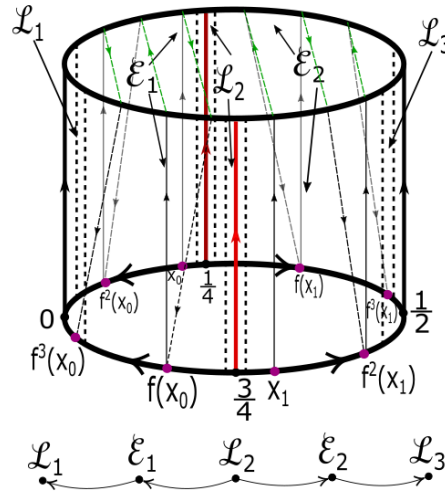


Рис. 6. Надстройка над $\phi_{2,1}$, поведение траекторий при склейке
и соответствующий ей граф $Y_{\phi_{2,1}^t}$

Для диффеоморфизма $\phi_{2,-1}$ надстройка $\phi_{2,-1}^t$ производится таким же образом, как и в случае, когда $v' = 1$, за исключением того, что $x = 0$ и $x = \frac{1}{2}$ теперь источники. Тогда для графа $Y_{\phi_{2,-1}^t}$ справедливы те же рассуждения, что и для графа надстройки $\phi_{2,1}^t$, только в данном случае направление рёбер меняется в противоположную сторону (см. рис. 7).



Рис. 7. Граф $Y_{\phi_{2,-1}^t}$

Заметим, что графы $Y_{\phi_{3,0}^t}$, $Y_{\phi_{2,1}^t}$ и $Y_{\phi_{2,-1}^t}$ попарно не изоморфны, так как ни один из них нельзя получить переименованием вершин другого так, чтобы их тип и ориентация ребер не менялась. Количество вершин графа $Y_{\phi_{3,0}^t}$ отличается от количества вершин других двух построенных графов в примере, а у графов $Y_{\phi_{2,1}^t}$ и $Y_{\phi_{2,-1}^t}$ различается направление

рёбер. Соответственно, пользуясь снова критерием об эквивалентности потоков с помощью построения изоморфизма графов, получим, что надстройки $\phi_{3,0}^t$, $\phi_{2,1}^t$ и $\phi_{2,1}^t$ не эквивалентны.

Перейдём к общему случаю для потоков $\phi_{q,v}^t$ и $\phi_{q',v'}^t$ для $q \neq q'$ и $v \neq v'$. Опираясь на рассуждения, приведенные выше в примерах, получим графы данных надстроек с различными q и v . Количество ячеек в графе надстройки над диффеоморфизмом $\phi_{q,v}$ равно $2q + 1$. Графы для различных q и v представлены на рисунках 8–10 ниже. Если оба числа q и q' нечётные ($v = v' = 0$), причем $q \neq q'$, то количество вершин графов надстроек с такими данными не совпадает, значит, нельзя построить биекцию между вершинами.

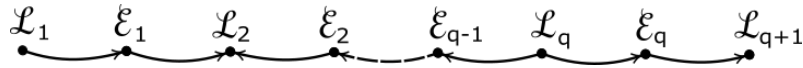


Рис. 8. Граф $Y_{\phi_{q,0}^t}$ для $q = 2n + 1$.

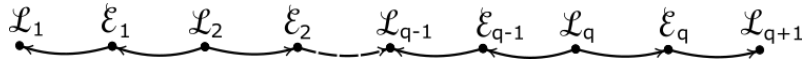


Рис. 9. Граф $Y_{\phi_{q,1}^t}$ для $q = 2n, v = 1$.

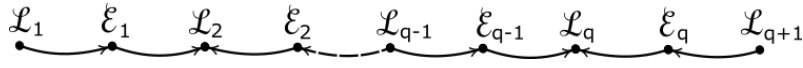


Рис. 10. Граф $Y_{\phi_{q,-1}^t}$ для $q = 2n, v = -1$.

Аналогично для чётных и не равных друг другу q и q' получим различное количество вершин у графов. Если же в данном случае $q = q'$, но $v \neq v'$, то количество вершин, конечно, совпадает, а направление рёбер – нет, как это показано на рисунках 9 и 10 для общего случая.

Для случая, когда $q = 2n$ и $q' = 2n + 1$, $n = 1, 2, \dots$, имеем и разное количество вершин, и меняющуюся ориентацию ребер (см. рисунки 8-10).

Значит, во всех рассмотренных случаях при $q \neq q'$ и $v \neq v'$ имеем не изоморфные графы $Y_{\phi_{q,v}^t}, Y_{\phi_{q',v'}^t}$. Следовательно надстройки $\phi_{q,v}^t$ и $\phi_{q',v'}^t$ не эквивалентны.

Значит, необходимым и достаточным условием того, чтобы $\phi_{q,v}^t$ и $\phi_{q',v'}^t$ были эквивалентны, является условие $q = q'$ и $v = v'$.

4. Заключение. Таким образом, делая вывод о классификации надстроек над градиентно-подобными диффеоморфизмами, мы получили доказательство того, что из эквивалентности надстроек $\phi_{n,k,l}^t$ и $\phi_{n',k',l'}^t$ не следует сопряженность диффеоморфизмов $\phi_{n,k,l}$ и $\phi_{n',k',l'}$. Тем не менее, если надстройки над меняющими ориентацию диффеоморфизмами $\phi_{q,v}^t$ и $\phi_{q',v'}^t$ эквивалентны, то и сами $\phi_{q,v}$ и $\phi_{q',v'}$ сопряжены.

То есть результатом данной работы можно считать следующее утверждение:

Предложение 4.1. Дiffeоморфизмы $\phi_{q,\nu}$ и $\phi_{q',\nu'}$ сопряжены тогда и только тогда, когда надстройки $\phi_{q,\nu}^t$ и $\phi_{q',\nu'}^t$ эквивалентны.

СПИСОК ЛИТЕРАТУРЫ

1. Grines V., Medvedev T., Pochinka O. Dynamical Systems on 2- and 3- Manifolds. – Switzerland: Springer International Publishing, 2016. – 313 p.
2. Kruglov V., Malyshev D., Pochinka O. Topological Classification of Ω -stable Flows on Surfaces by Means of Effectively Distinguishable Multigraphs // Discrete and Continuous Dynamical Systems. – 2018. – Vol. 38. No. 9. – P. 4305– 4327.
3. Майер А. Г. Грубое преобразование окружности в окружность // Ученые записки Горьк. гос. ун-та. – 1939. – Т. 12. – С. 215–229.
4. Peixoto M. M. On structural stability // Ann. Math. – 1959. – Vol. 69. – P. 199–222.

БАРИНОВА М. К., ГОГУЛИНА Е. Ю., ПОЧИНКА О. В.
РЕАЛИЗАЦИЯ АЦИКЛИЧЕСКОЙ ДИАГРАММЫ СМЕЙЛА
ОМЕГА-УСТОЙЧИВЫМ ДИФФЕОМОРФИЗМОМ ПОВЕРХНОСТИ¹

Аннотация. Диаграммой Смейла Ω -устойчивого диффеоморфизма является граф, вершины которого соответствуют базисным множествам, а ориентированные ребра последовательно соединяют вершины максимальных цепей. Одной из проблем, поставленных Смейлом, является описание допустимых диаграмм — диаграмм, реализуемых каким-либо диффеоморфизмом. В настоящей работе доказано, что любой ациклический граф реализуется Ω -устойчивым диффеоморфизмом поверхности с нетривиальными базисными множествами.

Ключевые слова: Диаграмма Смейла, Ω -устойчивость, базисное множество, DA-диффеоморфизм, ациклический граф.

BARINOVA M. K., GOGULINA E. YU., POCHINKA O. V.
REALIZATION OF THE ACYCLIC SMALE DIAGRAM
BY AN OMEGA-STABLE SURFACE DIFFEOMORFISM

Abstract. Smale diagram of Ω -stable diffeomorphism is a graph, which vertices correspond to basic sets and oriented edges sequentially connect vertices of maximal chain. Smale formulated description of acceptable diagrams, which are realized by a diffeomorphism. Present paper proves that an acyclic graph is implemented by an Ω -stable diffeomorphism on surface with nontrivial basic sets.

Keywords: Smale diagram, Ω -stability, basic set, DA-diffeomorphism, acyclic graph.

Введение и формулировка результатов. Пусть f — диффеоморфизм замкнутого n -многообразия M^n . Говорят, что диффеоморфизм f удовлетворяет аксиоме A , если его неблуждающее множество является гиперболическим, и периодические точки плотны в нем. Для A -диффеоморфизмов справедлива теорема о спектральном разложении С. Смейла [2], устанавливающая единственное представление неблуждающего множества в виде конечного объединения попарно непересекающихся множеств, называемых *базисными*, каждое из которых является компактным, инвариантным и топологически транзитивным.

На множестве базисных множеств любого A -диффеоморфизма f можно ввести отношение частичного порядка С. Смейла [2]. Именно, пусть Λ_i, Λ_j — базисные множества

¹ Исследование поддержано Лабораторией ДСП, НИУ ВШЭ, грант Правительства РФ, договор 075-15-2019-1931.

