

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
«ПОЛОЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

**ЭЛЕКТРОННЫЙ СБОРНИК
ТРУДОВ МОЛОДЫХ СПЕЦИАЛИСТОВ
ПОЛОЦКОГО ГОСУДАРСТВЕННОГО
УНИВЕРСИТЕТА**

Выпуск 25 (95)

ПРОМЫШЛЕННОСТЬ

Информационные технологии

Машиностроение

Химические технологии

Обновляется 1 раз в год

Новополоцк

ПГУ

2018

РЕДАКЦИОННАЯ КОЛЛЕГИЯ

Главный редактор:
проф., д.т.н. Д.Н. Лазовский

Заместитель главного редактора:
доц., к.т.н. Д.О. Глухов

Председатель ред. совета проф., д.т.н. В.П. Иванов
Зам. председателя ред. совета проф., д.т.н. Н.Н. Попок

Информационные технологии:

доц., к.т.н. В.Ф. Янушкевич
доц., к.т.н. Р.П. Богуш
доц., к.ф.-м.н. С.А. Вабищевич
доц., к.ф.-м.н. А.А. Козлов

Машиностроение:

доц., к.т.н. А.В. Дудан
доц., к.т.н. Т.В. Вигерина
доц., к.т.н. А.М. Долгих
доц., к.т.н. В.Э. Завистовский

Химическая техника и технологии:

доц., к.т.н. А.В. Митинов
доц., х.т.н. С.Ф. Якубовский
доц., к.пед.н. И.В. Бурая
доц., к.т.н. А.А. Ермак

ЭЛЕКТРОННЫЙ СБОРНИК ТРУДОВ МОЛОДЫХ СПЕЦИАЛИСТОВ ПОЛОЦКОГО ГОСУДАРСТВЕННОГО УНИВЕРСИТЕТА [Электронный ресурс]. – Новополоцк : Полоцкий государственный университет, 2018. Вып. 25 (95). Промышленность. – 1 электрон. опт. диск.

Издается с 2014 года (в печатном виде – с 2004 г.)

Сборник включен в Государственный регистр информационного ресурса. Регистрационное свидетельство № 3201404563 от 30.11.2014 г.

В сборнике представлены труды молодых специалистов Полоцкого государственного университета по направлениям прикладных исследований, проводимых в университете: информационные технологии, машиностроение, химические технологии.

Предназначен для работников науки, образования и производства, будет полезен студентам, магистрантам и аспирантам университетов.

Учредитель: учреждение образования «Полоцкий государственный университет»
211440, ул. Блохина, 29, г. Новополоцк, Беларусь
тел. 8 (0214) 39 40 46, e-mail: vestnik@psu.by

Компьютерный дизайн *М. С. Мухоморовой*

Техническое редактирование и компьютерная верстка *О. П. Михайловой,*

Д. М. Севастьяновой, Т. А. Дарьяновой

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

УДК 004.051

ЦЕЛЕСООБРАЗНОСТЬ РАЗРАБОТКИ ИГРОВОГО ПРИЛОЖЕНИЯ «FLAMY – НЕОБЫЧНАЯ ГОЛОВОЛОМКА» ДЛЯ МОБИЛЬНЫХ УСТРОЙСТВ ПОД УПРАВЛЕНИЕМ ОПЕРАЦИОННОЙ СИСТЕМЫ ANDROID

А.П. ЛАБУЗОВА

(Представлено: канд. физ.-мат. наук, доц. О.В. ГОЛУБЕВА)

Рассматривается целесообразность разработки игрового приложения «Flamy – необычная головоломка» для платформы Android. Показано, что разработка игрового приложения «Flamy – необычная головоломка» целесообразна, так как платформа занимает ведущие позиции на рынке мобильных приложений, а игровые приложения жанра «Головоломки» входят в топ самых популярных.

Мобильные телефоны играют важную роль в жизни современного человека. Высокий ритм жизни требует от нас большей мобильности, именно поэтому практически у каждого сейчас есть мобильный телефон. На данной стадии технического развития это уже не просто средство связи, а карманный персональный компьютер. Пользователей смартфонов с каждым годом становится все больше. Люди не просто носят их с собой, а активно ими пользуются: звонят, фотографируют, общаются в социальных сетях, просматривают новости, ищут информацию в Интернете, покупают товары, бронируют номера в отелях, заказывают такси, управляют своими финансами и, конечно же, играют.

На данный момент мобильная игровая индустрия развита очень хорошо и продолжает развиваться еще большими темпами. Каждый день на рынке появляется огромное количество новых продуктов, поэтому имеет смысл изучить популярность платформ, проанализировать спрос пользователей на игры различных жанров и изучить возможных конкурентов.

Согласно данным сайта w3schools.com [1] на август 2018 года около 81,3% мобильных устройств на данный момент работают именно под управлением Android, что делает разработку под эту платформу весьма привлекательной.

На рисунке 1 представлено процентное соотношение используемых операционных систем на мобильных устройствах.

Mobile Devices Statistics					
The values below are extracted from W3Schools' OS Platform Statistics :					
*iOS is the operating system for Apple mobile devices (like iPhone, iPad and iPod).					
2018	Total	iOS*	Android	Windows	Others
August	8.70 %	1.43 %	7.05 %	0.18 %	0.04 %

Рисунок 1. – Статистика используемых операционных систем на мобильных устройствах

Также разработку под Android привлекательной делает и тот факт, что комплект средств разработки для данной платформы находится в свободном доступе, и, следовательно, для того, чтобы приступить к созданию продукта, не потребуются больших денежных вложений.

Сейчас на рынке представлены игры различных категорий, по данным статистики AppBrain [2] самыми популярными в Google Play из них являются казуальные, симуляторы и головоломки. Рейтинг популярных категорий представлен на рисунке 2.

То, что головоломки входят в тройку самых востребованных мобильных игр, совсем неудивительно, ведь люди всегда любили разгадывать различного рода головоломки. Самыми древними головоломками известных человечеству на данный момент можно назвать логические задачи, найденные на стенах египетских пирамид, в древнегреческих манускриптах и в других исторических памятниках. Широкая же популяризация головоломок началась с «Пятнашек», и позднее подкрепилась «Кубиком Рубика» [3]. Что же касается компьютерных головоломок, то здесь одной из первых и заслуженно одной из самых популярных является «Тетрис».

Category	All apps	Average rating	Apps with >50K downloads	Paid apps	Average price	Low quality apps
Entertainment	190240	4.12	16399 (9%)	6458 (3%)	\$ 10.72	29990 (16%)
Personalization	144743	4.32	14192 (10%)	19928 (14%)	\$ 2.18	14233 (10%)
Tools	141175	4.06	14174 (10%)	10241 (7%)	\$ 6.06	11554 (8%)
Education	242358	4.27	12279 (5%)	18712 (8%)	\$ 5.00	30743 (13%)
Lifestyle	184473	4.22	10570 (6%)	5045 (3%)	\$ 16.67	32572 (18%)
Casual	63589	4.17	10275 (16%)	3001 (5%)	\$ 3.59	3776 (6%)
Simulation	30381	3.95	9691 (32%)	1271 (4%)	\$ 6.96	1105 (4%)
Photography	68779	4.08	8509 (12%)	1503 (2%)	\$ 5.78	9742 (14%)
Books & Reference	146179	4.32	8406 (6%)	10136 (7%)	\$ 6.65	28182 (19%)
Music & Audio	171659	4.35	8053 (5%)	5084 (3%)	\$ 3.70	38754 (23%)
Puzzle	74009	4.26	6834 (9%)	4783 (6%)	\$ 3.53	6362 (9%)
Action	33573	4.15	6067 (18%)	1616 (5%)	\$ 3.34	1916 (6%)
Finance	68269	4.07	5533 (8%)	2033 (3%)	\$ 37.00	5533 (8%)
Arcade	69358	4.34	4858 (7%)	3386 (5%)	\$ 2.82	4204 (6%)

Рисунок 2. – Рейтинг популярных категорий в Google Play

Современная индустрия в области головоломок стремительно прогрессирует. Каждый год на рынке появляются все новые сборники и конструкции, которые призваны держать интеллект человека в тонусе, развивать логику, тренировать нестандартное мышление и повышать интеллектуальный уровень в целом.

На основании того, что головоломки популярны и данная область стремительно прогрессирует, каждый день в Google Play появляется множество различных приложений в данной категории. Однако ниша головоломок, при решении которых пользователю необходимо использовать аппаратные и физические возможности своего мобильного устройства, по-прежнему практически пуста.

Поскольку смартфоны и планшеты всё прочнее входят в жизнь современного человека, вполне закономерно будет всё больший интерес к возможностям данных устройств со стороны пользователя. И игра-головоломка, которая сможет познакомить его поближе со своим устройством, сможет стать весьма полезной и востребованной.

Анализ рынка аналогов и конкурентов показал, что у разрабатываемого игрового приложения есть два аналога:

- «Неординарная головоломка – Раскуси меня!» [4].
- «Каверзный тест» [5].

К сильным сторонам «Неординарная головоломка – Раскуси меня!» можно отнести интересные, необычные и хорошо продуманные уровни. К слабым – чересчур скромный дизайн и плохо продуманная монетизация от рекламы. Также данное приложение уже более года не обновлялось и не расширялось.

Если анализировать приложение «Каверзный тест», то к его сильным сторонам можно отнести качественную графику, интересные уровни, хорошо продуманная монетизация через рекламу. Однако в нем почти отсутствуют такие задачи, при решении которых пользователь использует аппаратные и физические возможности своего мобильного устройства.

Поскольку в разрабатываемом приложении будет делаться упор именно на уровни, при прохождении которых используются аппаратные и физические свойства мобильного устройства, то в совокупности с приятным графическим интерфейсом и хорошо продуманной и лаконично встроенной монетизацией, можно говорить об актуальности разрабатываемого приложения.

После изучения популярности платформ, анализа спроса пользователей на игры различных жанров и изучения возможных конкурентов, можно прийти к выводу, что разработка игрового приложения «Flamy – необычная головоломка» для мобильных устройств под управлением операционной системы Android является целесообразной и что в дальнейшем данное приложение может пользоваться спросом у пользователей.

ЛИТЕРАТУРА

1. Mobile Devices Statistics [Электронный ресурс]. – Режим доступа: https://www.w3schools.com/browsers/browsers_mobile.asp. – Дата доступа: 14.09.2018.
2. Top categories. Most popular Google Play categories [Электронный ресурс]. – Режим доступа: <https://www.appbrain.com/stats/android-market-app-categories/>. – Дата доступа: 14.09.2018.
3. Головоломки – самые древние игры [Электронный ресурс]. – Режим доступа: http://pomozgui.ru/stati/?golovolomki_-_samie_drevnie_igri_na_zemle. – Дата доступа: 15.09.2018.
4. Неординарная головоломка – Раскуси меня! [Электронный ресурс]. – Режим доступа: <https://play.google.com/store/apps/details?id=ru.allyteam.puzzle>. – Дата доступа: 12.09.2018.
5. Каверзный тест [Электронный ресурс]. – Режим доступа: <https://play.google.com/store/apps/details?id=air.trickytest>. – Дата доступа: 12.09.2018.

УДК 004.415.2

**ПРОЕКТИРОВАНИЕ ИГРОВОГО ПРИЛОЖЕНИЯ
«FLAMY – НЕОБЫЧНАЯ ГОЛОВОЛОМКА» ДЛЯ МОБИЛЬНЫХ УСТРОЙСТВ
ПОД УПРАВЛЕНИЕМ ОПЕРАЦИОННОЙ СИСТЕМЫ ANDROID**

*А.П. ЛАБУЗОВА**(Представлено: канд. физ.-мат. наук, доц. О.В. ГОЛУБЕВА)*

Рассматривается проектирование игрового приложения «Flamy – необычная головоломка» для мобильных устройств под управлением операционной системы Android.

Игровое приложение «Flamy – необычная головоломка» предназначено для приятного и интересного времяпрепровождения пользователя. Пользователю будет предлагаться ряд уровней, в которых необходимо решить различные задачи и ребусы, используя не только свои логику и сообразительность, но и аппаратные и физические возможности устройства.

Определение основных функциональных возможностей приложения

При запуске приложения на экране отображается главное меню. Данное меню предоставляет пользователю следующие возможности:

- приступить к прохождению игры;
- оценить игру;
- выйти из игры.

Если пользователь переходит к прохождению игры, то открывается экран с текущим для игрока уровнем. На экране уровня предоставляется информация о количестве жизней и очков. Также пользователь здесь может получить доступ к меню подсказок и бонусов или вернуться в главное меню. При верном прохождении уровня, игрок получает доступ к следующему уровню. Следует также реализовать поздравление с успешным прохождением уровня и кратким описанием верного решения. В случае неверного решения «тратится» одна «жизнь». Если все остается нуль «жизней», то открывается диалоговое окно, предоставляющее следующий выбор:

- перейти на пять уровней назад;
- просмотреть рекламу для восстановления «жизней».

При выборе первого варианта текущий уровень уменьшается на пять, второго – происходит восстановление всех жизней, а текущий уровень остается прежним. В случае успешного прохождения всей игры следует реализовать отображение поздравления.

Выбор пункта главного меню «Оценить игру» пользователю предоставляется возможность дать оценку данной игре. При этом открывается экран, где можно выставить оценку от единицы до 5 включительно; если выставленная оценка будет от 1 до 3, то предлагается возможность написать разработчику о причине недовольства игрой, иначе – написать отзыв в Google Play. После оценки приложения пользователь получает дополнительные очки в благодарность, независимо от его решения.

Если пользователь выбирает пункт выхода из игры, то приложение должно закрыться.

Варианты использования

Диаграмма вариантов использования, как правило, является исходной моделью, с которой начинается процесс моделирования на языке UML. Она описывает функциональное назначение системы в самом общем виде с точки зрения всех ее пользователей и заинтересованных лиц.

При создании диаграмм вариантов использования нужно выделить два вида сущностей – актеры и прецеденты. Актер – это множество логически связанных ролей, исполняемых при взаимодействии с прецедентами. Прецедент – это описание множества последовательных событий, включая варианты, выполняемых системой, которые приводят к наблюдаемому актером результату [2].

Для игры-головоломки можно выделить одного актера - игрок. Основные требования к разрабатываемому приложению отображены с помощью диаграммы вариантов использования, представленной на рисунке 1.

Игрок может взаимодействовать с программой различными образами: проходить и оценивать игру. При прохождении игры он может использовать подсказки, аппаратные и физические возможности своего мобильного устройства.

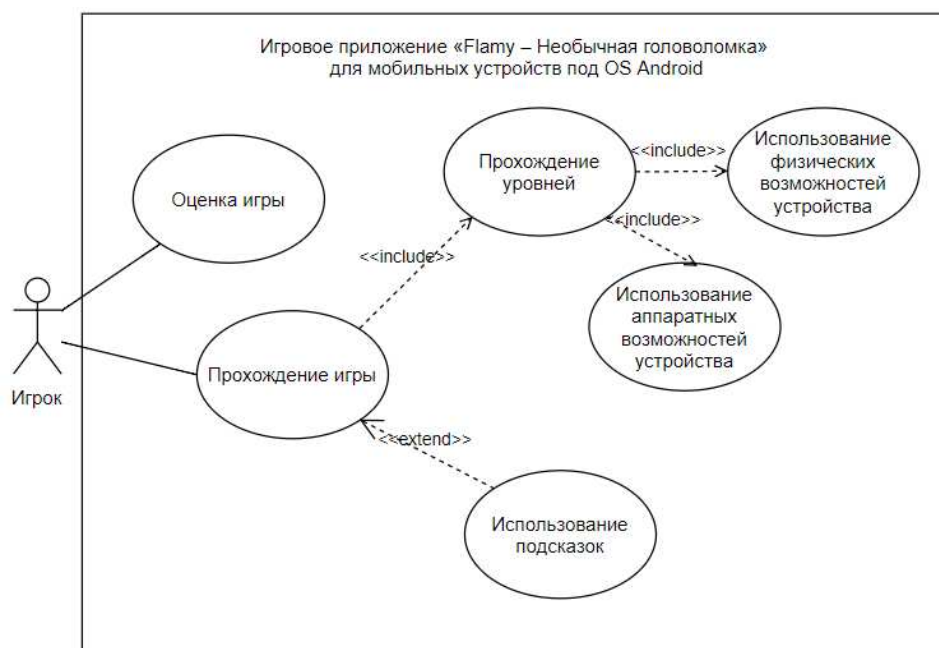


Рисунок 1. – Диаграмма вариантов использования

Разработка пользовательского интерфейса

При проектировании приложения немаловажную часть занимает разработка пользовательского интерфейса. Пользовательский интерфейс является своеобразным коммуникационным каналом, по которому осуществляется взаимодействие пользователя и его мобильного устройства. Поэтому от качества проектирования пользовательского интерфейса во многом зависит эффективность работы с приложением.

Немаловажной задачей является разработка графического интерфейса пользователя. Игровое приложение разрабатывается для мобильных устройств под управлением ОС Android, поэтому оно будет строиться согласно спецификации Material Design.

Под графическим интерфейсом пользователя (Graphical User Interface - GUI) подразумевается тип экранного представления, при котором пользователь имеет произвольный доступ ко всем видимым экранным объектам (элементам интерфейса) и осуществляет непосредственное манипулирование ими. Чаще всего элементы интерфейса в GUI реализованы на основе метафор и отображают их назначение и свойства, что облегчает понимание и освоение программ неподготовленными пользователями.

Графический интерфейс пользователя является частью пользовательского интерфейса и определяет взаимодействие с пользователем на уровне визуализированной информации [1].

Любое приложение под Android представляет собой набор Activity и их взаимодействие между собой. Activity представляют собой экраны, которые создаются при помощи XML разметки.

Для удобной работы пользователя с игрой разработаем дружелюбный графический интерфейс, который поможет игроку безошибочно ориентироваться в приложении. Основными Activity являются главное меню, экран с уровнем и меню подсказок. Чтобы наглядно представить внешний вид данных экранов, создадим их прототипы. Прототип главного меню представлен на рисунке 2.

Как видно из рисунка 2 экран главного экрана должен представлять собой следующее:

- крупный логотип с названием игры по центру;
- три крупных кнопки, расположенных вертикально друг за другом под логотипом с названием игры.

Логотип статичен и некликабелен.

При нажатии условной кнопки для перехода к прохождению игры должен открыться экран с уровнем. Его прототип представлен на рисунке 3.

Теперь на основе макета, предоставленного на рисунке 3, разработаем концепцию графического пользовательского интерфейса. Как видно из рисунка сверху располагается информация о текущем количестве «жизней» и количестве очков игрока, а также кнопка для возврата в главное меню. Ниже располагается основная область уровня, где находится задание, его условие и кнопка перехода в меню подсказок и бонусов. Еще ниже располагается место для рекламы. При включенном Интернете там будет отображаться баннер рекламы, иначе – однотонная пустая область.



Рисунок 2. – Прототип главного меню



Рисунок 3. – Прототип экрана для уровней игры

При нажатии на условную кнопку перехода в меню подсказок и бонусов, открывается диалоговое окно с данным меню. Его прототип представлен на рисунке 4.



Рисунок 4. – Прототип меню подсказок и бонусов

Исходя из макета, представленного на рисунке 4, разработаем концепцию графического пользовательского интерфейса. Меню подсказок и бонусов представляет собой диалоговое окно, состоящее из надписи «Меню» и четырех кнопок, расположенных друг под другом:

- «Подсказка -10»;
- «Пропустить -30»;
- «Видео рекламы +10»;
- «Заккрыть».

При нажатии на условную кнопку «Подсказка -10» отображается небольшое диалоговое окно с текстом подсказки и кнопкой «X» для закрытия, а также происходит вычитание десяти очков из текущего счета игрока. Условная кнопка «Пропустить -30» позволяет игроку пропустить текущий уровень

и перейти к следующему, при этом из актуального счета игрока вычитается тридцать очков. Если игрок кликает на условную кнопку «Видео рекламы +30», то на весь экран воспроизводится видео с рекламой после просмотра данного видео игрок получает плюс тридцать очков к своему текущему счету. При нажатии на условную кнопку «Закрыть» окно закрывается, и игрок возвращается к прохождению уровня.

Таким образом, в данной статье были определены основные функциональные возможности, составлена диаграмма вариантов использования, а также разработан пользовательский интерфейс игрового приложения «Flamy – необычная головоломка».

ЛИТЕРАТУРА

1. Графический интерфейс пользователя [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/Графический_интерфейс_пользователя. – Дата доступа: 16.09.2018.
2. Лекция 3 : Элементы графической нотации диаграммы вариантов использования [Электронный ресурс]. – Режим доступа: <https://www.intuit.ru/studies/courses/32/32/lecture/1004%25252525253Fpage%25252525253D2/>. – Дата доступа: 16.09.2018.

УДК 004.738.5

КРАТКИЙ ОБЗОР БИБЛИОТЕКИ MATERIAL DESIGN LITE**А.В. ВОЙТОВ***(Представлено: канд. техн. наук, доц. А.Ф. ОСЬКИН)**Рассматриваются возможности библиотеки Material Design Lite.*

При разработке приложения интерфейс играет очень важную роль. Так как именно по интерфейсу пользователь оценивает программу в целом. Чтобы сделать его приятным для человеческого глаза используют разные подходы при проектировании дизайна. Одним из таких является использование концепта Material Design от Google.

Material Design Lite (MDL) это библиотека компонентов для веб-разработчиков на основанная на философии Google Material Design. Данная библиотека занимает очень мало места, поэтому позволяет загружаться сайтам очень быстро. Ее можно использовать при оформлении статей, блогов, сайтов-визиток и т. п. Она отлично подойдет, если нужно просто добавить стиль Material Design.

При построении интерфейса первым делом необходимо определиться с макетом. Для этого Material Design Lite предлагает некоторые способы по оформлению частей навигации, основной сетки, футера.

Примеры представлены на рисунках 1 и 2.

**Рисунок 1. – Шапка сайта****Рисунок 2. – Выезжающее меню слева**

MDL позволяет веб-приложению адаптироваться под различные размеры экранов. Для этого можно использовать компонент-контейнер grid. Его можно представить в виде сетки, которая имеет 12 колонок для компьютеров, 8 для планшетов и 4 для телефонов. Для каждого внутреннего элемента можно установить ширину: количество колонок, которое будет занимать элемент. При чем для каждого экрана ширину можно установить разной. На рисунках 3 и 4 представлено как будут отображаться элементы на планшете и смартфоне.

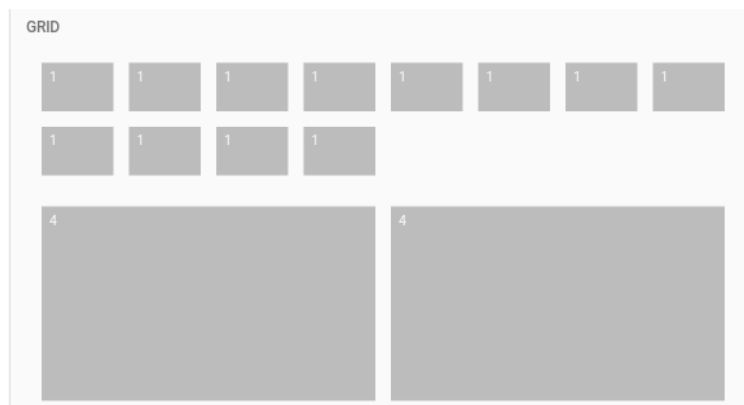


Рисунок 3. – Расположение элементов на планшете

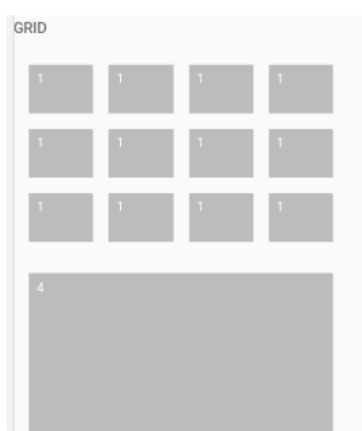


Рисунок 4. – Расположение элементов на смартфоне

Важным элементов в MDL являются карточки (Card), который представляет собой виртуальный лист бумаги содержащий такие данные как фото, текст, ссылка. Карточки позволяют удобно отображать контент, который состоит из различных типов объектов. А их использование в контейнере grid позволяет съезжать им на новую строку на меньших экранах, что и делает сайт адаптивным. Пример карточек изображен на рисунке 5.



Рисунок 5. – Карточки

Иногда необходимо предоставить пользователю способ выбрать опцию из некоторого списка, где так же элементы могут быть активные или нет. И на рисунке 6 представлено как это можно сделать, используя Material Design Lite.

Так же стоит упомянуть о таких элементах как кнопки, переключатели, текстовые поля (рис. 7). Все элементы анимированы, а кнопки имеют такое свойство как Ripple effect. Так же текстовые поля могут быть настроены на валидацию данных.

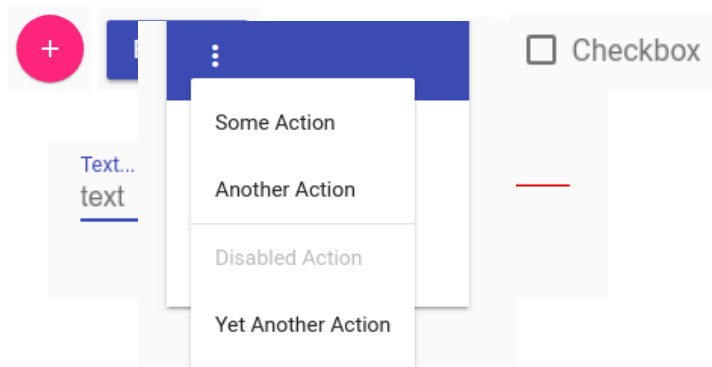


Рисунок 6. – Всплывающее меню

Одной из интересных особенностей является то, что MDL можно настроить путем изменения палитры цветов. Все что нужно это выбрать первичный и вторичный цвета в палитре цветов на официальном сайте (рис. 8).

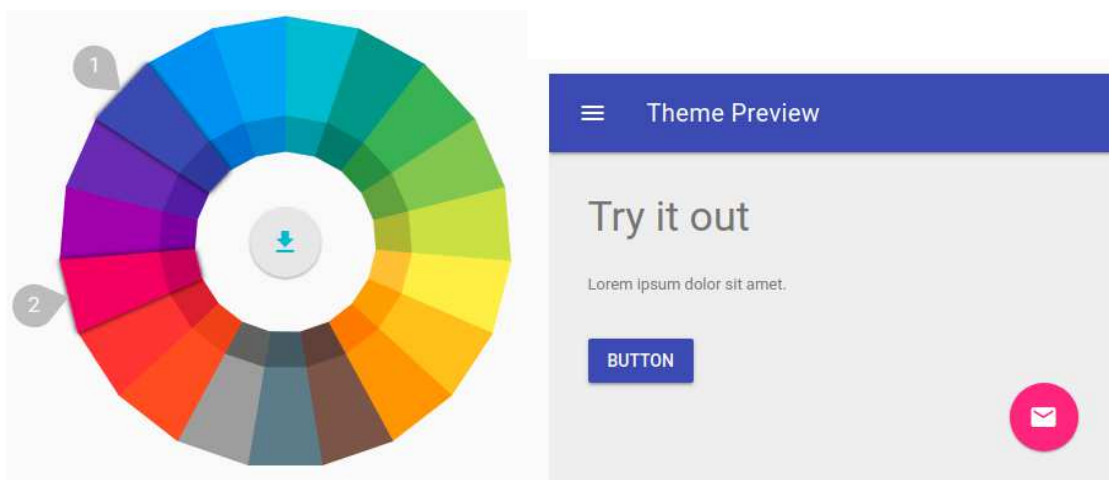


Рисунок 8. – Палитра цветов

В данной статье были рассмотрены некоторые возможности библиотеки Material Design Lite от Google, которые помогут оформить веб-приложение в стиле Material Design, зная только html, css и js.

ЛИТЕРАТУРА

1. Material Design [Электронный ресурс] / Википедия. – Режим доступа: https://ru.wikipedia.org/wiki/Material_Design. – Дата доступа: 08.09.18.
2. Material Design Lite [Электронный ресурс] / Официальный сайт Material Design Lite. – Режим доступа: <https://getmdl.io/>. – Дата доступа: 08.09.18.

УДК 004.932.72'1

ОБРАБОТКА ИЗОБРАЖЕНИЙ С ПОМОЩЬЮ БИБЛИОТЕКИ OPENIMAJ**А.В. ВОЙТОВ***(Представлено: канд. техн. наук, доц. А.Ф. ОСЬКИН)**Рассматриваются возможности библиотеки OpenIMAJ.*

Иногда возникают задачи, при которых необходимо анализировать изображения в своих программах. Иногда необходимо найти лицо, иногда некий предмет. Для решения подобных задач может помочь библиотека OpenIMAJ.

OpenIMAJ - это удостоенный наград набор библиотек и инструментов для анализа мультимедийного контента и создания контента. OpenIMAJ очень широк и содержит все: от современного компьютерного видения (например, дескрипторы SIFT, обнаружение выявляемого региона, распознавание лиц и т. п.) И расширенную кластеризацию данных до программного обеспечения, которое выполняет анализ содержимого, компоновки и структуры веб-страниц.

OpenIMAJ доступна для языка java и представляет собой множество Jar модулей, каждый из которых решает разные задачи, такие как обработка изображений, обработка видео обнаружение лиц и т. п.

Для обнаружения лиц OpenIMAJ предоставляет интерфейс FaceDetector с методом detectFaces, который принимает изображения и возвращает список с областями где он обнаружил лица. Основными реализациями данного интерфейса являются HaarCascadeDetector и FKEFaceDetector.

HaarCascadeDetector использует алгоритм признаков Хаара для нахождения лиц. Пример использования данного класса представлен в листинге 1. Конструктор данного класса принимает размер минимального квадрата в пикселях, которым может быть лицо. Алгоритм признаков хаара работает с одним каналом изображения и поэтому используется метод calculateIntensity чтобы привести картинку к градациям серого.

```
FaceDetector<DetectedFace, FImage> fd = new HaarCascadeDetector(40);
List<DetectedFace> faces = fd.detectFaces( Trans-
forms.calculateIntensity(frame));
for( DetectedFace face : faces ) {
    frame.drawShape(face.getBounds(), RGBColour.RED);
}
```

Листинг 1. – пример использования каскадов HaarCascadeDetector

Реализация FKEFaceDetector может быть более полезна, так как она находит такие лицевые точки как уголки глаз, рта, носа. Поэтому объект который он возвращает это KEDetectedFace, который содержит массив лицевых точек(FacialKeypoint). На листинге 2 представлен пример выделения глаз используя реализацию FKEFaceDetector.

```
FaceDetector<KEDetectedFace, FImage> fd = new FKEFaceDetector();
List<KEDetectedFace> faces = fd.detectFaces( Transforms.calculateIntensity(
frame ) );
for( KEDetectedFace face : KEDetectedFace ) {
    for( FacialKeypoint fk : face.getKeypoints() ) {
        if( fk.type == EYE_LEFT_CENTER )
            frame.drawShape(new Circle(fk.position, 3), RGBColour.GREEN);
        if( fk.type == EYE_RIGHT_CENTER )
            frame.drawShape(new Circle(fk.position, 3), RGBColour.GREEN);
    }
}
```

Листинг 2. – пример использования каскадов FKEFaceDetector

Еще одной из интересных задач является нахождение предмета на изображении. Для этого используется алгоритм SIFT. Идея этого алгоритма заключается в следующем, необходимо найти «интересные точки» на изображении, описать их, затем найти такие точки на втором изображении и сравнить их между собой. Под «интересными точками» обычно подразумевают места на изображении,

где резко меняется цвет, яркость и т. п. Их особенность заключается в том, что они не зависят от того, где находятся, их размера или поворота.

Для поиска таких точек в OpenIMAJ используется реализация DoGSIFTEngine. Его использование показано в листинге 3.

```
MBFImage query = ImageUtilities.readMBF(new
URL("http://static.openimaj.org/media/tutorial/query.jpg"));
MBFImage target = ImageUtilities.readMBF(new
URL("http://static.openimaj.org/media/tutorial/target.jpg"));
DoGSIFTEngine engine = new DoGSIFTEngine();
LocalFeatureList<Keypoint> queryKeypoints = en-
gine.findFeatures(query.flatten());
LocalFeatureList<Keypoint> targetKeypoints = en-
gine.findFeatures(target.flatten());
```

Листинг 3. – Поиск «интересных точек»

Когда у нас есть точки на исходном изображении и точки на целевом изображении их необходимо сравнить. Для этого есть несколько реализаций интерфейса LocalFeatureMatcher.

BasicMatcher находит все похожие точки на изображении, и совпадений получается очень много. В таком случае обычно используют класс ConsistentLocalFeatureMatcher2d. Основной его особенностью является то, что он сравнивает не просто точки, а еще и сами плоскости моделей, тем самым отсекая лишние точки. Пример его использования приведен в листинге 4, а его работа проиллюстрирована на рисунке 1.

```
RobustAffineTransformEstimator modelFitter = new RobustAffineTransformEstima-
tor(5.0, 1500,
new RANSAC.PercentageInliersStoppingCondition(0.5));
matcher = new ConsistentLocalFeatureMatcher2d<Keypoint>(
new FastBasicKeypointMatcher<Keypoint>(8), modelFitter);
matcher.setModelFeatures(queryKeypoints);
matcher.findMatches(targetKeypoints);
MBFImage consistentMatches = MatchingUtilities.drawMatches(query, target,
matcher.getMatches(),
RGBColour.RED);
```

Листинг 4. – Пример использования ConsistentLocalFeatureMatcher2d

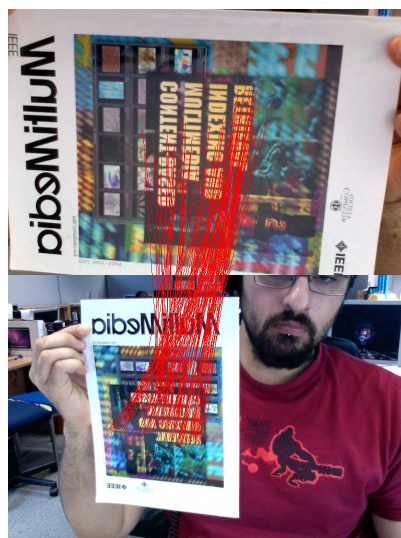


Рисунок 1. – Пример работы ConsistentLocalFeatureMatcher2d

Класс RobustAffineTransformEstimator имеет метод getModel(), который возвращает матрицу внутреннего аффинного преобразования. Данная матрица поможет преобразовать форму исходного

изображения в новую найденную. Пример этого приведен в листинг 5, а пример работы проиллюстрирован на рисунке 2.

```
target.drawShape(query.getBounds().transform(modelFitter.getModel()  
.getTransform().inverse()), 3, RGBColour.BLUE);  
DisplayUtilities.display(target);
```

Листинг 5. – Пример выделения контуров

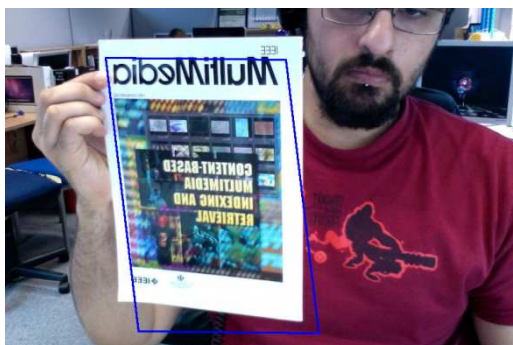


Рисунок 2. – Пример выделения контуров

В данной статье были рассмотрены такие возможности библиотеки OpenIMAJ, как нахождение лица на изображении и нахождение предметов при помощи алгоритма SIFT.

ЛИТЕРАТУРА

1. Material Design [Электронный ресурс] / Википедия. – Режим доступа: https://ru.wikipedia.org/wiki/Material_Design. – Дата доступа: 08.09.18.
2. Material Design Lite [Электронный ресурс] / Официальный сайт Material Design Lite. – Режим доступа: <https://getmdl.io/>. – Дата доступа: 08.09.18.

УДК 004.55:004.514+737.1

**СОЗДАНИЕ ЭЛЕКТРОННОГО КАТАЛОГА ИЗОБРАЖЕНИЙ КОЗЬЯНКОВСКОГО КЛАДА
АРАБСКИХ КУФИЧЕСКИХ ДИРХАМОВ IX–X ВЕКОВ
ИЗ МУЗЕЙНОГО СОБРАНИЯ
НАЦИОНАЛЬНОГО ПОЛОЦКОГО ИСТОРИКО-КУЛЬТУРНОГО МУЗЕЯ-ЗАПОВЕДНИКА****В.В. ГАВРИЛОВ***(Представлено: канд. тех. наук, доц. А.Ф. ОСЬКИН)*

Рассматриваются вопросы разработки (выбор технологии, определение функциональных возможностей, проектирование графического интерфейса) и создания оболочки электронного каталога изображений Козьянковского клада арабских куфических дирхамов IX–X веков из музейного собрания Национального Полоцкого историко-культурного музея-заповедника.

В рамках проекта № Г09-217 Белорусского республиканского фонда фундаментальных исследований Научно-исследовательское и просветительное учреждение культуры «Национальный Полоцкий историко-культурный музей-заповедник» совместно с Институтом физики им. Б. И. Степанова НАН Беларуси проводило исследовательскую работу по изучению уникального клада куфических дирхамов IX–X вв., состоящего из 7663 серебряных монет. Исследования велись как специалистами-нумизматами – работа с композицией Козьянковского клада осуществлялась научным сотрудником отдела нумизматики Государственного Эрмитажа Вяч. С. Кулешовым и научными сотрудниками музея-заповедника, так и физиками – химическое исследование лазерным спектральным анализатором состава монет проводилось сотрудниками Института физики НАН Беларуси канд. физ.-мат. наук М.В. Бельковым и доктором физ.-мат. наук С.Н. Райковым [1]. Конечным результатом проекта стало издание «Электронного каталога изображений Козьянковского клада арабских куфических дирхамов IX–X веков из музейного собрания Национального Полоцкого историко-культурного музея-заповедника» [2]. Его создание и будет рассмотрено далее.

При разработке электронного каталога был выдвинут ряд технических требований к его реализации:

- электронный каталог должен работать с внешнего носителя – оптического диска без установки дополнительного программного обеспечения на компьютер пользователя;
- для удобства использования и удешевления производства тиража количество дисков должно быть минимально возможным;
- использованная технология программирования должна позволить без существенных затрат перенести электронный каталог в сеть Интернет;
- так как большая часть изображений монет публикуется впервые, сделать невозможным копирование напрямую исходного изображения монет.

Вышеперечисленные требования и обусловили использование для создания оболочки языка гипертекстовой разметки HTML (HyperText Markup Language) [3] совместно с языком написания сценариев JavaScript [4]. Изначальное использование данной связки при разработке электронного каталога на оптическом носителе позволяет без каких-либо затрат и осложнений перенести его в сеть Интернет.

Непосредственно для визуализации оболочки используются возможности НТА (HTML Application) — приложения Microsoft Windows, являющегося документом HTML, отображаемым в отдельном окне без элементов интерфейса обозревателя, таких как строка меню, строка адреса, панель инструментов («безбраузерное приложение») с помощью движка Microsoft Internet Explorer [5]. На НТА не распространяется большинство ограничений безопасности Internet Explorer. Возможность создания НТА появилась с выпуском Microsoft Internet Explorer 5.0, что гарантирует совместимость со всеми актуальными версиями операционной системы Microsoft Windows.