А-дiffeоморфизма f . Говорят, что Λ_i, Λ_j находятся в отношении $\prec$ ($\Lambda_i \prec \Lambda_j$), если $W_{\Lambda_i}^s \cap W_{\Lambda_j}^u \neq \emptyset$.

Последовательность, состоящая из базисных множеств $\Lambda_i = \Lambda_{i_0}, \Lambda_{i_1}, \dots, \Lambda_{i_k} = \Lambda_j$ ($k \geq 1$), такая что $\Lambda_{i_0} \prec \Lambda_{i_1} \prec \dots \prec \Lambda_{i_k}$ называется *цепью длины $k \in \mathbb{N}$* , соединяющей периодические орбиты Λ_i и Λ_j .

Такая цепь называется *максимальной*, если в нее нельзя добавить ни одного нового базисного множества. Цепь называется *циклом*, если $\Lambda_i = \Lambda_j$.

Дiffeоморфизм $f: M^n \rightarrow M^n$ называется Ω -устойчивым, если C^1 -близкие к f диффеоморфизмы топологически сопряжены на неблуждающих множествах. Согласно [3], диффеоморфизм $f: M^n \rightarrow M^n$ является Ω -устойчивым тогда и только тогда, когда он удовлетворяет аксиоме А и не имеет циклов.

Диаграммой Смейла Δ_f Ω -устойчивого диффеоморфизма $f: M^n \rightarrow M^n$ называется граф, вершины которого соответствуют базисным множествам, а ориентированные ребра последовательно соединяют вершины максимальных цепей.

В дальнейшем под *направленным циклом* ориентированного графа будем понимать замкнутый путь, пройденный в соответствии с направлениями ребер ([6]). Граф называют *ациклическим*, если у него нет направленных циклов. При этом не исключаются "параллельные" пути, которые выходят из одной начальной вершины и приходят в одну конечную, но при этом не совпадают.

В работе ([2]), в качестве проблемы сформулирован следующий вопрос: какие диаграммы могут соответствовать Ω -устойчивым диффеоморфизмам?

Настоящая работа дает частичный ответ на этот вопрос в виде доказательства следующей теоремы.

Теорема. *Любой связный ациклический граф реализуется Ω -устойчивым диффеоморфизмом поверхности.*

Построение базисных множеств для диффеоморфизма двумерного тора.
Диффеоморфизм Аносова на 2-торе. Пусть $C \in GL(2, \mathbb{R})$ – гиперболическая матрица с собственными значениями λ_1, λ_2 такими что $|\lambda_1| > 1$ и $|\lambda_2| = 1/|\lambda_1|$. Так как матрица C имеет определитель, равный 1, то она индуцирует гиперболический автоморфизм $\hat{C}: T^2 \rightarrow T^2$ с неподвижной точкой О. Этот диффеоморфизм является диффеоморфизмом Аносова, обладающий двумя трансверсальными инвариантными слоениями (устойчивым и

неустойчивым), любой слой каждого из которых всюду плотен на торе. Кроме того, множество периодических точек диффеоморфизма $\hat{C}$ также плотно на T^2 .

«Хирургическая операция» Смейла. Пусть $\sigma: \square \rightarrow [0,1]$ – функция, заданная формулой (см. рис. 1)

$$\sigma(x) = \begin{cases} 0, & x, \lambda^{-3}, \\ \frac{1}{\frac{\lambda^{-3}+1}{2}-x}, & \lambda^{-3} < x < 1, \\ 1 + e^{\frac{2}{(x-\lambda^{-3})^2(x-1)^2}}, & \\ 1, & x..1. \end{cases}$$

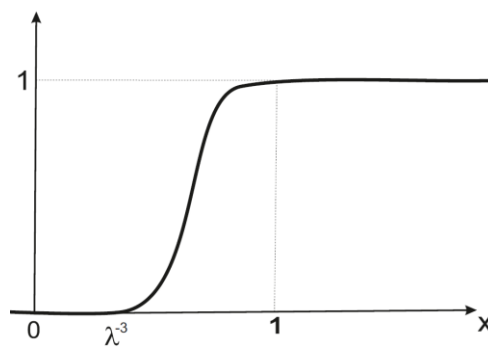


Рис. 1. График функции $\sigma(x)$.

Определим функцию $\nu: [0,1] \rightarrow [0,1]$ формулой (см. рис. 2)

$$\nu(x) = \begin{cases} \lambda^2 x, & 0, x, \lambda^{-3}, \\ \sigma(x)x + (1-\sigma(x))\lambda^2 x, & \lambda^{-3} < x, 1. \end{cases}$$

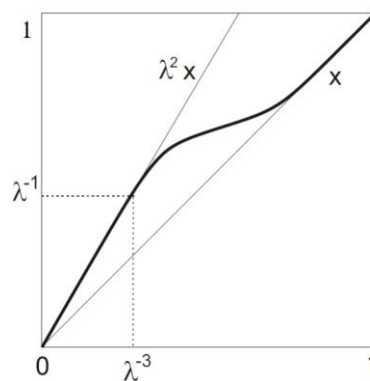


Рис. 2. График функции $\nu(x)$.

Продолжим функцию ν нечетным образом на отрезок $[-1,1]$, положив $\nu(-x) = -\nu(x)$ для $x \in [0,1]$.

Положим $D = \{(x, y) \in \mathbb{R}^2 \mid x^2 + y^2 \leq 2\}$. Определим функцию $\gamma_A : D \rightarrow [0, 1]$ формулой

$$\gamma_A(x, y) = \begin{cases} \nu(x), & 0 \leq |y| \leq \lambda^{-3}, \\ \sigma(|y|)x + (1 - \sigma(|y|))\nu(x), & \lambda^{-3} < |y| \leq \sqrt{2}. \end{cases}$$

Определим диффеоморфизм $B_A : D \rightarrow D$ формулой $B_A(x, y) = (\gamma_A(x, y), y)$. По построению, $B_A(x, y) = (\lambda^2 x, y)$, если $x^2 + y^2 \leq \lambda^{-6}$ и является тождественным на ∂D .

Пусть x, y – локальные координаты в окрестности $U(O)$ точки на T^2 такие, что диффеоморфизм $\hat{C}$ в этих координатах имеет форму $\hat{C}(x, y) = (x/\lambda, \lambda y)$. Тогда $Ox \subset W_O^s$ и $Oy \subset W_O^u$, а также $\{y = \text{const}\}$ и $\{x = \text{const}\}$ являются устойчивым и неустойчивым слоениями. Определим диффеоморфизм $\hat{B}_A : T^2 \rightarrow T^2$, совпадающий с диффеоморфизмом B_A внутри окрестности $U(O)$ и тождественный вне этой окрестности. Тогда согласно [4], [1], диффеоморфизм $\hat{\Psi}_A = \hat{B}_A \circ \hat{C}$ является DA-диффеоморфизмом, неблуждающее множество которого состоит из одномерного аттрактора A , обладающего единственной связкой степени 2 с двумя различными граничными неподвижными p_1 и p_2 , и источниковой неподвижной точкой α (см. рис. 3).

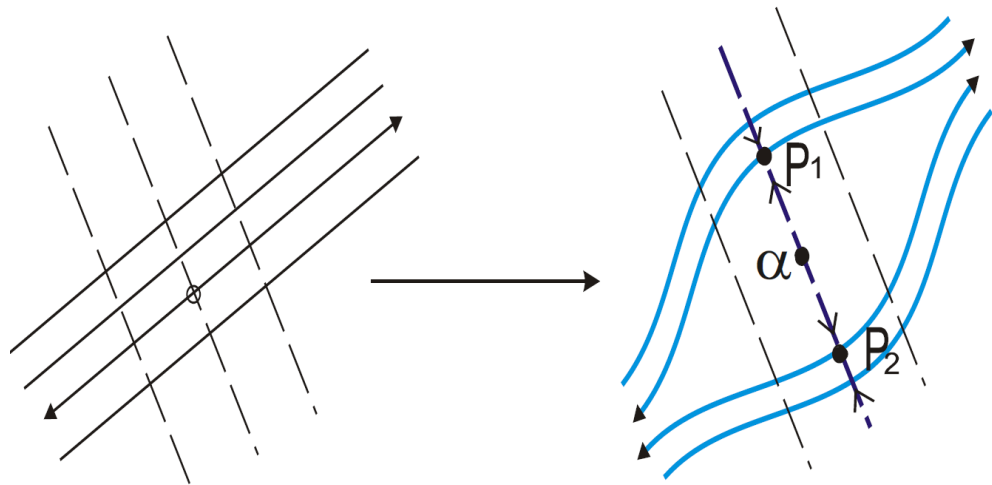


Рис. 3. «Хирургическая операция» Смейла.

Модельные диффеоморфизмы на торе. В этом разделе, используя хирургическую операцию Смейла, мы построим модельные диффеоморфизмы на двумерном торе.

Диффеоморфизм $F_{m_1, \dots, m_k}$. Выберем $k \in \mathbb{N}$ периодических орбит $O_1, \dots, O_k$ диффеоморфизма Аносова $\hat{C}$ периодов $m_1, \dots, m_k$, соответственно. В окрестности каждой орбиты произведем хирургическую операцию Смейла вдоль устойчивого слоения.

Полученный диффеоморфизм $F_{m_1, \dots, m_k} : T^2 \rightarrow T^2$ имеет единственный нетривиальный одномерный аттрактор $\Lambda_{m_1, \dots, m_k}$ и k периодических источниковых орбит $\alpha_{m_1}, \dots, \alpha_{m_k}$ периодов $m_1, \dots, m_k$, соответственно.

Диффеоморфизм $F^{n_1, \dots, n_l}$. Выберем $l \in \mathbb{N}$ периодических орбит $O^1, \dots, O^l$ диффеоморфизма Аносова $\hat{C}$ периодов $n_1, \dots, n_l$, соответственно. В окрестности каждой орбиты произведем хирургическую операцию Смейла вдоль неустойчивого слоения. Полученный диффеоморфизм $F^{n_1, \dots, n_l} : T^2 \rightarrow T^2$ имеет единственный нетривиальный одномерный репеллер $\Lambda^{n_1, \dots, n_l}$ и l периодических стоковых орбит $\omega_{n_1}, \dots, \omega_{n_l}$ периодов $n_1, \dots, n_l$, соответственно.

Диффеоморфизм $F_{m_1, \dots, m_k}^{n_1, \dots, n_l}$. Выберем $l+k \in \mathbb{N}$ периодических орбит $O_1, \dots, O_k, O^1, \dots, O^l$ диффеоморфизма Аносова $\hat{C}$ периодов $m_1, \dots, m_k, n_1, \dots, n_l$, соответственно. В окрестности каждой орбиты $O_1, \dots, O_k$ произведем хирургическую операцию Смейла вдоль устойчивого слоения и в окрестности каждой орбиты $O^1, \dots, O^l$ произведем хирургическую операцию Смейла вдоль неустойчивого слоения. Полученный диффеоморфизм $F_{m_1, \dots, m_k}^{n_1, \dots, n_l} : T^2 \rightarrow T^2$ имеет единственное нетривиальное нульмерное базисное множество $\Lambda_{m_1, \dots, m_k}^{n_1, \dots, n_l}$, k периодических источниковых орбит $\alpha_{m_1}, \dots, \alpha_{m_k}$ периодов $m_1, \dots, m_k$ и l периодических стоковых орбит $m_1, \dots, m_k$ периодов $n_1, \dots, n_l$, соответственно.

Примеры модельных диффеоморфизмов $F_{2,3}, F^{1,4}, F_2^4$ изображены на рисунке 4.

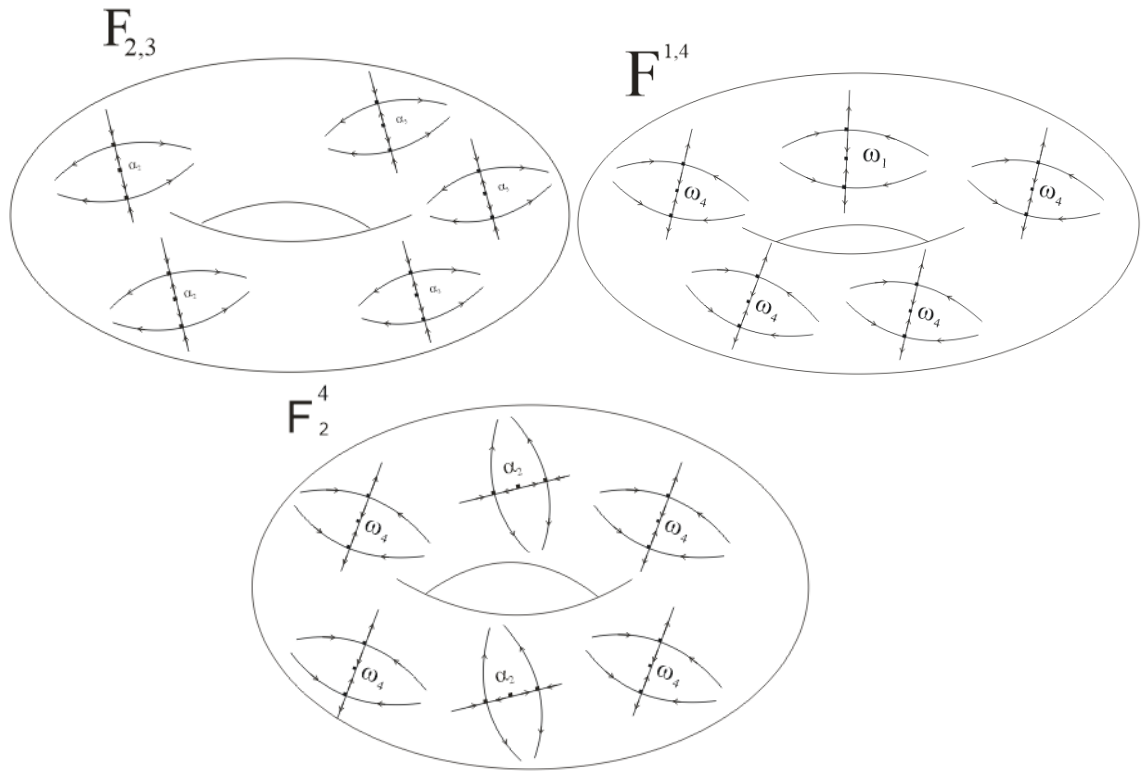


Рис. 4. Модельные диффеоморфизмы $F_{2,3}, F^{1,4}, F_2^4$.

Доказательство основного результата. В настоящем разделе мы реализуем диаграмму Смейла Ω -устойчивым диффеоморфизмом поверхности.

Распределение вершин графа по уровням. Так как в графе Γ нет направленных циклов, на нем можно ввести частичный порядок. Упорядочим вершины графа по следующему принципу:

- уровень B_0 состоит из вершин $b_0^0, \dots, b_0^{q_0}$, в которые не входит ни одно ребро;
- уровень B_r состоит из вершин $b_r^0, \dots, b_r^{q_r}$, в которые входят, только ребра из вершин уровней $B_0, \dots, B_{r-1}$,
- последний уровень B_p состоит из вершин $b_p^0, \dots, b_p^{q_p}$, из которых не выходят ребра (см. рис. 5).

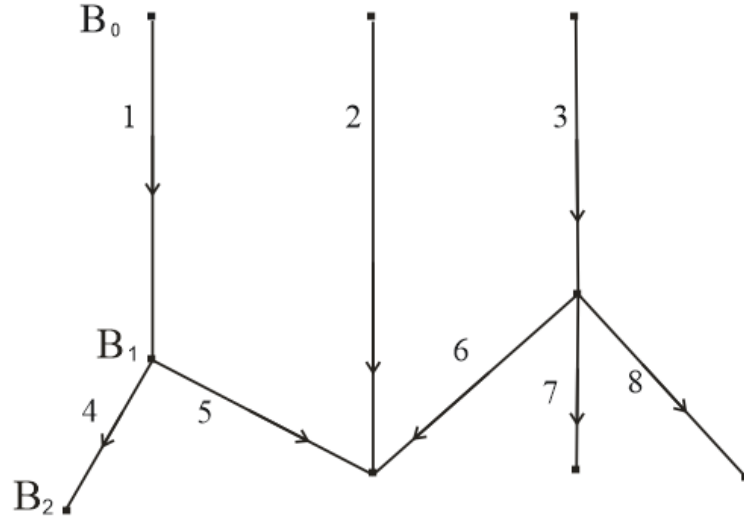


Рис. 5. Распределение вершин по уровням.

Перенумеруем ребра графа в произвольном порядке: каждому ребру $(b_{r_1}^{j_1}, b_{r_2}^{j_2})$ поставим в соответствие некоторый номер $k_{r_1, r_2}^{j_1, j_2}$, то есть каждое ребро будет определяться тройкой $(b_{r_1}^{j_1}, b_{r_2}^{j_2}, k_{r_1, r_2}^{j_1, j_2})$.

Реализация вершин уровня B_0 . Для каждой вершины b_0^j и каждого выходящего ребра положим $n_i = k_{0, r_i}^{j, j_i}$ и $f_{b_0^j} = F^{n_1, \dots, n_{\nu_0^j}}$. Обозначим через ω_{0, r_i}^{j, j_i} стоковую орбиту, соответствующую выходящему ребру $(b_0^j, b_{r_i}^{j_i}, k_{0, r_i}^{j, j_i})$. В примере графа на рисунке 5, $f_{b_0^0} = F^1, f_{b_0^1} = F^2, f_{b_0^2} = F^3$.