За счёт использования указанных выше технологий программирования размер непосредственно файла оболочки получился небольшим, ≈ 10 Кб, что сыграло важную роль, так как размер всей базы изображений монет, а это 7663 единицы, занял почти всё имеющееся пространство диска — 7,62 Гб. В качестве носителя каталога был выбран двухслойный DVD (DVD-DL), который соответствовал техническому условию и позволил уместить весь электронный каталог.

Одной из трудоёмких задач в создании каталога была обработка большого массива графической информации, полученной после фотографирования аверса и реверса каждой из монет клада. Так, для обработки, обрезки каждого изображения и «склейки» аверса и реверса монеты в единый файл были дополнительно разработаны и созданы Action для Adobe Photoshop.

Основным поисковым критерием отбора изображения стал идентификационный номер монеты — существующий музейный номер предмета по Книге поступлений (КП) основного фонда. Также в каталоге отдельным файлом представлена инвентарная опись монет клада. На JavaScript были написаны разнообразные функции, отвечающие за подсказки при выборе изображения, за определение ошибочности ввода данных для поиска. Были подключены «горячие клавиши» для работы с поисковой формой (рис. 1).

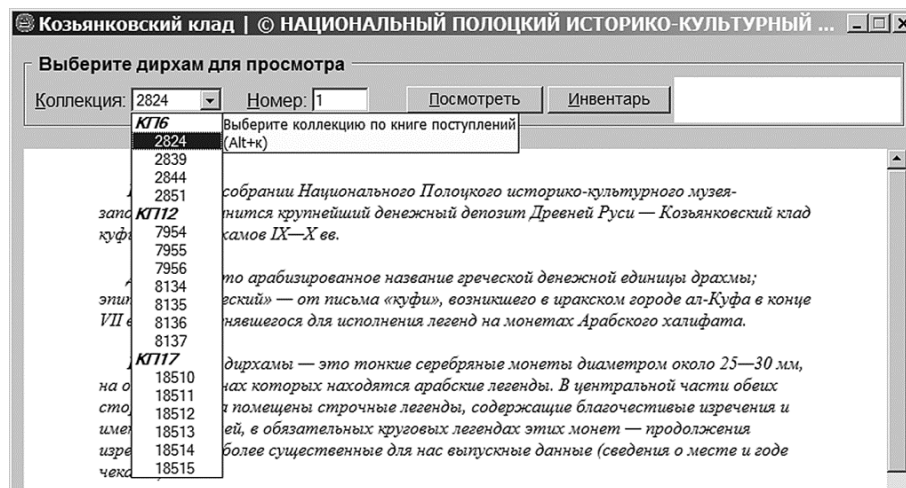


Рисунок 1. – Поисковый модуль оболочки электронного каталога

Для модуля, отвечающего за вывод выбранного изображения, была использована разработка Zoomify [6]. Zoomify работает с изображениями в формате JPG, создавая возможность масштабирования и панорамирования для быстрого, интерактивного просмотра. Она преобразует исходное графическое изображение в набор «плиток» (tiled) разного разрешения, что позволяет загружать не всё изображение целиком, а только выбранную область. Для электронного каталога изображений на оптическом диске был выбран Flash модуль Zoomify Viewer, что позволило не зависеть от конфигурации компьютера пользователя и работать электронному каталогу напрямую с диска (рис. 2).

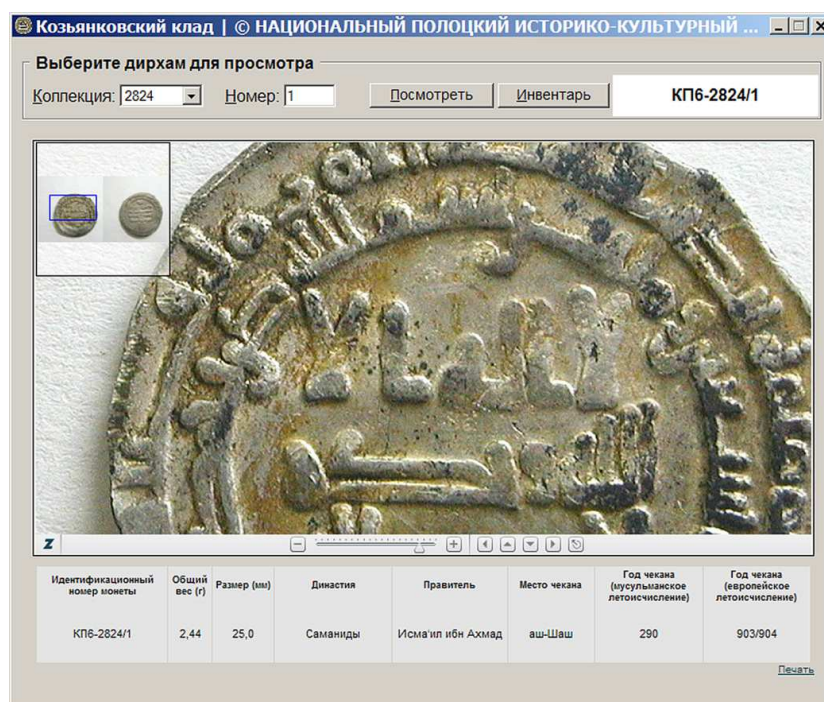


Рисунок 2. – Пользовательский интерфейс оболочки электронного каталога

Для обновления версии Adobe Flash Player, если она не соответствует минимально требуемой было заключено соглашение с Adobe на распространение дистрибутива Adobe Flash Player на оптических дисках. Наличие аналогичного модуля Zoomify Viewer, использующего HTML5 и JavaScript, позволяет поместить созданный каталог в сеть Интернет с минимальными затратами.

Таким образом, автором создан готовый программный продукт «Электронный каталог изображений Козьянковского клада арабских куфических дирхамов IX—X веков из музейного собрания Национального Полоцкого историко-культурного музея-заповедника» в соответствии с заданными техническими требованиями: возможность работы с оптического диска без установки на компьютер пользователя; выбранная технология программирования предусматривает возможность размещения каталога в сети Интернет без существенных затрат; использованный модуль Zoomify позволяет оптимизировать загрузку изображений и делает затруднительным прямое скачивание исходного изображения монет.

ЛИТЕРАТУРА

1. Козьянковский клад арабских куфических дирхамов IX—X веков из музейного собрания Национального Полоцкого историко-культурного музея-заповедника [Электронный ресурс]. – Режим доступа: <http://polotsk.museum.by/node/25135>. – Дата доступа: 10.09.2018.
2. Электронный каталог изображений Козьянковского клада арабских куфических дирхамов IX—X веков из музейного собрания Национального Полоцкого историко-культурного музея-заповедника [Электронный ресурс]. – Электрон. дан. (7,62 ГБ). – Полоцк : НПИКМЗ, 2011. – 1 электрон. опт. диск (DVD-ROM DL).
3. Муссиано, Ч. HTML и XHTML. Подробное руководство / Ч. Муссиано, Б. Кеннеди. – 6-е изд. – СПб. : Символ-Плюс, 2008. – 752 с.
4. Флэнаган, Д. JavaScript. Подробное руководство / Д. Флэнаган. – 4-е изд. – СПб. : Символ-Плюс, 2004. – 960 с.
5. Introduction to HTML Applications (HTAs) [Электронный ресурс]. – Режим доступа: <https://msdn.microsoft.com/en-us/library/ms536496%28v=vs.85%29.aspx>. – Дата доступа: 10.09.2018.
6. Zoomify [Электронный ресурс]. – Режим доступа: <http://www.zoomify.com/index.html>. – Дата доступа: 10.09.2018.

УДК 004.55

**ИСПОЛЬЗОВАНИЕ HTML APPLICATION
ДЛЯ СОЗДАНИЯ ОБОЛОЧКИ ЭЛЕКТРОННОГО КАТАЛОГА ИЗОБРАЖЕНИЙ
КОЗЬЯНКОВСКОГО КЛАДА АРАБСКИХ КУФИЧЕСКИХ ДИРХАМОВ IX–X ВЕКОВ
ИЗ МУЗЕЙНОГО СОБРАНИЯ
НАЦИОНАЛЬНОГО ПОЛОЦКОГО ИСТОРИКО-КУЛЬТУРНОГО МУЗЕЯ-ЗАПОВЕДНИКА**

В.В. ГАВРИЛОВ

(Представлено: канд. тех. наук, доц. А.Ф. ОСЬКИН)

Рассматривается вопрос разработки оболочки электронного каталога изображений Козьянковского клада арабских куфических дирхамов IX–X веков из собрания Национального Полоцкого историко-культурного музея-заповедника с использованием HTML Application (HTA). Описываются возможности данной технологии.

При разработке электронного каталога изображений Козьянковского клада арабских куфических дирхамов IX–X веков из музейного собрания Национального Полоцкого историко-культурного музея-заповедника был выдвинут ряд условий, которым должен соответствовать конечный продукт. Одно из условий технического задания было следующим: используемая технология программирования для создания каталога должна позволить без существенных затрат перенести электронный каталог с оптического носителя, на котором изначально предполагалось его издавать, в сеть Интернет. Это и обусловило использование HTML Application (HTA) в связке с языком гипертекстовой разметки HTML (HyperText Markup Language) и языком написания сценариев JavaScript.

Первые упоминания о возможности создания приложений на основе кодов HTML появились ещё в середине 1999 г. По времени это совпадает с выпуском 5-й версии браузера Microsoft Internet Explorer. Начиная именно с этой версии, корпорация Microsoft вводит в обращение термин HTML Application и в её продуктах появляются средства поддержки новой технологии.

HTML Application позволяет создавать собственные приложения, основанные на скриптовых языках VBScript, JavaScript и стандартном HTML-коде. В его основе лежит формат файлов HTA (производное от HTML Application).

Windows определяет подобные файлы как исполняемые. Возможность HTA исполнять JavaScript и VB-скрипты делает возможным создавать приложения различного уровня сложности. Простота написания позволяет, обладая опытом создания веб-страниц и написания скриптов, создавать такие HTA-файлы, которые ни по функциональности, ни по дизайну не будут уступать небольшим программам, написанным на более сложных языках.

Создать HTA-приложение можно просто написав HTML-страницу и сохранив её с расширением .hta, которое обрабатывается программой mshta.exe. Её задача – обеспечить связь программы с браузером. Так как такие приложения используют движок браузера, то они объединяют в себе все их возможности: их объектную модель, разметку страницы (HTML), каскадные листы стилей (CSS), скрипты (VBS и JS). Также на такие приложения не накладывается ограничение безопасности, они работают как любой исполняемый файл.

В то время как расширение .hta указывает системе, как работать с приложением, новый тег <hta:application> и атрибуты указывают окну, как себя вести, будучи приложением. Этот тег предоставляет ограниченный набор атрибутов, управляющих всем от стиля границы окна до программной иконки и меню. Для создания функционирующего HTA этот тег должен быть описан внутри тега <head> html-страницы. Пример разметки HTA представлен в Листинге.

```
<hta:application id="DRH"  
ApplicationName="Klad"  
Border="dialog"  
BorderStyle="normal"  
Caption="yes"  
ContextMenu="no"  
Icon="dirham.ico"  
InnerBorder="no"  
MaximizeButton="no"  
MinimizeButton="yes"
```

```
Scroll="no"  
Selection="no"  
ShowInTaskBar="yes"  
SingleInstance="no"  
SysMenu="yes"  
Version="1.2"  
WindowState="normal">  
<script type="text/javascript" language="JavaScript1.5">  
window.resizeTo(800, 676);  
window.moveTo(screen.width/2-400, screen.height/2-338);  
</script>
```

**Листинг. – Разметка HTA с кодом JavaScript
для создания окна оболочки электронного каталога**

При помощи HTML создаётся абсолютно любой пользовательский интерфейс, причём с минимальными усилиями и без каких-либо существенных ограничений. Логика программы организуется кодом на JavaScript или Visual Basic Script, а само приложение выходит за рамки браузера. Программа на стадии выполнения ничем не будет отличаться от обычных оконных приложений для Windows: запущенное приложение будет присутствовать на панели задач, иметь собственное окно, меню и значок. Хотя здесь нужно оговориться: программа не будет видна среди запущенных приложений в диспетчере задач, единственное средство, которым можно контролировать запущенную программу, – это процесс mshta.exe.

Общие затраты времени на разработку HTML-приложения оказываются меньшими по сравнению с классическими C++ или Visual Basic, даже несмотря на отсутствие визуальных средств проектирования и интеллектуальную поддержку языка. Дело в том, что HTML-приложения не имеют никаких промежуточных состояний и не требуют установки у пользователя.

Конечно, выигрывая в скорости разработки, HTML-приложения значительно уступают exe-файлам в быстродействии. Однако существует по крайней мере один класс задач, где HTML-приложения явно предпочтительнее. Речь идёт о генерации или анализе HTML-страниц и использовании языка HTML и JavaScript для обработки подобных же кодов и объектной модели.

Существует и другое важное обстоятельство, делающее HTML-приложения весьма привлекательными, – это практически полное отсутствие специальных требований к среде проектирования и системным ресурсам.

Применение технологии Microsoft HTML Application (HTA) в создании оболочки электронного каталога изображений Козьянковского клада арабских куфических дирхамов IX–X веков из музейного собрания Национального Полоцкого историко-культурного музея-заповедника позволило создать программу, работающую непосредственно с оптического диска без установки на компьютер пользователя. Размер полученного файла не превысил 10 Кб, что сыграло важную роль, так как размер всей базы изображений монет клада, а это 7663 единицы, занял почти всё имеющееся пространство диска – 7,62 Гб. Условия технического задания на разработку были выполнены – использованная технология программирования позволяет без существенных затрат перенести электронный каталог в сеть Интернет.

ЛИТЕРАТУРА

1. Introduction to HTML Applications (HTAs) [Электронный ресурс]. – Режим доступа: <https://msdn.microsoft.com/en-us/library/ms536496%28v=vs.85%29.aspx>. – Дата доступа: 12.09.2018.
2. HTML Applications Reference [Электронный ресурс]. – Режим доступа: <https://technet.microsoft.com/ru-ru/sqlserver/ms536473%28v=vs.110%29>. – Дата доступа: 12.09.2018.
3. Сизов, В. HTML-приложения [Электронный ресурс]. – Режим доступа: <https://www.bytemag.ru/articles/detail.php?ID=8900>. – Дата доступа: 12.09.2018.

УДК 004.021

МЕТОДЫ ОПТИМИЗАЦИИ ПРОИЗВОДИТЕЛЬНОСТИ МОБИЛЬНЫХ ИГР В UNITY3D**А.Д. КАРПОВИЧ***(Представлено: Д.В. ПЯТКИН)*

Рассматриваются наиболее эффективные способы оптимизации мобильных игр, написанных в Unity3d.

Unity3d [1] – один из самых популярных игровых движков для мобильных платформ. Множество разработчиков используют его для создания и выпуска игр. При разработке игр для мобильных устройств важно учитывать их ограниченные технические возможности. Существуют слабые и мощные по производительности телефоны. Новые поколения мобильной GPU могут быть в 5 раз более производительнее своих предшественников. Несмотря на это важно вовремя оптимизировать приложение, для того чтобы добиться максимальной производительности на большинстве устройств и охватить большее количество пользователей.

Основной раздел

Оптимизация [2] — модификация системы для улучшения её эффективности.

Некоторые задачи часто могут быть выполнены более эффективно. Например, программный код, который вычисляет выражение, можно записать следующим образом:

```
int a = b*c + b*c + b*c;
```

Листинг 1. – Вычисление выражения без оптимизации

Здесь мы видим, что выполняется 3 операции умножения и 2 операции сложения. Этот код можно улучшить, вычислив произведение заранее. Теперь вместо 3 операций умножения мы обошлись только одной.

```
int d = b*c;  
int a = d + d + d;
```

Листинг 2. – Вычисление выражения с использованием математической оптимизации

Обычной практикой считается оптимизация после реализации основного функционала игры. Преждевременная оптимизация кода может вызвать множество проблем и сильно замедлить разработку. Так как оптимизированный код менее гибкий и его сложнее читать.

Британский ученый Майкл А. Джексон часто цитирует свои правила оптимизации программ: Первое правило оптимизации программы: не делайте ее. Второе правило оптимизации программы (только для экспертов!): не делайте ее пока что [3]. Он обосновал это тем, что учитывая рост скорости компьютеров, программа будет достаточно быстрой. Кроме того, если пытаться слишком много оптимизировать, то код сильно усложнится, и появится много ошибок.

Однако при разработке мобильной игры оптимизацию не следует считать последней стадией разработки проекта: аппаратное обеспечение, представленное сейчас на рынке, сильно ограничено по сравнению с компьютерами. Поэтому высок риск того, что игра не будет работать на большинстве устройств и про оптимизацию следует думать с самого начала разработки.

Для оптимизации требуется найти узкое место: критическую часть кода, которая является основным потребителем необходимого ресурса. Улучшение примерно 20 % кода иногда влечёт за собой изменение 80 % результатов, согласно *принципу Парето* [4]. Утечка ресурсов (памяти, дескрипторов и т.д.) также может привести к падению скорости выполнения программы. Для поиска таких утечек используются специальные отладочные инструменты, а для обнаружения узких мест применяются программы – *профайлеры* [5].

Профайлер – инструмент, позволяющий программисту увидеть, сколько времени у программы уходит на выполнение каждой функции и ранжировать их по порядку. Когда верхняя функция занимает 3% времени, это означает, что если удастся вдвое сократить время ее выполнения, то общая производительность программы ускорится на 1,5%.

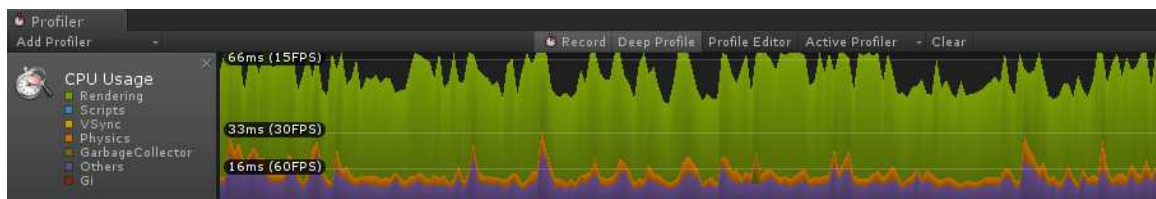


Рисунок. – Окно профайлера в Unity

Профилрование важно, потому что оно поможет выяснить, какие оптимизации действительно приведут к большому приросту производительности, а какие являются пустой тратой времени. Благодаря тому, что рендеринг обрабатывается на отдельном чипе – GPU, отрисовка одного кадра занимает в два раза меньше времени (только GPU, а не CPU + GPU). Это означает, что если CPU замедляет работу, оптимизация графики не повысит частоту кадров, и если GPU замедляет работу, не помогут оптимизация физики и скриптов.

Оптимизация в основном фокусируется на одиночном или повторном времени выполнения, использовании памяти, дискового пространства, пропускной способности или некотором другом ресурсе. Это обычно требует компромиссов: один параметр оптимизируется за счёт других. Например, увеличение размера программного кэша улучшает производительность времени выполнения, но также увеличивает потребление памяти. Прозрачность кода и его выразительность получается за счёт деоптимизации. Сложные специализированные алгоритмы требуют больше усилий по отладке и увеличивают вероятность ошибок.

Элементы больше всего нуждающиеся в оптимизации:

- графика – 70%;
- расчет физики – 20%;
- скрипты – 10%.

Выделение памяти для новых объектов – дорогая операция. Использование методов создания новых объектов во время игры нужно максимально минимизировать. Для большого количества однотипных объектов можно использовать пул объектов. Один раз инициализировав определенное количество объектов, они будут использоваться повторно, вместо создания и уничтожения каждый раз, что существенно снизит нагрузку на CPU и GPU.

Все, что будет использоваться больше одного раза лучше *кешировать* [6]. Операции типа `GameObject.Find()` или `GetComponent()` достаточно ресурсозатратны, а если они вызываются где-нибудь в цикле или в `Update()`, то производительность может упасть. Пример кеширования можно увидеть в листинге 3.

```

Transform _cachedTransform;

void Awake () {
    _cachedTransform = transform;
}

void Update () {
    _cachedTransform.localPosition = _newPosition;
}

```

Листинг 3. – Кеширование компонента Transform

Для оптимизации отрисовки графики для начала нужно уменьшить количество *draw-call* [7].

Draw-call – команда графическому API (например, OpenGL или Direct3D) на отрисовку. Графический API производит значительную работу для каждого draw-call, что сильно влияет на производительность CPU.

Для мобильных устройств рекомендуется иметь до 100 draw-calls. Для уменьшения их количества можно использовать *батчинг* [7] – отрисовка однотипных объектов использующих один и тот же материал за один подход.

Для графики упаковывать все спрайты в *атласы* [8]. Для этого есть как встроенный инструмент Sprite Packer, так и продвинутый TexturePacker. Таким образом уменьшается количество вызовов отрисовки спрайтов.

Шейдеры управляют всеми визуальными элементами в игре. Нужно знать, что оптимизированные шейдеры могут значительно повысить производительность, поскольку они не требуют большого количества вычислений.

Еще один способ ускорить игру – использовать *мобильные шейдеры* [9], даже на более высокопроизводительных платформах. При работе над мобильной игрой, обязательно нужно использовать мобильные шейдеры, потому что они требуют меньше вычислений.

Так же для улучшения отрисовки графики можно использовать *occlusion culling* [10].

Occlusion Culling – функция, отключающая рендеринг тех объектов, которые в данный момент не видит камера. Используя виртуальную камеру, происходит проход по сцене для построения иерархии потенциально видимых объектов. Эти данные используются каждой камерой для определения того, что она видит, и что нет. Опираясь на полученную информацию, происходит рендеринг только видимых объектов. Это уменьшает количество draw-calls и увеличивает производительность игры.

В Unity есть возможность использовать разные типы *коллайдеров* [11] – компонентов, позволяющих обрабатывать столкновения объектов. По возможности лучше использовать примитивные коллайдеры такие как Box Collider и Circle Collider для ускорения расчёта физики.

Компонент RigidBody обычно используются для добавления веса к объекту. Если объект привязан к RigidBody, он может зависеть от физики, такой как сила тяжести других сил. Большое количество объектов RigidBody, отрицательно повлияет на производительность.

Unity поддерживает несколько типов аудио, по умолчанию он будет импортировать аудиоклипы для использования типа загрузки *Decompress On Load* [12] вместе со сжатием *Vorbis* [13]. Звуковые эффекты обычно короткие и, следовательно, имеют небольшие требования к памяти. Для них настройка *Decompress on Load* будет работать лучше всего, но тип сжатия должен быть либо *PCM*, либо *ADPCM* [14]. PCM обеспечивает более высокое качество, но поставляется с большим размером файла, что отлично подходит для очень короткого, но важного звукового эффекта. ADPCM имеет коэффициент сжатия в 3,5 раза меньше, чем PCM, и лучше всего используется для аудио эффектов, которые используются очень часто.

Для более длинных аудиоклипов, таких как фоновая музыка или другие большие файлы, лучше использовать сжатыми в памяти, что приводит к распаковке файла прямо перед воспроизведением.

Таким образом, в данной статье рассмотрены основные методы оптимизации мобильных игр, однако не стоит забывать, что методы оптимизации сильно зависят от типа разрабатываемой игры. Некоторые методы оптимизации применимы только к узкому спектру задач и могут негативно сказаться на итоговой производительности. Так же не стоит пытаться полностью оптимизировать программу. Оптимизацию стоит заканчивать тогда, когда достигнута необходимая производительность

ЛИТЕРАТУРА

1. Unity (игровой движок) [Электронный ресурс] / Wikipedia – The Free Encyclopedia. – Режим доступа: [https://ru.wikipedia.org/wiki/Unity_\(игровой_движок\)](https://ru.wikipedia.org/wiki/Unity_(игровой_движок)). – Дата доступа: 14.09.18.
2. Оптимизация (информатика) [Электронный ресурс] / Wikipedia – The Free Encyclopedia. – Режим доступа: [https://ru.wikipedia.org/wiki/Оптимизация_\(информатика\)](https://ru.wikipedia.org/wiki/Оптимизация_(информатика)). – Дата доступа: 14.09.18.
3. Руководство по оптимизации [Электронный ресурс] / Docs Unity3d. – Режим доступа: <https://docs.unity3d.com/ru/current/Manual/MobileOptimizationPracticalGuide.html>. – Дата доступа: 14.09.18.
4. Закон Парето или Принцип 80 на 20 [Электронный ресурс] / Элитариум – центр дополнительного образования. – Режим доступа: <http://www.elitarium.ru/zakon-pareto-princip-80-na-20-pravilo-jurana-kachestvo-resursy-raspredelenie-rabota-vazhnost/>. – Дата доступа: 14.09.18.
5. The Profiler Window [Электронный ресурс] / Docs Unity3d. – Режим доступа: <https://docs.unity3d.com/ru/current/Manual/Profiler.html>. – Дата доступа: 14.09.18.
6. Оптимизация скриптов [Электронный ресурс] / Docs Unity3d. – Режим доступа: <https://docs.unity3d.com/ru/current/Manual/MobileOptimizationPracticalScriptingOptimizations.html>. – Дата доступа: 15.09.18.
7. Батчинг вызовов отрисовки (Draw Call Batching) [Электронный ресурс] / Docs Unity3d. – Режим доступа: <https://docs.unity3d.com/ru/current/Manual/DrawCallBatching.html>. – Дата доступа: 15.09.18.
8. Sprite Atlas [Электронный ресурс] / Docs Unity3d. – Режим доступа: <https://docs.unity3d.com/Manual/SpriteAtlas.html>. – Дата доступа: 15.09.18.
9. Использование и производительность встроенных шейдеров [Электронный ресурс] / Docs Unity3d. – Режим доступа: <https://docs.unity3d.com/ru/current/Manual/shader-Performance.html>. – Дата доступа: 16.09.18.

10. Occlusion Culling [Электронный ресурс] / Docs Unity3d. – Режим доступа: <https://docs.unity3d.com/ru/current/Manual/OcclusionCulling.html>. – Дата доступа: 16.09.18.
11. Коллайдеры (Colliders) [Электронный ресурс] / Docs Unity3d. – Режим доступа: <https://docs.unity3d.com/ru/current/Manual/CollidersOverview.html>. – Дата доступа: 16.09.18.
12. Audio Clip [Электронный ресурс] / Docs Unity3d. – Режим доступа: <https://docs.unity3d.com/Manual/class-AudioClip.html>. – Дата доступа: 16.09.18.
13. Vorbis [Электронный ресурс] / Docs Unity3d. – Режим доступа: <https://docs.unity3d.com/ScriptReference/AudioCompressionFormat.Vorbis.html>. – Дата доступа: 16.09.18.
14. Audio compression format [Электронный ресурс] / Docs Unity3d. – Режим доступа: <https://docs.unity3d.com/ScriptReference/AudioCompressionFormat.html>. – Дата доступа: 16.09.18.

УДК 004.021

СРАВНЕНИЕ АРХИТЕКТУРНЫХ ПАТТЕРНОВ ПРОЕКТИРОВАНИЯ ИГР

А.Д. КАРПОВИЧ

(Представлено: Д.В. ПЯТКИН)

Производится сравнение архитектурных паттернов проектирования, чаще всего использующихся при программировании игр. Так же в статье описываются преимущества компонентно-ориентированного программирования в сравнении со стандартным объектно-ориентированным подходом.

Большие игры требуют создание сложной архитектуры. В результате чего, создаются сложные сущности и сложное взаимодействие между классами. *Объектно-ориентированный подход (ООП)* [1] программирования помогает в решении многих задач. Однако при разработке более сложных систем, в том числе систем, моделирующих процессы окружающего мира, появляется ряд других проблем, для решения которых необходим другой подход. Разработка игр, с использованием ООП, влечет за собой постоянный рефакторинг кода и увеличение длительности разработки. Грамотно выбранная архитектура позволяет сильно упростить и ускорить процесс разработки игры.

Архитектура компьютерной игры [2] – система организации программы, которая определяет внутреннюю логику построения кода, выбор структурных элементов и определение связей между ними.

Сложная система игрового приложения состоит из нескольких подсистем – функциональных модулей, сервисов, слоёв, подпрограмм, соединённых в определённой последовательности. При таком функциональном разбиении разработчики получают не сильно связанный код, а набор понятных элементов, взаимодействующих по простым правилам.

Важно в процессе разработке следовать признакам хорошей архитектуры:

- В него легко вносить правки. Новые фрагменты не требуют переписывания уже существующих.
- Система эффективна. Код решает поставленные задачи и работает в любых условиях.
- Срок разработки можно уменьшить, увеличив команду. Задачи легко делятся между разработчиками.
- Фрагменты кода можно повторно использовать в других системах.

Одним из самых простых архитектурных шаблонов используемых в играх, а также одним из самых старых, считается *Game Loop* [3]. Этот шаблон применяется практически в каждой игре. Однако стоит заметить, что вне игровой индустрии этот шаблон не нашёл широкого применения

Game Loop (Игровой цикл) - это общий контроль потока для всей игровой программы. Каждая итерация игрового цикла известна как кадр. Большинство игр в режиме реального времени обновляются несколько раз в секунду: 30 и 60 являются двумя наиболее распространёнными интервалами. Если игра работает со скоростью 60 кадров в секунду, то игровой цикл завершает 60 итераций каждую секунду.

```
while game is running
    process inputs
    update game world
    generate outputs
loop
```

Листинг 1. – Псевдокод игрового цикла

Game Loop работает на протяжении всей игры. На каждой итерации игровой цикл обрабатывает пользовательский ввод без блокировки, обновляет состояние игры и осуществляет рендеринг игры.

Ещё одним часто встречающимся архитектурным решением в программировании игр является Model-View-Controller(MVC) [4]. Он часто используется в приложениях с графическим интерфейсом пользователя (GUI) [5]. MVC разделяет программу на Контроллеры (Controllers), Представления (Views) и Модели (Models):

- Модель предоставляет данные и реагирует на команды контроллера, изменяя своё состояние.
- Представление отвечает за отображение данных модели пользователю, реагируя на изменения модели.
- Контроллер интерпретирует действия пользователя, оповещая модель о необходимости изменений.

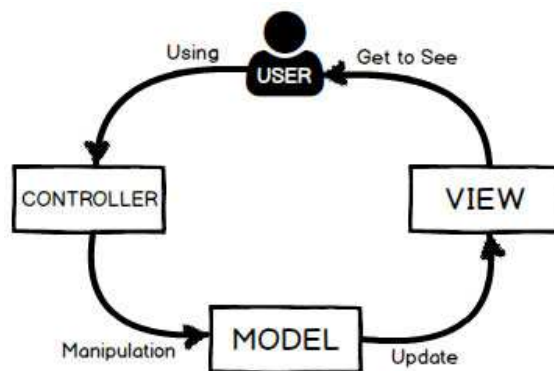


Рисунок 1. – Диаграмма MVC

Основная цель применения этой концепции состоит в отделении бизнес-логики (модели) от её визуализации (представления, вида). За счёт такого разделения повышается возможность повторного использования кода. Так же это разделение позволяет программистам, разрабатывающим бизнес-логику, работать независимо от разработчиков представления.

Однако в игровых приложениях возможности MVC обычно не используются в полной мере. Роль Контроллера в управлении потоком данных может взять на себя игровой цикл и набор Систем, а количество разнородных Компонентов гораздо больше двух и не описывается простым делением на Модель и Представление. Кроме того, многие компоненты являются необязательными.

В игровых приложениях лучше всего отдать предпочтение использованию архитектурных решений, реализующих *компонентно-ориентированный подход (КОП)* [6].

Компонентно-ориентированный подход – парадигма программирования, существенным образом опирающаяся на понятие *компонента* [7] – независимого модуля исходного кода программы, предназначенного для повторного использования и развёртывания и реализующегося в виде множества языковых конструкций (например, «классов» в объектно-ориентированных языках программирования), объединённых по общему признаку и организованных в соответствии с определёнными правилами и ограничениями.

Этот подход решает одну из главных проблем объектно-ориентированного подхода – проблему хрупких базовых классов – ситуации, когда изменить реализацию типа-предка невозможно, не нарушив корректность функционирования типов-потомков.

В ООП подходе объект определяется описываемым его классом. В КОП подходе, объект определяется компонентами, из которых он состоит. Такой подход упрощает повторное использование написанного кода – использование одного компонента в разных объектах. Также из различных комбинаций уже существующих компонентов, можно собрать новый тип объекта.

Для примера, возьмем объект «персонаж». С точки зрения ООП – это был бы один большой класс. С точки зрения КОП – это набор компонентов из которых состоит объект «персонаж»:

- характеристики персонажа – компонент «Stats»;
- управление персонажем – компонент «CharacterController»;
- анимация персонажа – «CharacterAnimationController»;
- обработчик столкновений – «CharacterCollisionHandler».

Шаблон проектирования *Component-Entity-System* [8] стал эффективным дополнением КОП подхода и Game Loop паттерна. Этот шаблон проектирования разделяет различные проблемы и задачи между сущностями (Entities), компонентами (Components) и системами (Systems):

– Entity(Сущность) - это контейнерные объекты, к которым могут быть присоединены компоненты. Entity являются основой для всех объектов на сцене. Такие игровые объекты, как юниты, декорации или пули, являются сущностями. Сущности ничем не отличаются друг от друга, и являются всего лишь вспомогательным контейнером для компонентов. Без компонентов Сущности не выполняют никаких функций и ничего не делают.

– Component(Компонент) – повторно используемый модуль или контейнер данных, который может быть присоединён к объектам для обеспечения внешнего вида, поведения или функциональности. Вся логика реализуется через компоненты. Различные типы объектов определяются путем смешивания, сопоставления и конфигурирования компонентов. Например: компонент ИИ (Искусственного интеллекта), компонент физического движения, графический компонент;

– System(Система) – обеспечивает управление классами компонентов. Системы часто являются необязательными, но их можно использовать для разделения логики и данных. Системы не владеют ни сущностями, ни компонентами. Они имеют доступ к ним через независимые объекты-диспетчеры, которые в свою очередь управляют жизненным циклом сущностей и компонентов.

Сущности компоненты и системы обмениваются информацией через Entity-Component-System Manager который является диспетчером событий.

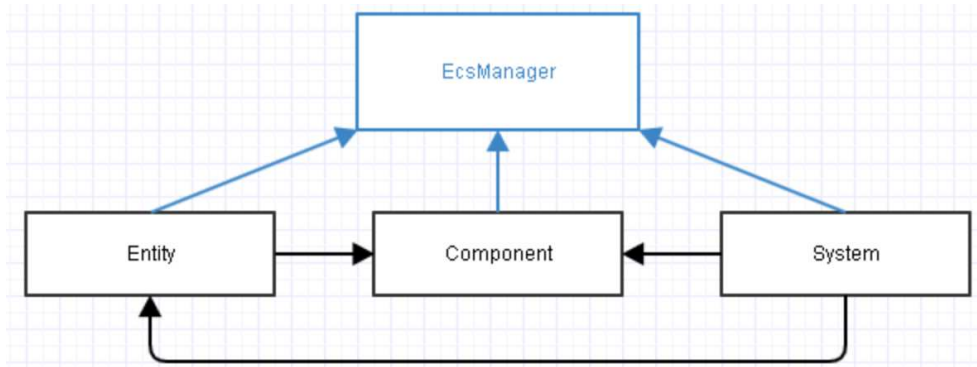


Рисунок 2. – Диаграмма Entity-Component-System

К примеру: мы имеем 3 компонента: Transform, задающий положение и масштабирование объекта на игровой сцене, Velocity, позволяющий объекту перемещаться и Material, который отображает объект на экране:

```

class TransformComponent : BaseComponent {
    Vector3 position;
    Vector3 rotation;
    Vector3 scale;
}

class VelocityComponent : BaseComponent {
    float x;
    float y;
    float speed;
}

class MaterialComponent : BaseComponent {
    Color color;
    Shader shader;
}
  
```

Листинг 2. – Пример компонентов

На основе этих компонентов с помощью композиции можно создать произвольное количество различных сущностей. Сущности по своей сути являются простыми контейнерами для компонентов:

```

class Player : BaseEntity {
    TransformComponent position;
    VelocityComponent velocity;
    MaterialComponent material;
}

class Floor : BaseEntity {
    TransformComponent position;
    MaterialComponent material;
}
  
```

Листинг 3. – Пример сущностей

Системы управляют логикой игры. Они принимают сущности и запускают операции над объектами, которые имеют конкретные компоненты, требуемые системой.

```
class SystemRender : BaseSystem {
    void update( Entity entities ) {
        for (auto entity : entities) {
            TransformComponent position = entity.GetComponent<TransformComponent>();
            MaterialComponent material = entity.GetComponent<MaterialComponent>();
            Render( position, material );
        }
    }
}
```

Листинг 4. – Пример Системы

В данной статье рассмотрены архитектурные паттерны проектирования игр, а также техники их использования. Подводя итоги, можно сделать вывод, что компонентно-ориентированный подход при разработке игр – наиболее инновационный. КОП делает процесс разработки более интуитивным и гибким.

ЛИТЕРАТУРА

1. Объектно-ориентированное программирование [Электронный ресурс] / Wikipedia – The Free Encyclopedia. – Режим доступа: https://ru.wikipedia.org/wiki/Объектно-ориентированное_программирование. – Дата доступа: 16.09.18.
2. Архитектура, производительность и игры [Электронный ресурс] / Game Programming Patterns. – Режим доступа: <https://martalex.gitbooks.io/gameprogrammingpatterns/content/chapter-1/1.1-architecture-performance-and-games.html>. – Дата доступа: 16.09.18.
3. Игровой цикл [Электронный ресурс] / Game Programming Patterns. – Режим доступа: <https://martalex.gitbooks.io/gameprogrammingpatterns/content/chapter-3/3.2-game-loop.html>. – Дата доступа: 18.09.18.
4. Model-View-Controller [Электронный ресурс] / Wikipedia – The Free Encyclopedia. – Режим доступа: <https://ru.wikipedia.org/wiki/Model-View-Controller>. – Дата доступа: 18.09.18.
5. Графический интерфейс пользователя [Электронный ресурс] / Wikipedia – The Free Encyclopedia. – Режим доступа: https://ru.wikipedia.org/wiki/Графический_интерфейс_пользователя. – Дата доступа: 19.09.18.
6. Компонентно-ориентированное программирование [Электронный ресурс] / Wikipedia – The Free Encyclopedia. – Режим доступа: https://ru.wikipedia.org/wiki/Компонентно-ориентированное_программирование. – Дата доступа: 19.09.18.
7. Компонент [Электронный ресурс] / Game Programming Patterns. – Режим доступа: <https://martalex.gitbooks.io/gameprogrammingpatterns/content/chapter-5/5.1-component.html>. – Дата доступа: 20.09.18.
8. Entity-component-system [Электронный ресурс] / Wikipedia – The Free Encyclopedia. – Режим доступа: <https://en.wikipedia.org/wiki/Entity%E2%80%93component%E2%80%93system>. – Дата доступа: 20.09.18.

УДК 004.932

МОБИЛЬНОЕ ПРИЛОЖЕНИЕ ДЛЯ РАСПОЗНАВАНИЯ РЕКВИЗИТОВ БАНКОВСКИХ КАРТ ПОД УПРАВЛЕНИЕМ iOS

А.Ю. КУРИЛОВИЧ*(Представлено: канд. техн. наук, доц. Р.П. БОГУШ)*

Представлено разработанное мобильное iOS-приложение для распознавания реквизитов банковских карт, которое предполагает предварительную обработку входных изображений для улучшения их качества и выделения блоков символов, распознавание которых выполняется с помощью библиотеки Tesseract. При разработке использовался язык программирования Objective-C, библиотека OpenCV 2.4.13, фреймворк iPhone SDK.