Реализация вершин уровня B_r . Для каждой вершины b_r^j и каждого входящего ребра $(b_{r_i}^{j_i}, b_r^j, k_{r_i, r}^{j_i, j})$, $i = 1, \dots, \mu_r^j$ положим $m_i = k_{r_i, r}^{j_i, j}$. Для каждого выходящего ребра $(b_r^j, b_{r_i}^{j_i}, k_{r, r_i}^{j, j_i})$, $i = 1, \dots, \nu_r^j$ положим $n_i = k_{r, r_i}^{j, j_i}$. Положим $f_{b_r^j} = F^{m_1, \dots, m_{\mu_r^j}, n_1, \dots, n_{\nu_r^j}}$. Обозначим через $\alpha_{r_i, r}^{j_i, j}$ источникную орбиту, соответствующую входящему ребру $(b_{r_i}^{j_i}, b_r^j, k_{r_i, r}^{j_i, j})$ и через ω_{r, r_i}^{j, j_i} — стоковую орбиту, соответствующую выходящему ребру $(b_r^j, b_{r_i}^{j_i}, k_{r, r_i}^{j, j_i})$. В примере графа на рисунке 5, $f_{b_0^0} = F_1^{4,5}, f_{b_1^1} = F_3^{6,7,8}$.

Реализация вершин уровня B_p . Для каждой вершины b_p^j и каждого входящего ребра $(b_{r_i}^{j_i}, b_p^j, k_{r_i, r}^{j_i, j}), i=1, \dots, \mu_p^j$ положим $m_i = k_{r_i, r}^{j_i, j}$ и $f_{b_p^j} = F_{m_1, \dots, m_{\mu_p^j}}$. Обозначим через $\alpha_{r_i, p}^{j_i, j}$ источниковую орбиту, соответствующую входящему ребру $(b_{r_i}^{j_i}, b_p^j, k_{r_i, r}^{j_i, j})$.

В примере графа на рисунке 5, $f_{b_2^0} = F_4, f_{b_2^1} = F_{2,5,6}, f_{b_2^2} = F_7, f_{b_2^3} = F_8$.

Склейка динамики вдоль ребра графа Γ . Каждому ребру $(b_{r_1}^{j_1}, b_{r_2}^{j_2}, k_{r_1, r_2}^{j_1, j_2})$ графа Γ однозначно соответствует стоковая орбита $\omega_{r_1, r_2}^{j_1, j_2}$ диффеоморфизма $f_{b_{r_1}^{j_1}}$ и источниковая орбита $\alpha_{r_1, r_2}^{j_1, j_2}$ диффеоморфизма $f_{b_{r_2}^{j_2}}$. Обе орбиты имеют одинаковый период $k_{r_1, r_2}^{j_1, j_2}$. Покажем, как склеить динамику вдоль ребра $(b_{r_1}^{j_1}, b_{r_2}^{j_2}, k_{r_1, r_2}^{j_1, j_2})$.

Выберем окрестности $U_{r_1, r_2}^{j_1, j_2}, V_{r_1, r_2}^{j_1, j_2}$, содержащие орбиты $\omega_{r_1, r_2}^{j_1, j_2}, \alpha_{r_1, r_2}^{j_1, j_2}$, соответственно, и такие, что $f_{b_{r_1}^{j_1}}(U_{r_1, r_2}^{j_1, j_2}) \subset \text{int} U_{r_1, r_2}^{j_1, j_2}, V_{r_1, r_2}^{j_1, j_2} \subset \text{int} f_{b_{r_2}^{j_2}}(V_{r_1, r_2}^{j_1, j_2})$. Положим $X_{r_1, r_2}^{j_1, j_2} = U_{r_1, r_2}^{j_1, j_2}, \text{int} f_{b_{r_1}^{j_1}}(U_{r_1, r_2}^{j_1, j_2}), Y_{r_1, r_2}^{j_1, j_2} = f_{b_{r_2}^{j_2}}(V_{r_1, r_2}^{j_1, j_2}), \text{int} V_{r_1, r_2}^{j_1, j_2}$. Оба множества состоят из одинакового количества двумерных колец. Обозначим через $H_{r_1, r_2}^{j_1, j_2} : X_{r_1, r_2}^{j_1, j_2} \rightarrow Y_{r_1, r_2}^{j_1, j_2}$ диффеоморфизм такой, что $H_{r_1, r_2}^{j_1, j_2} \circ f_{b_{r_1}^{j_1}}|_{\partial U_{r_1, r_2}^{j_1, j_2}} = f_{b_{r_2}^{j_2}} \circ H_{r_1, r_2}^{j_1, j_2}|_{\partial U_{r_1, r_2}^{j_1, j_2}}$.

На рисунке 6 введенные объекты изображены для ребра $(b_0^0, b_1^0, 1)$ графа, приведенного на рисунке 5.

Положим

$$Q_{r_1, r_2}^{j_1, j_2} = (\mathbb{T}^2, \text{int} f_{b_{r_1}^{j_1}}(U_{r_1, r_2}^{j_1, j_2})) \cup (\mathbb{T}^2, \text{int} V_{r_1, r_2}^{j_1, j_2})$$

и

$$\hat{Q}_{r_1, r_2}^{j_1, j_2} = (\mathbb{T}^2, \text{int} f_{b_{r_1}^{j_1}}(U_{r_1, r_2}^{j_1, j_2})) \cup_{H_{r_1, r_2}^{j_1, j_2}} (\mathbb{T}^2, \text{int} V_{r_1, r_2}^{j_1, j_2}).$$

Обозначим через $p_{r_1, r_2}^{j_1, j_2} : Q_{r_1, r_2}^{j_1, j_2} \rightarrow \hat{Q}_{r_1, r_2}^{j_1, j_2}$ естественную проекцию.

Склейкой динамики вдоль ребра $(b_{r_1}^{j_1}, b_{r_2}^{j_2}, k_{r_1, r_2}^{j_1, j_2})$ графа Γ назовем диффеоморфизм $f_{r_1, r_2}^{j_1, j_2} : \hat{Q}_{r_1, r_2}^{j_1, j_2} \rightarrow \hat{Q}_{r_1, r_2}^{j_1, j_2}$, совпадающий с диффеоморфизмом $p_{r_1, r_2}^{j_1, j_2} f_{b_{r_2}^{j_2}}(p_{r_1, r_2}^{j_1, j_2})^{-1}|_{p_{r_1, r_2}^{j_1, j_2}(\mathbb{T}^2, V_{r_1, r_2}^{j_1, j_2})}$ на $p_{r_1, r_2}^{j_1, j_2}(\mathbb{T}^2, V_{r_1, r_2}^{j_1, j_2})$ и с диффеоморфизмом $p_{r_1, r_2}^{j_1, j_2} f_{b_{r_1}^{j_1}}(p_{r_1, r_2}^{j_1, j_2})^{-1}|_{p_{r_1, r_2}^{j_1, j_2}(\mathbb{T}^2, U_{r_1, r_2}^{j_1, j_2})}$ на $p_{r_1, r_2}^{j_1, j_2}(\mathbb{T}^2, U_{r_1, r_2}^{j_1, j_2})$.

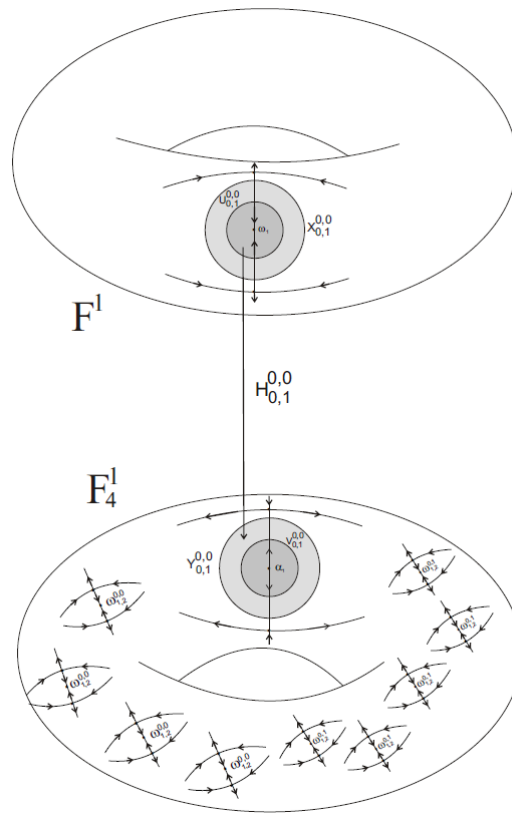


Рис. 6. Объекты, связанные с ребром $(b_0^0, b_1^0, 1)$.

Построение результирующего диффеоморфизма. В построенных по графу Γ моделях существует взаимно однозначное соответствие между стоковыми и источниковыми орбитами.

Именно стоковая орбита $\omega_{r_1, r_2}^{j_1, j_2}$ диффеоморфизма $f_{b_{r_1}^h}$ соответствует источниковой орбите $\alpha_{r_1, r_2}^{j_1, j_2}$ диффеоморфизма $f_{b_{r_2}^{j_2}}$. Обе орбиты имеют одинаковый период $k_{r_1, r_2}^{j_1, j_2}$.

Результирующий диффеоморфизм $f: M^2 \rightarrow M^2$ представляет из себя склейку динамики вдоль всех ребер графа Γ .