Для мобильного iOS-приложения распознавания реквизитов банковских карт используется алгоритм обработки видеоизображений, который состоит из этапов: детектирование области карты; сегментация изображения банковской карты; преобразование цветных сегментов к оттенкам серого; улучшение контрастности; подчеркивание границ символов с использованием операций математической морфологии; уточнение границ сгруппированных блоков символов; распознавание блоков символов библиотекой Tesseract.

В качестве основного языка для написания программной реализации взят за основу Objective-C, с использованием библиотеки OpenCV 2.4.13 (данная версия является самой актуальной для iOS), из основных фреймворков iPhone SDK, были использованы такие как: CoreMedia и AVFoundation - менеджмент медиа-данных [1, 2]; UIKit (UI) - работа с интерфейсами приложений; CoreGraphics (CG) - низкоуровневая, легковесная обработка 2D изображений на базе движка Quartz [3].

Интерфейс мобильного приложения (рисунок 1, а-в), снабжен областью просмотра изображения (1), захватываемого камерой устройства в реальном времени; областью вывода данных (3); меткой успешной фиксации на объекте распознавания (2). Область захвата имеет пропорцию равную 4:3, что является стандартом для вертикально ориентированного снимка iPhone/iPad. Область вывода содержит три вертикально расположенных текстовых поля, в которые последовательно выводится распознанная алгоритмом информация (номер банковской карты, дата срока ее истечения, фамилия и имя владельца карты). Метка детектирования карты, выполнена в виде рамки ярко-зеленого цвета с фиксированной толщиной границ, и отображается на экране в момент успешного определения положения карты, повторяя ее границы.

Захват видеопотока с основной камеры устройства реализован в отдельном потоке с помощью фреймворка AVFoundation и объекта ввода AVCaptureDeviceInput, который инициализируется объектом устройства захвата AVCaptureDevice в режиме видео съемки. Создается новая сессия AVCaptureSession, с задающимися через параметр размерами кадра. Далее при помощи метода addInput к ней добавляется ранее определенный объект ввода и через вызов startRunning - сессия запускается. Получение данных кадра осуществляется посредством метода обратного вызова didOutputSampleBuffer:fromConnectio с отображением фреймов на текущем контексте контроллера представления с помощью AVCaptureVideoDataOutput.

Согласно алгоритму работы, в первую очередь определяется ориентация кадра. После получения объекта изображения типа UIImage из данных потока, используя мета-данные от камеры устройства получим информацию о его ориентации типа UIDeviceOrientaton.

Используя объект CIDetector, и ряд опций: CIDetectorTypeRectangle (для обнаружения прямоугольных областей), CIDetectorAccuracyHigh (высокую точность распознавания), получим на выходе массив объектов типа CIRectangleFeature, с размерами и позициями детектированных на изображении прямоугольников. Вычисляя последовательно соотношение сторон каждого, выбираем только те, которые с наибольшей точностью подходят к эталонным размерам карты. Параллельным шагом становится поворот изображения кадра на угол, соответствующий вертикальному его отображению.

Отделить изображение карты по найденным выше параметрам типа CIRectangleFeature, из вертикально ориентированного снимка можно с помощью функции CGImageCreateImageInRect(), с первым параметром изображения-донора и вторым - области которую необходимо вырезать. Передавая в качестве донора - изображение банковской карты, а в качестве второго параметра последовательно вычисленные области для блоков номера карты, блока даты истечения срока действия и блока имени владельца карты получим изображения сегментов представляющих собой области интереса текущего алгоритма в формате указателей на объекты CGImage. Каждый из полученных сегментов поступает на обработку в качестве аргумента для следующего метода processBlock:cardBlockType.

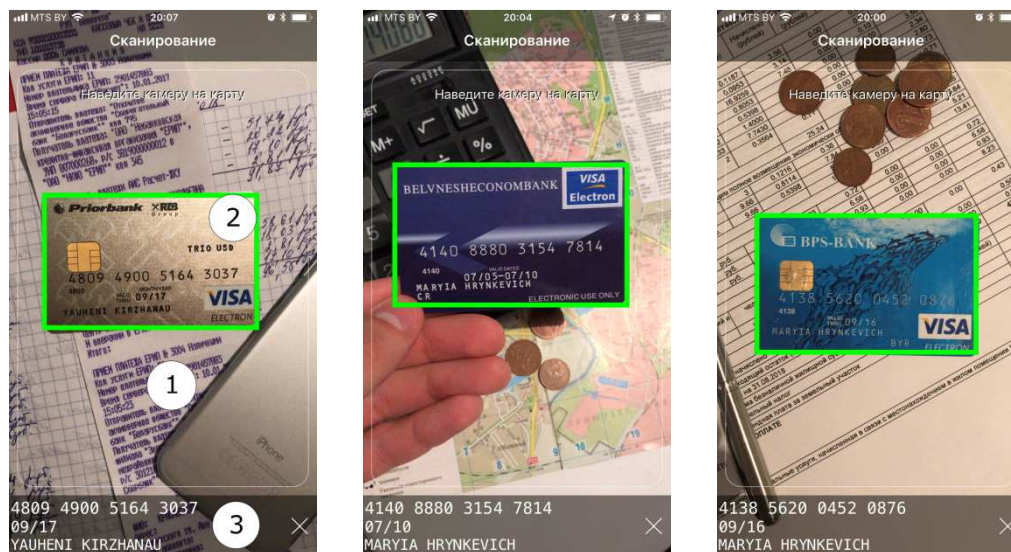


Рисунок 1. – Снимки экранов внешнего вида мобильного приложения, реализующего алгоритм распознавания реквизитов:

1 – область просмотра изображения; 2 – индикация успешного захвата карты; 3 – область вывода данных

Для поступившего на обработку изображения вычисляется его представление в формате массива OpenCV, типа Mat. Далее с помощью метода `cvtColor()` и параметра `CV_BGR2GRAY` выполняется его конвертация из цветного в оттенки серого.

Процедура повышения контрастности изображения методом нормализации гистограмм, выполняется в два этапа. Сперва вычисляется стандартная гистограмма изображения путем подсчета количества пикселей с заданными уровнем яркости в промежутке от 0 до 255, определяется средняя яркость, минимальные и максимальные значения яркости. Затем, принимая во внимание параметр погрешности ϵ , а также минимальные и максимальные уровни яркости, полученные на первом этапе, циклично с обеих сторон гистограммы выполняется подсчет количества пикселей начиная с их первоначальных экстремальных значений, когда это количество превосходит заданный в рамках погрешности порог, цикл прерывается и выбирается значения равное одному уровню яркости меньше чем текущий обрабатываемый. На выходе получаем два скорректированных значения минимальной и максимальной яркости с помощью которых вычисляем первый коэффициент контрастирования α и второй β . Далее с помощью функции OpenCV `convertTo(OutputArray, type, alpha, beta)` получаем изображение с нормализованной гистограммой [4].

Оценивая среднее значение яркости на изображении принимается решение о цвете символов на изображении, в случае если символы белые выполняется морфологическое преобразование `White Top Hat`, с помощью вызова функции `morphologyEx(inputArr, outputArr, operation, kernel)`, где параметр выбранной операции `MORPH_TOPHAT`; для темного цвета символов выполняется та же функция, но с параметром `MORPH_BLACKHAT` которая означает применение морфологической операции `Black Top Hat`. Ядро фильтра задается функцией `getStructuringElement()`, с указанием прямоугольной формы ядра `MORPH_RECT` и размеров через функцию `Size()`.

Адаптивная бинаризация выполняется путем выполнения функции OpenCV, `adaptiveThreshold()` в качестве параметров которой, передаются указатели на входной и выходной массивы данных, значение максимальной яркости для выходного изображения (всегда 255), параметр типа бинаризации `CV_ADAPTIVE_GAUSSIAN_C`, что означает применение алгоритма вычисления порога бинаризации по корреляции локально рассматриваемого блока данных с окном Гаусса (Гауссианом) [5]; `CV_TRESH_BINARY` указывает на прямой способ указания выходных значений (без инверсии). Через параметр функции `blockSize` задается значение размера рассматриваемого для определения порога бинаризации блока данных, а значение константы C носит характер смещения, по умолчанию равно 0 и может быть вычтено в процессе вычисления порога из весовой суммы [5].

Снова применяя функцию `morphologyEx(inputArr, outputArr, operation, kernel)` с параметром операции морфологии `MORPH_CLOSE`, а затем с параметром `MORPH_ERODE` получим последовательно сначала изображение к которому было применено морфологическое преобразование замыкания, а затем

операция эрозии. В обоих случаях в качестве ядра выступает матрица типа `Mat`, полученная от функции `getStructuringElement()`, с указанием формы эллипса `MORPH_ELLIPSE` и размера заданного через `Size()`.

В методе `calculateVerticalProjection:cardBlockType`: создается область окна с размером ширины изображения и с высотой равной высоте символа. Для каждого блока высота символов различна. Далее, смещая в цикле позицию окна на позицию вниз, с помощью функции `cv::countNonZero()` высчитывается количество фоновых пикселей попавших в область окна. Та позиция окна, при которой количество фоновых пикселей в ней минимально и считается позицией расположения символов в рассматриваемом блоке. Метод возвращает объект типа `CGRect` прямоугольной области занимаемой символами.

Далее выполняется процедура распознавания символов в указанной области изображения. Создается объект `G8Tesseract` с указанием в качестве инициализатора дескриптора языка для распознавания. Для случая распознавания блока номера карты, и блока даты истечения ее срока действия задается только английский язык ("`eng`"), для блока имени держателя - русский и английский ("`rus+eng`").

Экземпляр объекта `G8Tesseract` на следующем шаге получает по два варианта списков для каждого сеанса распознавания. Первый список `White List` содержит в себе символы с большим доверием, которые в процессе распознавания будут иметь больший вес для принятия позитивного решения в их сторону, и задается через параметр `charWhiteList`. Второй список `Black List`, противоположность первого варианта, содержащая набор символов с наименьшей степенью доверия. Задается этот список через параметр `charBlackList`.

При распознавании блока символов в области номера карты в качестве `White List` выступает набор содержащий только цифры: от 0 до 9. При распознавании даты срока истечения - цифры от 0-9 и символ косой черты `"/"`. В список недоверенных символов для поля держателя карты входят знаки препинания (пунктуации), а также спецсимволы, например, такие как знак равно `"="` и фигурные скобки `"{"`.

В параметр `rect` записывается экземпляр полученного на предыдущих этапах уточненного значения позиции и размера блоков символов. Указывается максимальное время на распознавание одного фрагмента, `maximumRecognitionTime`. Объекту `Tesseract` присваивается делегат на `self`, для получения результирующих данных через функцию обратного вызова, и выполняется метод `recognize`, дающий сигнал начать распознавание.

После завершения процесса распознавания системой `Tesseract`, данные результата можно получить вызвав метод `recognizedText`, экземпляра `G8Tesseract`.

После того как все блоки будут распознаны, текстовый результат каждого из них поступает в основной цикл алгоритма, там в свою очередь принимается решение о его выводе на экран с помощью объектов `NSRegularExpression` и `NSTextCheckingResult`, а также предопределенных регулярных выражений для каждого из распознаваемых блоков, принимая во внимания конкретные особенности каждого из них.

ЛИТЕРАТУРА

- 1 Developer Apple Docs [Электронный ресурс] / Core Media Framework. – Режим доступа: <https://developer.apple.com/documentation/coremedia>, свободный. – Дата доступа: 15.11.2017.
- 2 Developer Apple Docs [Электронный ресурс] / AV Foundation Framework – Режим доступа: <https://developer.apple.com/av-foundation/>, свободный. – Дата доступа: 18.11.2017.
- 3 Developer Apple Docs [Электронный ресурс] / Core Graphics Framework. – Режим доступа: <https://developer.apple.com/documentation/coregraphics/>, свободный. – Дата доступа: 22.11.2017.
- 4 OpenCV documentation [Электронный ресурс] / Miscellaneous Image Transformations. AdaptiveThreshold. – Режим доступа: https://docs.opencv.org/2.4/modules/imgproc/doc/miscellaneous_transformations.html, свободный. – Дата доступа: 30.11.2017.
- 5 OpenCV documentation [Электронный ресурс] / Image Filtering. getGaussianKernel. – Режим доступа: <https://docs.opencv.org/2.4/modules/imgproc/doc/filtering.html>, свободный. – Дата доступа: 30.11.2017.

УДК 004.932

**ПРЕДВАРИТЕЛЬНАЯ ОБРАБОТКА ИЗОБРАЖЕНИЙ БАНКОВСКИХ КАРТ
ДЛЯ РАСПОЗНАВАНИЯ ИХ РЕКВИЗИТОВ
С ИСПОЛЬЗОВАНИЕМ МОБИЛЬНЫХ УСТРОЙСТВ ПОД УПРАВЛЕНИЕМ iOS**

А.Ю. КУРИЛОВИЧ

(Представлено: канд. техн. наук, доц. Р.П. БОГУШ)

Предложен алгоритм предварительной обработки изображений банковских карт для мобильных устройств под управлением iOS, который включает такие этапы, как: детектирование области карты; сегментация изображения банковской карты; преобразование цветных сегментов к оттенкам серого; улучшение контрастности; подчеркивание границ символов с использованием операций математической морфологии; уточнение границ сгруппированных блоков символов. Выходными данными алгоритма являются также координаты уточненной области символов на изображении и используемый на карте язык.

В настоящее время большинство приложений мобильного банкинга требуют ручного ввода реквизитов банковской карты (данных всех информационных полей, нанесенных на карту) в систему для их наглядного представления на экране смартфона и совершения, в последующем, расчетных операций. Процесс заполнения каждого из этих полей в мобильных приложениях требует от человека определенного терпения, усидчивости и особой внимательности, кроме этого занимает время. Поэтому, разработка алгоритмического и программного обеспечения распознавания реквизитов банковских карт для мобильных устройств является актуальной.

Предлагаемый алгоритм предварительной обработки изображений банковских карт для мобильных устройств под управлением iOS требует выполнения следующих шагов: детектирование области карты; сегментация изображения банковской карты; преобразование цветных сегментов к оттенкам серого; улучшение контрастности; подчеркивание границ символов с использованием операций математической морфологии; уточнение границ сгруппированных блоков символов.

В качестве входных данных алгоритма выступает полученный с камеры устройства видеопоток, разделенный во времени на кадры, представляющие собой изображения в цветовом пространстве RGB, без альфа-канала. Учитывая характеристику устройства iPhone 6 (2014 г.в.), камера обладает 8-мегапиксельной ПЗС-матрицей, вспышкой из двух светодиодов и автофокусировкой. Размеры кадра при этом могут варьироваться в зависимости от конкретной модели устройства, и достигать размеров 3840×2160 пикселей. Все устройства младше вышеуказанного обладают более совершенными характеристиками. Операционная система (выше iOS 2) способна выполнять автоматическое удержание фокуса на объекте съемки, автоматическую регулировку яркости и баланса белого на фото. Вышеперечисленное говорит о том, что изображение, поступающее напрямую от системы с фотокамеры, будет иметь оптимальные входные параметры: глубину резкости, яркость, достаточный размер кадра для передачи деталей банковской карты.

Отделение изображения банковской карты от фона осуществляется путем обработки его алгоритмами iOS, которые базируются на методе скоростного обнаружения объектов Виолы—Джонса [1] и глубоком обучении по методу OverFeat [2].

Критерием выбора контура карты из множества детектированных прямоугольников R , где количество найденных прямоугольных областей N , выступает соотношение размеров ее сторон. Поскольку размеры сторон карты ($m_o \times n_o$) определены ISO/IEC 7811-1:2017, соотношение ее сторон является постоянной величиной $\frac{n_o}{m_o}$. Представим объект найденного прямоугольника r_i как $R_i(m_i, n_i)$, где $i = 0, 1, \dots, N-1$. Тогда относятся к регионам интереса будут прямоугольники r для которых выполняется условие (1). Точным ответствием считается прямоугольник с наибольшей длинной стороны n_{max} из всех отображенных.

$$\frac{n_i}{m_i} = \frac{n_o}{m_o} \quad (1)$$

Размеры прямоугольной области $r_o(m_{max}, n_{max})$ и ее расположение относительно целого изображения $p_o(x_o, y_o)$ используются для извлечения области карты из оригинального кадра.

Из изображения карты I с размерами $m_{max} \times n_{max}$ (далее $m_i \times n_i$) выделяются области интереса алгоритма, содержащие информацию о номере банковской карты, дате истечения срока ее действия, владельце. Размер и расположение этих областей определено ISO/IEC 7811-1:2017 и может быть записано в следующем представлении:

- область $C(x_C, y_C, n_C, m_C)$ – номер карты;
- область $D(x_D, y_D, n_D, m_D)$ – дата срока истечения;
- область $E(x_E, y_E, n_E, m_E)$ – имя держателя карты.

Поскольку размеры $m_I \times n_I$ полученного изображения карты I могут варьироваться, необходимо преобразовать C , D и E к виду, который будет применим для их корректного отделения от изображения I . Для этого необходимо определить два коэффициента масштабирования: по ширине (2) и по высоте (3).

$$C_{scale.width} = \frac{n_I}{n_C} \quad (2)$$

$$C_{scale.height} = \frac{m_I}{m_C}, \quad (3)$$

Умножив каждый из параметров обозначенных выше областей на соответствующий ему коэффициент масштабирования получим размеры этих областей относительно размера полученного изображения карты I .

$$\begin{aligned} C_I(x_C \cdot C_{scale.width}, y_C \cdot C_{scale.height}, n_C \cdot C_{scale.width}, m_C \cdot C_{scale.height}) \\ D_I(x_D \cdot C_{scale.width}, y_D \cdot C_{scale.height}, n_D \cdot C_{scale.width}, m_D \cdot C_{scale.height}) \\ E_I(x_E \cdot C_{scale.width}, y_E \cdot C_{scale.height}, n_E \cdot C_{scale.width}, m_E \cdot C_{scale.height}) \end{aligned}$$

Отделив фрагменты областей C_I , D_I , E_I от изображения карты I , получим изображения: I_C , I_D , I_E соответственно номера карты, время истечения ее срока действия и имени владельца.

Согласно ISO/IEC 7810 - 2006 номер банковской карты содержит 16 цифр распределенных на 4 равные группы по 4 цифры в каждой. Исходя из этого выделенное на предыдущем этапе изображение I_C с областью номера карты C_I , разделяется на 4 равные области C_{I1} , C_{I2} , C_{I3} , C_{I4} (4).

$$C_{Ij}(x_{Cj} + (\frac{n_{Cj}}{4} \times (j - 1)), y_{Cj}, \frac{n_{Cj}}{4}, m_{Cj}) \quad (4) \\ j=1,2,3,4$$

Выделив рассчитанные области C_{I1} , C_{I2} , C_{I3} , C_{I4} на изображении I_C получим соответственно изображения I_{C1} , I_{C2} , I_{C3} , I_{C4} - групп номера карты.

Следующим шагом является преобразование изображения в полутоновое. Далее для повышения яркостной разницы между контурами символов и фоном изображений применяется процедура повышения контрастности. Наиболее эффективным для данной задачи является метод нормализации гистограммы, так как выполняет растяжку не всего диапазона интенсивностей, а только его наиболее информативного участка. Это позволяет не рассматривая истинные экстремальные значения яркостей назначить условия их определения с заданной погрешностью, что усиливает эффект контрастности за счет потери шумовых областей с редко встречающимися интенсивностями [3].

Для определения цвета фона изображения и цвета символов мы вычисляем среднюю яркость изображения Y_{aver} . Если $Y_{aver} > 127.5$ то принимаем цвет фона полученного изображения как светлый, а цвет символов как темный, в противном случае – наоборот. С использованием операций математической морфологии, осуществляется подчеркивание границ символов. Если цвет символов светлый, используется функция морфологического преобразования White Top Hat [4]. Для темных символов – функция Black Top Hat ($T_b(I)$). Структурирующий элемент для ядер обоих фильтров имеет прямоугольную форму и размер $n_b \times m_b$, рассчитывающиеся по формуле (5), где n_I , ширина поступившего для морфологического преобразования изображения.

$$\begin{aligned} n_b &= n_I \cdot 0.06 \\ m_b &= \frac{n_b}{3} \end{aligned} \quad (5)$$

На следующем этапе выполняется бинаризация изображений с адаптивным порогом на основе анализа локального региона [5, 6]. Метод преобразует изображение в градациях серого I к монохромному изображению согласно (6).

$$bin_{x,y} = \begin{cases} g_{max}, & \text{if } Y_{xy} > T(Y_{xy}, block_size, c) \\ 0, & \text{else} \end{cases} \quad (6)$$

где $Y_{x,y}$ значение яркости пикселя с координатами x, y изображения I ; $g_{max}=255$ - значение максимальной яркости; $T(Y_{x,y}, block_size, c)$ - адаптивный порог бинаризации, рассчитываемый индивидуально для каждого пикселя; где функция T - представляет собой весовую сумму [5] (взаимную корреляцию с окном Гаусса, Гауссианом (7) [6]) блока пикселей размером $block_size \times block_size$ соседних с обрабатываемым пикселем (с координатами x, y) минус коэффициент c [5].

$$G_i = \alpha \cdot e^{-\frac{(i - \frac{(block_size - 1)}{2})^2}{2 \cdot \sigma^2}} \quad (7)$$

где $i=0..block_size-1$, α выбрана таким образом, что $\sum_i G_i = 1$

Для уменьшения количества шумов, удаления из бинаризованного изображения лишних неинформативных деталей, и грубого приближения символов к изолированным закрасленным блокам, отличающимся от фона изображения - к нему применяются последовательно морфологические операции закрытия и эрозии.

Структурный элемент b ядер обоих фильтров имеет эллипсную форму, что лучше подходит при обработке символов шрифт которых, OCR-B, согласно ISO/IEC 7811-1:2017, имеет сглаженные края.

Размер ядер $n_b \times n_b$, где n_b - не должен превышать размера равного трети толщины символа w_{syml} , иначе появляется вероятность утери контура символа при обработке.

Толщина символа w_{syml} не зависит от типа обрабатываемого фрагмента изображения карты, и является строго определенной для OCR-B. Проекция значения ширины шрифта (мм.) на размер в пикселях относительно размера области полной банковской карты, рассчитывается на основании ее полной ширины (8), и является константой в рамках обработки текущей области карты.

$$w_{syml} = n_0 \cdot 0.004884 \quad (8)$$

Для уточнения краев области символов на обработанном изображении, используется метод скользящего вертикально окна. Высота окна варьируется от размера шрифта символов. Для банковской карты это значения определяемые ISO/IEC 7811-1:2017, H_{cn} - высота шрифта номер карты (4,0мм), H_{exd} - высота шрифта поля даты срока истечения карты (2,85мм), H_{mn} - высота шрифта поля держателя карты (2,65мм). Тогда их проекции на высоту в пикселях относительно размеров области полной карты h_{cn} , h_{exd} , h_{mn} . Общий вид расчета представлен формулой (9),

$$h = \frac{H \cdot m_0}{54.0} \quad (9)$$

где 54.0 мм - высота карты по ISO 7810 ID-1, а m_0 - высота детектированной области карты (пкс.).

Выходными данными алгоритма являются обработанные и улучшенные изображения полей реквизитов банковской карты с параметрами: уточненная область символов, полученная на предыдущем этапе; язык.

В данной статье решена задача разработки алгоритма предварительной обработки изображений банковских карт для распознавания их реквизитов. Предложенный алгоритм представляет собой описание обработки полученного с камеры устройства кадра видеопоследовательности и включает в себя: определение контуров карты по алгоритму скоростного обнаружения объектов Виолы—Джонса и методу OverFeat; выделение областей интереса; улучшение качества символов в этих областях с помощью метода нормализации гистограммы и морфологических преобразований Top Hat; уточнение границ символов через применение бинаризации с адаптивным порогом, выполнение морфологических операций над результатами, а после поиска максимально подходящей области с помощью вертикально скользящего окна.

ЛИТЕРАТУРА

1. Viola, P. and Jones, M.J. Robust Real-time Object Detection Using a Boosted Cascade of Simple Features. In Proceedings of the Computer Vision and Pattern Recognition Conference, 2001.
2. Sermanet, Pierre, David Eigen, Xiang Zhang, Michael Mathieu, Rob Fergus, and Yann LeCun. OverFeat: Integrated Recognition, Localization and Detection Using Convolutional Networks. arXiv:1312.6229 [Cs], December, 2013.

3. Кустикова, В.Д. Разработка мультимедийных приложений с использованием библиотек OpenCV и IPP : учеб. пособие / В.Д. Кустикова. – Н. Новгород : Нижегород. гос. ун-т им. Н.И. Лобачевского, 2012.
4. Гонсалес, Р. Цифровая обработка изображений / Р. Гонсалес, Р. Вудс. – М. : Техносфера, 2005. – 1072 с.
5. OpenCV documentation [Электронный ресурс] / Miscellaneous Image Transformations. AdaptiveThreshold. – Режим доступа: https://docs.opencv.org/2.4/modules/imgproc/doc/miscellaneous_transformations.html, свободный. – Дата доступа: 30.11.2017.
6. OpenCV documentation [Электронный ресурс] / Image Filtering. getGaussianKernel. – Режим доступа: <https://docs.opencv.org/2.4/modules/imgproc/doc/filtering.html>, свободный. – Дата доступа: 30.11.2017.

УДК 004.81

НЕЙРОННЫЕ СЕТИ И ЭКСПЕРТНЫЕ СИСТЕМЫ**И.М. ПЕРЕЛЫГИН***(Представлено: канд. техн. наук, доц. И.Б. БУРАЧЕНОК)*

Изучены понятия искусственного интеллекта. Рассмотрены подходы к разработке искусственного интеллекта. Разобрано понятие искусственной нейронной сети и принципы ее работы. Изучено понятие экспертной системы принципы и режимы работы. Проведено сравнение нейронных сетей и экспертных систем.

С появлением компьютеров, все больше функций переходит с человека на машину. Но компьютер может выполнять только действия по заранее созданному алгоритму. Для решения нетривиальных задач, которые можно также назвать интеллектуальными, и ведутся разработки по созданию искусственного интеллекта. Джон Маккарти в 1956 году во время дармутской конференции дал следующее понятие: «Искусственный интеллект – это наука и сопутствующие технологии по созданию интеллектуальных машин. А именно – интеллектуальных компьютерных программ» [1]. И вот спустя полвека, с появлением достаточно мощных аппаратных средств, искусственный интеллект активно используется людьми в различных сферах: промышленность, сельское хозяйство, медицина и многие другие сферы деятельности человека. Есть мнение, что в век информационных технологий, с тем огромным количеством информации, объемы которой постоянно растут, искусственный интеллект может стать одной из ключевых технологий.

Во-первых, следует рассмотреть, что взято за основу искусственного интеллекта. Эта наука связана с психологией, нейрофизиологией, трансгуманизмом, философией и другими, казалось бы, никак не связанными с информационными технологиями.

Можно выделить два подхода к разработке искусственного интеллекта, а именно, нисходящий и восходящий. Нисходящий подход подразумевает создание символьных систем, моделирующих высокоуровневые психические процессы: мышление, рассуждения. Восходящий подход подразумевает изучение нейронных сетей человека, и создание подобных им моделей, имитирующих интеллектуальное поведение, на основе более мелких «неинтеллектуальных» элементов. При использовании восходящего подхода, по сути, имитируется работа правого полушария головного мозга. Нисходящий подход ориентирован на имитацию обработки информации левым полушарием мозга человека. При данном подходе, система должна была воспроизводить ход логических рассуждений эксперта, специалиста в конкретной предметной области.

Искусственная нейронная сеть (ИНС), которую чаще называют просто нейронной сетью, представляет из себя математическую или компьютерную модель, построенную по принципу работы биологических нейронных сетей. В большинстве случаев ИНС является адаптивной системой, изменяющей свою структуру, основываясь на обработке входящей и исходящей информации, подаваемой во время фазы обучения. ИНС состоит из связанной группы искусственных нейронов. Искусственный нейрон – узел искусственной нейронной сети, являющийся упрощенной моделью естественного нейрона. Математически искусственный нейрон представляет собой взвешенный сумматор, единственный выход которого определяется через его входы и матрицу весов следующим образом: $y = f(\sum_{i=1}^n w_i \cdot x_i + b \quad n_i = 1)$. Здесь x_i сигналы на входах нейрона, w_i веса входов, а функция f называется функцией активации нейрона. Функция активации f определяет зависимость сигнала на выходе нейрона от взвешенной суммы сигналов на его входах. Искусственный нейрон полностью характеризуется своей функцией активации. Результат поступает в пороговую функцию, которая представляет собой перепад, пока взвешенный сигнал на входе нейрона не достигает некоторого порогового уровня – сигнал на выходе равен нулю. Обучение нейронной сети сводится к изменению весов входов нейронов сети.

В последние годы популярность ИНС достигла колоссальных высот, они применяются практически во всех сферах хозяйства человека. Ведущие компании мира вкладывают огромные деньги на исследования в данной области. Обученные нейронные сети проходят тест Тьюринга []. Постоянно на рынок выходят продукты, с использованием технологий ИНС.

Сейчас как нейронные сети, так и экспертные системы, относят к направлению искусственного интеллекта, хотя этот термин появился в 70-е годы в связи с экспертными системами, как направления альтернативного нейронным сетям. Экспертная система (ЭС) – это программное средство, использующее знания экспертов, для высокоэффективного решения задач в интересующей пользователя предметной

области [2]. ЭС претендовали именно на решение важных прикладных задач в узких областях знания. Основными компонентами экспертной системы является: база знаний и «решатель проблем». База знаний содержит данные (факты), и правила (знания). Экспертные системы используют правила для принятия решений на основе фактов. У экспертных систем есть два основных режима работы: режим приобретения знаний и режим решения задачи. Режим приобретения знаний, аналогично процессу обучения нейронной сети, служит для наполнения системы данными, которые, впоследствии, позволят системе решать поставленные перед ней задачи. В этом режиме эксперт, наполняет систему данными (знаниями), которые в свою очередь позволяют системе решать задачи из данной предметной области уже без участия эксперта.

К преимуществам ЭС над ИНС можно отнести то, что ЭС в режиме работы может объяснить причину сделанного выбора, показать ход решения поставленного вопроса. В свою очередь ИНС работают как черный ящик, нельзя узнать, как был получен данный результат. Принято считать, что обучение нейронной сети гораздо более быстрый процесс, чем создание экспертной системы, но в условиях узкоспециализированной области, для которой необходим сбор и структуризация данных, создание экспертной системы никак не проигрывает по временным затратам нейронным сетям, а возможно даже имеет преимущество. Теперь, когда огромное количество данных собирается, систематизируется, и хранится в базах данных, возможно создание очень обширных баз знаний, на основании которых, экспертные системы смогут очень качественно решать поставленные перед ними задачи. Совмещение знаний экспертов из разных отраслей позволяет ЭС делать заключения, к которым сложно было бы придти даже при анализе человеком. Например, системе PROSPECTOR, сочетающей знания девяти экспертов, удалось обнаружить залежи руды, стоимостью в миллион долларов, при этом, наличие этих залежей не предполагал ни один из экспертов.

Экспертные системы хорошо зарекомендовали себя во многих областях: медицина, экономика, промышленность. Но, к сожалению, не получили такой популярности как нейронные сети. Давно не происходило крупных событий, связанных с экспертными системами. Новые экспертные системы, как правило, создаются для частных компаний, и их создание не поддается широкой огласке. Также на развитие экспертных систем не выделяется такое большое финансирование, как на развитие нейронных сетей.

Экспертные системы использовались при создании приложения виртуальной реальности для обучения специалистов газового хозяйства, позволяющего получить практические навыки по пуску и продувке газорегуляторного пункта (ГРП). Выбор был основан на том, что область узконаправленная, и сбор информации совместно с ее структуризацией для обучения нейронной сети увеличил бы время создания программного продукта в сравнении с ЭС. Так же использование ЭС позволяет быстро адаптировать алгоритм под различные конфигурации ГРП. При использовании нейронной сети, при изменении конфигурации ГРП, пришлось бы «переобучать» нейронную сеть, а при использовании экспертной системы достаточно просто дописать факты и правила, касающиеся нового узла. Например, правило: «если газ перед регулятором, взведена ручка регулятора, кран на импульс закрыт, то допущена ошибка». Подобные правила просты, и понятны не только разработчику, благодаря чему система легко может объяснить пользователю, какую ошибку он совершил. Так же использование нейронной сети не может гарантировать точность определения ошибки, так как результат будет наверняка верным, только если данный случай был описан в обучающей выборке.

В заключении исследования можно сделать вывод, что нейронные сети в отличие от экспертных, не производят реальных рассуждений, а по сути, запоминают закономерности, в соответствии с которыми, и выдают результат. Для создания настоящего искусственного интеллекта необходима интеграция описанных направлений, создание систем, которые сочетали бы качества, как нейронных сетей, так и экспертных систем.

ЛИТЕРАТУРА

1. **McCarthy, John.** What is artificial intelligence? / John McCarthy. – 2007.
2. Инфоградия [Электронный ресурс] / – Режим доступа: <https://infopedia.su/>. – Дата доступа: 14.09.2018.

УДК 004.89

РАЗРАБОТКА ЛОГИЧЕСКОЙ СТРУКТУРЫ И ТЕСТОВОГО ЗАДАНИЯ
ДЛЯ ПРИЛОЖЕНИЯ ВИРТУАЛЬНОЙ РЕАЛЬНОСТИ

И.М. ПЕРЕЛЫГИН, А.А. АНАНЕНКО

(Представлено: канд. техн. наук, доц. И.Б. БУРАЧЕНОК)

Сформированы требования к алгоритму тестового задания для приложения виртуальной реальности по обучению специалистов газового хозяйства. Рассмотрены возможные подходы к построению логической структуры тестового задания, проблемы, связанные с использованием этих подходов. Обоснован выбор подхода, на основании которого спроектирован алгоритм. Сделан вывод об эффективности использования данного подхода.

В результате проделанной работы было разработано приложение виртуальной реальности для обучения специалистов газового хозяйства, позволяющее получить практические навыки по пуску и продувке газорегуляторного пункта (ГРП). При реализации проекта заказчиком поставлена задача: спроектировать логическую структуру программы, предусмотрев возможность производить пуск и продувку ГРП различными способами. Также стоит принять во внимание, что ГРП несмотря на то, что являются типовыми сооружениями, могут отличаться по конфигурации, оборудованию и способу подключения к газовой сети.

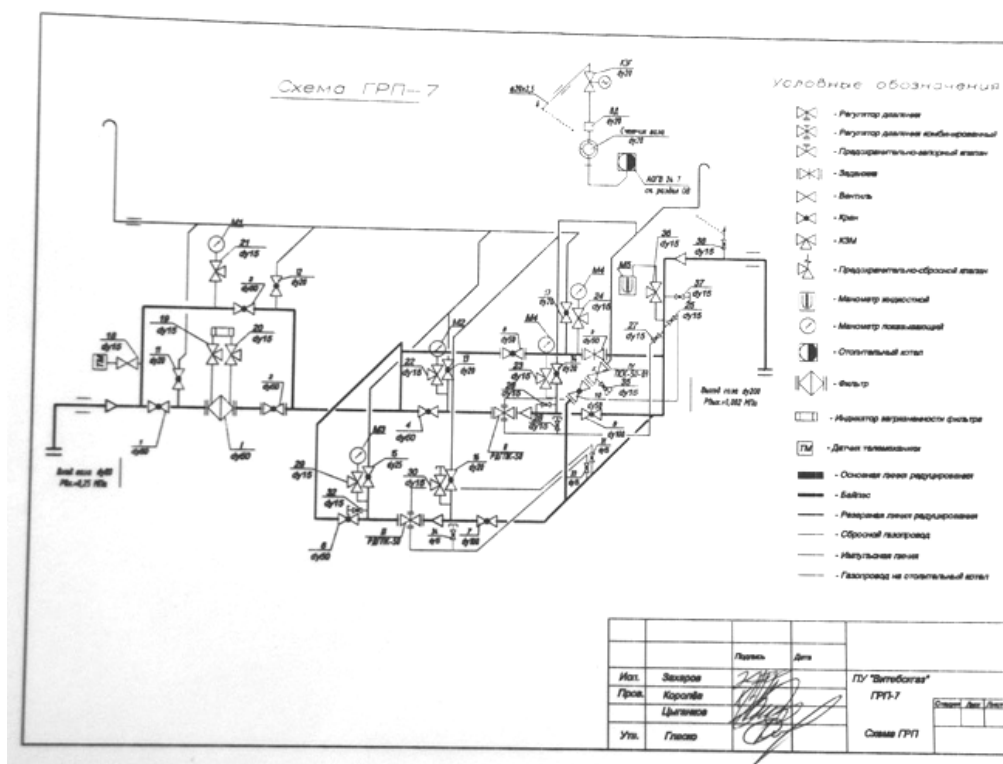


Рисунок. – Схема газорегуляторного пункта

Все эти факторы в значительной мере могут повлиять на алгоритм производимых работ. Следовательно, возникает необходимость в разработке логики программы с возможностью адаптации алгоритма выполнения задания в сжатые сроки под новые ГРП и виды оборудования. Также крайне важным условием для реализации алгоритма являлась задача точного соответствия технологической инструкции по пуску продувке ГРП, поскольку даже незначительные ошибки, допущенные во время выполнения данных операций, могут привести к серьезным последствиям.

Был сформирован список требований, предъявляемых к алгоритму, в соответствии с приведенными ранее условиями. В первую очередь, возможность легкой адаптации алгоритма, под различные конфигурации ГРП и абсолютной его точности по определению верности действий пользователя, а также,

при условии использования технологий виртуальной реальности и ограниченности аппаратных возможностей оборудования, требования к вычислительным ресурсам системы для выполнения алгоритма должны быть минимальными.

Отметим, то что традиционные методы программирования, для достижения поставленной цели, не подходят, так как алгоритм, основанный на данном подходе крайне трудно адаптировать под различные типы ГРП, и его разработка отнимает много времени.

Создание алгоритма на основе графов состояний также не целесообразно, так как необходимо предусмотреть, и описать в матрице все возможные варианты, при которых допущена ошибка. В этом случае необходимо составить такой граф для каждого ГРП. При данном подходе велика вероятность возникновения ошибки и, он также не соответствует требованию адаптивности алгоритма и не рационален в плане временных затрат.

Была рассмотрена возможность применения нейронных сетей. Искусственная нейронная сеть (ИНС), которую чаще называют просто нейронной сетью, представляет из себя математическую или компьютерную модель, построенную по принципу работы биологических нейронных сетей [1]. При условии качественного обучения нейронной сети, она могла бы достаточно хорошо подойти для решения поставленной задачи, но, к сожалению, область, которую должен описывать наш алгоритм, достаточно узкая и готовых данных для обучения ИНС нет. Следовательно, обучение нейронной сети потребует больших временных затрат. Также адаптация алгоритма под различные ГРП, заставляет столкнуться с проблемами, возникающими при переобучении нейронных сетей. Переобучение (overfitting) – одна из проблем глубоких нейронных сетей, состоящая в следующем: модель хорошо объясняет только примеры из обучающей выборки, адаптируясь к обучающим примерам, вместо того чтобы учиться классифицировать примеры, не участвовавшие в обучении (теряя способность к обобщению) [2]. Так как, по сути, нейронные сети не делают логических заключений, а просто запоминают закономерности, то нельзя гарантировать, что будет наверняка определено, допустил пользователь ошибку при пуске продувке ГРП или нет.