СПИСОК ЛИТЕРАТУРЫ

1. Каток А. Б., Хасселблатт Б. Введение в современную теорию динамических систем. — М.: Факториал, 1999. — 767 с.
2. Смейл С. Дифференцируемые динамические системы // Успехи мат. наук. — 1970. — Т. 25, В. 1. — С. 113–185.
3. Smale S. The Ω -stability theorem // Same Proceedings. — 1970. — Vol. 14. — P. 289–297.

4. Williams R. The “ DA ” maps of Smale and structural stability // Global Analysis (Proc. Sympos. Pure Math., Vol. XIV, Berkeley, Calif., 1968), Amer. Math. Soc. – 1970. – P. 329–334.
5. Карпов Д. В. Теория графов. – СПб.: Санкт-Петербургское отделение Математического института им. В. А. Стеклова РАН. – 2017. – 482 с.

ФИРСОВА С. А., ВДОВИН П. С., КАРПУШКИНА И. С.

**ПРИМЕНЕНИЕ ТЕХНОЛОГИИ БЛОКЧЕЙН
ДЛЯ ДИСТАНЦИОННОГО ГОЛОСОВАНИЯ**

Аннотация. В статье рассматривается созданный авторами прототип программной системы, предназначенной для принятия коллегиальных решений путем проведения дистанционного голосования на основе технологии блокчейн. Реализовано как открытое, так и тайное голосование, автоматизировано получение отчетной документации по результатам голосования, а также предусмотрена возможность генерации протоколов заседаний, выписок из протоколов и т. п.

Ключевые слова: прототип программной системы, системы дистанционного голосования, технология блокчейн, диаграммы UML.

FIRSOVA S. A., VDOVIN P. S., KARPUSHKINA I. S.

APPLICATION OF BLOCKCHAIN TECHNOLOGY FOR REMOTE VOTING

Abstract. The article presents a software system prototype developed by the authors and designed for making collective decisions by remote voting based on blockchain technology. As a result, both open and secret voting have been implemented. The receipt of reporting documents based on voting results has been automated. It is also possible to generate minutes of meetings, extracts from minutes, etc.

Keywords: software system prototype, remote voting systems, blockchain technology, UML diagrams.

Введение. Дистанционное голосование – процесс принятия решения, путём обычного голосования, с применением специальных электронных средств голосования и технических электронных средств для подсчета голосов и оглашения результата.

Проанализировав существующие программные решения для систем дистанционного голосования, можно сформулировать следующие недостатки:

- недостаточная прозрачность процесса голосования;
- недостаточная отказоустойчивость системы;
- недостаточная устойчивость к взлому.

Поэтому всё чаще для программных систем подобного назначения предлагается использовать протокол на основе технологии блокчейн, который поможет устранить сформулированные выше недостатки.

Блокчейн – это распределенная база данных, которая может храниться на рабочих станциях, географически удаленных друг от друга. Блокчейн можно рассматривать как цепочку блоков с тремя столбцами, где каждая строка представляет отдельную транзакцию (операцию), которую совершил участник сети. В первом столбце хранится метка времени о транзакции; во втором – сведения о транзакции; третий столбец содержит в себе свой хэш и хэш предыдущего блока, так все блоки представляют собой единую цепочку, то есть историю всех действий в системе с начала её старта. Транзакция является основным событием в технологии блокчейна, которое разрешено базовым протоколом. При проведении этого события идет обновление информации в блоке цепочки, передающееся к блокчейн-сети по всему миру.

Среди базовых принципов технологии блокчейн выделяют:

- распределенность – нет единой централизованной базы данных, взломав которую, можно удалить или фальсифицировать информацию;
- безопасность – даже если удастся взломать несколько блоков и изменить в них данные, то безопасность системы не пострадает. Для изменения блокчейн-системы необходимо взломать все блоки, что маловероятно. Любая попытка взлома будет замечена участниками сети, кроме того, система защищена шифрованием;
- прозрачность – вся база данных открыта, посмотреть данные блока может любой желающий, т.е. увидеть транзакцию может каждый, однако узнать непосредственных участников транзакции будет возможно, только если они сами пожелают обнародовать этот факт;
- монолитность – технология дает возможность обмениваться данными напрямую между отправителем и получателем. Подлинность операций в системе проверяется исключительно участниками транзакции.

Архитектура прототипа программной системы для проведения дистанционного голосования. Авторами статьи был создан прототип программной системы, предназначенной для принятия коллегиальных решений путем проведения дистанционного голосования на основе технологии блокчейн. Данный прототип позволяет проводить дистанционное голосование (открытое и тайное), а также генерировать различную отчетную документацию по результатам голосования для заседаний кафедр, советов факультетов, ректората и т.п. в Мордовском государственном университете.

Архитектура разработанного прототипа описана с помощью диаграмм UML, при этом можно выделить следующие компоненты спецификации системы: концептуальную модель, модель размещения, модель реализации.

Для описания концептуальной модели была разработана диаграмма вариантов использования:

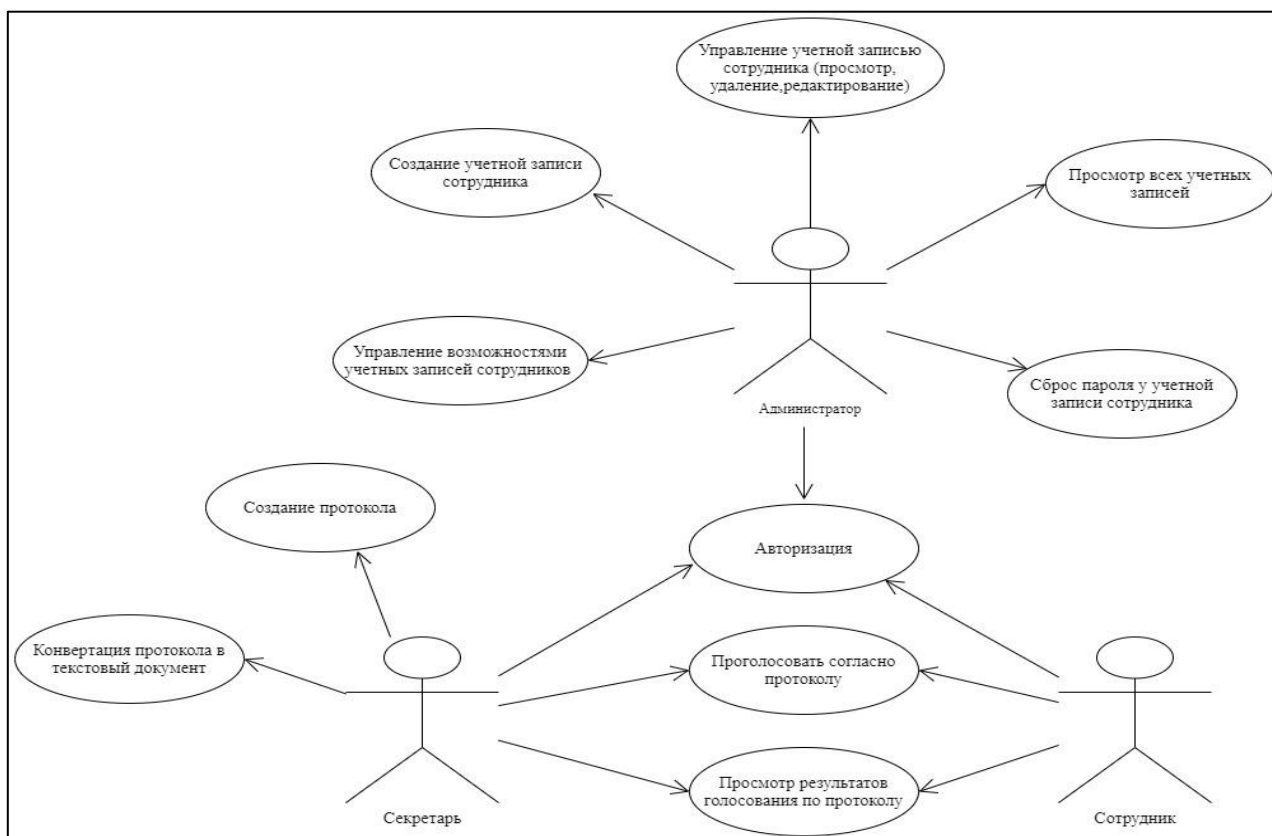


Рис. 1. Диаграмма вариантов использования.

В результате моделирования были выявлены следующие актеры:

- администратор – пользователь, осуществляющий управление учетными записями сотрудников;
- секретарь – пользователь, выполняющий функции создания заседаний пользователей, осуществляющий работу с протоколами голосования, а также формирующий различные выписки из протоколов заседаний;
- сотрудник – пользователь, осуществляющий голосование согласно протоколу.

Модель размещения описывает варианты физического размещения элементов системы, для представления этой модели удобно использовать UML-диаграмму развертывания (см. рис. 2).

Как видно из диаграммы развёртывания – результаты голосования по протоколам передаются на сервер, а затем сохраняются в базе данных. База данных создана и в настоящий момент управляется СУБД MS SQL Server 2019.

При тайном голосовании результаты помещаются в хранилище блокчейн- платформы Waves, а затем считываются программой сервера и дублируются в базу данных.