Другое направление в области искусственного интеллекта. Экспертная система (ЭС) – это программное средство, использующее знания экспертов, для высокоэффективного решения задач в интересующей пользователя предметной области [3]. Данный подход к созданию алгоритма, имеет дополнительный плюс, так как можно отследить ход решения поставленного вопроса ЭС. В отличии от нейронной сети, которая выступает как черный ящик. Использование ЭС позволяет легко и быстро адаптировать алгоритм под различные конфигурации ГРП. Все основные принципы работы должны быть описаны в правилах, при адаптации алгоритма под новую конфигурацию ГРП нужно только описать правила лишь для отличающихся узлов. Описание правил работы ГРП в ЭС, позволит точно определять наличие ошибки в действиях пользователя. Принято считать, что обучение нейронной сети более быстрый процесс, чем создание ЭС, но в условиях узкоспециализированной области, для которой необходим сбор и структуризация данных, создание ЭС никак не проигрывает по временным затратам нейронным сетям, а возможно даже имеет преимущество.

Таким образом, было принято решение разделить логическую структуру программы условно на базу знаний и ЭС. База знаний состоит из данных (фактов) и правил (знаний). Факт имеет форму строки, в которой перечислены все условия. Последним записывается правило, которое должно выполниться при наличии всех требуемых условий. Правило представляет собой метод, который выполняет определенное действие. ЭС после каждого действия проверяет все факты и запускает соответствующие правила. Если факт был использован, то он становится неактивным. Если условие из неактивного факта пропадет, то факт снова становится активным. Все факты имеют равный приоритет, за исключением фактов, связанных с допуском ошибки. Такие факты имеют более высокий приоритет.

В результате проведенного анализа вариантов построения логической структуры тестового задания, установлено, что при выявленных к алгоритму требованиях, наиболее оптимальным является использование модели логической структуры программы основанной на принципах реализации ЭС. При тестировании программного продукта с заданной структурой тестового задания получены положительные результаты. Скорость работы алгоритма соответствует заданным требованиям и на контрольном компьютере в худшем случае составляет менее 0,001 секунды.

ЛИТЕРАТУРА

1. Fandom [Электронный ресурс]. – Режим доступа: http://ru.science.wikia.com/wiki/Искусственная_нейронная_сеть/. – Дата доступа: 17.09.2018.
2. Habr [Электронный ресурс]. – Режим доступа: <https://habr.com/company/wunderfund/blog/330814/>. – Дата доступа: 17.09.2018.
3. Инфопедия [Электронный ресурс]. – Режим доступа: <https://infopedia.su>. – Дата доступа: 14.09.2018.

УДК 004.021

**ПРОЕКТИРОВАНИЕ ГРАФИЧЕСКОГО ИНТЕРФЕЙСА
ДЛЯ АВТОМАТИЗИРОВАННОЙ ИНФОРМАЦИОННОЙ СИСТЕМЫ
ТУРИСТИЧЕСКОГО ЦЕНТРА УЧРЕЖДЕНИЯ ОБРАЗОВАНИЯ
«ПОЛОЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

А.И. ПОПОВ

(Представлено: канд. тех. наук, доц. И. Б. БУРАЧЕНОК)

Рассмотрены принципы построения графического интерфейса пользователя в приложении на языке программирования Java. Проведен сравнительный анализ двух наиболее популярных фреймворков по созданию пользовательского интерфейса, с их кратким описанием, анализом преимуществ и недостатков каждого. Приведено краткое описание спроектированного интерфейса для разрабатываемого приложения.

В каждом современном языке программирования существует собственный набор методов и средств для разработки графического интерфейса. В языке программирования Java, для настольных приложений, чаще всего используются такие библиотеки как JavaFX с проектированием интерфейса на основе fxml-файлов и Swing с непосредственным созданием элементов в коде программы.

В представленной статье приведем сравнение некоторых технологий проектирования интерфейса настольных приложений, написанных на языке Java.

Как правило, современные приложения строятся по шаблону MVC (Model-View-Controller или Модель-Вид-Контроллер). Идея данного шаблона проста – разделение обязанностей: задача контроллера – обработка действий пользователя (нажатие по кнопкам, обработка запросов к серверу и т.д.); модель – предоставляет контроллеру представление данных, запрашиваемых пользователем; вид – обеспечивает представление данных, полученных из модели.

В Swing элементы пользовательского интерфейса размещаются непосредственно при помощи кода приложения. При этом указываются параметры, присваиваемые элементу управления, от его координат относительно родительского элемента, до различных свойств элемента. Возможно выбирать из предоставляемого набора средств, реализованных в стандартных классах библиотеки Swing, либо реализовать необходимые свойства самостоятельно [1].

Перечислим основные преимущества библиотеки Swing:

- содержит достаточно богатый и удобный набор элементов пользовательского интерфейса;
- практически не зависит от той платформы, на которой должна выполняться программа, следовательно, меньше подвержена ошибкам, характерным для конкретной платформы;
- обеспечивает одинаковое восприятие конечными пользователями приложений с графический пользовательский интерфейс на разных платформах;
- встроенный редактор форм почти во всех средах разработки;
- на базе свинга есть много расширений типа SwingX;
- поддержка различных стилей (Look And Feel).

Однако можно указать на следующие недостатки:

- окно с множеством компонентов начинает подтормаживать;
- работа с менеджерами компоновки становится затруднительной в сложных интерфейсах.

Листинг 1 демонстрирует пример создания контекстного меню с одним элементом с использованием библиотеки Swing.

```
JMenuBar mb = new JMenuBar();
JMenu m1 = new JMenu("FILE");
mb.add(m1);
JMenuItem m22 = new JMenuItem("Save as");
m1.add(m22);
```

Листинг 1. – Пример создания меню приложения на Swing

JavaFX является более современным решением для разработки пользовательского интерфейса настольных приложений на Java. Для отрисовки используется графический конвейер, что значительно ускоряет работу приложения. Набор встроенных компонентов обширен, есть отдельные компоненты для отрисовки графиков. Реализована поддержка мультимедийного контента, множество эффектов отображения, анимации. Внешний вид всех компонентов можно легко изменить с помощью CSS-стилей.

Для создания компонентов используются специальные xml/fxml-файлы, в которых при помощи тегов, параметров и свойств создаются элементы пользовательского интерфейса. Также существует большое количество графических редакторов для проектирования интерфейса на JavaFX с последующей интерпретацией в xml формат. Листинг 2 демонстрирует пример создания контекстного меню с одним элементом с использованием библиотеки JavaFX. Данный код размещается в отдельном fxml файле, доступ ко всем элементам происходит по присвоенному им id.

```
<MenuBar layoutY="2.0" prefHeight="25.0" prefWidth="600.0">
  <Menu mnemonicParsing="false" text="File" id="menuBar">
    <items>
      <MenuItem mnemonicParsing="false", text="Close" id="menuItem"/>
    </items>
  </Menu>
</MenuBar>
```

Листинг 2. – Создание меню с помощью JavaFX

Достоинства фреймворка JavaFX:

- быстрая работа за счет графического конвейера при компиляции и запуске проекта это ускоряет процесс обработки кода и работу приложения;
- множество различных компонентов. Фреймворк позволяет использовать для создания интерфейса, как набор стандартных элементов, так и сторонние пользовательские элементы;
- поддержка стилей. JavaFX поддерживает CSS стилизацию всех компонентов интерфейса;
- утилиты для создания программы. Существует большое количество графических утилит, позволяющих моделировать интерфейс при помощи графического отображения элементов с последующим преобразованием в fxml;
- приложение можно запускать как десктопное и в браузере как часть страницы. Построение интерфейса при помощи fxml файлов и CSS стилей позволит при необходимости перенести интерфейс приложения на веб-форму.

Недостатки JavaFX:

- фреймворк активно развивается, поэтому случаются и падения, и некоторые «глюки», избежать непредвиденных проблем можно не используя элементы, добавленные в последнем обновлении и не злоупотребляя различными пользовательскими расширениями для фреймворка;
- JavaFX полностью привязан к платформе языка Java, однако так как данный проект будет разрабатываться как настольное приложение на языке Java. Однако данный минус не является существенным. Проблемы установки программы на компьютеры пользователей решаются установкой на них JVM [2].

Приложение будет иметь несколько форм, одна из которых считается главной и при запуске программы отображается первой. При закрытии главной формы приложения, прекращается работа всего приложения, при этом также закрываются все открытые окна приложения.

Главная форма приложения будет представлять из себя окно, с меню выбора таблиц из базы данных. После выбора таблицы будут выводиться все хранимые данные в таблицу, расположенную в форме. При помощи боковой панели пользователь сможет управлять данными: заносить новые, редактировать или удалять старые, осуществлять поиск по какому-либо параметру. Кроме того, с помощью кнопок можно будет вызывать дополнительные формы для просмотра графика работы сотрудников и проведения определенным сотрудником мероприятий.

Макеты разработанных форм продемонстрированы на рисунках 1, 2.

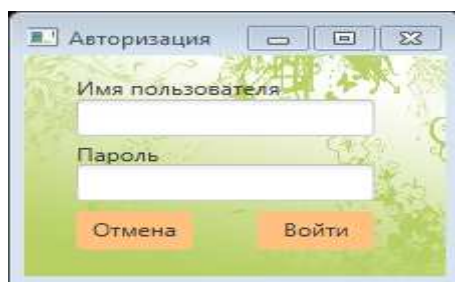


Рисунок 1. – Окно авторизации пользователя

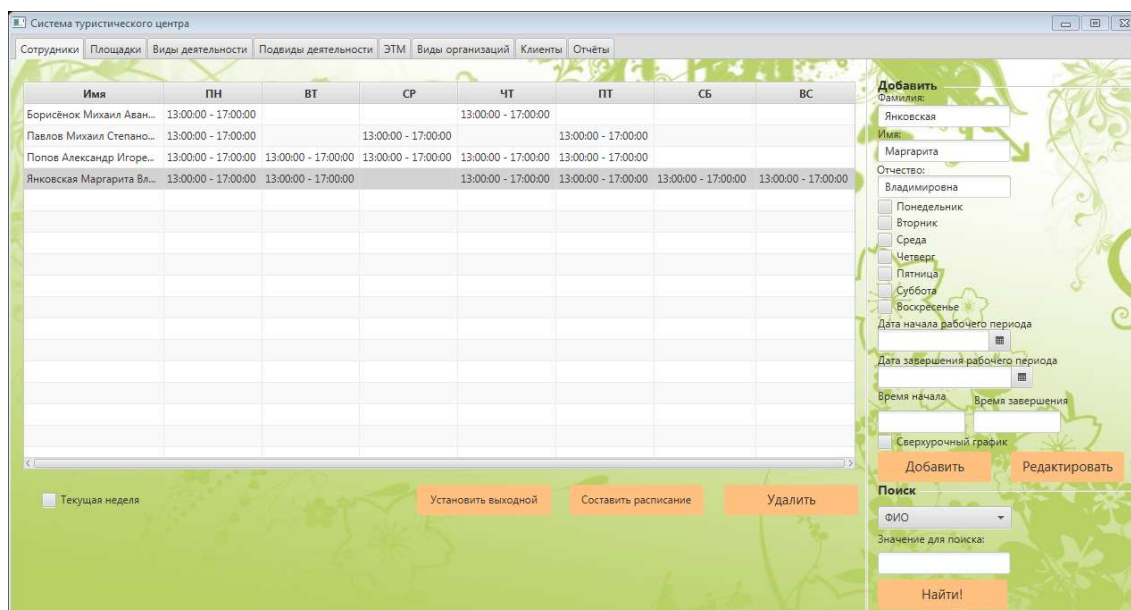


Рисунок 2. – Рабочее окно приложения

Кроме того, приложение должно генерировать отчёты по хранимым данным. Пример формы отчётов продемонстрирован на рисунке 3.

	A	B	C	D	E
1	Отчёт о проводимых мероприятиях				
2	Название мероприятия	Сотрудник	Дата проведения	Начало экскурсии	Конец экскурсии
3	Экскурсия в кадетский корпус	Янковская Маргарита Владимировна	2018-04-30	08:45:00	09:15:00
4	Экскурсия в кадетский корпус	Янковская Маргарита Владимировна	2018-05-01	08:30:00	09:00:00
5	Обзорная экскурсия	Попов Александр Игоревич	2018-05-22	15:00:00	16:00:00
6					
7					
8					
9					
10	Подпись руководителя ТЦ:	М.В. Янковская			
11					
12					
13					
14					

Рисунок 3. – Пример формы сгенерированного отчёта

Таким образом, после анализа наиболее популярных фреймворков для реализации пользовательского интерфейса настольных приложений было принято решение использовать библиотеку JavaFX. Для облегчения проектирования интерфейса будет использоваться графическая утилита JavaFX SceneBuilder. После выбора средств разработки интерфейса составлены макеты основных экранных форм для приложения и примеры формы генерируемых приложением отчётов.

ЛИТЕРАТУРА

1. Краткий обзор GUI-фреймворков для Java [Электронный ресурс] / EAX. – Режим доступа: <https://eax.me/java-swing/>. – Дата доступа: 09.09.18.
2. GUI на Java [Электронный ресурс] / Хакер. – Режим доступа: <https://xaker.ru/2014/09/10/java-gui/>. – Дата доступа: 09.09.18.

УДК 004.021

**ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ
ДЛЯ АВТОМАТИЗИРОВАННОЙ ИНФОРМАЦИОННОЙ СИСТЕМЫ
ТУРИСТИЧЕСКОГО ЦЕНТРА УЧРЕЖДЕНИЯ ОБРАЗОВАНИЯ
«ПОЛОЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

А.И. ПОПОВ*(Представлено: канд. тех. наук, доц. И. Б. БУРАЧЕНОК)*

Рассмотрены основные принципы проектирования базы данных для автоматизированной информационной системы Туристического центра учреждения образования «Полоцкий государственный университет». Описаны задачи, которые должна реализовывать разрабатываемая система. На основании анализа выделенных преимуществ осуществлен выбор системы управления базы данных для реализации программного обеспечения.

Базы данных значительно помогают облегчить процесс мониторинга больших объёмов данных, тем самым отлично подходят для внедрения в структуру различных автоматизированных информационных систем (АИС).

В представленной статье приводится описание основных методов проектирования базы данных для АИС Туристического центра учреждения образования «Полоцкий государственный университет».

Цель моделирования данных состоит в обеспечении разработчика концептуальной схемой базы данных в форме одной модели или нескольких локальных моделей, которые относительно легко могут быть отображены в любую систему баз данных. Проектирование базы данных является важным этапом в разработке приложения, так как разработанная модель должна целиком охватывать целевую предметную область и позволять полностью решать все поставленные бизнес-задачи. Основной задачей разрабатываемого проекта будет автоматизация и контроль процесса бронирования дат для проведения различных мероприятий. Исходя из этого, можно выделить следующие основные задачи, которые позволит решить разрабатываемая система:

- организовать возможность удобного хранения информации и её последующей обработки (добавление новых записей, редактирование и удаление существующих записей, поиск данных по ключевым полям);
- организовать контроль процесса бронирования мероприятий;
- наладить процесс закрепления мероприятий за определённой площадкой проведения;
- организовать контроль процесса закрепления сотрудников за проводимыми мероприятиями;
- организовать контроль рабочего времени сотрудников;
- автоматизировать процесс составления статистических графиков работы туристического центра за выбранный период;
- реализовать все необходимые бизнес и логические ограничения при работе с данными.

После анализа предметной области и поставленных задач был выявлен необходимый набор основных сущностей:

- Employee (сотрудники);
- WorkStatus (рабочие графики);
- Event (мероприятия);
- Place (места проведения мероприятий);
- Client (клиенты);
- Reservation (забронированные дни).

Кроме основных сущностей также был создан ряд вспомогательных подсущностей (таблиц) для хранения дополнительных данных. После составления полного списка необходимых таблиц была спроектирована концептуальная схема базы данных, увидеть которую можно на рисунке.

Реляционная модель представляет базу данных в виде множества взаимосвязанных отношений (таблиц), которые используются для хранения информации об объектах, представленных в базе данных.

В соответствии с выделенными выше сущностями, определён набор необходимых таблиц базы данных, представленный в таблице.

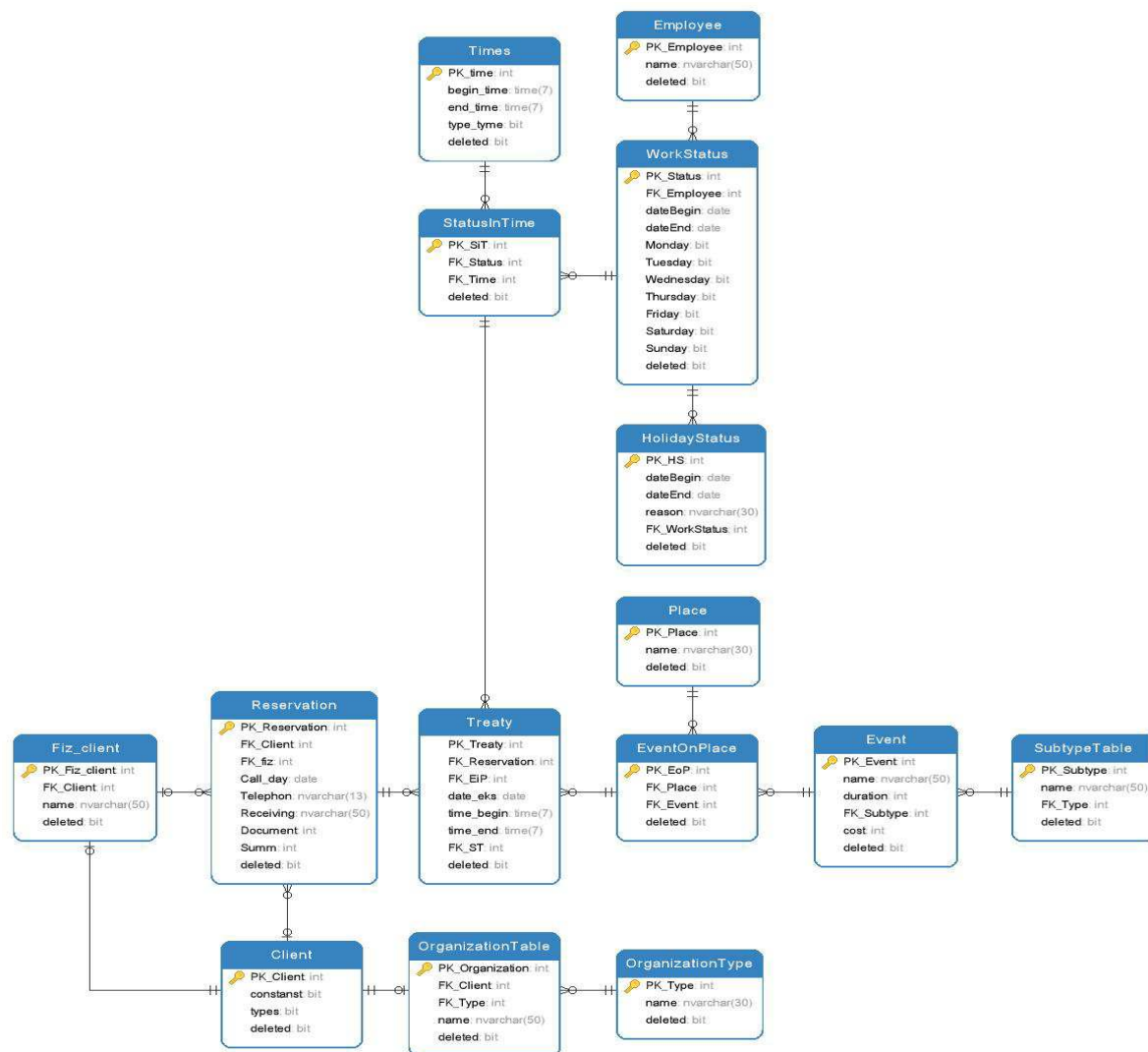


Рисунок. – Концептуальная схема базы данных

Таблица. – Перечень необходимых таблиц

Название таблицы	Описание предназначения
Employee	Хранение данных о всех сотрудниках туристического центра
WorkStatus	Хранение данных о рабочем графике каждого сотрудника
HolidayStatus	Хранение данных о внеплановых выходных и не выходах сотрудника на работу
Times	Хранение данных о рабочем времени каждого из сотрудников
StatusInTime	Вспомогательная таблица, является связующим звеном таблиц WorkStatus и Times
Event	Хранение данных о проводимых мероприятиях
SubtypeTable	Хранение данных о подтипах мероприятий
Types	Хранение данных о категориях мероприятий
Reservation	Хранение данных о бронировании даты мероприятия
Treaty	Хранение данных о проведении заказанного мероприятия
Place	Хранение данных о площадках, на которых могут проводиться мероприятия
EventOnPlace	Закрепление мероприятия за определённой площадкой.
Client	Хранение общей информации о клиентах, обращающихся в туристический центр
OrganizationTable	Хранение данных о клиентах центра, которые являются организациями
OrganizationType	Хранение данных о типах организаций, обращающихся в центр
Fiz_client	Хранение данных о клиентах центра, которые являются физическими лицами

Важное достоинство баз данных – возможность хранить сгруппированные данные в разных таблицах и устанавливать связи между этими таблицами посредством ключей и последующее объединение этих данных в единую базу. Это позволяет снизить избыточность данных, которые хранятся и упростить организацию запросов для их получения.

Первичный ключ является уникальным идентификатором записи, хранимой в базе данных, в связи с этим поле первичного ключа не может повторяться или быть пустым. Важное условие – каждая таблица может содержать только один первичный ключ, при этом первичный ключ может быть представлен как одним полем, так и объединённой группой полей таблицы.

Для организации связей между двумя и более таблицами, используются вторичные ключи. Они служат ограничителями целостности связей нескольких таблиц, т.к. подчиненная таблица не может ссылаться на несуществующие записи главной таблицы. Это позволяет строить целостные модели данных [1].

Так же очень важно выбрать систему управления базой данных (СУБД). Для разработки представленной базы данных удобно использовать СУБД Microsoft SQL 2014. Основными преимуществами данной СУБД, которые послужили причиной выбора её являются:

- поддержка режима Windows Server Core, благодаря которой уменьшается время простоя баз при загрузке патчей;
- усовершенствованная политика доступа. Теперь пользователям назначаются роли, по умолчанию выставляются схемы доступа для различных групп пользователей, что позволяет максимально повысить уровень безопасности хранимой информации;
- появились более широкие возможности для интеграции и синхронизации данных. Упростился процесс загрузки, извлечения и преобразования данных в/из Excel;
- внедрены in-memoгу технологии, которые увеличивают производительность любых вычислений в 15-30 раз;
- модернизировано колоночное хранилище памяти, которые впервые было внедрено в Microsoft SQL Server 2012. Теперь хранилище стало обновляемым, повысился уровень сжатия данных и увеличилась скорость обработки запросов [2].

Таким образом с учетом проведенного исследования для решения поставленных задач по созданию автоматизированной информационной системы Туристического центра учреждения образования «Полоцкий государственный университет» была спроектирована модель базы данных и для ее реализации выбрана СУБД Microsoft SQL 2014.

ЛИТЕРАТУРА

- 1 Семантическое моделирование данных, ER-диаграммы [Электронный ресурс] / CitForum. – Режим доступа: http://citforum.ru/database/osbd/glava_24.shtml/. – Дата доступа: 07.09.18.
- 2 Новые возможности MS SQL Server 2014 [Электронный ресурс] / Softpointplus. – Режим доступа: <http://www.softpointplus.ru/bazi-dannih-sql/novie-vozmognosty-ms-sql-server-2014.html/>. – Дата доступа: 07.09.18.

УДК 004.05

АРХИТЕКТУРА КЛИЕНТ-СЕРВЕРНЫХ ИГРОВЫХ ПРИЛОЖЕНИЙ

Д.В. СУЩЕВСКИЙ

(Представлено: канд. тех. наук И.Б. БУРАЧЁНОК)

Исследована архитектура клиент-серверных игровых приложений и проанализированы подходы, которые помогают избавиться от нечестных игроков, а также избежать проблемы с синхронизацией состояния игры на клиентских компьютерах с сервером.

Одной из самых сложных задач клиент-серверных игровых приложений является синхронизация всех игроков с состоянием сервера. Также острым вопросом всегда являлась тема «читеров», так называемых жульничающих игроков, ведь при разработке однопользовательской игры особо нет разницы, если игрок будет жульничать, ведь эти действия повлияют только на него. Но если это многопользовательская игра, нечестный игрок упрощает ее для себя, но ухудшает игровой процесс для других пользователей, таким образом это становится большой проблемой, с которой необходимо бороться, так как такое поведение отталкивает других игроков.

Решением проблем с «читерами» служит авторитарный сервер с наивным клиентом [1]. То есть, вся игровая логика происходит под контролем игрового сервера, а клиенты лишь отображают актуальное состояние, полученное от сервера и отправляют команды в виде нажатых клавиш. С таким подходом, игрок может подменять урон или здоровье, но сервер будет знать актуальное значение и это никак не скажется на игровом процессе других игроков, так как они будут видеть актуальные данные. Также нельзя доверять получаемым позициям игрока, так как он может прислать не верные или не корректные данные. Ситуации с неверными данными легко обработать. Отправленные данные могут содержать информацию, что игрок прошёл сквозь какую-то поверхность, хотя это невозможно или переместился на недостижимое расстояние, но если мы знаем где он находился и его новую позицию, то можно проверить корректность этих данных, а затем предпринять определенные действия в зависимости от их корректности.

Самый простой способ синхронизации, это чтобы клиенты отправляли серверу изменения с определенным интервалом. Обновления содержат пользовательские данные, которые также могут содержать информацию о том, что за прошедший период времени введенных данных не было. Но сразу можно заметить недостаток такого подхода. Например, если интервал составляет 50 мс, сервер получает значение, обрабатывает и возвращает назад результат, но во время ожидания ответа от сервера, у игрока просто застынет игровой процесс. Также общение может происходить с задержкой, из-за скорости сети, где получение или отправка может быть от 50 мс до 500 мс и более, это увеличит время простоя, что просто неприемлемо для современных игр. Это единственная проблема подхода с фиксированным интервалом. Например, есть сервер и два игрока А и В, у игрока А скорость интернета на порядок выше игрока В и при отправлении событий они приходят с различным интервалом, поэтому сервер продолжает расчёт только при получении обновлений от всех клиентов. Отсюда следует, что задержка игры равна задержке игрока с самым медленным интернетом, так как серверу приходится ожидать данных от клиента, чтобы обработать их и отправить обратно результат ожидающим игрокам.

Предсказание на стороне клиента является отличным улучшением подхода с фиксированным интервалом. Этот подход используется при условии, что игра достаточно детерминирована, т.е. результат зависит от предыдущего состояния и не является хаотичным, как например в так называемых «шутерах» или других динамических играх. При таком подходе клиент отправляет серверу свою команду, но локально продолжает эмитировать игровой процесс, так как в большинстве случаев предсказания, которые просчитаны сервером будут совпадать с действительностью. Проблема в том, что игрок видит мир в настоящий момент времени, но из-за задержки обновления, сервер отправляет состояние мира в прошлом, поэтому происходят скачки в предыдущее состояние, что хорошо отражается на позиции персонажа. Но при таком подходе можно не волноваться о нечестных игроках, так как все некорректные данные отображаются только у игрока, который жульничает, но не влияет на общее состояние игры, которое наблюдают другие клиенты [2].

Следующим этапом в совершенствовании архитектуры – согласование с сервером. Решением проблемы с предсказанием, является хранение на клиенте копий всех команд, которые отправлялись до этого на сервер. Теперь при отправке сообщений, у клиента хранится список команд, которые совершались им, когда приходит ответ по обработанной команде, то игрок восстанавливает игровой процесс до этого состояния, удаляет команду из своей очереди и выполняет команды, которые следовали за ней, чтобы вернуться в корректное состояние. Когда приходит последняя команда, клиент удаляет все копии команд до той, которая пришла включительно и применяет последнее состояние, в результате чего, ничего не изменится. Если необработанных команд больше нет, то на этом всё завершится.

Ранее описанный подход хорошо работал для небольшого количества пользователей. Но что, если клиентов много, и они часто отправляют команды? Обновление игры для каждой команды и оповещение всех игроков об изменении состояния основательно нагрузит сервер и сеть. Чтобы исправить эту ситуацию, можно обновлять мир с низкой частотой, например, 20 раз в секунду. Во время обновления, все накопленные команды применяются и меняется состояние игры клиентов на актуальное. Частота обновления мира зависит от динамики игры, если это шахматы, то имеет смысл поставить обновление меньше 20 раз в секунду, если игры является «шутером», то может потребоваться 100 или больше. Клиент будет также, как и в подходе согласования видеть гладкий игровой процесс, вне зависимости от скорости обновления, но клиент редко получает информацию о состоянии других игроков, что приводит к рывкам в геймплее [3]. Есть два самых распространенных способа для борьбы с этой ситуацией:

- экстраполяция;
- интерполяция.

Экстраполяция хорошо подходит для игр, где легко предсказать дальнейшее состояние игры. Например, в игре «Марио» мы знаем с какой скоростью движется персонаж и что находится перед ним на определенном расстоянии. Следующее положение персонажа зависит от предыдущего, и мы почти наверняка можем предсказать его новое положение в игре. Клиент получает скорость и направление каждого игрока и в течении какого-то времени он не будет получать новых данных. В зависимости от полученных данных, игрок должен эмитировать движения других игроков. Самым простым способом является предположить, что направление и ускорение будут константными между обновлениями и локально воспроизводить физику игры с учетом этих данных, позже, когда придут новые данные, необходимо скорректировать состояние других игроков. Этот метод подойдет для объектов, которые имеют большую инерцию.

Интерполяция идеально подходит для тех ситуаций, когда скорость и направление меняются быстро. Где необходимо быстро изменять позиции и дальнейшее поведение невозможно предугадать, используя экстраполяцию. Идея состоит в том, что есть позиции игроков за определенным промежутком времени и можно показывать, что происходило между обновлениями, то есть не в настоящий момент, а в прошлом. Игроки видят слегка разное состояние игрового мира, себя игроки видят в настоящий момент времени, но состояние других игроков, является состоянием из прошлого [4].

С интерполяцией могут возникнуть проблемы, если необходима высокая пространственная и временная точность, например, при выстреле. Игрок целится с задержкой и поэтому при выстреле у себя, игрок мог попасть в противника, но в действительности он не попал в цель, так как, то состояние противника, что он видел, было его состояние в прошлом. Здесь может помочь компенсация задержки. Клиент при выстреле, отправляет событие на сервер, с серверным временем состояния, которое отображает момент выстрела и направление оружия. Так как сервер рассылает состояние клиентам, то он может восстановить любой момент из прошлого, в том числе во время выстрела. Таким образом сервер воспроизводит ситуацию выстрела, проверяет, что произошло и с учетом полученного результата, он обновляет состояние мира и рассылает его клиентам. При таком подходе, подобные ситуации будут просчитаны верно. Правда, может возникнуть ситуация, когда игрок прошёл зону поражения, но через доли секунды погиб. Это альтернатива, на которую готовы идти разработчики, потому что лучше уничтожить игрока в прошлом, чем невозможность этого сделать вообще.

Таким образом, в данной статье представлены основные принципы проектирования архитектуры клиент-серверных игровых приложений, а также описанные способы борьбы с нечестными игроками и способы синхронизации, такие как интерполяция и интервальное обновление игрового мира. Детальное изучение представленных в данной статье вопросов позволило исключить ошибки при разработке собственного проекта.

ЛИТЕРАТУРА

1. Unity. Высокоуровневые понятия сетевого взаимодействия [Электронный ресурс] / Unity. – Режим доступа: <https://docs.unity3d.com/ru/current/Manual/net-HighLevelOverview.html/>. – Дата доступа: 17.09.2018.
2. Сетевое программирование в Source [Электронный ресурс] / Developer Community. – Режим доступа: https://developer.valvesoftware.com/wiki/Source_Multiplayer_Networking:ru/. – Дата доступа: 17.09.2018.
3. Мультиплеер в быстрых играх. Хабр [Электронный ресурс] / Хабрахабр. – Режим доступа: <https://habr.com/post/302834/>. – Дата доступа: 17.09.2018.
4. Player.IO [Электронный ресурс] / Player.IO. – Режим доступа: <http://www.ant-karlov.ru/PlayerIO-interpolyatsiya-ili-udivitelnyi-mir-obmana.html/>. – Дата доступа: 17.09.2018.

УДК 004.05

АРХИТЕКТУРНЫЙ ШАБЛОН ПРОЕКТИРОВАНИЯ
ENTITY-COMPONENT-SYSTEM

Д.В. СУЩЕВСКИЙ

(Представлено: канд. тех. наук, доц. И.Б. БУРАЧЁНОК)

Исследован архитектурный шаблон проектирования Entity-Component-System. Описаны его основные возможности, улучшающие архитектуру игрового приложения и решающие проблемы, связанные с производительностью и гибкостью программного обеспечения.

Одной из главных проблем настоящего времени является постоянно растущая сложность программного обеспечения (ПО). Это приводит к увеличению вероятности допустить ошибку. Для решения этих проблем создаются новые подходы к написанию ПО, шаблоны проектирования и правила, облегчающие поддержку системы. Игры, как и другие приложения должны работать быстро, а их дизайн должен позволять легко расширять функционал.

В представленной статье подробнее рассмотрим один из интересных подходов Entity Component System (ECS) в основе которого лежит композиция. ECS – это архитектурный шаблон, который в основном используется при разработке игр. ECS придерживается композиции, а не наследования, что позволяет повысить гибкость в определении объектов. Объекты состоят из одного или нескольких компонентов, которые добавляют дополнительное поведение или функциональность. Поэтому поведение объекта легко изменить во время выполнения, путём добавления или удаления компонентов [1]. Многие крупные компании, такие как, Unity, Epic, Cytetek используют этот шаблон. Данный подход является ориентированным на данные и позволяет уменьшить количество кэш промахов (cache miss) [2].

Компонентный подход получает всё большее признание в разработке игр. Главная идея подхода состоит в том, чтобы разделять функционал на отдельные компоненты, которые в основном не зависят друг от друга. Стандартные глубокие иерархии наследования не используются. Вместо традиционных иерархий объектов создаются совокупности, коллекции независимых компонентов. Каждый объект обладает только теми функциями, которые ему необходимы. Любой новый функционал легко реализуется путём добавления новых компонентов.

Контроллер кэш памяти управляет содержанием кэша, получая данные из оперативной памяти, передаёт ее процессору и возвращает результаты вычислений в оперативную память. При обращении ядра процессора к контроллеру за данными, контроллер проверяет есть ли эти данные в кэш-памяти. Если данные содержатся в кэш-памяти, то они отдаются процессору. Если данные не найдены, то ядру необходимо ожидать момента, когда эти данные будут загружены из оперативной памяти, что является весьма ресурсоёмким процессом. Ситуация, когда в кэш-памяти не оказывается необходимых данных, называется кэш-промахом. Контроллер пытается свести кэш-промахи к минимуму. Размер кэша процессора по сравнению с оперативной памятью очень мал. Обычно в нём хранится небольшая часть данных, взятых из оперативной памяти. Как правило у всех современных процессоров есть кэши различных уровней (рис.), чем выше уровень, тем у него меньше памяти и эффективней скорость получения данных.

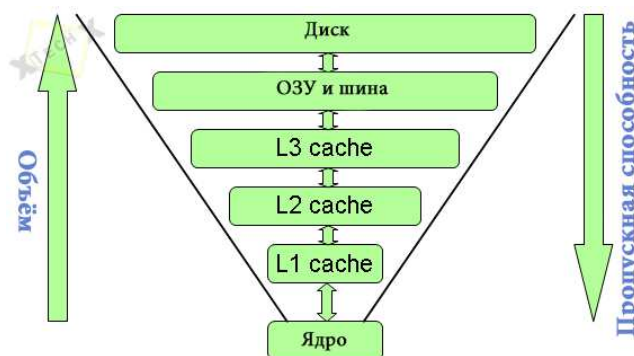


Рисунок. – Уровни памяти компьютера

Если данные не найдены в самом верхнем уровне L1, то идёт обращение к уровню, что находится ниже L2 и происходит поиск нужных данных, если этих данных нет и там, то происходит обращение к последнему уровню L3, если данных не найдено и в нём, то контроллер осуществляет обращение к оперативной памяти. Такие сложные архитектурные решения для работы с памятью были выбраны по причине того, что обычно самое узкое место в программах, это работа с памятью. Поэтому очень важно, чтобы работа с памятью была максимально эффективной. Для этого программа должны быть максимально кэш дружелюбной (cache friendly), чтобы достичь максимальной производительности.

Шаблон можно разделить на три составляющие.

1. Сущности (Entity) – объекты контейнеры, не обладающими свойствами, которые выступают в роли хранилища для «компонентов». Сущность представляет собой неявное агрегирование компонентов. Как правило, хранилище – простой контейнер данных. Обычно совместно с сущностями используется подход, называемый пулинг объектов (object pool). Это позволяет сократить расходы на выделение и перераспределение памяти, что является ресурсоёмким процессом. Идея данного подхода состоит в том, чтобы выделить большой размер памяти, а при необходимости брать её из резерва и использовать для создания объектов, а после того как объект не нужен, необходимо вернуть его назад.

2. Компоненты (Component) – это блоки данных, определяющие возможные свойства любых игровых объектов или событий. Все эти данные сгруппированы и обрабатываются определенной логикой. Они являются объектами с простой структурой данных (plain old object, POD). Каждый тип компонента можно прикрепить к сущности, чтобы определить её характеристики. Компоненты не содержат логики, иногда они являются пустыми маркерами для обработки системы. Компонент можно сравнить со структурой в языке программирования C, он не имеет методов и способен хранить данные. Каждый компонент описывает определённый аспект объекта и его параметры. Сами по себе компоненты практически не имеют смысла, но в сочетании с сущностями и системами, они становятся чрезвычайно мощным средством решения задач в разработке. Например, сущности можно присвоить свойство «здоровье», которое является обычным целочисленным или дробным значением в памяти. Компонент не должен иметь большой размер, т.к. это будет вызывать проблемы в скорости обработки.

3. Система (System) – отвечает за обработку компонентов, в них и происходит работа всей логики. У системы имеется список компонентов с определенными типами, где она просматривает их и обрабатывает. Как правило, у системы никогда нет одного элемента, она имеет дело с коллекцией, и обрабатывает элементы поочередно, но это не значит, что коллекция не может быть пуста или иметь всего один элемент. Такой подход избавляет от проблем, если в дальнейшем необходимо будет добавить, например, нового персонажа. Например, система может работать с позицией, скоростью. Каждая система будет обновляться в логическом порядке.

Межсистемная связь может осуществляться множеством способов. Например, способом отправки данных между системами – хранить определенные данные в компонентах. В игре положение объекта может постоянно обновляться. Однако данный подход не всегда хорош, когда события происходят редко и необходимо каким-то образом хранить текущее состояние. Самое распространенное – флаги состояния, но это имеет большой недостаток. Системы на каждой итерации будут читать флаги и проверять доступность события, что может быть неэффективным, одной из причин может стать предсказатель переходов (branch prediction).