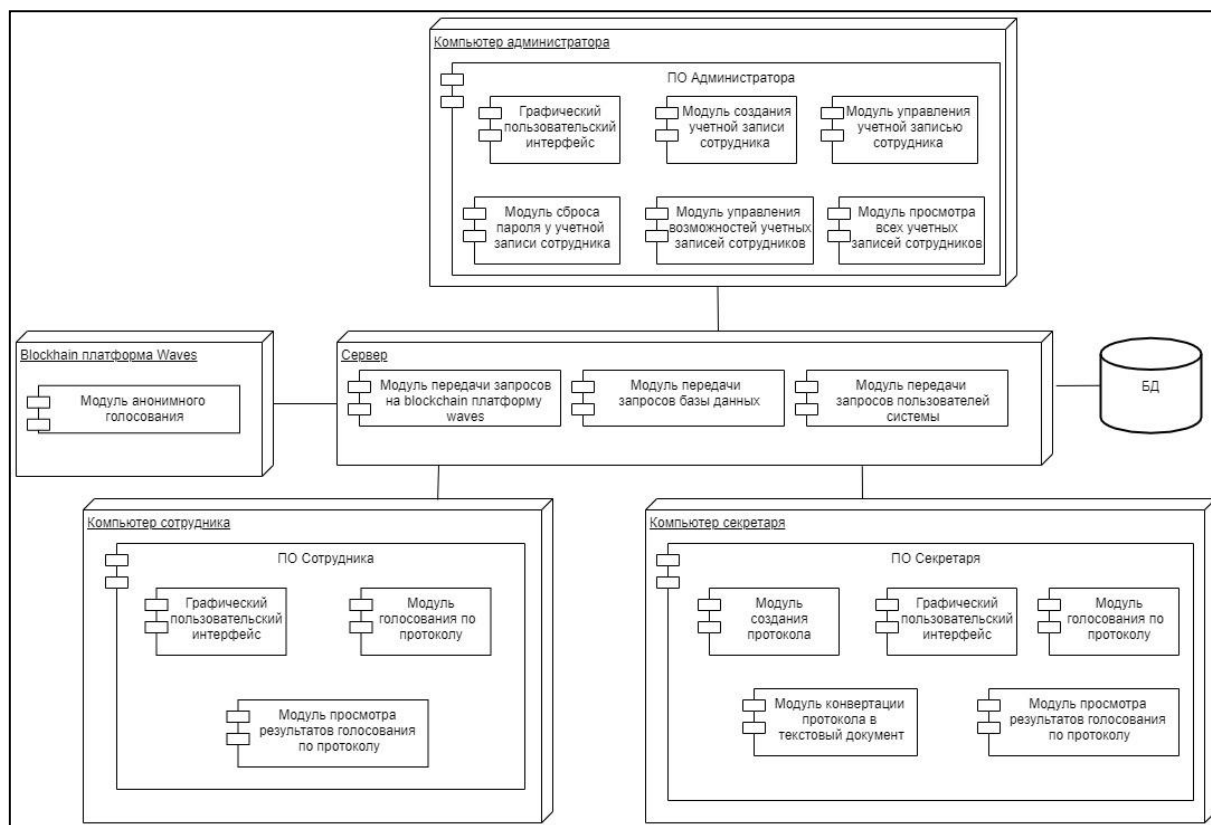


Рис. 2. Диаграмма развертывания.

Модель реализации описывает разделение системы на отдельные компоненты, независимые задачи, подпрограммы, информационные и управляющие потоки и связи между элементами системы. Для представления модели реализации используется диаграмма компонентов, представленная на рисунке 3.

Архитектура программной системы для принятия коллегиальных решения делится на три уровня:

- уровень представления информации;
- уровень логики;
- уровень данных.

На уровне представления информации осуществляется взаимодействие системы с пользователем. Через графический пользовательский интерфейс производится ввод данных в систему и вывод запрашиваемой пользователем информации.

Все вычисления системы производятся на уровне логики. Здесь обрабатываются команды и запросы, принимаемые от пользователя системы.

Уровень данных содержит базу данных, в которой находятся все данные о пользователях системы, протоколах, результатов голосований по протоколам. Также в этот уровень входит хранилище данных блокчейн-платформы Waves, которое содержит результаты при тайном голосовании.

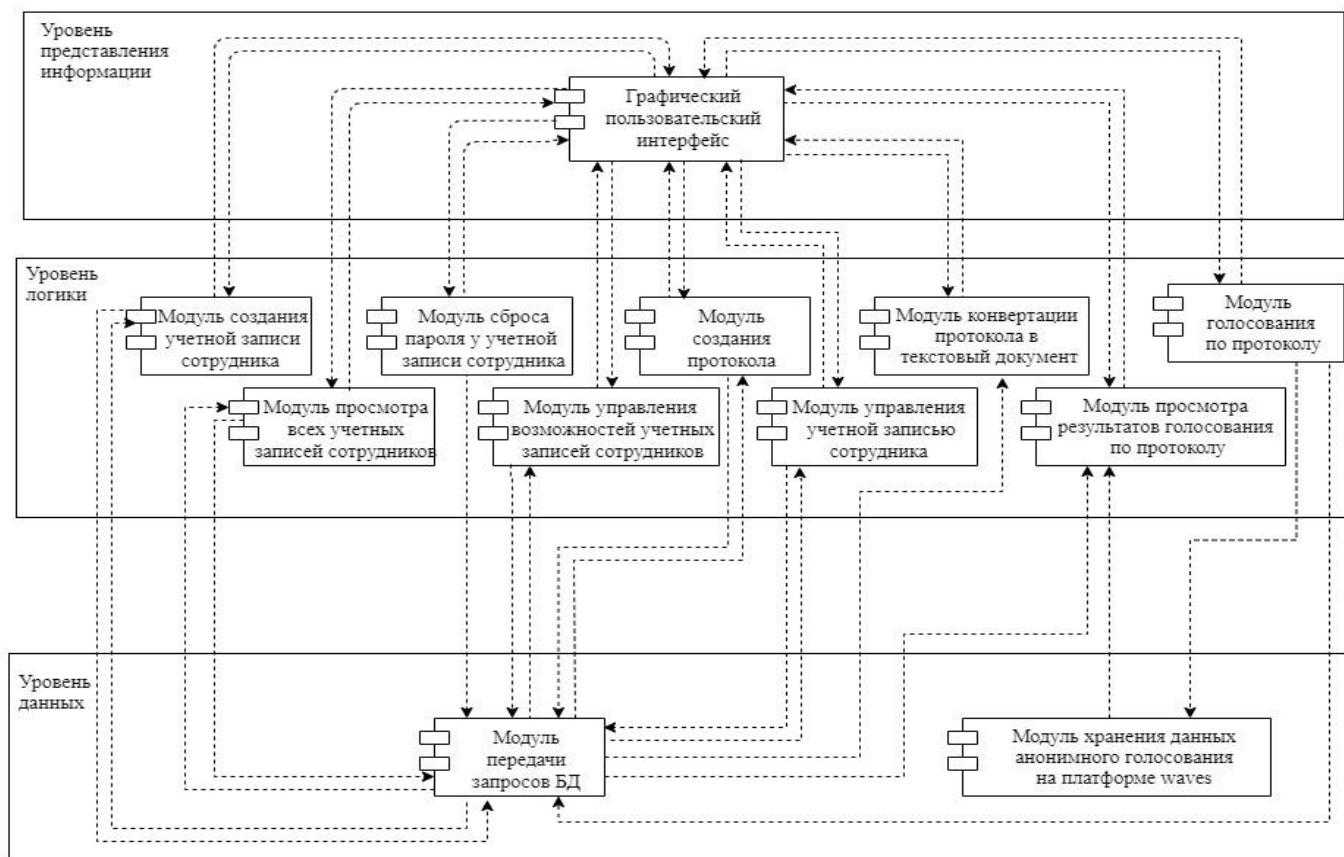


Рис. 3. Диаграмма компонентов.

Для хранения информации системы была выбрана система управления базами данных (далее СУБД) MS SQL Server 2019. База данных прототипа программной системы для принятия коллегиальных решений содержит 5 таблиц:

- «users» (Пользователи): содержит сведения о пользователях системы;
- «info_protocol» (Протоколы): содержит подробную информацию о протоколах;
- «answers_protocol» (Ответы на протоколы): содержит ответы пользователей на вопросы, по которым проходит голосование по протоколам;
- «user_protocol» (Доступ для пользователей): содержит информацию о протоколах, которые доступны определённому пользователю;
- «result_protocol» (Результаты голосования): содержит результаты по завершившимся голосованиям.

Схема базы данных представлена на рисунке 4.