Предсказатель переходов – это механизм, входящий в состав микропроцессоров, с конвейерной архитектурой, где осуществляется предсказание, будет ли выполнен условный переход в исполняемой программе. Причиной является то, что современные процессоры выполняют многие операции параллельно, что позволяет сократить время простоя конвейера за счёт предварительной загрузки и исполнения инструкций, которые должны выполняться после выполнения условного перехода. Прогнозирование ветвлений играет критическую роль [3]. Это может стать проблемой в ECS, так как компоненты обрабатываются, как непрерывный конвейер с маленькими задержками. Например, в Unity метод FixedUpdate выполняется 50 раз в секунду. Также есть метод Update, скорость которого может варьироваться от мощности компьютера, т.е. может быть, как 100, так и 10 вызовов в секунду. В таком случае, проблема может быть критичной. Поэтому, как правило в системе присутствует несколько подсистем, которые обновляют сущности. Для решения этой проблемы может быть использован шаблон проектирования – наблюдатель. Все системы, зависящие от события, подписываются на него. Таким образом, действие из события будет выполняться только однажды в тот момент, когда это произойдёт и не требуется постоянный опрос с проверками.

Подход Entity-Component-System решает проблему с множественными вызовами методов обновления, которые окружают все игровые приложения. Хороший пример – игровой движок Unity. В нём происходят нативные вызовы языка программирования C++. Существует тест производительности одного

из разработчиков Unity, в котором он показал то, как нативные вызовы влияют на производительность. В этих тестах разработчик Unity осуществил 10000 вызовов метода Update и вызов метода обновления, не являющийся нативным. Ссылки на компоненты были добавлены в коллекцию, где вызывались поочередно [4]. В таблице представлены результаты замеров производительности.

Таблица. – Замеры производительности нативного и не нативного методов в Unity

Mono			IL2CPP		
Методы	iPhone 6	iPhone 4s	Методы	iPhone 6	iPhone 4s
Update	2.8ms		Update	5.4ms	10.91ms
Manager	0.52ms	2.1ms	Manager (Dynamic array)	1ms	2.52ms
			Manager (Array)	0.22ms	1.15ms

В результате исследования можно сделать вывод, что использование подхода ECS не только позволяет уменьшить связанность кода, но и существенно увеличить производительность программы. Предложенный подход позволяет упростить дальнейшую поддержку и обеспечить простой способ связи игровых компонентов между собой.

Из достоинств подхода ECS можно отметить: гибкость, масштабируемость, эффективное использование памяти, простой доступ к объектам и лёгкость тестирования.

В дальнейшем планируется внедрить данный подход в разработку казуального мультиплеерного игрового приложения. Так как в Unity изначально используется не чистый не полностью реализованный в ECS подход, из-за чего присутствуют определенные проблемы с производительностью, при большом количестве объектов.

ЛИТЕРАТУРА

1. Wikipedia. Entity-Entity-Component-System [Электронный ресурс] / Wikipedia. – Режим доступа: <https://en.wikipedia.org/wiki/Entity-component-system/>. – Дата доступа: 10.09.2018.
2. GameProgrammingPatterns [Электронный ресурс]. – Режим доступа: <http://gameprogrammingpatterns.com/data-locality.html/>. – Дата доступа: 10.09.2018.
3. Branch prediction [Электронный ресурс] / Danluu. – Режим доступа: <https://danluu.com/branch-prediction/>. – Дата доступа: 10.09.2018
4. 10 000 вызовов Update [Электронный ресурс] / Unity. – Режим доступа: <https://blogs.unity3d.com/ru/2015/12/23/1k-update-calls/>. – Дата доступа: 10.09.2018.

УДК 004.432.2

ПРОБЛЕМА ИСПОЛЬЗОВАНИЯ ЧИСТЫХ ФУНКЦИЙ В ЯЗЫКЕ ПРОГРАММИРОВАНИЯ JAVASCRIPT

К.И. ЛАПКОВСКИЙ

(Представлено: Д.В. ПЯТКИН)

Рассмотрено использование чистых функций в языке программирования, который не разрабатывался для написания программ в функциональном стиле. Представлена применимость языка JavaScript в функциональном программировании на примере главной концепции функционального программирования – использовании чистых функций. После анализа примеров различных функций в языке JavaScript, сформулированы правила, которым необходимо следовать при написании программы в функциональном стиле.

Функциональное программирование с каждым годом приобретает все большую популярность. Одна из концепций функционального программирования – использование чистых функций. Написание программы, опираясь эту на парадигму и используя специальные языки программирования (Scala, Haskell), позволяет создать приложение, которое одинаково легко справляется с любым объемом данных, работая при этом очень быстро. Однако насколько возможно использование не специализированных языков программирования для такой цели? Рассмотрим применение чистых функций в популярном языке программирования JavaScript.

JavaScript – это язык программирования с слабой динамической типизация, реализующий прототипную модель наследования [1].

Прототипное наследование. Прототип – хранилище методов [2]. Если в объекте вызывается метод, то этот метод сначала ищется в самом объекте, потом в прототипе этого объекта. Прототипы могут формировать цепочки, создавая при этом сколь угодно сложную иерархию. Технически в прототипе можно хранить не только методы, но и свойства, но этой возможностью пользуются редко. Использование прототипов помогает в переиспользовании программный код [3]. Классическое наследование решает ту же задачу.

Чистые функции. Чистыми называют функции, которые не имеют побочных эффектов ввода-вывода и памяти (они зависят только от своих параметров и возвращают только свой результат). Чистые функции обладают несколькими полезными свойствами, многие из которых можно использовать для оптимизации кода:

- если результат чистой функции не используется, её вызов может быть удален без вреда для других выражений;
- результат вызова чистой функции может быть мемоизирован, то есть сохранен в таблице значений вместе с аргументами вызова. Если в дальнейшем функция вызывается с этими же аргументами, её результат может быть взят прямо из таблицы, не вычисляясь (иногда это называется принципом прозрачности ссылок). Мемоизация, ценой небольшого расхода памяти, позволяет существенно увеличить производительность и уменьшить порядок роста некоторых рекурсивных алгоритмов;
- если нет никакой зависимости по данным между двумя чистыми функциями, то порядок их вычисления можно поменять или распараллелить (говоря иначе вычисление чистых функций удовлетворяет принципам thread-safe);
- если весь язык не допускает побочных эффектов, то можно использовать любую политику вычисления. Это предоставляет свободу компилятору комбинировать и реорганизовывать вычисление выражений в программе (например, исключить древовидные структуры) [1].

Примеры функций в JavaScript

В листинге 1 представлены четыре функции. Проанализируем являются ли они чистыми.

```
// A: Простое умножение
function doubleA(n) {
    return n * 2
}
// B: С переменной
var two = 2
function doubleB(n) {
    return n * two
}
```

```
// C: С вспомогательной функцией
function getTwo() {
  return 2
}
function doubleC(n) {
  return n * getTwo()
}
// D: Преобразование массива
function doubleD(arr) {
  return arr.map(n => n * 2)
}
```

Листинг 1. – Функции JavaScript

Функция В не является чистой, так как читает переменную из внешней области видимости, тем самым становится зависимой от этой переменной. Если в функцию В два раза передать одинаковые параметры, но перед вторым вызовом функции изменить переменную two, возвращаемые значения после первого и второго вызова функции doubleB будут отличаться.

Функция doubleC так же нельзя назвать чистой. Функция doubleC зависит от функции getTwo, которая может быть переопределена. Если getTwo присвоить значение отличное от функции, то doubleC выбросит исключение, так как нельзя вызвать не функцию.

Рассмотрим функцию doubleD. Внутри функции вызывается метод map, который хранится в прототипе. Так как все экземпляры классов ссылаются на один и тот же прототип, то при изменении прототипа поведение всех экземпляров класса тоже изменится. Помимо этого функцию можно “перекрыть”, то есть определить функцию с таким же названием, как и в прототипе. Функция doubleD не чистая.

Чтобы понять является ли doubleA чистой функцией необходимо знать механизм, который использует JavaScript для преобразования объектов в примитивы. Когда интерпритатору необходимо преобразовать объект в примитив, для объекта вызывается скрытый метод toPrimitive [4]. ToPrimitive вызывает метод valueOf. Если valueOf возвращает примитив, то этот примитив и используется как результат преобразования. Если valueOf вернул объект, то вызывается метод toString, возвращаемое значение метода toString и будет результатом преобразования. Методы toString, toPrimitive и valueOf можно переопределить или перекрыть как и обычные методы прототипа.

Таким образом, возможность изменять прототип и перекрывать функции мешает использовать чистые функции в JavaScript.

Однако можно обезопасить свою программу от перекрытия функций, если явно вызывать методы прототипа, и не доверять поиск метода интерпритатору JavaScript. То есть для вызова метода map рекомендуется использовать синтаксис, представленный в листинге 2.

```
Array.prototype.map.call(array, ()=>{})
```

Листинг 2. – Вызов метода прототипа

ЛИТЕРАТУРА

1. Функциональное программирование [Электронный ресурс] // Wikipedia. – Режим доступа: https://ru.wikipedia.org/wiki/Функциональное_программирование#Чистые_функции. – Дата доступа: 24.09.2018.
2. Прототип объекта [Электронный ресурс] // Современный учебник JavaScript. – Режим доступа: <https://learn.javascript.ru/prototype>. – Дата доступа: 24.09.2018.
3. Object.prototype [Электронный ресурс] // MDN web docs. – Режим доступа: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object/prototype. – Дата доступа: 24.09.2018.
4. Coercion [Электронный ресурс] // You don't know JS. – Режим доступа: <https://github.com/getify/You-Dont-Know-JS/blob/master/types%20%26%20grammar/ch4.md>. – Дата доступа: 25.09.2018.

УДК 004.432.2

ПЕРЕОПРЕДЕЛЕНИЕ КОНТЕКСТА ВЫЗОВА ПО УМОЛЧАНИЮ В ЯЗЫКЕ ПРОГРАММИРОВАНИЯ JAVASCRIPT

К.И. ЛАПКОВСКИЙ
(Представлено: Д.В. ПЯТКИН)

Рассматривается переопределение контекста вызова функций в JavaScript. Представлены все возможные случаи задания контекста функций. Анализируется, насколько эффективны уже существующие механизмы. В результате сформулирован альтернативный способ задания контекста функций.

JavaScript – мультипарадигменный язык программирования. Такая универсальность языка влечет за собой некоторые проблемы. Например, если рассматривать JavaScript как объектно-ориентированный язык программирования, можно заметить, что на уровне языка не реализованы такие механизмы как абстрактный класс (хотя программист сам без труда может запрограммировать класс, экземпляр которого нельзя создать), а прототипное наследование усложняет написание программы в функциональном стиле.

Неправильное задание контекста вызова скорее приведет к ошибкам во время выполнения программы, чем к невозможности реализовать какие-либо принципы разных парадигм программирования. Однако если программа не была должным образом протестирована, исключительные ситуации могут появляться редко, воспроизводиться сложно, при этом сильно влиять на работу программы.

Контекст вызова – это значение, в контексте которого была вызвана функция. Любая функция в JavaScript может обращаться к ключевому слову `this` внутри себя [1]. Ключевое слово `this` ссылается на контекст вызова функции. Функция в момент вызова связывается с объектом, и `this` внутри этой функции получает соответствующее значение. Рассмотрим четыре возможных случая определения контекста в порядке их приоритета:

Связывание с помощью оператора `new`. Если функция была вызвана с оператором `new`, то `this` получит ссылка на только что созданный, пустой объект. Пример представлен в листинге 1.

```
var bar = new foo()
```

Листинг 1. – Определение контекста для функции `foo` с помощью оператора `new`

Явное связывание. Если функция вызвана с помощью методов `call` или `apply`, или к функции привязан контекст с помощью функции `bind`, тогда контекстом вызова будет первый аргумент, переданный в эти функции. Пример представлен в листинге 2.

```
var bar = foo.call(obj2)
```

Листинг 2. – Определение контекста функции методом `call`

Неявное связывание. Если функция вызвана в контексте какого-то объекта, так же известного как объекта-контейнера. То в `this` попадет ссылка на этот объект. Пример в листинге 3.

```
var bar = obj1.foo()
```

Листинг 3. – Неявное задание контекста для функции

Связывание по умолчанию. Если ни один из вышеописанных случаев не применим к вызову функции, то в `this` попадет значение по умолчанию. В строгом режиме - это `undefined`, в не строгом – это глобальный объект [2]. Пример представлен в листинге 4.

```
var bar = foo()
```

Листинг 4. – Задание контекста функции по умолчанию

Проблема жесткого связывания и связывания по умолчанию

Контекст функции теряется, если использовать функцию как переменную [3]. Обычно для привязки контекста используется жесткую привязка, или стрелочную функцию, которая получает контекст из замыкания.

Жесткая привязка контекста вызова к функции (через метод bind), уменьшает гибкость функции, не позволяя переопределить контекст с помощью неявного или даже явного задания контекста вызова.

Привязка функции к контексту по умолчанию редко является намеренной, так как в случае с привязкой к undefined, любая попытка считать свойство у this, или вызов метода, приведет к ошибке во время выполнения программы. Если же использовать не строгий режим, то в контекст вызова попадет глобальный объект, и изменение любого свойства или метода объекта приведет к изменению глобального объекта. Сценарий с глобальным объектом в качестве контекста вызова часто приводит к сложно уловимым ошибкам.

Реализация переопределения контекста по умолчанию

В качестве решения проблемы жесткого связывания и связывания по умолчанию можно использовать переопределение контекста вызова. Реализация функции представлена в листинге 5.

```
if (!Function.prototype.softBind) {
  Function.prototype.softBind = function(obj) {
    var fn = this,
        curried = [].slice.call( arguments, 1 ),
        bound = function bound() {
          return fn.apply(
            (!this ||
             (typeof window !== "undefined" &&
              this === window) ||
             (typeof global !== "undefined" &&
              this === global)
            ) ? obj : this,
            curried.concat.apply( curried, arguments )
          );
        };
    bound.prototype = Object.create( fn.prototype );
    return bound;
  };
}
```

Листинг 5. – Реализация переопределения контекста по умолчанию

Таким образом, рассмотрены различные способы привязки контекста, и проблемы их использования. Разработана функция, задающая контекст по умолчанию, позволяющая решить проблему потери контекста, при этом сохраняя возможность его переопределения далее в программе.

ЛИТЕРАТУРА

1. JavaScript.info [Электронный ресурс] // Object methods, this. – Режим доступа: <https://javascript.info/object-methods>. – Дата доступа: 24.09.2018.
2. You don't know JS this & Objects Prototypes [Электронный ресурс]. – Режим доступа: <https://github.com/getify/You-Dont-Know-JS/blob/master/this%20%26%20object%20prototypes/ch2.md>. – Дата доступа: 24.09.2018.
3. Dmitri Pavlutin Blog [Электронный ресурс] // Gentle explanation of 'this' keyword in JavaScript. – Режим доступа: <https://dmitripavlutin.com/gentle-explanation-of-this-in-javascript/>. – Дата доступа: 24.09.2018.
4. MDN web docs this [Электронный ресурс]. – Режим доступа: <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/this>. – Дата доступа: 24.09.2018.

УДК 004.4'22

**ЭФФЕКТИВНЫЕ МЕТОДЫ SEO-ПРОДВИЖЕНИЯ САЙТОВ,
ПОСТРОЕННЫХ НА БАЗЕ CMS WORDPRESS****Н.И. ЛАПКОВСКИЙ***(Представлено: Д.В. ПЯТКИН)*

Представлены практические способы по работе с сайтами, построенными на базе CMS Wordpress, которые помогут достичь максимальных результатов в их SEO-продвижении. Рассмотрены, как встроенный в CMS Wordpress функционал, так и функционал, представленный сторонними плагинами.

Обычно поисковые системы — самый крупный источник трафика для большинства сайтов. Google, Яндекс, Bing и другие поисковые роботы используют продвинутые алгоритмы, чтобы проанализировать и упорядочить страницы в результатах поиска. Но даже такие сложные алгоритмы не идеальны — им нужна помощь разработчиков и владельцев сайтов, чтобы понять, какой контент расположен на их сайте. Если контент сайта плохо оптимизирован для SEO, то поисковые системы не смогут верно и полностью его проанализировать, и тогда у них будет меньше доверия к такому сайту. Следовательно, плохо SEO-оптимизированный сайт будет располагаться в конце списка результатов поиска, и, как итог, посещаемость такого сайта будет низкой.

Поисковая система — это компьютерная система, предназначенная для поиска информации [1].

Поисковая оптимизация (англ. search engine optimization, SEO) — комплекс мер по внутренней и внешней оптимизации для поднятия позиций сайта в результатах выдачи поисковых систем по определённым запросам пользователей, с целью увеличения сетевого трафика (для информационных ресурсов) и потенциальных клиентов (для коммерческих ресурсов) и последующей монетизации (получения дохода) этого трафика [2].

Система управления содержимым (контентом) (англ. Content management system, CMS) — информационная система или компьютерная программа, используемая для обеспечения и организации совместного процесса создания, редактирования и управления содержимым, иначе — контентом (от английского content) [3].

Проверка настроек отображения сайта в сети. Wordpress имеет встроенную опцию сокрытия вашего сайта от поисковых систем. Эта опция нужна, чтобы дать разработчику время для работы над сайтом перед тем, как он будет готов к публикации в сети Интернет. Иногда эта опция может быть выбрана случайно, что делает сайт невидимым для поисковых систем. Если сайт не показывается в результатах поиска, то эта опция — первое, что стоит проверить. Найти её можно перейдя в Настройки → Чтение [4].

Использование SEO-дружественных URL. Единый указатель ресурса (англ. Uniform Resource Locator, URL) — единообразный локатор (определитель местонахождения) ресурса.

SEO-дружественные URL содержат слова, которые чётко описывают контент страницы, из-за чего такие URL проще воспринимать как людям, так и поисковым системам. Примеры SEO-дружественных URL представлены ниже [5]:

- <https://www.example.com/how-to-install-windows>;
- <https://www.example.com/epic-burning-man-sunrise-dance-party>.

Такие URL более читабельны, чем обычные, поэтому пользователи могут понять, что они увидят на странице, просто посмотрев на текст URL. Обычные (SEO-недружественные) URL выглядят следующим образом:

- <https://www.example.com/?p=123>;
- <https://www.example.com/posts/123>.

Добавление карты сайта. Карта сайта — XML-файл с информацией для поисковых систем о страницах веб-сайта, которые подлежат индексации.

Добавление карты сайта не продвигает сайт в результатах выдачи поисковых систем, однако это помогает поисковым системам быстро находить страницы сайта и начать их индексацию.

Например, при использовании плагина Yoast (анализ которого описан ниже в статье) SEO карта сайта создаётся автоматически. Найти её можно перейдя по следующему URL ([example.com](https://www.example.com) нужно заменить доменным именем вашего сайта) [6]:

https://www.example.com/sitemap_index.html.

Оптимизация постов блога (сайта) для SEO. Часто начинающие Wordpress-разработчики допускают ошибку, полагая, что установки и активации SEO плагина достаточно для качественной оптимизации сайта. Однако это не совсем так. SEO-оптимизация — постоянный процесс, который должен выполняться на протяжении всего времени работы с сайтом, если требуется достичь хороших результатов.

Wordpress-плагин Yoast SEO позволяет добавить заголовок, описание и ключевое слово к каждому посту и странице. Так же плагин имеет функцию предпросмотра ссылки поисковой системы, которая позволяет заранее просмотреть, что увидит посетитель при поиске вашего сайта [7].

Правильное использование категорий и тегов. Wordpress позволяет сортировать посты по категориям и по тегам. Это помогает владельцу и посетителям быстрее находить интересующую их информацию. Также категории и теги помогают поисковым системам понимать структуру сайта и его контент.

Категории используются для обширной группировки постов. Если бы сайт был книгой, то его категории были бы содержанием этой книги.

Теги — это более специфичные ключевые слова, описывающие контент отдельного поста. Например, пост категории «Еда» мог бы иметь теги: «завтрак», «салат», «блины» и т.д.

Оптимизация комментариев. Комментарии могут быть значимым показателем взаимодействия с пользователем на сайте. Заинтересованные пользователи означают больше ссылок на сайт, больше трафика и улучшение SEO. Однако для хорошей SEO-оптимизации сайта нужно быть уверенным, что комментарии на нём настоящие, а не спам. Спаммеры оставляют комментарии с недействующими или ненужными ссылками, что отрицательно сказывается на доверии поисковых систем к сайту, на котором такие комментарии расположены. Для борьбы со спамом в комментариях лучшим решением будет использование плагина Akismet, одного из двух плагинов, изначально предустановленных в CMS Wordpress [8].

Спам (англ. spam) — массовая рассылка корреспонденции рекламного характера лицам, не выражавшим желания её получать [9].

Правильное использование внешних ссылок. Как сказано выше, ссылки помогают поисковым системам решить, какие страницы сайта наиболее важны. При переходе на сайт по ссылке используется некоторый SEO-ресурс этого сайта, называемый ссылочный вес. Для правильной работы SEO на сайте ссылочного веса «приходящего» на ваш сайт с других сайтов должно быть больше, чем «исходящего» на другие сайты.

Добавление атрибута «nofollow» к внешним ссылкам (ссылкам на другие сайты, которыми вы не обладаете) даёт инструкцию поисковым системам не индексировать эти ссылки. Это помогает сохранить ссылочный вес сайта.

Оптимизация изображений. Изображения занимают больше компьютерной памяти чем текст, а также требуют более длительного времени при загрузке на сайте. Скорость загрузки изображений влияет на показатели SEO-оптимизации сайта. Чем меньше времени для загрузки изображений требуется, тем лучше, поэтому следует разумно выбирать размер и качество изображений для сайта.

Другим способом оптимизации изображений для поисковых систем является использование title- и alt-тегов. Эти теги помогают поисковику понять о чём данное изображение.

Таким образом, в ходе данного исследования были рассмотрены наиболее эффективные способы SEO-продвижения сайта, созданного на базе CMS Wordpress. Косвенно был затронут вопрос оптимизации скорости загрузки сайта, также влияющий на SEO. Приведённые в статье методы могут быть использованы и на других сайтах, а также в других CMS-системах с аналогичным функционалом.

ЛИТЕРАТУРА

1. Поисковая система [Электронный ресурс] // Материал из Википедии – свободной энциклопедии. – Режим доступа: https://ru.wikipedia.org/wiki/Поисковая_система. – Дата доступа: 23.09.2018.
2. Поисковая оптимизация [Электронный ресурс] // Материал из Википедии – свободной энциклопедии. – Режим доступа: https://ru.wikipedia.org/wiki/Поисковая_оптимизация. – Дата доступа: 23.09.2018.
3. Система управления содержимым [Электронный ресурс] // Материал из Википедии – свободной энциклопедии. – Режим доступа: https://ru.wikipedia.org/wiki/Система_управления_содержимым. – Дата доступа: 23.09.2018.
4. Wordpress – Reading Setting [Электронный ресурс]. – Режим доступа: https://www.tutorialspoint.com/wordpress/wordpress_reading_setting.htm – Дата доступа: 23.09.2018.

5. How to Make Your Wordpress URLs SEO Friendly [Электронный ресурс]. – Режим доступа: <https://www.searchenginejournal.com/seo-friendly-wordpress-urls/203926/amp/>. – Дата доступа: 23.09.2018.
6. XML Sitemaps in the Yoast SEO plugin [Электронный ресурс]. – Режим доступа: <https://kb.yoast.com/kb/enable-xml-sitemaps-in-the-wordpress-seo-plugin/amp/>. – Дата доступа: 23.09.2018.
7. How To Use Yoast SEO On WordPress: Complete Tutorial [Электронный ресурс]. – Режим доступа: <https://kinsta.com/blog/yoast-seo/#settings-saved>. – Дата доступа: 25.09.2018.
8. 13 Simple Ways to Boost SEO on Your Wordpress Website [Электронный ресурс]. – Режим доступа: <https://www.dreamhost.com/blog/boost-seo-wordpress-website/>. – Дата доступа: 25.09.2018.
9. Спам [Электронный ресурс] // Материал из Википедии – свободной энциклопедии. – Режим доступа: <https://ru.wikipedia.org/wiki/Спам>. – Дата доступа: 25.09.2018.
10. Working with Images [Электронный ресурс]. – Режим доступа: https://codex.wordpress.org/Using_Images. – Дата доступа: 25.09.2018.

УДК 004.421.5

**ГЕНЕРАЦИЯ СЛУЧАЙНЫХ (ПСЕВДОСЛУЧАЙНЫХ) ЧИСЕЛ
В ЯЗЫКЕ ПРОГРАММИРОВАНИЯ PHP****Н.И. ЛАПКОВСКИЙ**
(Представлено: Д.В. ПЯТКИН)

Рассмотрена генерация случайных(псевдослучайных) чисел в языке программирования PHP. Приведём примеры применения псевдослучайных чисел в PHP. Приведены возможные способы нахождения псевдослучайного числа. Исследуем процесс генерации псевдослучайного числа.

Случайные числа — это неотъемлемая часть программирования, особенно, если это касается систем безопасности. К примеру, криптография основывается на генерации случайных значений, которые невозможно предугадать. Конечно же и в PHP случайные числа играют огромную роль: с их помощью можно генерировать токены, идентификаторы и другие значения.

Псевдослучайные числа — вырабатываемая алгоритмически последовательность чисел, обладающих свойствами случайных чисел и используемых взамен последних при решении на ЭВМ ряда классов задач.

Как сказано выше, генерация псевдослучайных чисел основывается на специальных алгоритмах [1]. Входным параметром, от которого будет отталкиваться алгоритм, может быть как случайное значение, так и заранее определённое.

В PHP псевдослучайные числа используются для различных целей. В основном, они связаны с безопасностью. На их основе генерируются токены, ключи, значения для аутентификации, значения для сброса паролей и многое другое. Всё это делается для того, чтобы получаемые значения невозможно было предугадать.

Некоторые важные примеры применения псевдослучайных значений:

- Генерация значений для криптографии — псевдослучайное число используется, например, для шифрования «в одну сторону», а также для хэширования паролей. Также псевдослучайное значение используется как вектор инициализации в криптографии;

- Генерация псевдослучайных значений, таких как ID сессии — PHP используется для создания огромного количества приложений, где безопасность стоит на первом месте. Многий функционал базируется на работе с сессиями и генерированными ID сессий;

- Генерация токенов для аутентификации, которые практически невозможно предугадать — многие PHP приложения базируются на работе с другими системами через специальные API и интерфейсы. Обычно перед использованием API нужно пройти процесс аутентификации. Получать трудно-подбираемые значения для токенов очень сложно. Именно поэтому в данных задачах используются псевдослучайные числа.

API (программный интерфейс приложения) (англ. application programming interface, API) — набор готовых классов, процедур, функций, структур и констант, предоставляемых приложением (библиотекой, сервисом) или операционной системой для использования во внешних программных продуктах[2].

Генераторы псевдослучайных чисел. Псевдослучайные числа, использующиеся в случаях, описанных выше, в PHP генерируются псевдо-генераторами. Всего доступно несколько алгоритмов:

- Линейный конгруэнтный метод, при использовании функции `lcg_value()`. Функция `lcg_value()` возвращает псевдослучайное значение в диапазоне (0,1). Функция комбинирует два КГ с периодами $2^{31} - 85$ и $2^{31} - 249$. Период этой функции равен произведению базовых [3];

- Вихрь Мерсенна, используется функцией `mt_rand()`. Функция `mt_rand()` представляет собой замену старой функции `rand()`. Она использует генератор псевдослучайных чисел с известными характеристиками, основанный на Вихре Мерсенна, который генерирует псевдослучайные числа в среднем в 4 раза быстрее, чем функция `rand()`. Вызванная без обязательных параметров `min` и `max`, функция `mt_rand()` возвращает псевдослучайное значение между 0 и `RAND_MAX`. Если нужно, например, псевдослучайное число между 5 и 15(включительно), следует использовать вызов `mt_rand(5, 15)`[4];

- Функция `rand()`, использующая аналогичную функцию в языке Си [5].

Фактически данные функции возвращают не случайные числа, а числа, распределённые таким образом, что они выглядят как случайные. Последовательность этих чисел зависит от базового случайного числа внутри реализованного алгоритма.

Базовые числа для генераторов. Базовые числа или вектора таких чисел — это наборы данных, которые используются для генерации псевдослучайных чисел. Псевдо-генераторы случайных чисел работают, только отталкиваясь от них. Если злоумышленник узнает это базовое число, то в будущем сможет предугадать значения ваших псевдослучайных чисел.

В PHP можно задать базовые числа двумя способами. Первый — это используя функцию `mt_srand()`. Этот способ в основном используется при юнит тестах случайного ряда.

Второй способ — это предоставление PHP право самому генерировать базовые числа. Начиная с версии 4.2, PHP предоставляет эту возможность. В последствии для генерации псевдослучайных чисел будет задействован Вихрь Мерсенна.

PHP генерирует базовое число, в зависимости от операционной системы. На платформах Linux можно воспользоваться функциями `mcrypt_create_iv()` или `openssl_pseudo_random_bytes()` в `/dev/urandom`. Windows предоставляет специальный псевдо-генератор, к которому можно получить доступ через функции `openssl_pseudo_random_bytes()` и `mcrypt_create_iv()`.

Таким образом, в статье рассмотрены различные способы генерации псевдослучайных чисел и особенности их использования. Исследован процесс генерации псевдослучайного числа в языке программирования PHP.

ЛИТЕРАТУРА

1. Псевдослучайные числа [Электронный ресурс]. – Режим доступа: https://economic_mathematics.academic.ru/3716/Псевдослучайные_числа. – Дата доступа: 24.09.2018.
2. API [Электронный ресурс] // Материал из Википедии – свободной энциклопедии. – Режим доступа: <https://ru.wikipedia.org/wiki/API>. – Дата доступа: 24.09.2018.
3. `lcg_value` [Электронный ресурс]. – Режим доступа: <http://php.net/manual/ru/function.lcg-value.php>. – Дата доступа: 24.09.2018.
4. `mt_rand` [Электронный ресурс]. – Режим доступа: <http://php.net/manual/ru/function.mt-rand.php>. – Дата доступа: 25.09.2018.
5. `rand` [Электронный ресурс]. – Режим доступа: <http://php.net/manual/ru/function.rand.php>. – Дата доступа: 25.09.2018.

УДК 004.056.55

ПЯТЬ УРОВНЕЙ УПРАВЛЕНИЯ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ В ОРГАНИЗАЦИИ

Р.Ю. КАРАБАНОВ

(Представлено: канд. физ.-мат. наук, доц. Ю.Ф. ПАСТУХОВ)

Чтобы корректно защитить информацию в организации, необходимо иметь грамотную программу управления уязвимостями. Без такого пункта не обходятся ни одни рекомендации по обеспечению кибербезопасности на предприятии.

При рассмотрении этих пяти этапов будем основываться на модели зрелости возможностей (СММ, Capability Maturity Model), что в итоге должно дать вам представление о том, как вывести вашу организацию на новый уровень обеспечения информационной безопасности.

СММ переводится как "модель зрелости возможностей". Подразумевает модель зрелости процессов в компании. Изначально СММ была разработана для измерения эффективности работы подрядчиков, выполняющих заказы правительства США на разработку и поставку ПО. Позже модель была обобщена для любых компаний-разработчиков программного обеспечения.

Структура СММ. За основу при оценке способности организации качественно выполнять работу, которая (способность) была названа зрелостью, создатели модели взяли процессы организации. Дальше они сделали несколько нетривиальных предположений, которые впоследствии были приняты и признаны справедливыми многими ИТ-специалистами (а может быть, и большинством из них).

Предположение 1. Существуют качественно отличающиеся уровни зрелости проектной организации, разрабатывающей информационные системы (в модели СММ таких уровней пять).

Предположение 2. Всякая организация-разработчик заинтересована в переходе на более высокий уровень зрелости (не только для того, чтобы повысить свои шансы в борьбе за контракты Министерства обороны, но и в целях собственного совершенствования).

Предположение 3. Переход возможен только на следующий по порядку уровень. "Перескочить" через уровень нельзя (точнее, риски для организации при этом резко возрастают).

Таким образом, уровни образуют "лестницу", по которой поднимается организация по мере собственного развития. Каждый уровень характеризуется определенными составом и свойствами процессов организации. "Лестница уровней" СММ получила широкое признание и распространение. Вот как она выглядит (рис.).

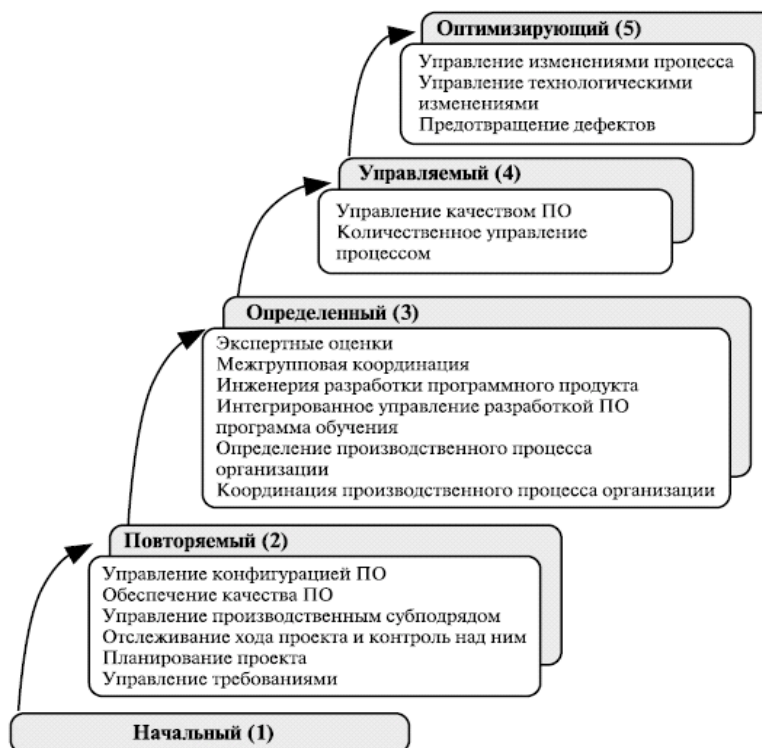


Рисунок. – Распределение групп ключевых процессов по уровням зрелости

Уровень 1: Начальный.

На начальном этапе программы управления уязвимостями процессы и процедуры либо отсутствуют полностью, либо присутствуют в минимальном наличии. Сканирование на уязвимости выполняется сторонним вендором как часть пентеста или внешнего сканирования. Такие тестирования обычно выполняются от одного до четырех раз в год по требованию аудитора, либо согласно нормативным требованиям.

Уровень 2: Повторяемый.

На этом этапе программы управления уязвимостями сканирование дыр в безопасности осуществляется внутри компании. Организация определяет набор процедур для сканирования брешей – обычно приобретается решение для управления уязвимостями, после чего сканирование проводится раз в неделю или в месяц, но на регулярной основе.

Большинство организаций, находящихся на этом этапе, не располагают поддержкой своего руководства, что приводит к тому, что бюджет на сканирование уязвимостей бывает крайне ограничен. Соответственно, приобретается относительно дешевое решение, либо используется бесплатный сканер уязвимостей с открытым исходным кодом.

Уровень 3: Установленный.

Этот уровень подразумевает, что программа управления уязвимостями качественно продумана и вся организация понимает ее. Команда компьютерной безопасности имеет поддержку со стороны руководства, а также располагает доверием системных администраторов. На этом этапе команда компьютерной безопасности уже выбрала надежное и безопасное решение для сканирования сети организации.

Согласно рекомендациям Center for Internet Security (CIS), на этом этапе сканирования проводятся минимум раз в неделю, а отчеты составляются индивидуально и доставляются на разные уровни организации. Таким образом, системные администраторы получают конкретные отчеты об уязвимостях, а менеджмент о тенденциях в этой области.

Данные о дырах в безопасности затем расшариваются с остальными отделами, завязанными на информационной безопасности, что позволяет обеспечить наличие оперативной информации. Например, если на внешнем брандмауэре обнаружен эксплойт, SIEM позволит быстро определить, какие системы уязвимы для этого эксплойта.

Уровень 4: Управляемый.

Здесь оцениваются конкретные атрибуты программы, а показатели предоставляются команде менеджмента. Можно выделить следующие показатели уязвимости, которые должна оценивать каждая организация:

Какой процент систем организации еще не подвергался сканированию системой управления уязвимостями?

Каков средний показатель уязвимости каждой из систем организации?

Каков общий показатель уязвимости каждой из систем организации?

Сколько времени в среднем занимает полное обновление прикладных программ для системы?

Эти показатели можно применять как к организации в целом, так и к отдельным подразделениям, что поможет понять, какие подразделения снижают риски, а какие отстают в этом вопросе.

Уровень 5: Оптимизированный.

На оптимизированном уровне метрики, определенные на предыдущем этапе, модернизируются. Оптимизация каждого из показателей способна обеспечить уменьшение поверхности для атаки. Команда безопасников должна работать совместно с менеджментом, чтобы установить определенные цели для программы управления уязвимостями.

Таким образом, обеспечение постоянного совершенствования вашей программы управления уязвимостями является ключом к уменьшению поверхности для атаки вашей организации. Если вам удастся уменьшить поверхность, вы поставите киберпреступников в затруднительное положение, им придется искать новые подходы и лазейки.

ЛИТЕРАТУРА

1. Пять этапов управления уязвимостями в организации [Электронный источник]. – 2018. – Режим доступа: <https://www.anti-malware.ru/practice/methods/five-stages-of-vulnerability-management>. – Дата доступа: 23.09.2018.
2. НОУ ИНТУИТ | Лекция | Зрелость проектных организаций. Методология СММ [Электронный источник]. – 2018. – Режим доступа: <https://www.intuit.ru/studies/courses/2298/598/lecture/12857>. – Дата доступа: 23.09.2018.
3. Криптографические основы безопасности [Электронный источник]. – 2017. – Режим доступа: <http://www.intuit.ru/studies/courses/28/28/lecture/20412?page=4#sect17>. – Дата доступа: 23.09.2018.
4. Что такое СММ? [Электронный источник]. – 2018. – Режим доступа: <http://www.quizful.net/interview/qa/cmm>. – Дата доступа: 23.09.2018.
5. Capability Maturity Model [Электронный источник] // Википедия. – 2018. – Режим доступа: https://ru.wikipedia.org/wiki/Capability_Maturity_Model. – Дата доступа: 23.09.2018.

УДК 004.056.55

КАК ВЫПОЛНЯЕТСЯ ШИФРОВАНИЕ В МЕССЕНДЖЕРЕ TELEGRAM**Р.Ю. КАРАБАНОВ***(Представлено: канд. физ.-мат. наук, доц. Ю.Ф. ПАСТУХОВ)*

Несмотря на то что мессенджер Telegram вышел на рынок позднее своих главных конкурентов WhatsApp и Viber, он довольно быстро приобрел репутацию одного из самых безопасных сервисов. Шифрование Telegram с помощью собственной разработки – протокола MTProto – позволило создать хорошо защищенный от взлома продукт, благодаря чему он стал популярен во всем мире.

Рейтинг безопасности

Безопасность виртуального общения определяется рейтингом Фонда электронных рубежей (EFF). В постоянно обновляемой таблице каждому сервису выставляется оценка от 1 до 7 в зависимости от уровня защиты информации от потенциального взлома (рис.).



Рисунок. – Шифрование Telegram является самым защищенным среди других мессенджеров по версии Falcongaze Analytics Center

Секретные чаты Telegram, шифрование которых осуществляется по принципу end-to-end (E2E), имеют в данном рейтинге максимальный бал – 7, а стандартная переписка, используемая по умолчанию, – 4. Поскольку при обычном общении ключи сохраняются на серверах компании, считается, что потенциально они могут стать доступны третьи лицам.

До недавнего времени рейтинг WhatsApp и Viber в таблице EFF был не очень высоким и равнялся всего двум баллам. Конкуренция с Telegram заставила данные компании пересмотреть свою политику безопасности. В связи с этим, был введен принцип оконечного шифрования, который с 2106 года стал использоваться по умолчанию, и позволил получить 6 баллов от EFF. Его суть заключается в хранении ключей, необходимых для расшифровки сообщений, только на устройстве пользователя. Таким образом, чтобы получить доступ к информации, необходимо обладать физическим доступом к смартфону.

Возникает вопрос: если сервис позиционирует себя как самый безопасный мессенджер, почему не использовать секретные чаты по умолчанию, что позволит безоговорочно возглавить рейтинг? Дело в том, что E2E-шифрование имеет определенный недостаток – секретная переписка привязывается к конкретному устройству, поэтому и ее история хранится только на одном устройстве.

Политика компании заключается в предоставлении пользователям права выбора, ведь стандартный режим позволяет заходить в аккаунт с любого устройства.

Шифрование Telegram на основе MTProto

Протокол MTProto использует два слоя шифрования – сервер-сервер и клиент-сервер. Он работает на основе следующих алгоритмов:

- AES – симметричный 256-битный алгоритм, принятый правительством США в качестве стандарта.
- RSA – криптографический алгоритм, в основе которого лежит вычислительная сложность задачи факторизации крупных целых чисел.
- Метод Диффи-Хеллмана – позволяет получить двум и более собеседникам секретный ключ по незащищенному от прослушивания, однако защищенному от подмены каналу связи.
- SHA-1, MD5 – хеш-алгоритмы, используемые во многих криптографических протоколах и приложениях для безопасного хеширования. В отличие от протокола Double Ratchet, который применяется WhatsApp и уже успел получить одобрение известных специалистов в области защиты информации, разработчики MTProto не спешат предоставлять свой продукт для независимого аудита. С одной стороны, это делает алгоритм потенциально уязвимым для атак, с другой – на сегодняшний день не зафиксировано ни одного успешного действия, приведшего к расшифровке сообщений. Создатели мессенджера заявляют о гарантии безопасности в отношении передачи зашифрованных данных. Для подтверждения своих слов Павел Дуров периодически организывает конкурсы, в которых участникам предлагается расшифровать переписку двух собеседников. Призовой фонд составляет 200 тыс. долларов, однако до настоящего времени ни один хакер не смог прочитать зашифрованные сообщения. Справедливости ради следует отметить, что многие эксперты довольно скептически относятся к подобным конкурсам, считая их скорее продуктами пиара, чем реальным доказательством защищенности системы.

Реален ли взлом

Даже если принять в качестве аксиомы, что MTProto действительно имеет лучшие параметры защиты среди современных мессенджеров, злоумышленники все же имеют возможность взломать аккаунт пользователя. При этом сам протокол здесь абсолютно не причем.

Уязвимость заключается в способе авторизации пользователя. Для данной процедуры используется реальный номер телефона, на который отправляется СМС-код для подтверждения входа в аккаунт. В основе подобного метода передачи данных лежит технология SS7 (Signaling System #7), которая разрабатывалась 40 лет назад и обладает слабыми параметрами безопасности по современным меркам. Теоретически злоумышленники могут перехватить СМС с кодом и взломать аккаунт. А поскольку в обычном режиме Telegram хранит все сообщения на своих серверах, хакеры могут получить доступ ко всей переписке конкретного пользователя.

Проблему решает общение в секретных чатах. В этом случае прочитать переписку можно только посредством реальной кражи телефона, так как сообщения не хранятся на сервере, а передаются исключительно между двумя устройствами.

ЛИТЕРАТУРА

1. Как осуществляется шифрование Телеграмм и в чем его отличие от других мессенджеров [Электронный источник]. – 2018. – Режим доступа: <https://ru.telegram-store.com/blog/shifrovanie-telegramm/>. – Дата доступа: 23.09.2018.
2. [Перевод] Почему у Telegram не включено End-to-End шифрование по умолчанию? [Электронный источник]. – 2018. – Режим доступа: <https://medium.com/@tglive/telegram-end-to-end-e93554cb9e46>. – Дата доступа: 23.09.2018.
3. Безопасен ли Telegram? Или как я искал закладку в MTProto / Хабр [Электронный источник]. – 2018. – Режим доступа: <https://habr.com/post/206900/>. – Дата доступа: 23.09.2018.
4. Анализ безопасности Telegram [Электронный источник]. – 2018. – Режим доступа: <https://www.securitylab.ru/analytics/490726.php>. – Дата доступа: 23.09.2018.
5. Миллиард чатов на замок: как работает шифрование WhatsApp и в чем его недостатки перед Telegram [Электронный источник]. – 2018. – Режим доступа: <https://tjournal.ru/26022-milliard-chatov-na-zamok-kak-rabotaet-shifrovanie-whatsapp-i-v-chem-ego-nedostatki-pered-telegram>. – Дата доступа: 23.09.2018.

УДК 811; 378.147

**ВЫБОР СРЕДСТВ И МЕТОДОВ РЕШЕНИЯ ЗАДАЧ.
ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ ОБУЧАЮЩЕГО ПРИЛОЖЕНИЯ
ДЛЯ ИЗУЧЕНИЯ АНГЛИЙСКОГО ЯЗЫКА****В.С. РОГУЛЕВ***(Представлено: канд. техн. наук, доц. А.Ф. ОСЬКИН)*

Представлен объект разработки – обучающее приложение для изучения английского языка. Рассмотрены вопросы выбора и обоснования средств и методов решения задач, проектирования базы данных для обучающего приложения.

Выбор и обоснование средств и методов решения задач

Выбор системы управления базами данных (СУБД) представляет собой сложную многопараметрическую задачу и является одним из важных этапов при разработке клиент-серверных приложений. Определяющим здесь являются вид программного продукта и категории пользователей (или профессиональные программисты, или конечные пользователи, или и то, и другое).

Другими показателями, влияющими на выбор СУБД являются:

- удобство и простота использования;
- качество средств разработки, защиты и контроля базы данных;
- уровень коммуникационных средств в случае применения ее в сетях;
- фирма-разработчик;
- стоимость.

SQL Server 2014 обеспечивает встроенные в базы данных возможности обработки данных в оперативной памяти при любых рабочих нагрузках, более быстрое получение результатов анализа данных с использованием знакомых средств аналитики, а также использование решений по обработке больших данных на корпоративном уровне.

Система SQL Server 2014 позволяет обращаться к данным из любого приложения, разработанного с применением технологий Microsoft .NET и Visual Studio. MS SQL Server обеспечивает высочайший уровень безопасности, надежности и масштабируемости для критически важных приложений. Чтобы использовать новые возможности, постоянно возникающие в быстро меняющемся деловом мире, предприятиям нужно быть способными быстро создавать и развертывать решения, управляемые данными. MS SQL Server 2014 позволяет сократить затраты времени и средства, требуемые на управление и развертывание таких приложений. Следует учесть, что фирма-разработчик данной СУБД является также разработчиком самой распространенной ОС. В финансовом плане важным фактором является то, что существуют бесплатные сборки данной СУБД.

Microsoft SQL Server – система управления реляционными базами данных, разработанная корпорацией Microsoft. Основным используемым языком запросов – Transact-SQL, создан совместно Microsoft и Sybase. Transact-SQL является реализацией стандарта ANSI/ISO по структурированному языку запросов (SQL) с расширениями. Используется для небольших и средних по размеру баз данных, и в последние 5 лет – для крупных баз данных масштаба предприятия, конкурирует с другими СУБД в этом сегменте рынка.

Для реализации приложения была выбрана среда разработки Microsoft Visual Studio 2017 – язык программирования C#. Был выбран именно этот язык по причине следующих его достоинств:

- претендует на подлинную объектную ориентированность (а всякая языковая сущность претендует на то, чтобы быть объектом);
- призван практически реализовать компонентно-ориентированный подход к программированию, который способствует меньшей машинно-архитектурной зависимости результирующего программного кода, большей гибкости, переносимости и легкости повторного использования (фрагментов) программ;
- изначальная ориентация на безопасность кода (что особенно заметно в сравнении с языками C и C++);
- расширенная поддержка событийно-ориентированного программирования;
- является «родным» для создания приложений в среде Microsoft .NET.

Для реализации приложения была выбрана клиент-серверная архитектура. Основными преимуществами работы в клиент-серверном варианте являются:

- высокая производительность при обработке больших объемов данных информационной базы;
- удобство администрирования и упорядочивание доступа пользователей к информационной базе.

Учитывая выше сказанное, MS SQL Server 2014 и С# (с использованием среды разработки Microsoft Visual Studio 2017) подходят для реализации обучающего приложения для изучения английского языка.

Проектирование базы данных

Главной целью проектирования базы данных является выбор оптимального количества таблиц для хранения данных, полей, которые должны войти в ту или иную таблицу, а также планирование отношений между таблицами. Эту и многие другие проблемы при проектировании баз данных можно решить с помощью нормализации.

Нормализация – процесс преобразования отношений базы данных к виду, отвечающему нормальным формам.

Нормальная форма – свойство отношения в реляционной модели данных, характеризующее его с точки зрения избыточности, потенциально приводящей к логически ошибочным результатам выборки или изменения данных. Нормальная форма определяется как совокупность требований, которым должно удовлетворять отношение.

При проектировании баз данных могут применяться различные подходы. Основная задача проектирования баз данных – это сокращение избыточности хранимых данных, а, следовательно, экономия используемых ресурсов (оперативной и дисковой памяти), уменьшение затрат на обновление избыточности копий данных, а также снижение вероятности нарушения целостности данных.

Реляционная база данных – это совокупность отношений, в которых хранится вся информация баз данных. Для пользователя такая база данных представляется набором двумерных таблиц, что облегчает понимание структуры данных и управления ими. Таблицы реляционной базы данных связаны между собой отношениями.

Схема базы данных обучающего приложения для изучения английского языка представлена на рисунке.

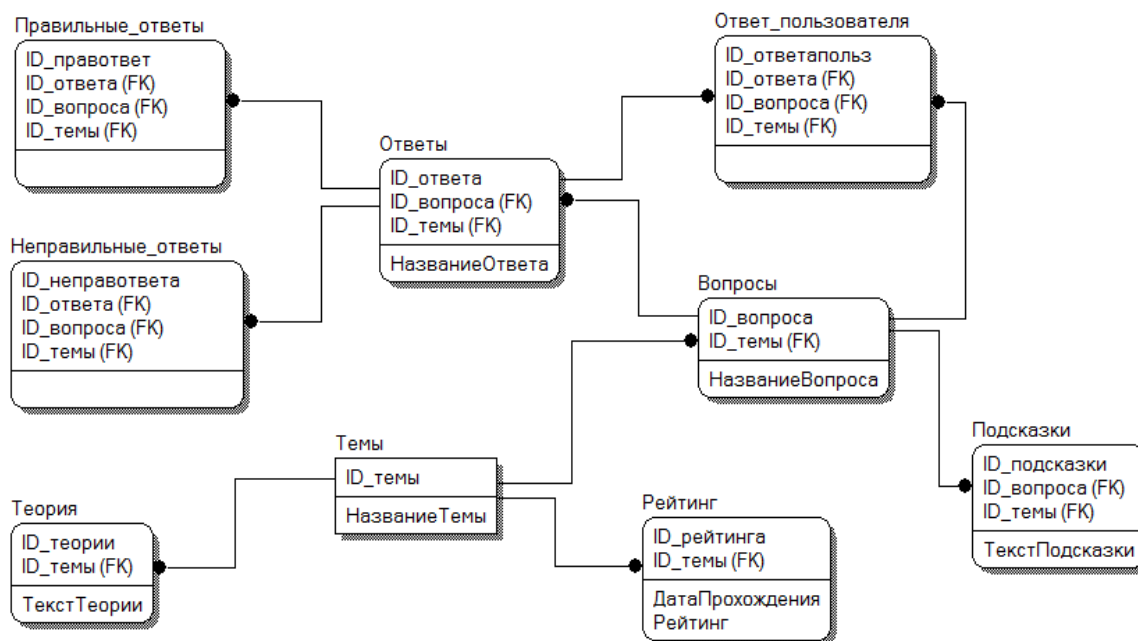


Рисунок. – Схема базы данных обучающего приложения

В результате проделанной работы была выбрана среда разработки приложения и спроектирована база данных, при помощи которой можно создать обучающее приложение для изучения английского языка, направленное на проверку знаний пользователя и хранение результатов прохождения тестов.

ЛИТЕРАТУРА

1. Раздел «Microsoft SQL Server 2008» на сайте Microsoft [Электронный ресурс]. – Режим доступа: <http://www.microsoft.com/sqlserver/2008/ru/ru/product-information.aspx>. – Дата доступа: 23.09.2018.
2. Малыгина, М. Базы данных: основы, проектирование, использование / М. Малыгина. – М.: BHV, 2004. – 512 с.
3. Арлоу, Д. С# для профессионалов : учеб. пособие / Д. Арлоу, И.И. Саливон. – Минск : Технология, 2007.

УДК 004.9

**ЭКОНОМИЧЕСКАЯ СОСТАВЛЯЮЩАЯ РАЗРАБОТКИ ОБУЧАЮЩЕГО ПРИЛОЖЕНИЯ
ДЛЯ ИЗУЧЕНИЯ АНГЛИЙСКОГО ЯЗЫКА.
ОЦЕНКА ВЛИЯНИЯ ИНФОРМАЦИОННОЙ СРЕДЫ НА ЭКОНОМИКУ****В.С. РОГУЛЕВ***(Представлено: канд. техн. наук, доц. А.Ф. ОСЬКИН)*

Представлена информация об экономической части разработки обучающего приложения. Также представлена информация об основных стадиях разработки приложения, видах работ и их экономической составляющей.

Программные средства подобно другим промышленным изделиям имеют определенный жизненный цикл. Под жизненным циклом программных средств вычислительной техники понимается период от начала разработки нового программного средства до снятия его с эксплуатации потребителем. Жизненный цикл включает три стадии: разработка (проектирование), производство (создание) и использование (сопровождение программного средства). Каждая стадия в свою очередь делится на фазы или этапы.

Стадия разработки программного средства может быть разделена на следующие этапы:

1. Проектирование. К этапу «Проектирование» относится анализ технического задания на разработку и проектирование. На данный этап отводится значительный период времени, так как качественно выполненное проектирование позволяет сократить временные затраты на последующие этапы разработки.

2. Реализация. К этапу «Реализация» относится непосредственно разработка программного продукта. Данный этап является самым продолжительным и составляет от 40 до 60% от всего времени.

3. Тестирование и испытание системы. На этапе «Тестирование и испытание системы» проводится проверка соответствия результата этапа реализации результатам, полученным на этапе проектирования. По продолжительности данный этап занимает около 20% всего времени разработки.

4. Доработка. На этап «Доработка» проводится заключительные приемо-сдаточные испытания, при необходимости – внесение незначительных исправлений в программный продукт, и оформление заключительных глав пояснительной записки.

Графическое отображение этапов разработки программы представлено на рисунке 1.

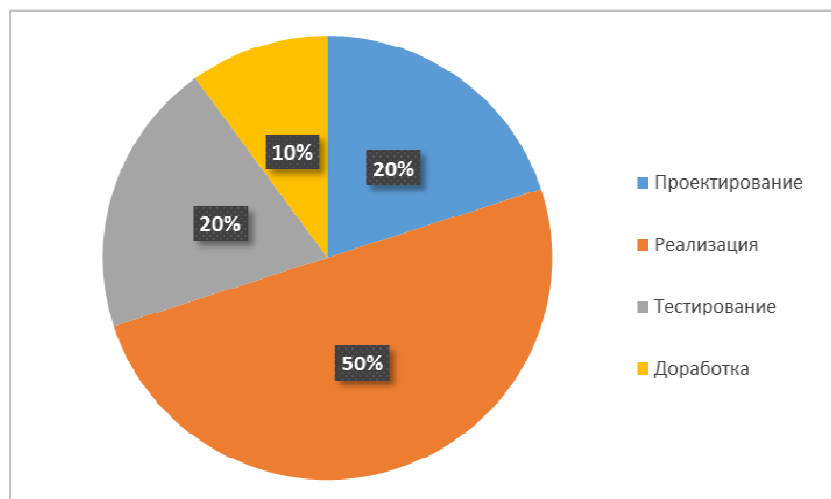


Рисунок 1. – Диаграмма распределения времени работы при разработке проекта

Виды работ. Кроме деления жизненного цикла ПС на стадии и фазы, можно выделить восемь видов работ, которые выполняются в процессе создания программного продукта:

1. Анализ требований и планирование. Предусматривает разработку спецификаций, анализ и модификацию функциональных, технических, интерфейсных требований.

2. Проектирование изделий. Включает в себя определение, спецификацию, анализ и модификацию аппаратно-программной архитектуры проекта программы и базы данных.

3. Программирование. Детальное проектирование, кодирование, автономная отладка и комплексирование отдельных компонентов программы, а также планирование работ программистов, разра-

ботка базы данных, документирование отдельных компонентов и организация программирования на уровне компонентов.

4. Планирование отладки
 5. Верификация. Процесс проверки корректности требований, отладки изделий и приемные испытания.
 6. Управление проектом. Планирование и контроль проекта, контроль и регулирование договоров, связь с пользователями.
 7. Контроль качества. Разработка и контроль, стандартные и технические проверки программных средств и процессов разработки.
 8. Документирование. Разработка и корректировка руководства для пользователя и операторов.
- Деление жизненного цикла программных средств на стадии, фазы и этапы, а также более подробное описание работ, выполняемых в процессе их разработки, производства и эксплуатации, необходимо для детального определения затрат, требуемых на эти цели.

Расчет стоимости продукта. В стоимость любого продукта включена стоимость его создания. В данном случае, программным продуктом является обучающее приложение. Расчет стоимости должен учитывать следующие статьи затрат:

1. Материалы и комплектующие – расходные материалы, потраченные при создании продукта;
2. Электроэнергия – так как продуктом является приложение, то при его разработке используются компьютеры, потребляющие электричество;
3. Основная заработная плата исполнителей – затраты на выдачу заработной платы сотрудникам, работающим над проектом;
4. Дополнительная заработная плата – затраты на оплату премий, рисков, отпусков и прочее;
5. Отчисления на социальные нужды – затраты на налоговые сборы, в фонд защиты населения, страхование, прочее;
6. Амортизация – затраты на непредвиденные обстоятельства;
7. Расходы на спецоборудование – затраты, связанные с приобретением специализированного оборудования;
8. Накладные расходы – расходы на общехозяйственные нужды;
9. Прочие прямые расходы – затраты на подготовку и приобретение различного рода информации.

Влияние статей затрат на окончательную себестоимость продукта можно увидеть на диаграмме, представленной на рисунке 2.

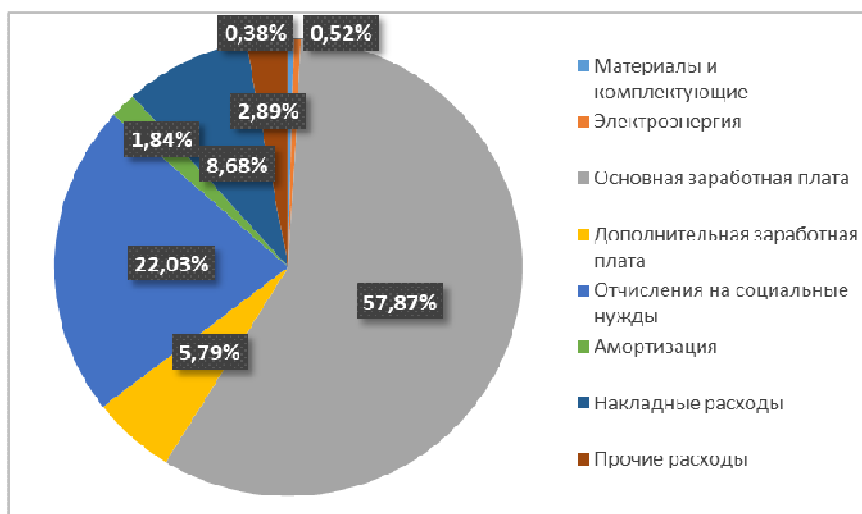


Рисунок 2. – Структура затрат на разработку обучающего приложения

Глядя на диаграмму можно сделать вывод, что основные затраты на разрабатываемый продукт, вне зависимости от сроков разработки, приходятся на выдачу заработной платы сотрудникам.

ЛИТЕРАТУРА

1. Habr.com [Электронный ресурс]. – Режим доступа: <https://www.habr.com>. – Дата доступа: 23.09.2018.
2. Richpro.ru – финансовый журнал [Электронный ресурс]. – Режим доступа: <http://richpro.ru>. – Дата доступа: 23.09.2018.

УДК 004.021

**ЗАЩИТА ДАННЫХ ПОЛЬЗОВАТЕЛЯ НА СТОРОНЕ СЕРВЕРА
ПОСРЕДСТВОМ ФРЕЙМВОРКА SPRING SECURITY И ТЕХНОЛОГИИ JSON WEB TOKEN****П.Д. ТАЛАЙКО***(Представлено: канд. тех. наук, доц. И. Б. БУРАЧЁНОК)*

Рассмотрены принципы работы фреймворка Spring Security и технологии JSON Web Token. Проведено основное описание реализации механизма авторизации и аутентификации пользователей в веб-приложении. Показаны реализация основных функций механизма защиты.

В современных веб-приложениях зачастую необходимо хранить пользовательскую информацию: настройки системы, личные данные, фильтры поиска и т.д. Поскольку веб-приложения являются многопользовательскими, то целесообразно идентифицировать пользователей посредством логина и пароля.

В представленной статье рассмотрим способ хранения личной информации пользователя с использованием фреймворка Spring Security и технологии JSON Web Token (JWT).

Spring Security – Java/Java EE фреймворк, предоставляющий механизмы построения систем аутентификации и авторизации, а также другие возможности обеспечения безопасности для промышленных приложений, созданных с помощью Spring Framework. Spring Security позволяет сделать разграничение доступа пользователей приложения к ресурсам, предоставляет необходимые для хэширования паролей средства. При этом Spring Security используется вместе с JSON Web Token.

JWT (JSON Web Token) – это открытый стандарт (RFC 7519) для создания токенов доступа, основанный на JSON формате. Как правило, используется для передачи данных авторизации в клиент-серверных приложениях. Токены создаются сервером, подписываются секретным ключом и передаются клиенту, который в дальнейшем использует данный токен для подтверждения своей личности. Клиент в дальнейшем использует данный токен для обращения к ресурсам веб-приложения.

Каждый пользователь при входе в систему обязан ввести свой логин и пароль. Пароли хранятся в базе данных в хэшированном виде, поэтому даже если злоумышленник каким-то образом получит доступ к таблице пользователей, он не сможет получить данные для входа в систему под именем другого пользователя.

Аутентификация – это процесс проверки подлинности каких-либо данных. Примером аутентификации может быть сравнение пароля, введенного пользователем, с паролем, который сохранен в базе данных сервера.

Пароли в базе данных не хранятся в открытом виде, а хешируются алгоритмом bcrypt, который позволяет сгенерировать разные хеш-значения для одинаковых данных. Если данные пользователя в базе данных совпадают с введенными, то пользователю выдается JWT-токен и процедуру аутентификации можно считать успешной.

Авторизация – предоставление определённому лицу или группе лиц прав на выполнение определённых действий. В разрабатываемом веб-приложении существуют две роли: роль пользователя (ROLE_USER) и роль администратора (ROLE_ADMIN). Соответственно существует необходимость разграничения доступа к возможностям веб-приложения и у пользователя не должно быть прав на выполнения части программного кода, если у него присутствует только роль пользователя. В качестве механизма разграничения ролей используется стандартная конфигурация Spring Security. Также каждый метод классов-контроллеров имеет аннотацию @PreAuthorize, которая позволяет либо допустить пользователя к исполнению программного кода, либо отвергнуть его запрос с ошибкой, если его роль не совпадает с той, что указана в аннотации.

Получение ролей сервером осуществляется из JWT токена, который хранится у пользователя локально в браузере в localStorage. При совершении какого-либо действия клиентское приложение передает JWT токен, в котором хранится информация о его ролях. Если пользователь локально изменил содержание токена, то при прохождении обработки на сервере и сравнения с секретным ключом он будет недействителен, что гарантируется спецификацией JWT.

При генерации JWT происходит установка времени жизни токена, после истечения которого токен становится не валидным и нуждается в обновлении. Построение токена можно привести в листинге 1.

```
private String doGenerateToken(Map<String, Object> claims, String subject) {
    final Date createdAt = clock.now();
    final Date expirationDate = calculateExpirationDate(createdAt);
    return Jwts.builder()
        .setClaims(claims)
        .setSubject(subject)
        .setIssuedAt(createdAt)
        .setExpiration(expirationDate)
        .signWith(SignatureAlgorithm.HS512, secret)
        .compact();
}
```

Листинг 1. – Функция построения токена

Как можно видеть в функции `doGenerateToken` из листинга 1, происходит построение JWT String токена, который включает в себя встроенную информацию (`claims`), имя пользователя (`subject`), дату создания токена (`createdAt`), дату истечения (`expirationDate`) и алгоритм шифрования с секретным значением. Как правило, секретным значением выступает текст, прописанный в контексте приложения.

По истечению времени токена необходимо его обновить, для чего будет использоваться функция `refreshToken`, приведённая в листинге 2.

```
public String refreshToken(String token) {
    final Date createdAt = clock.now();
    final Date expirationDate = calculateExpirationDate(createdAt);

    final Claims claims = getAllClaimsFromToken(token);
    claims.setIssuedAt(createdAt);
    claims.setExpiration(expirationDate);

    return Jwts.builder()
        .setClaims(claims)
        .signWith(SignatureAlgorithm.HS256, secret)
        .compact();
}
```

Листинг 2. – Функция обновления токена

В функцию `refreshToken` необходимо передать токен, время жизни которого уже истекло, установить с него `claims` и новую дату в обновлённый токен и вернуть.

Функция для проверки валидности токена приведена в листинге 3.

```
public Boolean validateToken(String token, UserDetails userDetails) {
    JwtUser user = (JwtUser) userDetails;
    final String username = getUsernameFromToken(token);
    final Date created = getIssuedAtDateFromToken(token);
    return (
        username.equals(user.getUsername())
            && !isTokenExpired(token)
            && isCreatedAfterLastPasswordReset(created,
user.getLastPasswordResetDate())
    );
}
```

Листинг 3. – Функция проверки валидности токена

Spring Security необходим для получения данных пользователя `UserDetails` и фильтров HTTP запросов. Фильтр для проверки авторизации пользователя приведён в листинге 4.

```
@Component
public class JwtAuthenticationEntryPoint implements AuthenticationEntryPoint,
Serializable {
    private static final long serialVersionUID = -8970718410437077606L;
    @Override
    public void commence(HttpServletRequest request, HttpServletResponse re-
sponse,
                        AuthenticationException authException) throws IOEx-
ception {
        response.sendError (HttpServletResponse.SC_UNAUTHORIZED, "Unauthor-
ized");
    }
}
```

Листинг 4. – Фильтр HTTP запроса

Для использования токена пользователю предстоит передать логин и пароль на указанный URL, где вызовется функция `doGenerateToken` в случае, если пользователь будет найден в базе данных. Функция `doGenerateToken` вернёт сгенерированный токен, который необходимо будет передавать при обращении к API. Фреймворк `Spring Security` позаботится о проверке валидности токена и доступа к частям кода благодаря перехватчикам (фильтрам) HTTP запросов, которые проверяют наличие заголовка `Autorization`.

ЛИТЕРАТУРА

1. Token-Based Authentication. [Электронный ресурс] / Git hub gist. – Режим доступа: <https://gist.github.com/zmts/802dc9c3510d79fd40f9dc38a12bccfc>. – Дата доступа: 09.09.18.
2. Using JWT with Spring. [Электронный ресурс] / Baeldung. – Режим доступа: <https://www.baeldung.com/spring-security-oauth-jwt>. – Дата доступа: 09.09.18.
3. Spring Security. [Электронный ресурс] / Spring. – Режим доступа: <https://spring.io/projects/spring-security>. – Дата доступа: 10.09.18.
4. JWT. [Электронный ресурс] / JWT. – Режим доступа: <https://jwt.io/> – Дата доступа: 10.09.18.

УДК 004.021

**ЗАЩИТА ИНФОРМАЦИИ ПОЛЬЗОВАТЕЛЯ НА КЛИЕНТСКОЙ ЧАСТИ ВЕБ-ПРИЛОЖЕНИЯ
ПОСРЕДСТВОМ JWT ТОКЕНА И ВСТРОЕННЫХ МЕХАНИЗМОВ ANGULAR 5****П.Д. ТАЛАЙКО***(Представлено: канд. тех. наук, доц. И. Б. БУРАЧЁНОК)*

Рассмотрены механизмы фреймворка Angular 5, которые позволяют проверить права доступа пользователя к тем или иным данным и разрешить или запретить доступ к определенным страницам веб-приложения.

В современных веб-приложениях на клиентских частях зачастую необходимо разграничить доступ пользователей к данным и запретить просмотр некоторых страниц для одних и разрешить для вторых.

В представленной статье расскажем о механизмах Angular 5 решающих задачи разграничения доступа.

Angular представляет фреймворк от компании Google для создания клиентских приложений. Прежде всего он нацелен на разработку SPA-решений (Single Page Application), то есть одностраничных приложений. В этом плане Angular является наследником другого фреймворка AngularJS. В то же время Angular это не новая версия AngularJS, а принципиально новый фреймворк.

Angular 5 предоставляет такую функциональность, как двустороннее связывание, позволяющее динамически изменять данные в одном месте интерфейса при изменении данных модели в другом, шаблоны, маршрутизация и так далее.

Одной из ключевых особенностей Angular является то, что он использует в качестве языка программирования TypeScript. Поэтому перед началом работы рекомендуется ознакомиться с основами данного языка.

Приложения на Angular можно описать с помощью таких языков как Dart или JavaScript. Однако основным языком для Angular все-таки является TypeScript.

Server написан на фреймворках Spring Boot, Spring Data и Spring Security, реализует авторизацию и аутентификацию пользователей благодаря технологии JWT.

JWT (JSON Web Token) – это открытый стандарт RFC 7519 для создания токенов доступа, основанный на JSON формате [4]. Он, как правило, используется для передачи данных авторизации в клиент-серверных приложениях. Токены создаются сервером, подписываются секретным ключом и передаются клиенту, который в дальнейшем использует данный токен для подтверждения своей личности. Токены создаются сервером, подписываются секретным ключом и передаются клиенту, который в дальнейшем использует данный токен для обращения к ресурсам веб-приложения.

При вводе логина и пароля на клиентской части, данные в формате JSON отправляются на сервер POST запросом на определённый URL для получения токена JWT токена, который хранится у пользователя локально в браузере в localStorage, при совершении какого-либо действия клиентское приложение передает JWT токен в заголовке Authorization, в котором хранится информация о его ролях, если пользователь локально изменил содержание токена, то при прохождении обработки на сервере и сравнения с секретным ключом он будет недействителен, данное гарантируется спецификацией JWT.

На клиентской части необходимо написать сервисы для отправления HTTP запросов на сервер. В Angular 5 таким сервисом является HttpClient, который предоставляет основные методы для общения по HTTP протоколу [2].

Сервер реализует REST API и разграничивает доступ пользователей к определенным функциям, а при возникновении исключительных ситуаций возвращает соответственный код ошибки. Задача клиента корректно обработать эти ошибки и сообщить о их наличии.

Для установления заголовка Authorization в HTTP запрос нужно воспользоваться классом HttpHeaders. Неправильно устанавливать для всех запросов HTTP свой заголовок. В свою очередь можем воспользоваться HttpInterceptor.

HttpInterceptor – это interface Angular 5 позволяющий обработать HTTP запрос перед его отправкой на сервер [3]. Благодаря этому интерфейсу можем манипулировать глобально запросами с клиента.

После того как пользователь получил свой личный токен идентификации в системе, он устанавливается в localStorage (локальное хранилище браузера). При обращении к серверу в заголовок с помощью класса, реализующего интерфейс HttpInterceptor, можно установить токен, и при после обработать ответ.

Пример класса AuthInterceptor, реализующего интерфейс HttpInterceptor приведён в листинге 1.

```

@Injectable()
export class AuthInterceptor implements HttpInterceptor {

    private apiTokenName = 'token';

    constructor(private tokenStorage: TokenStorageService, private router: Router, private userStore: AuthService) {
    }

    intercept(req: HttpRequest<any>, next: HttpHandler): Observable<HttpEvent<any>> {
        const token = TokenStorageService.token();

        if (token) {
            req = req.clone({setHeaders: {Authorization: `Bearer ${token}`}});
        }

        return next.handle(req).pipe(
            tap(
                (response: HttpResponse<any>) => {
                    if (response.body && response.body.token) {
                        this.userStore.setAuthentication(true);
                        TokenStorageService.setToken(response.body.token);
                        this.router.navigateByUrl('/');
                    }
                }, (error: HttpResponse) => {
                    if (error instanceof HttpResponse) {
                        switch (error.status) {
                            case 401: {
                                this.router.navigateByUrl('/auth');
                                this.userStore.setAuthentication(false);
                            }
                            break;
                            case 403: {
                                this.router.navigateByUrl('/auth');
                            }
                            break;
                        }
                    } else {
                        return Observable.throw(error);
                    }
                }
            ),
        );
    }
}

```

Листинг 1. – Реализация класса AuthInterceptor

Из кода можно видеть, что при возвращении ошибки с кодом 401 (Unauthorized) и 403 (Forbidden) происходит перенаправление на страницу авторизации. При успешном доступе к ресурсам происходит установка статуса `this.userStore.setAuthentication(true)`.

В Angular 5 существует ещё один полезный механизм, позволяющий ограничивать доступ к сайтам по тем или иным причинам и называется Guard [5]. Guard'ы позволяют по определенному условию давать доступ к сайту или нет. Реализация Guard изображена в листинге 2.

```

@Injectable({
    providedIn: 'root'
})
export class MainGuard implements CanActivate {

    constructor(private auth: AuthService, private route: Router) {
    }
}

```



```
    canActivate(next:   ActivatedRouteSnapshot,   state:   RouterStateSnapshot):  
Observable<boolean> | Promise<boolean> | boolean {  
    if (!this.auth.isAuthenticated()) {  
        this.route.navigateByUrl('/auth');  
        return false;  
    }  
    return true;  
}  
}
```

Листинг 2. – Реализация класса MainGuard

Guard'ы устанавливаются на ссылку страницы (route) приложения и при переходе между компонентами вызываются и разрешают или запрещают доступ. В листинге можем видеть, что класс MainGuard реализует интерфейс CanActivate, имеющий один метод canActivate. В данном примере показано что если пользователь авторизован, то запрещать переход на страницу авторизации, а при выходе и изменении статуса пользователя на неавторизованный, доступ на страницу авторизации будет разрешён.

В данной статье представлены механизмы для обеспечения удобства работы с авторизацией и аутентификацией пользователя и доступа его к определённым компонентам веб-приложения. А также приведены основы клиентской работы с токенами JWT.

ЛИТЕРАТУРА

1. Token-Based Authentication [Электронный ресурс] // Git hub gist. – Режим доступа: <https://gist.github.com/zmts/802dc9c3510d79fd40f9dc38a12bccfc/>. – Дата доступа: 09.09.18.
2. HTTP [Электронный ресурс] // Angular. – Режим доступа: <https://angular.io/tutorial/toh-pt6/>. – Дата доступа: 10.09.18.
3. Angular Authentication: Using the Http Client and Http Interceptors [Электронный ресурс] / Medium. – Режим доступа: https://medium.com/@gyanchenkie_40935/angular-authentication-using-the-http-client-and-http-interceptors-2f9d1540eb8/. – Дата доступа: 10.09.18.
4. JWT [Электронный ресурс] // JWT. – Режим доступа: <https://jwt.io/>. – Дата доступа: 10.09.18.
5. Routing & Navigation [Электронный ресурс] // Angular. – Режим доступа: <https://angular.io/guide/router/>. – Дата доступа: 10.09.18.

УДК 004.05

**ПРОТОКОЛ HYPERTEXT TRANSFER PROTOCOL SECURE.
БЕЗОПАСНОСТЬ WEB-СЕРВИСОВ****Д.С. ТАТАРИН***(Представлено: канд. тех. наук, доц. И.Б. БУРАЧЁНОК)*

Рассмотрены особенности протокола HyperText Transfer Protocol Secure (HTTPS) и проблемы, с которыми можно столкнуться при его использовании. Сделаны выводы о необходимых мерах для обеспечения полной безопасности при использовании протокола HTTPS.

При вводе названия сайта все привыкли видеть вначале ссылки `http://` или HyperText Transfer Protocol (HTTP) – «протокол передачи гипертекста» – стандартный протокол передачи данных от сервера, на котором находится сайт, к пользователю. Однако, не смотря на всю его популярность, все больше сайтов предпочитают использовать более продвинутый протокол – HTTPS, так как он защищает передаваемые данные от перехвата злоумышленниками путем их шифрования. В представленной статье рассмотрим этот протокол более подробно: как он работает, кому рекомендуется использовать и что нужно, чтобы его подключить к сайту.

HTTPS – это протокол, который обеспечивает конфиденциальность обмена данными между сайтом и пользовательским устройством. Безопасность информации обеспечивается за счет использования криптографических протоколов SSL (Cure Sockets Layer) – слой защищенных сокетов и его предшественника TLS (Transport Layer Security) – протокол защиты транспортного уровня, далее SSL/TLS, имеющих три уровня защиты:

- шифрование данных – позволяет избежать их перехвата;
- сохранность данных – любое изменение данных фиксируется;
- аутентификация – защищает от перенаправления пользователя.

SSL можно сравнить с «фантомом», в который заворачивают данные HTTP, чтобы скрыть их от посторонних. Протокол SSL/TLS помогает двум незнакомым друг с другом пользователям Интернета установить защищенное соединение через обычный, незащищенный канал. С помощью математических алгоритмов оба пользователя – клиент и сервер – договариваются о секретном ключе, не передавая его напрямую через соединение. Даже если кто-то сумеет подключиться к соединению и перехватить все передаваемые данные, расшифровать их ему не удастся.

Протокол SSL использует многослойную среду: с одной стороны, от него находится протокол программы-клиента, например, (Internet Message Access Protocol (IMAP) – протокол прикладного уровня для доступа к электронной почте, File Transfer Protocol (FTP) – протокол передачи файлов по сети и HTTP), а с другой – транспортный TCP/IP. Название TCP/IP происходит из двух важнейших протоколов семейства – Transmission Control Protocol (TCP)/Internet Protocol (IP) или протокол управления передачей/интернет протокол. Для SSL шифрования используются симметричные и ассиметричные ключи, полученные с помощью различных математических моделей [1].

Далее проведем сравнительный анализ использования протоколов HTTPS и HTTP.

Для HTTPS-соединений обычно используется TCP-порт 443. HTTPS широко используется для защиты информации от перехвата, а также, как правило, обеспечивает защиту от атак вида man-in-the-middle – в том случае, если сертификат проверяется на клиенте, и при этом приватный ключ сертификата не был скомпрометирован, пользователь не подтверждал использование неподписанного сертификата, и на компьютере пользователя не были внедрены сертификаты центра сертификации злоумышленника. Атаку man-in-the-middle можно увидеть на рисунке.