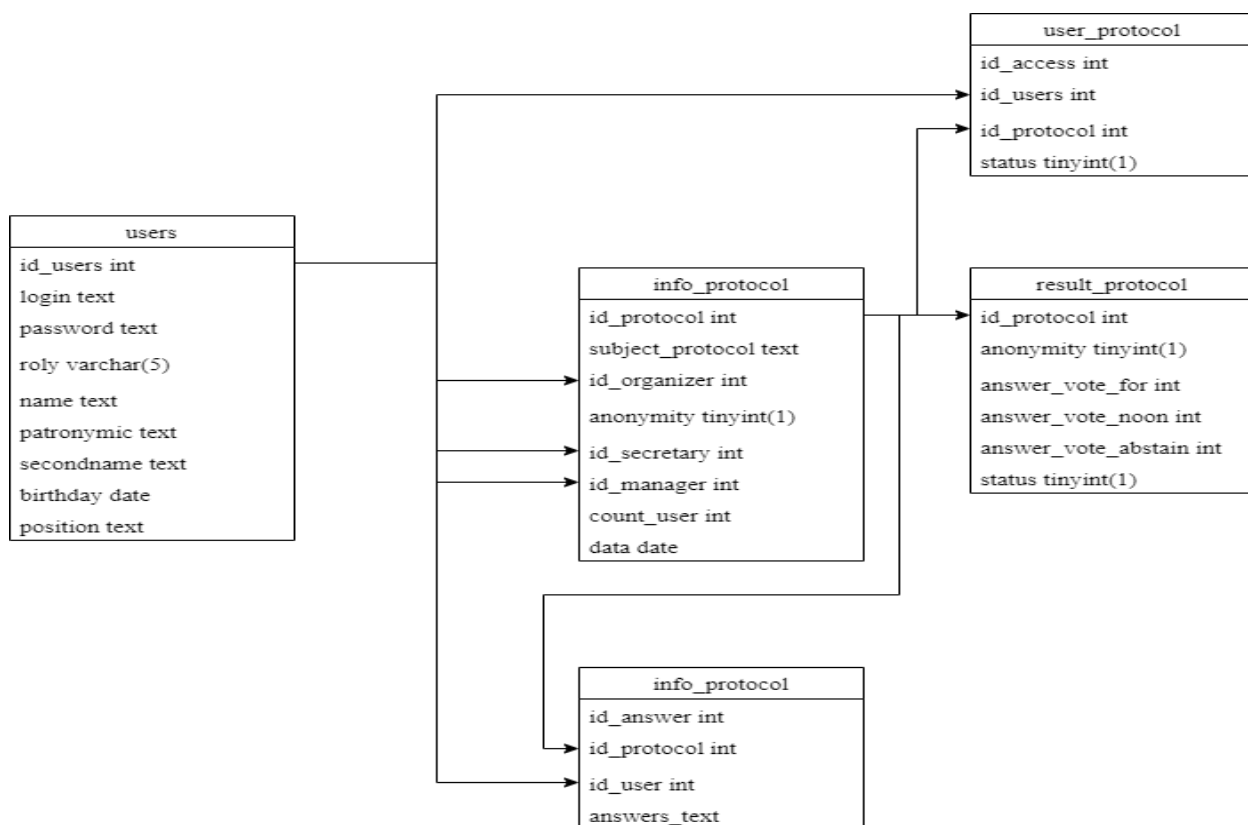


Рис. 4. Схема базы данных.

Технологии, использованные при разработке прототипа. При разработке прототипа программной системы использовались технология Windows Forms, платформы Microsoft.NET Framework и технология блокчейн на платформе Waves.

В настоящее время существует значительное количество реализаций блокчейнов, среди которых присутствуют такие платформы, как Bitcoin, Waves, Ethereum, zCash, Ripple, Stellar, IOTA и др. Каждая из этих платформ имеет свои положительные черты, но при разработке децентрализованных приложений следует обратить своё внимание на платформу Waves, поскольку платформа имеет тестовый режим блокчейна, и свою собственную среду разработки.

При разработке децентрализованного приложения можно выделить 2 основных этапа: написание смарт-контракта, развертывание смарт-контракта в блокчейн сети Waves. Смарт-контракт играет ключевую роль в разработке модуля, на нем описываются все функции, которые будут позволять пользователю выполнять те или иные манипуляции с данными.

Смарт-контракт – протокол взаимодействия между пользователями, пользователем и компьютером или компьютером и компьютером в соответствии с правилами, определенными в исходном коде смарт-контракта.

Waves IDE – онлайн-среда для разработки и тестирования смарт-контрактов на языке Ride. Основными возможностями Waves IDE являются:

- наличие библиотеки смарт-контрактов и тестов;
- компиляция и установка Ride-скриптов;
- подписание и отправка транзакций;
- наличие интерактивной консоли JavaScript со встроенными функциями работы с блокчейном;
- запуск JavaScript-тестов;
- отправка ссылки на файл.

Работа с Waves IDE. Для разработчиков платформа Waves создала тестовую блокчейн-сеть, для отладки своих смарт-контрактов. Основной особенностью тестовой сети является то, что не нужно разворачивать узел блокчейн-сети на своей виртуальной машине. Тестовая блокчейн-сеть в связке с IDE Waves является отличным инструментом разработки смарт-контракта.

Для начала разработки необходимо создать новый аккаунт в тестовой блокчейн-сети Waves, после чего создается смарт-контракт, в котором прописываются основные функции.

Основная функция будущего смарт-контракта – это функция верификации транзакций, которая позволяет обезопасить смарт-контракт от несанкционированного доступа. Для этого используется аннотация `Verifier` языка Ride. В аргументе аннотации находится переменная, которая содержит информацию о транзакции. После аннотации реализуется функция авторизации `ferify`, обычно в ней разрешают определенные типы транзакций, и происходит проверка публичного ключа, если транзакция типа `SetScriptTransaction`, пытается внести изменений в код смарт-контракта, которой расположено в блокчейн-сети.

```
@Verifier(tx)
func ferify() =
{ match (tx)
  {case t: SetScriptTransaction => sigVerify(tx.bodyBytes, tx
.proofs[0], ownerpublickey)
   case c: DataTransaction => true
   case d: InvokeScriptTransaction => true
   case => false}}
```

Рис. 5. Функция верификации транзакций `ferify`.

После успешной верификации транзакции, нужно её обработать, для этого используем аннотацию `Callable`. В аргументе аннотации, находится переменная, которая содержит информацию о верифицированной транзакции. После аннотации, реализуется функция обработки транзакции `vote`, она содержит единственный аргумент, типа `string`, в которой будет размещен номер протокола, и выбора ответа на него. Функция определяет,

голосовал ли раньше пользователь, если да, то функция выводит сообщение об ошибке, и не записывает ответ. Если пользователь не голосовал ранее, то функция записывает его ответ в Data Transactions. Смарт-контракт оплачивает комиссию пользователя, при отправке транзакции.

```
Callable(InvokeScriptTransaction)
func vote(str: String) =
{let dataFromStorage = this.getString
(InvokeScriptTransaction.callerPublicKey.toBase58String())
if (dataFromStorage.isDefined())
    then throw("Вы уже голосовали в данном опросе!")
    else
{ScriptResult(WriteSet([DataEntry
(InvokeScriptTransaction.callerPublicKey.toBase58String(), str)]),
TransferSet([ScriptTransfer
(InvokeScriptTransaction.caller, InvokeScriptTransaction.fee, unit
)]))}}
```

Рис. 6. Функция голосования vote.

Для развертывания смарт-контракта в тестовую блокчейн-сеть Waves, нам понадобится некоторое количество токенов Waves, чтобы оплатить комиссию при отправке транзакции. Разработчики платформы решили эту проблему, создав инструмент Waves Explorer, где можно бесплатно получить токены, и просмотреть все транзакции связанные с смарт-контрактом.

Для получения токенов нужно войти в меню инструмента, выбрать тестовую сеть «Testnet», в списке инструментов выбрать «Faucet», как показано на рисунке 7.

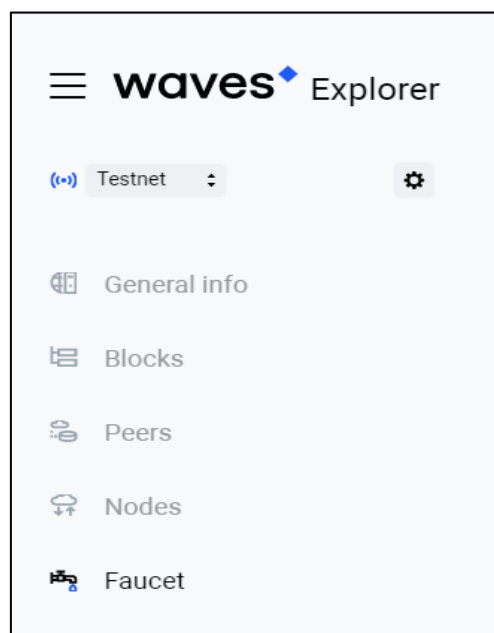


Рис. 7. Получение токенов в инструменте Waves Explorer.

После проделанных манипуляций открывается окно, в котором нужно ввести адрес смарт-контракта (см. рис. 8). Адресом является адрес аккаунта в тестовой блокчейн-сети Waves. После ввода адреса и проверки пользователя на робота, нужно нажать на кнопку «Request 10 Waves».