На данный момент HTTPS поддерживается всеми популярными веб-браузерами.

Обязательное использование защищенного протокола передачи данных требует вся информация, касающаяся проведения платежей в интернете: оплата товаров в интернет-магазинах любым способом (индивидуальная платежная карта, онлайн системы платежей и пр.), оплата услуг через интернет-банкинг, совершение платежей в онлайн сервисах (казино, online-курсы и т.п.) и многое другое. Использовать протокол HTTPS рекомендуется также на сайтах, которые для доступа к определенному контенту запрашивают личные данные пользователей, например, номер паспорта – такие данные необходимо защищать от перехвата злоумышленниками.

Если на вашем сайте используется что-либо похожее, то вам стоит серьезно задуматься над переходом на HTTPS. Поэтому далее мы рассмотрим, что для этого необходимо.

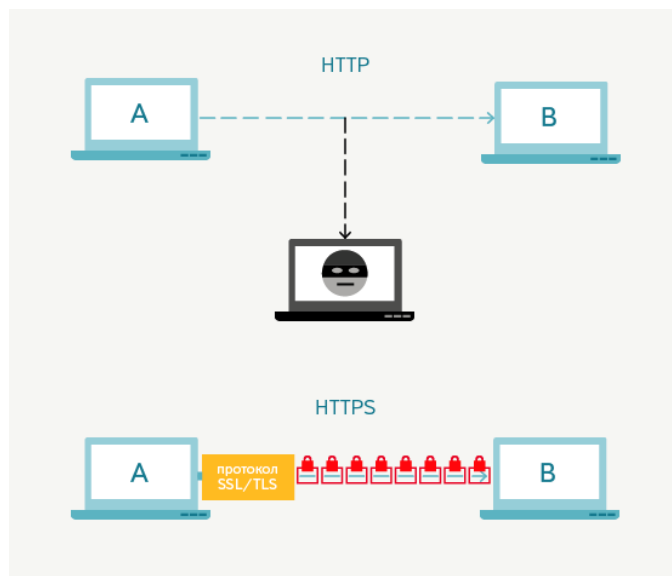


Рисунок. – Сравнение HTTP и HTTPS

Работа протокола HTTPS основана на том, что компьютер пользователя и сервер выбирают общий секретный ключ, с помощью которого и происходит шифрование передаваемой информации. Это ключ уникальный и генерируется для каждого сеанса. Считается, что его подделать невозможно, так как в нем содержится более 100 символов. Во избежание перехвата данных третьим лицом используется цифровой сертификат – это электронный документ (ЭД), который идентифицирует сервер. Каждый владелец сайта (сервера) для установки защищенного соединения с пользователем должен иметь такой сертификат. В этом ЭД указываются данные владельца и подпись. С помощью сертификата вы подтверждаете, что:

- лицо, которому он выдан, действительно существует;
- оно является владельцем сервера (сайта), который указан в сертификате.

Первое, что делает браузер при установке соединения по протоколу HTTPS, это проверку подлинности сертификата, и только в случае успешного ответа начинается обмен данными.

Подробнее остановимся на проблемах, с которыми можно столкнуться, используя HTTP.

1. Совместное использование HTTP и HTTPS. Когда сайты используют смешанный функционал HTTP и HTTPS, это потенциально приводит к информационной угрозе для пользователя. Например, если основные страницы некоторого сайта загружаются, используя HTTPS, а Cascading Style Sheets (CSS) и JavaScript загружаются по HTTP, то злоумышленник в момент загрузки последних может подгрузить свой код и получить данные HTML-страницы. Многие сайты, несмотря на такие уязвимости, загружают контент через сторонние сервисы, которые не поддерживают HTTPS. Механизм HSTS позволяет предотвратить подобные уязвимости, заставляя принудительно использовать HTTPS соединение даже там, где по умолчанию используется HTTP [2].

2. Атаки с использованием анализа трафика. В HTTPS были обнаружены уязвимости, связанные с анализом трафика. Атака с анализом трафика – это тип атаки, при которой выводятся свойства защищённых данных канала путём измерения размера трафика и времени передачи сообщений в нем. Анализ трафика возможен, поскольку шифрование SSL/TLS изменяет содержимое трафика, но оказывает минимальное влияние на размер и время прохождения трафика. В мае 2010 года исследователи из Microsoft Research и Университета Индианы обнаружили, что подробные конфиденциальные пользовательские данные могут быть получены из второстепенных данных, таких, например, как размеры пакетов. Анализатор трафика смог заполучить историю болезней, данные об использовавшихся медикаментах и проведённых операциях пользователя, данные о семейном доходе и пр. Все это было произведено несмотря на использование HTTPS в нескольких современных веб-приложениях в сфере здравоохранения, налогообложения и др. [3].

3. Человек посередине HTTPS. При атаке «человек посередине» используется то, что сервер HTTPS отправляет сертификат с открытым ключом в браузер. Если этот сертификат не заслуживает доверия, то канал передачи будет уязвимым к атаке злоумышленника. Такая атака заменяет оригинальный сертификат, удостоверяющий HTTPS-сервер, модифицированным сертификатом. Атака проходит успешно, если пользователь пренебрегает двойной проверкой сертификата, когда браузер отправляет

предупреждение. Это особенно распространено среди пользователей, которые часто сталкиваются с само заверенными сертификатами при доступе к сайтам внутри сети частных организаций.

В результате исследования можно сделать вывод о том, что протокол HTTPS достаточно гибок и удобен в использовании, он обеспечивает конфиденциальность обмена данными между сайтом и пользовательским устройством. Благодаря протоколу HTTPS стало возможным сохранять в тайне конфиденциальную информацию. Для организации общения клиента с сервером HTTPS использует для шифрования протокола SSL/TLS с достаточно высоким уровнем безопасности.

ЛИТЕРАТУРА

1. Как работает SSL? Принцип работы https соединения [Электронный ресурс]. – Режим доступа: <https://www.ipipe.ru/info/kak-rabotaet-ssl-sertificat.html>. – Дата доступа: 12.09.2018.
2. How to Deploy HTTPS Correctly [Электронный ресурс]. – Режим доступа: <https://www.eff.org/https-everywhere/deploying-https>. – Дата доступа: 13.09.2018.
3. Side-Channel Leaks in Web Applications: a Reality Today, a Challenge Tomorrow [Электронный ресурс]. – Режим доступа: <https://www.microsoft.com/en-us/research/publication/side-channel-leaks-in-web-applications-a-reality-to-ay-a-challenge-tomorrow> – Дата доступа: 12.09.2018.

УДК 004.021

**ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ
ДЛЯ МУЗЫКАЛЬНОГО СТРИМИНГОВОГО ВЕБ-СЕРВИСА «BLZMUSIC»****Д.С. ТАТАРИН***(Представлено: канд. тех. наук И.Б. БУРАЧЁНОК)*

Рассматриваются цели и принципы моделирования базы данных на примере музыкального стримингового веб-сервиса «BlzMusic». В спроектированной базе данных обосновывается выбор основных сущностей и связей между ними.

Трудно представить хороший сайт с большой аудиторией без разработанной базы данных. Они позволяют удобно хранить большие объемы информации не на устройстве пользователя, а на удаленном сервере, который может эти данные предоставить по запросу от приложения.

В представленной статье приведем описание базы данных для разработанного веб-сервиса «BlzMusic» и особенностей ее проектирования.

Цель моделирования данных состоит в обеспечении разработчика концептуальной схемой базы данных в форме одной модели или нескольких локальных моделей, которые относительно легко могут быть отображены в любую систему баз данных. Наиболее распространенным средством моделирования данных являются диаграммы «сущность-связь» или Entity-Relationship Model (ERD). С их помощью определяются важные для предметной области объекты (сущности), их свойства (атрибуты) и отношения друг с другом (связи). ERD непосредственно используются для проектирования реляционных баз данных. Нотация ERD была впервые введена П. Ченом и получила дальнейшее развитие в работах Баркера. Методология Integration Definition For Information Modeling (IDEF1), разработанная Т. Рэмей, основана на подходе П. Чена и позволяет построить модель данных, эквивалентную реляционной модели в третьей нормальной форме [1].

Сущность – это реальный или представляемый тип объекта, информация о котором должна сохраняться и быть доступной. В диаграммах сущность представляется в виде прямоугольника, содержащего имя сущности. При этом имя сущности – это имя типа, а не некоторого конкретного экземпляра этого типа. Каждый экземпляр сущности (объект) должен быть отличим от любого другого экземпляра той же сущности [2].

Для реализации музыкального стримингового веб-сервиса «BlzMusic» в качестве базы данных был выбран MS SQL Server 2017, это обусловлено следующим.

- Масштабируемость. Алгоритмы использования дискового пространства SQL Server 7.0 масштабируются в диапазоне от портативных компьютеров начиная с Windows версии 3.11.
- Превосходная производительность. Увеличенный до размер страниц способствует быстрому извлечению данных, позволяет использовать строки и столбцы большего размера, что открывает возможность эффективного хранения сложных, подробных данных.
- Простота использования. Динамическое самоадминистрирование. Выполнение многих рутинных задач администрирования теперь автоматизировано. Алгоритмы управления памятью и блокированием адаптируются динамически, размер файлов автоматически увеличивается и сокращается. Кроме того, средства автоматической настройки динамически настраивают алгоритмы использования ресурсов в зависимости от рабочей нагрузки.

Для построения самого приложения был выбран Java – объектно-ориентированный язык программирования, разработанный компанией Sun Microsystems (в последующем приобретённой компанией Oracle). Приложения Java обычно компилируются в специальный байт-код, поэтому они могут работать на любой виртуальной Java-машине (JVM) вне зависимости от компьютерной архитектуры. Программы на Java транслируются в байт-код, выполняемый виртуальной машиной Java (JVM) – программой, обрабатывающей байтовый код и передающей инструкции оборудованию как интерпретатор. Достоинство подобного способа выполнения программ – в полной независимости байт-кода от операционной системы и оборудования, что позволяет выполнять Java-приложения на любом устройстве, для которого существует соответствующая виртуальная машина [3]. Java является достаточно мощным инструментом разработки приложений, к тому же он имеет огромное комьюнити и большим количеством учебных курсов.

Для разработки интерфейса было решено использовать: html, css, js, что позволит создать приемлемый пользовательский интерфейс, который облегчит работу пользователю, который будет работать с приложением. Более того на данный момент сайты набирают все большую и большую популярность, потому что намного проще «зайти» на сайт, чем скачивать и ставить какое-либо приложение.

На первоначальном этапе разработки, с целью построения базы данных, был проведен анализ предметной области и выявление необходимого набора сущностей. База данных предназначена для хранения данных о музыке (исполняемых музыкальных произведениях), группах, артистах, концертах, жанрах, альбомах, пользователях. Для каждого трека и альбома должна присутствовать таблица с принадлежностью к какой-либо группе или артисту. У каждой группы должен присутствовать один или несколько жанров. Для каждого концерта должно быть указано к какой группе он относится. Также должна присутствовать таблица, где хранится логин и пароль пользователя и связь с его ролью, логином и корзиной. Следовательно, определяем роль, корзину и логин как отдельные сущности.

В результате краткого анализа предметной области выявлен необходимый набор сущностей:

- Music (Музыка);
- Band (Группа);
- Concert (Концерты группы);
- Genre (Жанры групп);
- Album (Альбом групп);
- Cart (Корзина пользователя);
- User (Пользователь);
- Login (Информация о логине пользователя);
- Role (Роль пользователя).

Связь – это графически изображаемая ассоциация, устанавливаемая между двумя сущностями. Связь может существовать между двумя разными сущностями или между сущностью и ей же самой (рекурсивная связь) [2].

Для реализации приложения, необходимо установить все связи между сущностями: необходимо рассмотреть всю информационную систему в совокупности и определить взаимное влияние сущностей.

Полученная схема базы данных представлена на рисунке.

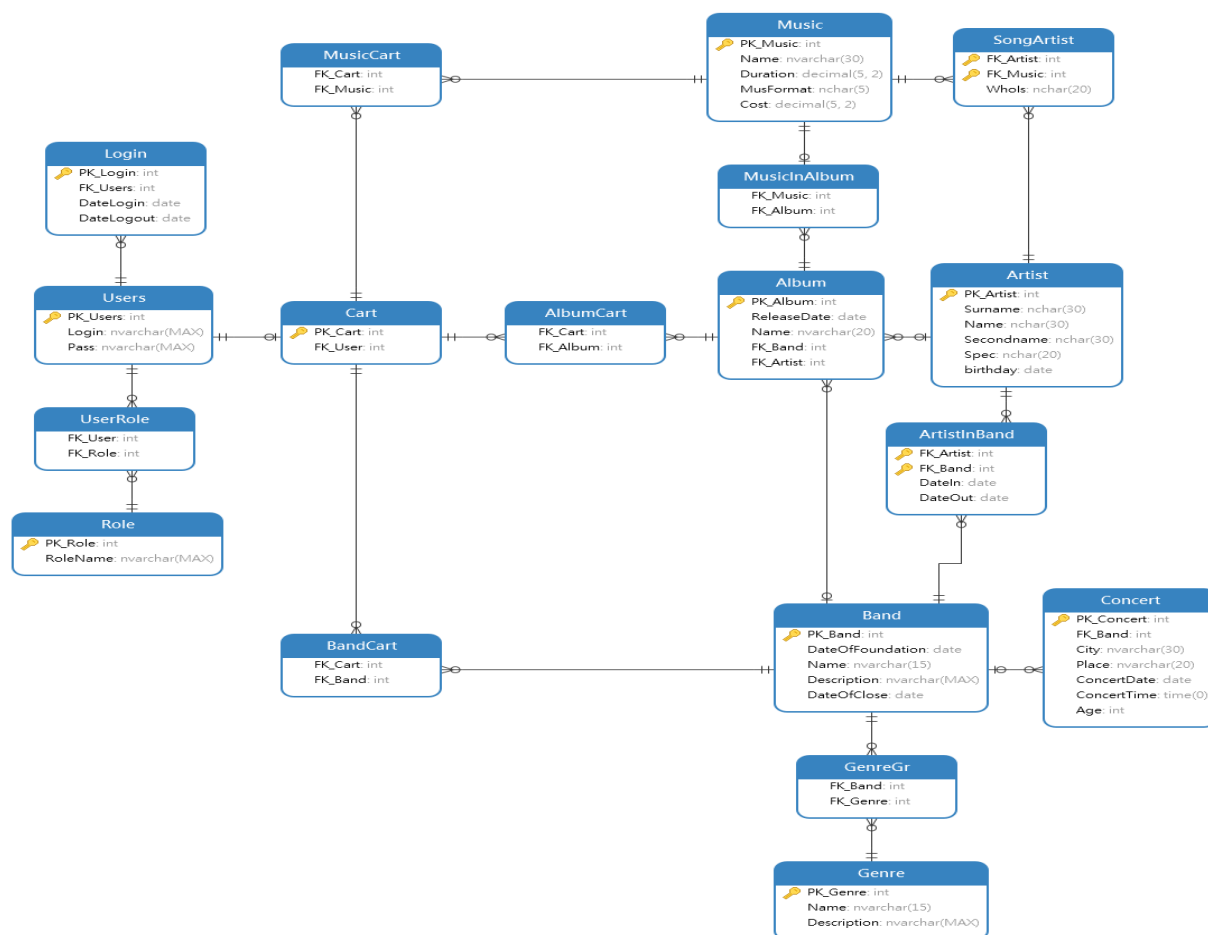


Рисунок. – Схема базы данных

Реляционная модель представляет базу данных в виде множества взаимосвязанных отношений. Отношения представлены в виде таблиц, строки которых соответствуют кортежам или записям, а столбцы – атрибутам отношений, доменам, полям. В соответствии выделенными выше сущностями, определен набор необходимых таблиц базы данных, представленный в таблице 1.

Таблица 1. – Перечень таблиц базы данных

Название таблицы	Описание
Music	Хранение данных о всех треках
MusicCart	Хранение данных о всех треках в корзине пользователя
MusicInAlbum	Хранение данных о треках состоящий в альбоме
Artist	Хранение данных о всех артистах
SongArtist	Хранение данных о треках артистов
ArtistInBand	Хранение данных о артистах в группе
Album	Хранение данных о всех альбомах
AlbumCart	Хранение данных о альбомах в корзине пользователя
Band	Хранение данных о всех группах
BandCart	Хранение данных о группах в корзине пользователя
Concert	Хранение данных о концертах группы
Genre	Хранение данных о жанрах
GenreGr	Хранение данных о жанрах группы
Cart	Хранение данных о корзине пользователя
Users	Хранение данных о пользователях
Login	Хранение данных о логине пользователя
Role	Хранение данных о роли
UserRole	Хранение данных о роли пользователя

Важное достоинство баз данных – возможность хранить сгруппированные данные в разных таблицах и устанавливать связи между этими таблицами посредством ключей и последующее объединение этих данных в единую базу. Это позволяет снизить избыточность данных, которые хранятся в базе данных, упростить организацию запросов для их извлечения.

Ключ сущности – первичный ключ – это атрибут (или множество атрибутов) уникальным образом идентифицирующих экземпляр сущности (объект). Например, ключ сущности Band – уникальный номер, ключ Music – уникальный код песни. Если ключ состоит из одного атрибута, его называют простым ключом. Если ключ сущности состоит из нескольких атрибутов, его называют составным ключом.

Для организации связей между двумя и более таблицами, используются вторичные ключи. Они служат ограничителями целостности связей нескольких таблиц, т.к. подчиненная таблица не может ссылаться на несуществующие записи главной таблицы (что позволяет строить целостные модели данных). Существует три вида связей между таблицами:

- один-к-одному;
- один-ко-многим;
- многие-ко-многим.

Отношение «один-ко-многим» имеет место, когда одной записи родительской таблицы может соответствовать несколько записей в дочерней таблице [4]. Связь «один-ко-многим» является самой распространенной для реляционных баз данных. Отношение «один-к-одному» имеет место, когда одной записи в родительской таблице соответствует одна запись в дочерней таблице [4]. Отношение «многие-ко-многим» имеет место, когда [4]:

- записи в родительской таблице может соответствовать больше одной записи в дочерней таблице;
- записи в дочерней таблице может соответствовать больше одной записи в родительской таблице.

Например, каждая группа имеет несколько жанров. К каждому жанру относятся несколько групп (табл. 2).

Таблица 2. – Первичные и внешние ключи отношений

Название таблицы	Первичный ключ	Внешние ключи
Album	PK_Album	FK_Band
Music	PK_Music	
Artist	PK_Artist	
Band	PK_Band	
Concert	PK_Concert	FK_Band
Genre	PK_Genre	FK_Subtype
Cart	PK_Cart	FK_User
User	PK_User	
Login	PK_Login	FK_User
Role	PK_Role	

Таким образом была спроектирована база данных музыкального стримингового веб-сервиса «BlzMusic». Были выделены 18 таблиц, которые находятся в ЗНФ. Написаны скрипты создания и наполнения базы данных, поэтому базу можно легко развернуть на любой СУБД.

ЛИТЕРАТУРА

1. Case-метод Баркера [Электронный ресурс]. – Режим доступа: http://citforum.ru/database/case/glava2_4_1.shtml. – Дата доступа: 14.09.2018.
2. Основные понятия баз данных [Электронный ресурс]. – Режим доступа: http://inf.susu.ac.ru/Klinachev/lc_sga_26.htm. – Дата доступа: 14.09.2018.
3. Основные понятия БД [Электронный ресурс]. – Режим доступа: <http://informatic.ugatu.ac.ru/lib/office/proekt.htm>. – Дата доступа: 14.09.2018.
4. Подробнее о технологии Java [Электронный ресурс]. – Режим доступа: <https://java.com/ru/about>. – Дата доступа: 14.09.2018.

УДК 004.021

ОСОБЕННОСТИ СОЗДАНИЯ ГРАФИЧЕСКИХ ИНТЕРФЕЙСОВ С ПОМОЩЬЮ ТЕХНОЛОГИЙ WINDOWS FORMS

М.А. ТРАЩЕНКО

(Представлено: канд. техн. наук, доц. И.Б. БУРАЧЁНОК)

Рассматриваются принципы построения графического интерфейса пользователя. Рассмотрены положительные стороны использования Windows Forms, его возможности для быстрого и гибкого в разработке интерфейса. Разобраны часто используемые при реализации графических интерфейсов свойства графических элементов.

Для создания графических интерфейсов с помощью платформы .NET применяются разные технологии – Window Forms, WPF, приложения для магазина Windows Store (для ОС Windows 8/8.1/10). Однако наиболее простой и удобной платформой до сих пор остается Window Forms или просто формы.

В представленной статье подробно остановимся на описании создания современных графических интерфейсов с помощью технологий Windows Forms.

Приложение Windows Forms представляет собой событийно-ориентированное приложение, поддерживаемое Microsoft .NET Framework. В отличие от пакетных программ, при работе приложения большая часть времени тратится на ожидание от пользователя каких-либо действий, как, например, ввод текста в текстовое поле или клика мышкой по кнопке.

Форма – это панель, на которой выводится информация для пользователя. Обычно приложение Windows Forms строится путем помещения элементов управления на форму и написания кода для реагирования на действия пользователя, такие как щелчки мыши или нажатия клавиш. Элемент управления – это отдельный элемент пользовательского интерфейса, предназначенный для отображения или ввода данных. При выполнении пользователем какого-либо действия с формой или одним из ее элементов управления создается событие. Приложение реагирует на эти события с помощью кода и обрабатывает события при их возникновении.

Большим недостатком Windows Forms является то, что командам программистов и дизайнеров приходится работать очень тесно, чтобы получился достойный проект. То есть, дизайнер рисует интерфейс, а программист его реализует. Причем программисту приходится постоянно отвлекаться от своей непосредственной задачи – написания логики программы. Он то и дело, подгоняет кнопки под размеры формы, вставляет картинки и т.п. Чтобы избавиться от этого недостатка Windows Forms, в .NET Framework включена технология Windows Presentation Foundation (WPF), являющаяся большим шагом в сторону улучшения разработки интерфейсов. По сравнению с Windows Forms – было сделано следующее: программист полностью погружается в разработку логики программы, а дизайнер может сразу же создавать дизайн программы, практически не зависимо от программиста.

Популярность Windows Forms постепенно падает, однако ее продолжают использовать в простых, не требующих особых представлений, интерфейсах, программах. Также во всех версиях .NET Framework существует поддержка Windows Forms, куда включены некоторые дополнения и улучшения.

Несмотря на то, что в Windows Forms весь интерфейс можно построить только с использованием мыши, программный код разметки также доступен для программиста, и при желании, может быть изменен.

Пример кода дизайнера Windows Forms представлен в листинге.

```
namespace HelloApp
{
    partial class Form1
    {
        private System.ComponentModel.IContainer components = null;
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
        }
    }
}
```

```

        base.Dispose(disposing);
    }
    #region Windows Form Designer generated code
    private void InitializeComponent()
    {
        this.SuspendLayout();
        this.AutoScaleDimensions = new System.Drawing.SizeF(6F,
13F);
        this.AutoScaleMode = Sys-
tem.Windows.Forms.AutoScaleMode.Font;
        this.ClientSize = new System.Drawing.Size(284, 261);
        this.Name = "Form1";
        this.Text = "Привет мир!";
        this.ResumeLayout(false);
    }
    #endregion
}
}

```

Листинг. – Пример кода дизайнера Windows Forms

С помощью специального окна Properties, представленного на рисунке, среда разработки Visual Studio предоставляет нам удобный интерфейс для управления свойствами элемента.

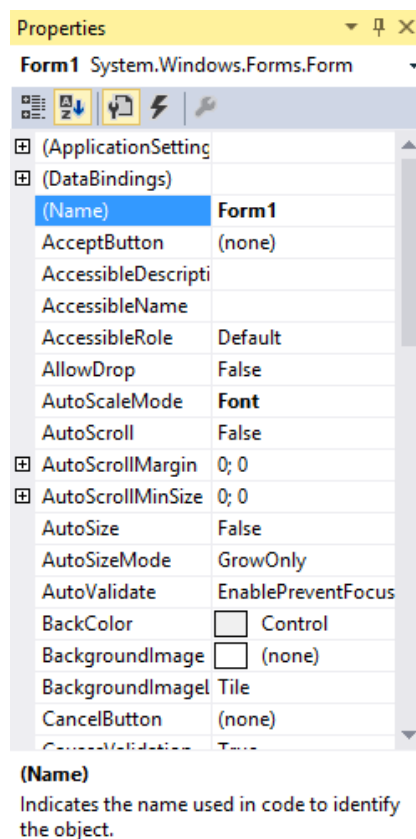


Рисунок. – Окно свойств элемента

Большинство этих свойств оказывает влияние на визуальное отображение формы. Разберем основные свойства:

- Name: устанавливает имя формы – точнее имя класса, который наследуется от класса Form;
- BackColor: указывает на фоновый цвет формы. Щелкнув на это свойство, мы сможем выбрать тот цвет, который нам подходит из списка предложенных цветов или цветовой палитры;

- `BackgroundImage`: указывает на фоновое изображение формы;
- `BackgroundImageLayout`: определяет, как изображение, заданное в свойстве `BackgroundImage`, будет располагаться на форме;
- `ControlBox`: указывает, отображается ли меню формы;
- `Cursor`: определяет тип курсора, который используется на форме;
- `Enabled`: если данное свойство имеет значение `false`, то она не сможет получать ввод от пользователя, то есть мы не сможем нажать на кнопки, ввести текст в текстовые поля и т.д.;
- `Font`: задает шрифт для всей формы и всех помещенных на нее элементов управления. Однако, задав у элементов формы свой шрифт, мы можем тем самым переопределить его;
- `ForeColor`: цвет шрифта на форме;
- `FormBorderStyle`: указывает, как будет отображаться граница формы и строка заголовка. Устанавливая данное свойство в `None` можно создавать внешний вид приложения произвольной формы;
- `HelpButton`: указывает, отображается ли кнопка справки формы;
- `Icon`: задает иконку формы;
- `Location`: определяет положение по отношению к верхнему левому углу экрана, если для свойства `StartPosition` установлено значение `Manual`;
- `MaximizeBox`: указывает, будет ли доступна кнопка максимизации окна в заголовке формы;
- `MinimizeBox`: указывает, будет ли доступна кнопка минимизации окна;
- `MaximumSize`: задает максимальный размер формы;
- `MinimumSize`: задает минимальный размер формы;
- `Opacity`: задает прозрачность формы;
- `Size`: определяет начальный размер формы;
- `StartPosition`: указывает на начальную позицию, с которой форма появляется на экране;
- `Text`: определяет заголовок формы;
- `TopMost`: если данное свойство имеет значение `true`, то форма всегда будет находиться поверх других окон;
- `Visible`: видима ли форма, если мы хотим скрыть форму от пользователя, то можем задать данному свойству значение `false`;
- `WindowState`: указывает, в каком состоянии форма будет находиться при запуске: в нормальном, максимизированном или минимизированном.

Таким образом, принято решение при разработке графического интерфейса для автоматизированной информационной системы учета товаров с целью продвижения малого бизнеса компании «GRANDDECOR» использовать технологии Windows Forms, что позволит сократить временные затраты на разработку интерфейса и уделить больше внимания логике автоматизированной информационной системы.

ЛИТЕРАТУРА

1. Metanit [Электронный ресурс] // Руководство по программированию в Windows Forms. – Режим доступа: <https://metanit.com/sharp/windowsforms/>. – Дата доступа: 13.09.18.
2. CitForum [Электронный ресурс] // Пошаговые руководства по Windows Forms. – Режим доступа: [https://msdn.microsoft.com/ru-ru/library/zftbwa2b\(v=vs.110\).aspx/](https://msdn.microsoft.com/ru-ru/library/zftbwa2b(v=vs.110).aspx/). – Дата доступа: 14.09.18.

УДК 004.021

**ПРИНЦИПЫ МОДЕЛИРОВАНИЯ БАЗЫ ДАННЫХ НА ПРИМЕРЕ
АВТОМАТИЗИРОВАННОЙ ИНФОРМАЦИОННОЙ СИСТЕМЫ УЧЕТА ТОВАРОВ****М.А. ТРАЩЕНКО***(Представлено: канд. техн. наук, доц. И.Б. БУРАЧЁНОК)*

Рассматриваются основные принципы моделирования базы данных на примере автоматизированной информационной системы под операционную систему Windows учета товаров для продвижения малого бизнеса компании «GrandDecor».

Трудно представить хорошее клиент-серверное приложение без разработанной базы данных. Она позволяет удобно хранить большие объемы информации не на устройстве пользователя, а на удаленном сервере, который может эти данные предоставить по запросу от приложения.

В представленной статье опишем методы проектирования базы данных на примере автоматизированной информационной системы (АИС) под операционную систему Windows учета товаров для продвижения малого бизнеса компании «GrandDecor».

Цель моделирования данных состоит в обеспечении разработчика концептуальной схемой базы данных в форме одной модели или нескольких локальных моделей, которые относительно легко могут быть отображены в любой системе баз данных.

Сущность структурного подхода к разработке АИС заключается в ее декомпозиции (разбиении) на автоматизируемые функции: система разбивается на функциональные подсистемы, которые в свою очередь делятся на подфункции, подразделяемые на задачи и так далее. Процесс разбиения продолжается вплоть до конкретных процедур. При этом автоматизируемая система сохраняет целостное представление, в котором все составляющие компоненты взаимосвязаны. Общая модель системы строится в виде некоторой иерархической структуры, которая отражает различные уровни абстракции с ограниченным числом компонентов на каждом из уровней. Одним из главных принципов структурного системного анализа является выделение на каждом из уровней абстракции только наиболее существенных компонентов или элементов системы [1].

В структурном анализе используется ряд средств, иллюстрирующих функции, выполняемые системой и отношения между данными. Каждой группе средств соответствуют определенные виды моделей (диаграмм). Наиболее распространенной диаграммой, предназначенной для моделирования данных, является диаграмма Entity-Relationship Diagrams (ERD) «сущность-связь».

В результате краткого анализа предметной области выявлен необходимый набор сущностей (таблиц), представленных в таблице.

Таблица. – Список всех таблиц, образующих базу данных

Наименование таблицы	Назначение
Брак	Хранит информацию о товарах, на которых выявлен брак
Категория товара	Хранит информацию о категориях товара, реализуемых на торговых точках
Номенклатура товара	Хранит название, идентификационный номер и категорию товара
Позиция ТТН	Хранит информацию о товаре, оформленном с номером данной ТТН
Поставщики	Хранит информацию о поставщиках товаров
Продажа	Хранит информацию о продажах товара
Производитель	Хранит условно-постоянную информацию (Справочник подстатей расходов)
Склад	Хранит информацию о производителях товара, поставляемого поставщиками
Тип поставщика	Хранит информацию о категории товара, который предоставляет данный поставщик
Товар	Хранит полную информацию о товаре
ТТН	Хранит информацию об оформленной ТТН

Базы данных могут содержать таблицы, которые связаны между собой различными связями. Связь (relationship) представляет ассоциацию между сущностями разных типов. При выделении связи выделяют главную или родительскую таблицу (primary key table / master table) и зависимую, дочернюю таблицу (foreign key table / child table). Дочерняя таблица зависит от родительской. Для организации связи используются внешние ключи. Внешний ключ представляет один или несколько столбцов из одной таблицы, который одновременно является потенциальным ключом из другой таблицы. Внешний ключ необязательно должен соответствовать первичному ключу из главной таблицы. Хотя, как правило, внешний ключ из зависимой таблицы указывает на первичный ключ из главной таблицы.

Реляционная схема базы данных представлена на рисунке.

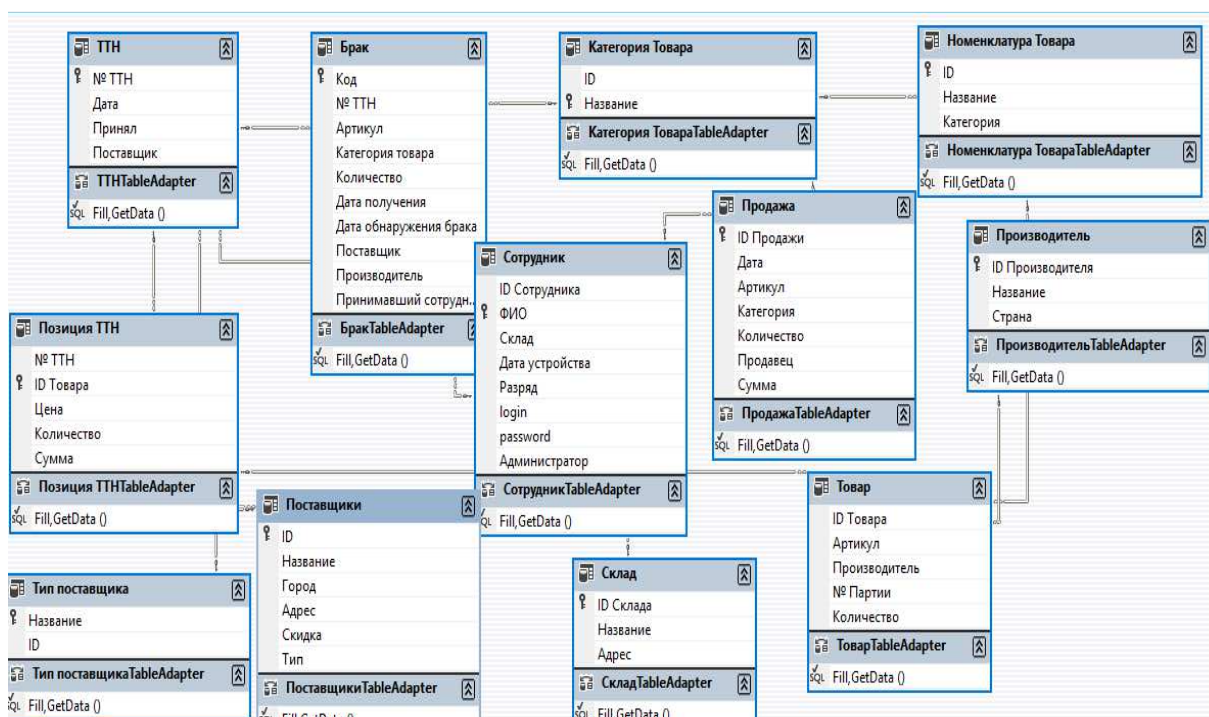


Рисунок. – Реляционная схема базы данных

Реляционная модель представляет базу данных в виде множества взаимосвязанных отношений (таблиц), которые используются для хранения информации об объектах, представленных в базе данных.

Важное достоинство баз данных – возможность хранить сгруппированные данные в разных таблицах и устанавливать связи между этими таблицами посредством ключей и последующее объединение этих данных в единую базу. Это позволяет снизить избыточность данных, которые хранятся, упростить организацию запросов для их получения.

Первичный ключ служит ограничителем целостности в рамках одной таблицы для идентификации, т.к. поле первичного ключа не может повторяться или быть пустым. Важное условие - каждая таблица может содержать только один первичный ключ.

Для организации связей между двумя и более таблицами, используются вторичные ключи. Они служат ограничителями целостности связей нескольких таблиц, т.к. подчиненная таблица не может ссылаться на несуществующие записи главной таблицы (что позволяет строить целостные модели данных).

Связи между таблицами бывают следующих типов:

1. Один к одному (One to one);
2. Один к многим (One to many);
3. Многие ко многим (Many to many).

Связь один к одному. Данный тип связи встречается не часто. В этом случае объекту одной сущности можно сопоставить только один объект другой сущности. Например, на некоторых сайтах пользователь может иметь только один блог. То есть возникает отношение один пользователь – один блог.

Связь один ко многим. Это наиболее часто встречаемый тип связи. В этом типе связи несколько строк, из дочерней таблицы, зависят от одной строки в родительской таблице. Например, в одном блоге

может быть несколько статей. В этом случае таблица блогов является родительской, а таблица статей – дочерней. То есть один блог – много статей.

Связь многие ко многим. При этом типе связей одна строка из таблицы А может быть связана с множеством строк из таблицы В. В свою очередь одна строка из таблицы В может быть связана с множеством строк из таблицы А. Типичный пример – студенты и курсы: один студент может посещать несколько курсов, и соответственно на один курс могут записаться несколько студентов [2].

Таким образом, рассмотренные варианты проектирования, несомненно, являются актуальными и, при верном подходе к выбору, позволяют произвести проектирование базы данных.

ЛИТЕРАТУРА

1. Helpiks [Электронный ресурс] // Моделирование данных. – Режим доступа: <https://helpiks.org/6-43047.html>. – Дата доступа: 13.09.17.
2. Metanit [Электронный ресурс] // Реляционные базы данных. – Режим доступа: <https://metanit.com/sql/tutorial/1.3.php>. – Дата доступа: 14.09.18.

УДК 004.021

**ПРОЕКТИРОВАНИЕ ИНТЕРФЕЙСА
АВТОМАТИЗИРОВАННОЙ ИНФОРМАЦИОННОЙ СИСТЕМЫ
«КОМИССИОННЫЙ МАГАЗИН» НА ПЛАТФОРМЕ 1С:ПРЕДПРИЯТИЕ 8.3**

В.А. ГОНЧАРОВ

(Представлено: канд. техн. наук, доц. А.Ф. ОСЬКИН)

Рассматриваются принципы проектирования пользовательского интерфейса в автоматизированной информационной системе «Комиссионный магазин» на платформе 1С:Предприятие 8.3.

В каждой автоматизированной информационной системе есть свои принципы проектирования интерфейса. 1С:Предприятие обладает рядом преимуществ построения прикладных решений, главным из которых является возможность приспосабливаться к особенностям конкретной области деятельности, т.е. конфигурируемость – возможность настройки системы на особенности конкретного предприятия и класса решаемых задач, в которой она применяется. Данная статья направлена на подробное описание технологий построения пользовательского интерфейса на платформе 1С.

Интерфейс пользователя – это все окна, меню, кнопки и прочее, с чем пользователь работает непосредственно в программе [1]. Запуск конфигурации в управляемом режиме начинается с рабочего стола. На рабочем столе должна быть представлена информация, актуальная для пользователя – задачи пользователя, текущие операции и т.д. Рабочий стол имеет возможности настройки, не прибегая к средствам конфигурирования. Для создания и редактирования форм объектов прикладного решения используется редактор формы. Формы объектов используются системой для визуального отображения данных в процессе работы пользователя. Параметры формы редактируются в списке. Разработчик имеет возможность добавить, удалить параметры формы и задать их свойства с помощью палитры свойств. Для редактирования модуля формы конфигуратор вызывает редактор текстов и модулей. Этот редактор предоставляет разработчику большое разнообразие возможностей для создания и изменения текста модуля [2].

Данная платформа имеет свои плюсы:

- «Трёхзвенка» – распределение нагрузки.
- Открытость базы данных.
- Отсутствие ограничений.
- Универсальность кода конфигураций.
- Платформа устанавливается очень просто.

Однако, есть и минусы:

– Цена: покупать придется отдельно клиентские лицензии на каждый клиентский компьютер, отдельно – одну лицензию на сервер 1С, отдельно лицензию на сервер SQL, если не использовать бесплатный

– Также иногда возникают сложности в работе БД. Они решаются гораздо быстрее за счет доступности данных и наличия внешних инструментов работы с базой данных [1]

Главное меню приложения изображено на рисунке 1.