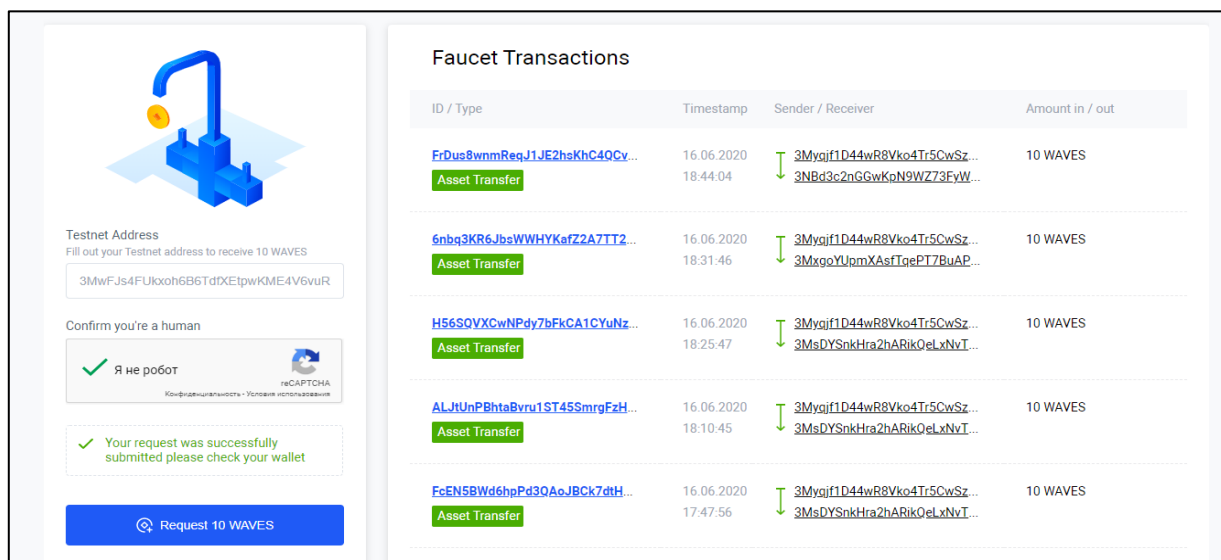


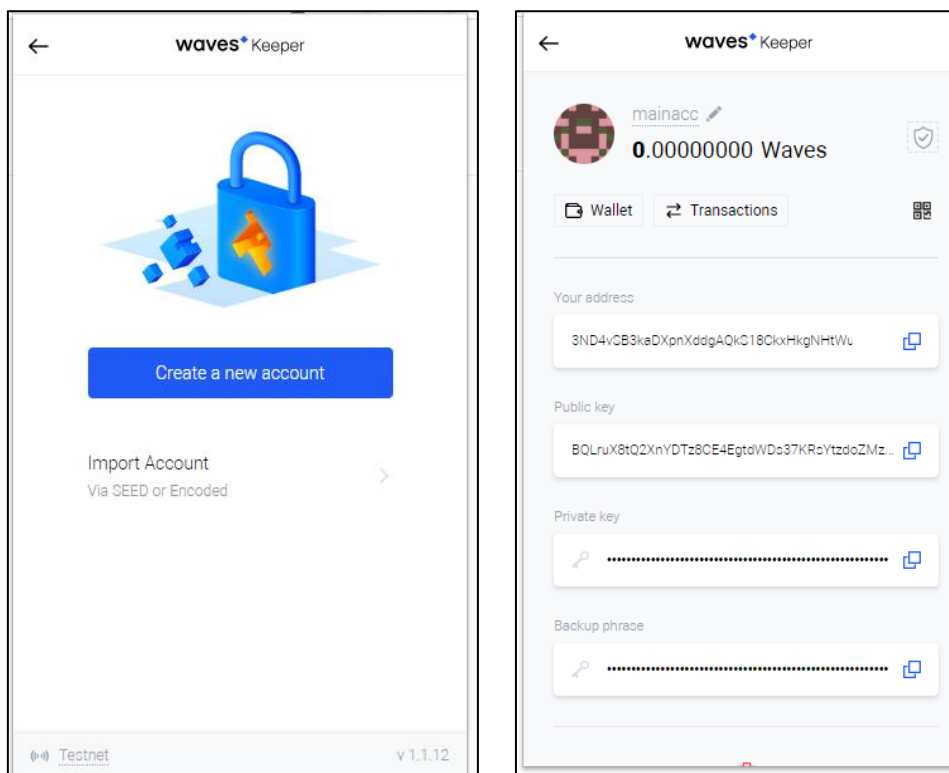
Рис. 8. Успешное получение токенов в инструменте Waves Explorer.

Полученные токены отразятся на аккаунте Waves в IDE.

После получения токенов, возможно развернуть созданный смарт-контракт в тестовую блокчейн-сеть Waves. Для этого нужно нажать на кнопку «Deploy» в IDE Waves, после нажатия открывается окно с подробным описанием транзакции, которая содержит всю основную информацию о смарт-контракте.

Для отправки транзакции следует с начала подписать транзакцию, для этого нужно нажать на кнопку «Add Sign», после этого для публикации транзакции в блокчейн-сеть необходимо нажать на кнопку «Publish». Далее при успешном развертывании транзакции в блокчейн-сеть появляется соответствующее уведомление.

Для проверки функциональности создадим тестовый протокол с тайным голосованием на несколько сотрудников. Для сотрудников нужно создать личные аккаунты на блокчейн-платформе Waves. Для этого существует браузерное расширение Waves Keeper, создание аккаунта в нем показано на рисунке 9а. Для дальнейшего голосования понадобится Private key созданного аккаунта, он находится в блоке с основной информацией об аккаунте (см. рис. 9б).



а)

б)

Рис. 9. а) создание аккаунта в Waves Keeper; б) основная информация созданного аккаунта.

Проголосуем с личных аккаунтов этих сотрудников. После окончания голосования, перейдём в инструмент Waves Explorer, и посмотрим всю информацию (см. рис. 10), которая находится в смарт-контракте, для этого нажмём на кнопку «Date».

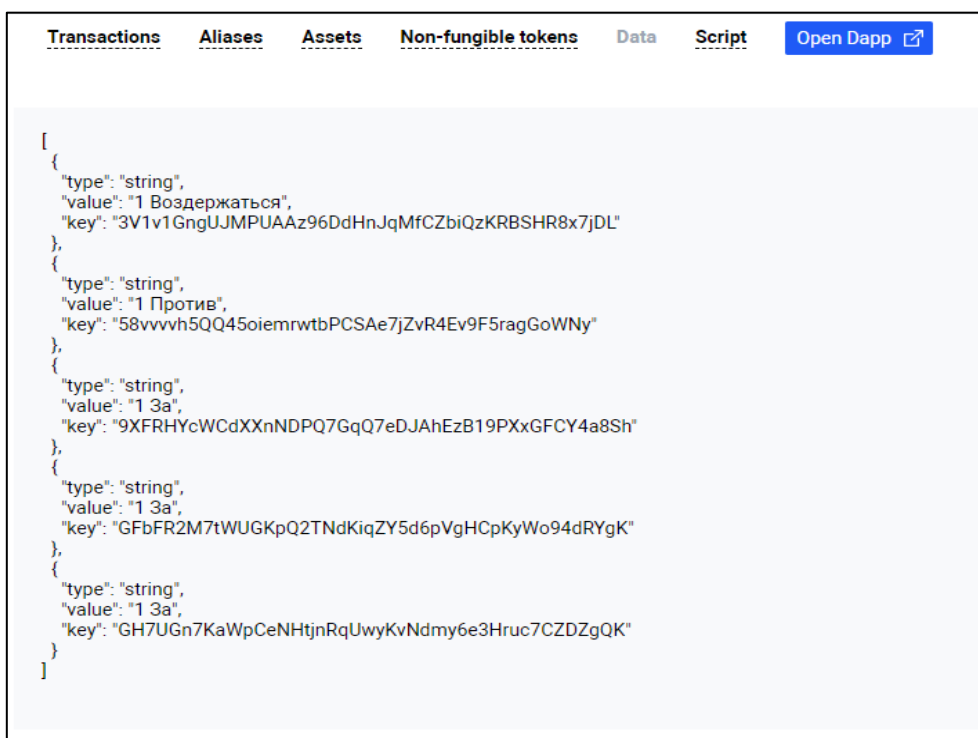


Рис. 10. Результат тестового голосования.

Заключение. Разработанный прототип программной системы позволяет проводить подразделениям университета открытое и тайное голосование по различным вопросам в дистанционном режиме и автоматизировать получение отчетной документации по результатам голосования, а также генерировать протоколы заседаний, выписки из протоколов и т.п.

СПИСОК ЛИТЕРАТУРЫ

1. Генкин А. Блокчейн и электронное голосование: о технологиях децентрализованной демократии // Самоуправление. – 2017. – № 2 (107). – С. 44–48.
2. Кутейников Д. Л. Особенности применения технологий распределенных реестров и цепочек блоков (блокчейн) в народных голосованиях // Актуальные проблемы российского права. – 2019. – № 9 (106). – С. 41–52.
3. Алексеев Р. А., Абрамов А. В. Проблемы и перспективы применения электронного голосования и технологии избирательного блокчейна в России и за рубежом // Гражданин. Выборы. Власть. – 2020. – № 1 (15). – С. 9–21.