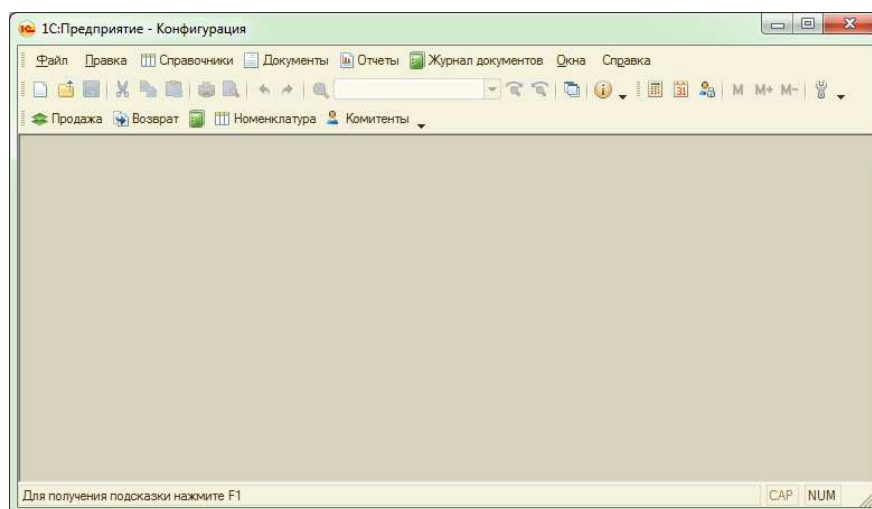


Рисунок 1. – Главное меню приложения

Как видно из рисунка 1, все главные модули находятся на экране, в том числе настройки и действия. В данном меню можно получить доступ ко всем компонентам: справочникам, документам, отчетам и др. Работа со справочниками заключается в сохранении данных для документов. Также, в реквизитах (комитент, подразделение и комиссионер) имеется 2 способа заполнения: ручной ввод или выбор из соответствующего справочника. Если в справочнике ещё нет нужного элемента – необходимо добавить его. Работа с документами заключается в создании отчётности и выводе на печать необходимого перечня документов.

При запуске программы в фоновом режиме выполняется автоматическая проверка документов на необходимость переоценки или возврата товаров. В константах хранятся сроки, по истечению которых необходимо выполнять переоценку или возврат. При необходимости уценки товара создается документ «Уценка товара», который заполняется данными из документа прихода товара, являющегося документом-основанием для уценки. В панели «Служебные сообщения» выводится уведомление пользователю о каждой проведенной уценке. Для продажи товаров создается документ «Продажа товара». В окне документа заполняются обязательные поля и указываются товары, которые будут проданы. После проведения данные попадают в регистр накопления «Партии товаров» и регистр сведений «Продажи». Для необходимой наглядности и удобства восприятия напечатанных документов в модулях увеличивается программный код с необходимым распределением данных на печати.

Конфигурация пользовательских интерфейсов изображена на рисунках 2–4.

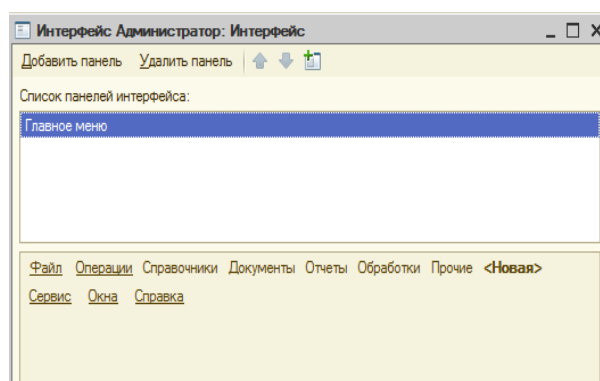


Рисунок 2. – Интерфейс администратора

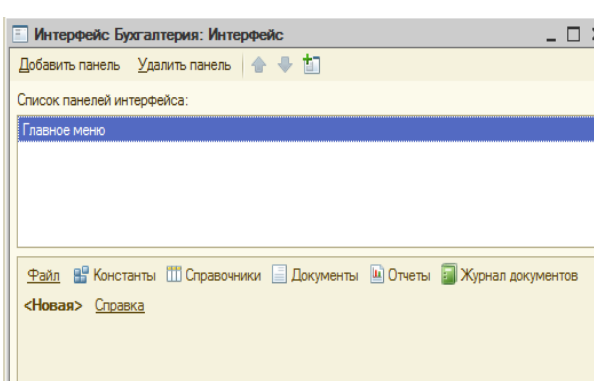


Рисунок 3. – Интерфейс бухгалтера

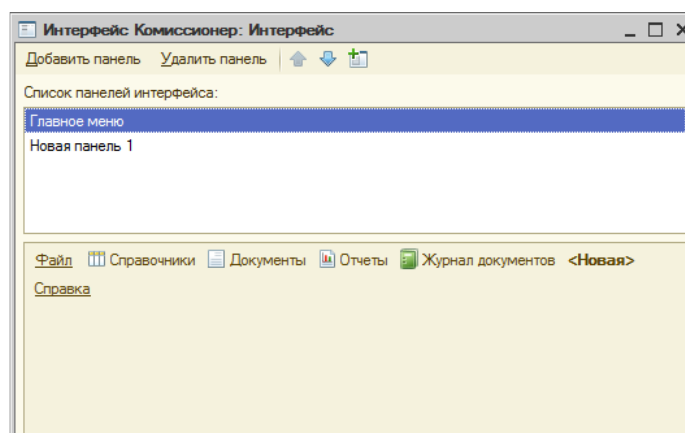


Рисунок 4. – Интерфейс комиссионера

Таким образом, рассмотрено построение интерфейса пользователя на платформе 1С:Предприятие 8.3, а также их плюсы и минусы. Представлены пользовательские интерфейсы работы с системой.

ЛИТЕРАТУРА

1. 1с-uroki [Электронный ресурс] // Бесплатные онлайн уроки 1С:Предприятие 8.3 и 8.2. – Режим доступа: http://1c-uroki.ru/kurs_1S_8.3.5/1_urok_po_platforme_1S_8.3. – Дата доступа: 10.09.18.
2. Forum-1c [Электронный ресурс] // Форум 1С. Задай вопрос – получи решение проблемы. – Режим доступа: <http://forum-1c.com.> – Дата доступа: 11.09.18.

УДК 004.021

**ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ
АВТОМАТИЗИРОВАННОЙ ИНФОРМАЦИОННОЙ СИСТЕМЫ
«КОМИССИОННЫЙ МАГАЗИН» НА ПЛАТФОРМЕ 1С:ПРЕДПРИЯТИЕ 8.3**

В.А. ГОНЧАРОВ

(Представлено: канд. техн. наук, доц. А.Ф. ОСЬКИН)

Рассматриваются цели и принципы моделирования базы данных для автоматизированной информационной системы «Комиссионный магазин» на платформе 1С:Предприятие 8.3.

Трудно представить хорошую автоматизированную информационную систему без разработанной базы данных. Они позволяют удобно хранить большие объемы информации на любом устройстве пользователя.

Данная статья направлена на описание базы данных для разработанной автоматизированной информационной системы и методов ее проектирования.

Цель моделирования данных состоит в обеспечении разработчика концептуальной схемой базы данных в форме одной модели или нескольких локальных моделей, которые относительно легко могут быть отображены в любую систему баз данных.

Наиболее распространенным средством моделирования данных является функциональная структура. С ее помощью определяются важные для предметной области модули (сущности) и их набор функций.

В результате краткого анализа предметной области выявлен необходимый набор модулей:

- Модуль прихода товаров.
- Модуль продажи товаров.
- Модуль уценки товаров.
- Модуль возврата товаров.
- Модуль формирования отчетов.
- Модуль накопления и печати документации.
- Модуль справочников.

Модуль – некий текстовый файл, который содержит программный код. Таким образом в 1С нет «единой» программы. Есть набор модулей для написания программного кода для каждого объекта конфигурации 1С. Модуль состоит из набора функций и процедур – т.е. обработчиков событий и методов [1]. Для реализации автоматизированной информационной системы необходимо установить все связи между модулями: необходимо рассмотреть всю информационную систему в совокупности и определить взаимное влияние модулей.

Структура системы представлена на рисунке.

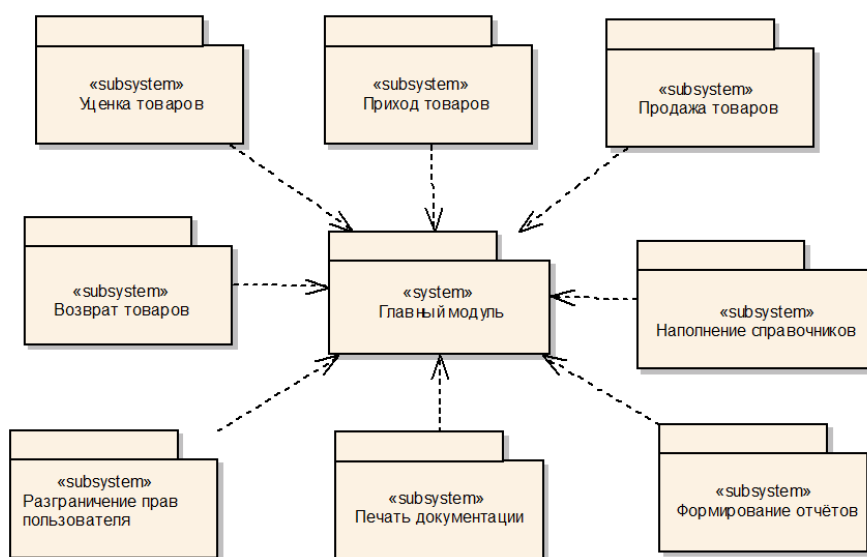


Рисунок. – Функциональная структура базы данных

Базы данных используются для решения аналитических задач, таких как финансовое моделирование и управление производительностью. База данных объединяет данные из нескольких источников и связывает наборы данных в единые модели.

В соответствии выделенными выше модулями, определён набор необходимых объектов базы данных, представленный в таблице 1.

Таблица. – Перечень объектов базы данных

№ п/п	Название объекта	Описание
	Справочник «Комитенты»	Информация о комитентах
	Справочник «Вид документа»	Список видов удостоверяющих документов
	Справочник «Подразделение»	Список всех подразделений организации
	Справочник «Комиссионеры»	Список комиссионеров
	Справочник «Номенклатура»	Информация о товаре
	Справочник «Единицы измерения»	Список используемых единиц измерения
	Справочник «Должности»	Список возможных должностей сотрудников
	Справочник «Сотрудники»	Полная информация о сотрудниках
	Справочник «Пользователи системы»	Список пользователей информационной системы
	Справочник «Организация»	Информация об организациях
	Справочник «Кассы организации»	Список перечня касс
	Справочник «Причины возврата»	Список причин возврата товаров
	Регистр сведений «Контактные лица организации»	Вносит изменения и хранит информацию об изменении должностей сотрудников.
	Регистр сведений «Цены продаж»	Информация об изменении цен товаров
	Регистр накопления «Партии товаров»	Список всех проводок, связанных с товаром
	Регистр накопления «Продажи»	Список оборотов товаров в магазине
	Документ «Приход товара»	Информация о поступлении товара
	Документ «Уценка товара»	Информация об уценённом товаре
	Документ «Возврат товара»	Информация о возвращаемых товарах
	Документ «Продажа товара»	Информация о проданных товарах
	Отчёт «Продажа по кассе»	Суммарную информацию о приходах и расходах
	Отчёт «Товарный отчёт»	Сведения об остатках, приходе и расходе товара
	Отчёт «Отчёт по приходным документам»	Информация о приходных документах
	Отчёт «Остатки товаров»	Информация о текущих остатках товаров
	Отчёт «Изменение цен на товары»	Информация об изменении цен на товар
	Отчёт «Отчёт по продажам»	Информация о продажах товаров
	Отчёт «Отчёт по продажам комитенту»	Информация о продажах товаров (комитента)
	Отчёт «Отчёт по возвратам»	Информация о возвратах товаров
	Отчёт «Счёт комитенту»	Информация о проданных товарах и общая сумма, необходимая для выплаты комитенту
	Журнал документов «Журнал документов»	Список документов: «Приход товара», «Продажа товара», «Уценка товара» и «Возврат товара»

Важнейшим достоинством применения баз данных в информационных системах является обеспечение независимости данных от прикладных программ [2]. Это позволяет снизить избыточность данных, которые хранятся, упростить организацию запросов для их получения. Так же очень важно выбрать СУБД. Для разработки представленной базы данных удобно использовать платформу 1С:Предприятие 8.3 и ее встроенный язык программирования. Хотя 1С:Предприятие 8.3 и является дорогостоящим продуктом, зато данная платформа является наиболее подходящей для расчетов в комиссионном магазине и поддерживает множество типов данных, необходимых при разработке.

Таким образом, были рассмотрены основные цели и принципы моделирования базы данных для автоматизированной информационной системы «Комиссионный магазин». Также были рассмотрены основные сущности спроектированной базы данных.

ЛИТЕРАТУРА

1. Howknow1c [Электронный ресурс] // Модуль 1С. – Режим доступа: <http://howknow1c.ru/programmirovanie-1c/modul-1s.html>. – Дата доступа: 13.09.18.
2. Bd-subd [Электронный ресурс] // Базы данных и СУБД. – Режим доступа: <http://bd-subd.ru/lekci/bazi-dannih-i-subd.htm>. – Дата доступа: 14.09.18.

где B_l – размер блока, l_i – размер исходного изображения в пикселях, для разрешения 4K максимальный размер длинны или ширины варьируется от 3600 до 4200 пикселей, l – размер входного слоя СНС, для СНС Yolo v3 он составляет [416×416] пикселей.

2. Обнаружение областей интереса на фрагментах кадра. Для этого каждый блок поступает на вход СНС Yolo v3. Выходным результатом для каждой области интереса, которая содержит объект или фрагмент объекта, является набор данных вектор $[X_1, Y_1, X_2, Y_2, Confidence, Class_index, B_n]$, где X_1, Y_1 — координаты левого верхнего угла области интереса, X_2, Y_2 — координаты правого нижнего угла области интереса, $Confidence$ — степень уверенности системы, $Class_index$ — индекс класса объекта, B_n — номер блока в котором осуществлялось обнаружение.

3. Нахождение координат обнаруженных областей на кадре. Обнаруженные координаты областей интереса на фрагментах кадра, записанные в текстовый документ, пересчитываются на координаты входного кадра:

$$\begin{aligned} Add_X &= B_n \bmod s; \\ Add_Y &= B_n \text{ div } s \end{aligned} \quad (2)$$

где Add_X и Add_Y – добавочные значения для X и Y координат соответственно, s – количество разбиений для одной стороны кадра

4. Объединение областей интереса. Для объединения соседних областей интереса рассчитывается коэффициент Жаккарда:

$$Iou = \frac{In}{B_1(x_2 - x_1) \cdot B_1(y_2 - y_1) + B_2(x_2 - x_1) \cdot B_2(y_2 - y_1) - In}, \quad (3)$$

где

$$\begin{aligned} In &= \left(\min(B_1(x_1) + B_1(x_2 - x_1), B_2(x_1) + B_2(x_2 - x_1)) - \max(B_1(x_1), B_2(x_1))) \right) \times \\ &\left(\min(B_1(y_1) + B_1(y_2 - y_1), B_2(y_1) + B_2(y_2 - y_1)) - \max(B_1(y_1), B_2(y_1))) \right), \end{aligned} \quad (4)$$

где B_1 – область интереса, $B_1(X_1, Y_1)$ – координаты левого верхнего угла области интереса, $B_1(X_2, Y_2)$ — координаты правого нижнего угла области интереса

Если рассчитанное значение коэффициента Жаккарда выше порогового и объекты относятся к одной группе, то данные области интереса объединяются в один объект. В данной работе пороговое значение выбиралось аналитически и составило 50%. Класс объекта определяется максимальной величиной $Confidence$ для объединяемых областей.

Алгоритм реализован с использованием языка программирования Python, с фреймворка Pytorch [10] и библиотеки OpenCV [11]. В качестве СНС была использована СНС Yolo v3 предварительно обученная на базе данных СОСО содержащей 80 классов объектов.

Рисунок 2 свидетельствует о том, что предложенный алгоритм позволил исключить ложное обнаружение автомобиля, как показано на рисунке 1.

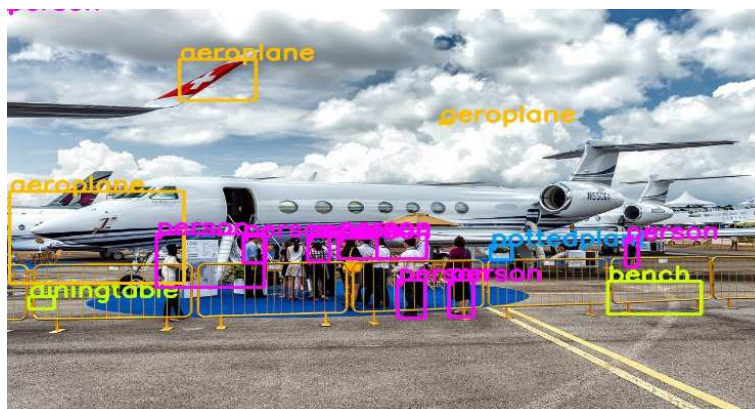


Рисунок 2. – Пример обнаружения и классификации объектов на основе предложенного алгоритма

Результаты экспериментов показали, что предложенный алгоритм позволяет уменьшить вероятность фрагментарной классификации одного объекта для граничных областей блоков, возникающих из-за разбиения входного кадра на фрагменты. Однако, наблюдается некоторое количество ложноположительных результатов обнаружения. Этот недостаток можно исключить, переобучив нейронную сеть на выборке, включающей фрагменты изображений.

ЛИТЕРАТУРА

1. Deep Residual Learning for Image Recognition / K. He [et al.] // Proc. of IEEE Conf. on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA. – 2016. – P. 770–778.
2. Szegedy, C. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning / C. Szegedy, S. Ioffe, V. Vanhoucke // Proc. of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17), 2017. – 4278–4284.
3. Object Detection via Region-based Fully Convolutional Networks [Electronic resource]. – Mode of access: <https://arxiv.org/pdf/1605.06409.pdf>. – Date of access: 20.12.17.
4. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks [Electronic resource]. – Mode of access: <http://papers.nips.cc/paper/5638-faster-r-cnn-towards-real-time-object-detection-with-region-proposal-networks.pdf>. – Date of access: 20.12.17.
5. You Only Look Once: Unified, Real-Time Object Detection / J. Redmon [et al.] // Proc. of IEEE Conf. on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA. – 2016. – P. 779–788.
6. YOLO9000 Better, Faster, Stronger [Electronic resource]. – Mode of access: <https://arxiv.org/pdf/1612.08242.pdf>. – Date of access: 20.12.17.
7. YOLOv3: An Incremental Improvement [Electronic resource]. – Mode of access: <https://arxiv.org/pdf/1804.02767.pdf>. – Date of access: 15.06.18.
8. An Effective Object Detection Algorithm for High Resolution Video by Using Convolutional Neural Network / Denis Vorobjov [et al.] // Huang T., Lv J., Sun C., Tuzikov A. (eds) Advances in Neural Networks. – ISSN 2018. Lecture Notes in Computer Science, vol 10878, Cham, Switzerland. – 2018. – P. 503–510.
9. COCO – Common Objects in Context [Electronic resource]. – Mode of access: <http://cocodataset.org>. – Date of access: 15.06.18.
10. PyTorch [Electronic resource]. – Mode of access: <https://pytorch.org>. – Date of access: 15.06.18.
11. OpenCV library [Electronic resource]. – Mode of access: <https://opencv.org>. – Date of access: 15.06.18.

УДК 004.932

**АНАЛИЗ АРХИТЕКТУР СВЕРТОЧНЫХ НЕЙРОННЫХ СЕТЕЙ
ДЛЯ ОБНАРУЖЕНИЯ И КЛАССИФИКАЦИИ ОБЪЕКТОВ
НА ВИДЕОПОСЛЕДОВАТЕЛЬНОСТЯХ БОЛЬШОГО РАЗРЕШЕНИЯ****И.Ю. ЗАХАРОВА, Д. ВОРОБЬЕВ***(Представлено: канд. техн. наук, доц. Р.П. БОГУШ)*

В приложении к задаче обнаружения объектов на видео большого разрешения рассмотрены архитектуры современных сверточных нейронных сетей. Выявлено, что применение масштабирования входного кадра большого разрешения к относительно малым размерам входного слоя СНС будет приводить к потере объектов небольшого размера, что ухудшает процедуру обнаружения в целом.

Последовательности видеоизображений высокого разрешения характеризуются тем, что содержат значительный объем информации, но при этом на обработку и извлечение необходимых данных требуют огромных вычислительных затрат. Однако, значительное повышение производительности вычислительной техники за последнее десятилетие предопределило развитие и использование ресурсоемких методов обработки информации, возможности которых ранее были ограничены аппаратными средствами, что привело к расширению спектра решаемых практических задач.

Обнаружение объектов на видеопоследовательностях и дальнейшее их распознавание являются актуальными задачами для автоматизированных систем управления и принятия решений, использующих техническое зрение различного назначения, а также видеонаблюдение. Эффективным инструментом для этого являются сверточные нейронные сети (СНС), включающие кроме традиционных полносвязных слоев, сверточные и слои субдискретизации. К первой СНС относят сверточную нейронную сеть LeNet-5 [1]. С учетом перспективы применения СНС и большого интереса к ним ученых, в последнее время предложено ряд архитектур, которые направлены на обнаружение и классификацию объектов на изображениях и могут быть использованы для обработки видеопоследовательностей большого разрешения.

Модель СНС AlexNet [2] включает восемь взвешенных слоев, из которых первые пять являются сверточными, а последние три представляют собой полносвязные слои. Данная модель была обучена на базе данных ImageNet ILSVRC-2010, содержащей 1.2 миллиона изображений, которые разбиты на 1000 классов. При этом, использовались искусственные расширения обучающей выборки, такие как сдвиг, вращение и удаление областей изображений. При тестировании сети на метриках top-1 и top-5 коэффициенты ошибок составили 67.4% и 40.9%, соответственно. Данная модель сети имеет ограничения при работе с многомасштабным анализом, подвержена переобучению из-за пропуска объектов на боковых выбросах, требует значительных вычислительных затрат.

Модель СНС R-CNN предполагает выделение областей интереса на основе предположений о местоположении объектов с использованием метода выборочного поиска. Далее размер регионов масштабируется к размеру входного слоя и обрабатывается СНС AlexNet. На последнем этапе выполняется бинарная классификация с использованием метода опорных векторов, которые были получены для каждого класса объектов. Повышение устойчивости модели к ошибочному делению объекта на фрагменты достигается за счет применения подавления не максимумов. При этом, совпадающие границы областей интереса на соседних фрагментах одного объекта удаляются из-за отсутствия существенного изменения величины градиента. Значение коэффициента ошибок при тестировании определяется величиной 15,3% в метрике ошибок top-5 [3]. Недостатком данной СНС является значительное время, затрачиваемое на обработку потока данных.

Для уменьшения временных затрат в 2015 году предложена модификация данного метода, Fast R-CNN [4], в которой используется усеченный метод опорных векторов. Особенностью Fast R-CNN является также представление регионов интереса в виде сверточной карты признаков. При этом используется 4-х размерный вектор $\{r, c, h, w\}$, где r, c - координаты верхнего левого угла, а h и w - высота и ширина региона соответственно. Далее карта признаков подается на слой субдискретизации с размером окна 7×7 , выходом которого является максимальное значение для каждого положения окна. Данный метод быстрее осуществляет классификацию в выбранных областях интереса, но не учитывает существенные временные затраты на выделение этих областей перед обработкой в СНС.

Дальнейшее уменьшение временных затрат было предложено в модели Faster R-CNN [5]. Следует отметить, что в отличие от Fast R-CNN данная модель учитывает временные затраты на выделение

объектов. В Faster R-CNN используется нейронная сеть для предположения о нахождении регионов интереса (Region Proposal Network, PRN). Вход сети RPN выделяет сверточные признаки, которые затем попадают на структуру, состоящую из двух полностью соединенных сверточных слоев. Первый слой является регрессионным и предсказывает прямоугольную область, ограничивающую объект. Второй слой является классификационным. Данная структура не одинаково хорошо показывает свои результаты работы с некоторыми группами объектов. Также Faster R-CNN не устойчива к мелким объектам и к зашумленным изображениям.

Модель GoogLeNet [6] состоит более чем из 100 слоев, однако полностью подключенные слои не используются совсем. Также, по сравнению с AlexNet, количество обнаруживаемых параметров уменьшено в 12 раз и сверточные ядра большого размера заменены на последовательность ядер меньшего размера. В модели данной СНС выделяют блок Inception, который включает операции свертки с ядрами размером 5×5 , 3×3 , 1×1 и субдискретизации для окна размером 3×3 . Данный блок последовательно повторяется в архитектуре сети девять раз. Применение указанных выше ядер свертки позволяет извлекать признаки различных размеров в одном блоке. В данной модели по сравнению с R-CNN на первом этапе улучшен подход к выделению регионов интереса за счет включения в метод выборочного поиска алгоритма генерации множества ограничивающих рамок (multibox) [7]. В алгоритме применяется СНС AlexNet, которая генерирует заданное количество ограничивающих рамок для каждой области интереса, далее используется подавление не максимумов для исключения наименее совпадающих ограничивающих рамок с действительной областью интереса. При обучении СНС GoogLeNet выборка искусственно расширялась путем многократного использования входных изображений, после чего результирующие значения многопеременной логической функции (softmax) усреднялись для идентичных изображений, что позволило улучшить результат классификации. Для данной модели СНС значение коэффициента ошибок в метрике top-5 составляет 6,7%. Однако, GoogLeNet не учитывает контекстную информацию обо всем изображении, так как на вход СНС подаются лишь области интереса. Также, выделение ограничивающих рамок расширенным методом выборочного поиска значительно замедляет обработку изображения.

В работе [8] предложена модификация блока Inception v2 для GoogLeNet путем факторизации сверточного слоя 5×5 на 2 слоя с размером 3×3 , что позволило увеличить скорость вычислений в 9,36 раза. Дальнейшее применение факторизации (разложения) для всех сверточных слоев $n \times n$ в два слоя с размерами $n \times 1$ и $1 \times n$ позволило уменьшить вычислительные затраты на 33%. Для исключения резкого уменьшения размерности карты признаков и снижения вычислительных затрат половина признаков, вычисленных на предыдущих этапах, подается на слой субдискретизации. Другая часть признаков поступает на последующий сверточный слой. Данная модель также предполагает начальное задание минимальных весовых коэффициентов, равномерно распределенных относительно каждого класса.

Модификация Inception v3 [9] характеризуется добавлением нормализации выборки из [11] на последних слоях вместо технологии отсева (Dropout). Модель СНС состоит из 11 чередующихся блоков Inception v3 и при тестировании достигает коэффициента ошибки 4,2% для метрики top-5. Однако, по сравнению со своим ранним аналогом, затрачивает в 2.5 раз больше вычислительной мощности.

Рассмотренная в [10] архитектура СНС ResNet содержит начальный сверточный слой с размерностью ядра $[7 \times 7]$, чередующиеся сверточные слои с ядрами $[3 \times 3]$ и $[1 \times 1]$, а также обеспечивает возможность соединения по технологии быстрого доступа между входами чередующихся слоев и их выходами. В работе отмечено, что использование технологии быстрого доступа позволяет исключить ухудшение качества работы детектора при увеличении количества слоев СНС. Коэффициент ошибки составляет 3,57% в метрике top-5.

В 2016 году компания Google представила четвертую версию блока Inception v4, а также его модификации Inception-ResNet [11]. Применение технологии TensorFlow позволило обучать модель GoogLeNet на одном устройстве, в отличие от предыдущих версий данной СНС, обучение которых выполнялось по частям с дальнейшим их соединением в полностью обученную модель.

Представленная в 2016 модель СНС Yolo (You Only Look Once) [12] использует сверточные слои с ядрами размером $[7 \times 7]$ и $[3 \times 3]$ для выделения признаков, а третий слой с ядром размером $[1 \times 1]$ для понижения размерности пространства предположений. Также используются слои субдискретизации размером $[2 \times 2]$. За каждым сверточным слоем $[3 \times 3]$ размещен полносвязный слой $[1 \times 1]$, передающий координаты ограничивающих рамок и выходные вероятности о нахождении объекта и классовой принадлежности этого объекта. Два конечных полносвязных слоя выполняют задачу классификации. Размеры областей интереса во время обучения выбираются вручную.

Модель делит входное изображение на сетку размером $[S \times S]$ частей (ячеек). Для каждой ячейки сетки предсказываются ограничивающие рамки и их весовые коэффициенты, которые характеризуют вероятность наличия объекта в ограничивающей рамке и величину этой вероятности ($\text{Pr}(\text{Object})$). Если центр объекта попадает в ячейку сетки, то для нее устанавливается максимальная вероятность наличия рассматриваемого объекта. Каждая ограничивающая рамка характеризуется 5 параметрами: координатами центра относительно границ ячейки сетки (x , y), шириной (w) и высотой (h) ячейки, и ($\text{Pr}(\text{Object})$). После разбиения каждая ячейка сети поступает на входной слой сети, масштабируясь под его размер. Во время тестирования вероятность наличия объекта в ограничивающей рамке, а также вероятность принадлежности объекта к конкретному классу умножаются, и их произведение дает классификационную оценку для каждого региона. Эти оценки кодируют как вероятность того, что этот класс появится в поле, так и насколько хорошо предсказанный регион подходит для объекта. По метрике Top-5 точность данной модели составляет 88%. Недостатком модели является ручной выбор размера рамок, множество ошибок локализации для объектов малого размера, а также при перекрывающихся рамках проявляется нестабильность модели.

Для улучшения сегментации и классификации были предложены следующие модификации Yolo v2, Yolo9000[13], в которых предусмотрены:

- нормализация данных, позволяющая не использовать технологию «отсева» без опасения возникновения переобучения;
- повышение размерности классификатора для Yolo v2 до $[448 \times 448]$ для 10 эпох ImageNet;
- использование сети, выносящей предположение о нахождении регионов интереса (RPN) по аналогии с моделью Faster R-CNN;
- применение метода k -средних ($k=5$) для уточнения размера ограничивающих рамок и предварительной сегментации объектов в каждой области интереса;
- более точное предсказание местоположения осуществляется за счет передачи на последующие слои не собственных координат ограничивающих рамок, а их смещения относительно верхнего левого угла исходного изображения;
- вычисление детализированных признаков с разрешением карты признаков $[26 \times 26]$;
- многомасштабное обучение, которое позволяет сети сегментировать и классифицировать объекты при разных разрешениях, исключаяющее полносвязные слои в архитектуре СНС;
- использование новой классификационной модели Darknet-19, которая включает 19 сверточных слоев и 5 слоев субдискретизации.

Yolo9000 имеет ту же архитектуру, что и Yolo v2, однако количество выходных гипотез ограничивается 3. Модификация архитектуры улучшенной версии Yolo заключается в замене 2 последних полносвязных слоев исходной архитектуры на сеть, выносящую предположения о наличии регионов интереса (Region Proposal Network, RPN) и применении метода k -средних (причем, $k=5$) для каждой ячейки. Таким образом, каждая ячейка может включать 5 объектов, в то время как первая версия Yolo ограничивалась одним объектом для ячейки. Далее выделенные области интереса поступают на вход классификатора Darknet-19, в котором используется 19 сверточных слоев размером $[3 \times 3]$ и $[1 \times 1]$, а также 5 слоев субдискретизации $[2 \times 1]$. Обучение Yolo 9000 выполнено на совместной базе данных (ImageNet и COCO), состоящей из 9000 классов. Вероятность правильного обнаружения для данной модели достигает значения 91,2% в метрике top-5, однако при этом вычислительные затраты значительно больше, чем у СНС Yolo. Для некоторых классов объектов, например, «человек», «одежда» вероятность правильной классификации уменьшается.

Представленный анализ современных архитектур СНС показал, что применение данных моделей для обработки видеоизображений большого разрешения требует применения масштабирования входного изображения к размерам входного слоя СНС, что будет приводить к потере объектов небольшого размера в кадре. Поэтому, для устранения данного недостатка необходимо использование специальных алгоритмов обработки [14].

ЛИТЕРАТУРА

1. LeCun, Y. Gradient-based learning applied to document recognition. Proceedings of IEEE / Y. LeCun [et al.]. – 1998. – 86(11). – P. 2278–2324.
2. Krizhevsky, I. Hinton. ImageNet classification with deep convolutional neural networks», Advances in Neural Information Processing Systems 25: Proc. of 26th Annual Conference on Neural Information Processing Systems / I. Krizhevsky, G.E. Sutskever. – 2012., Lake Tahoe, Nevada, United States. – P. 1106–1114.

3. R-CNN for Object Detection [Electronic resource]. – Mode of access: https://courses.cs.washington.edu/courses/cse590v/14au/cse590v_wk1_rcnn.pdf. – Date of access: 02.12.2017.
4. Ross B. Girshick, «Fast R-CNN», 2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December. – 2015. – P. 1440–1448.
5. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks [Electronic resource]. – Mode of access: <http://papers.nips.cc/paper/5638-faster-r-cnn-towards-real-time-object-detection-with-region-proposal-networks.pdf>. – Date of access: 20.12.17.
6. Christian Szegedy, Wei Liu, Yangqing Jia, Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich «Going Deeper with Convolutions [Electronic resource]. – Mode of access: <https://arxiv.org/pdf/1409.4842.pdf>.
7. D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov. «Scalable object detection using deep neural networks», 2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA. – 2014. – P. 2155–2162.
8. Rethinking the Inception Architecture for Computer Vision [Electronic resource]. – Mode of access: <https://arxiv.org/pdf/1512.00567.pdf>. – Date of access: 10.03.17.
9. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift [Electronic resource]. – Mode of access: <https://arxiv.org/pdf/1502.03167.pdf>. – Date of access: 10.03.17
10. He, K. Deep Residual Learning for Image Recognition / K. He, X. Zhang, S. Ren, J. Sun // Proc. of IEEE Conf. on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA. – 2016. – P. 770–778.
11. Szegedy, C. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning / C. Szegedy, S. Ioffe, V. Vanhoucke // Proc. of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17). – 2017. – 4278–4284.
12. Redmon, J. You Only Look Once: Unified, Real-Time Object Detection / J. Redmon, S.K. Divvala, R.B. Girshick, A. Farhadi // Proc. of IEEE Conf. on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA. – 2016. – P. 779–788.
13. YOLO9000 Better, Faster, Stronger [Electronic resource]. – Mode of access: <https://arxiv.org/pdf/1612.08242.pdf>. – Date of access: 15.06.17.
14. Vorobjov, D. An effective object detection algorithm for high resolution video by using convolutional neural network / D. Vorobjov, I. Zakharava, R. Bohush, S. Ablameyko // Proc. of the 15th Int. Symposium on Neural Networks (ISNN 2018), June 25–28 2018, Minsk. – 2018. – P. 503–508.

УДК 517.929.7

**РАСЧЕТ ТЕМПЕРАТУРНЫХ ПОЛЕЙ
ДВУХСЛОЙНОГО ПОЛОГО ЦИЛИНДРА
МЕТОДОМ РАЗДЕЛЕНИЯ ПЕРЕМЕННЫХ**

О.Н. МЕДВЕДЕВА

(Представлено: канд. техн. наук, доц. И.Б. СОРОГОВЕЦ)

В работе [1] для решения краевых задач теплопроводности в случае двухслойного полого цилиндра был применен метод разделения переменных. Рассмотрено девять начально-краевых задач на собственные значения и собственные функции. При решении однородных задач получено девять ортогональных систем функций. Решения неоднородных задач получены в виде рядов Фурье по указанным системам. В данной работе на основе результатов из [1] произведены расчеты температурных, построены различные графики. Все расчеты производились в среде Mathcad.

Постановка задачи. Полый цилиндр большой длины (бесконечный полый цилиндр) с внутренним радиусом R_0 и внешним R_1 , окружен оболочкой толщиной $H = R - R_1$, где R – внешний радиус оболочки. Теплофизические характеристики полого цилиндра и оболочки считаем различными: λ_1, c_1, ρ_1 ; λ_2, c_2, ρ_2 . Здесь c – теплоемкость, ρ – плотность, λ – коэффициент теплопроводности, индекс 1 относится к полуму цилиндру, индекс 2 – к оболочке. Требуется найти распределение температуры в системе двух цилиндрических тел, которая будет зависеть от начальной температуры и от способа взаимодействия системы с окружающей средой. Предполагается, что температурное поле в системе тел радиальное, т. е. не зависит от полярного угла φ и координаты z . Температуру полого цилиндра обозначаем $T_1(r, t)$ (t – время), температуру оболочки – $T_2(r, t)$.

Расчеты производим в безразмерных величинах (α_0, α_2 – коэффициенты теплообмена)

$$x = \frac{r}{R}, \quad x_0 = \frac{R_0}{R}, \quad x_1 = \frac{R_1}{R}, \quad a_1 = \frac{\lambda_1}{c_1 \rho_1}, \quad a_2 = \frac{\lambda_2}{c_2 \rho_2}, \quad K_a^2 = \frac{a_1}{a_2}, \quad K_\lambda = \frac{\lambda_1}{\lambda_2}, \quad F = \frac{a_2 t}{R^2}, \quad Bi = \frac{\alpha_2}{\lambda_2}, \quad Bi_0 = \frac{\alpha_0}{\lambda_1}.$$

Для проведения расчетов выбраны следующие значения указанных величин:

$$x_0 = 0,3; \quad x_1 = 0,7; \quad Ka = 1,4; \quad K\lambda = 1,9; \quad Bi_0 = 2; \quad Bi = 3.$$

В [1] поставлено девять различных задач, которые можно охарактеризовать как задача $(i - j)$ ($i = 1, 2, 3; j = 1, 2, 3$). Это означает, что при $x = x_0$ задано граничное условие i -го рода, а при $x = 1$ – граничное условие j -го рода. Ниже ссылки на формулы будут производиться без указания на то, что они из [1].

В задачу расчета входят:

- 1) определение корней характеристического уравнения $\Delta(\mu) = 0$, где $\Delta(\mu)$ определяется выражением (13);
- 2) вычисление коэффициентов $A_{1,k}, A_{2,k}, B_{1,k}, B_{2,k}$ (k – порядковый номер характеристического числа) функций (15), (17), представляющих решения однородных задач теплопроводности;
- 3) построение решений неоднородных задач методами, разработанными в [1];
- 4) построение графиков решений;
- 5) выводы по результатам расчетов.

Все расчеты, в том числе и построение графиков, производим в среде Mathcad [2].

Определение корней характеристических уравнений. Для решения уравнения $\Delta(\mu) = 0$ строим график функции $y = \Delta(x)$ и по графику определяем приближенное значение каждого корня, которое уточняется с помощью функции **root**. Для задач $(i - j)$ ($i = 1, 2, 3; j = 1, 2, 3$) массивы корней обозначены mij . Значения массивов в интервале (0; 60) :

$m11^T =$		1	2	3	4	5	6	7	8	9	10	11
	1	5.1	10.605	16.05	21.273	26.853	32.048	37.619	42.391	48.403	53.418	59.169
$m12^T =$		1	2	3	4	5	6	7	8	9	10	11
	1	2.326	7.53	13.529	18.407	24.298	29.189	35.041	39.954	45.757	50.716	56.468
$m21^T =$		1	2	3	4	5	6	7	8	9	10	11
	1	3.01	8.611	13.353	19.166	24.009	29.827	34.716	40.514	45.456	51.201	56.19
$m22^T =$		1	2	3	4	5	6	7	8	9	10	11
	1	5.586	10.951	16.137	21.609	26.784	32.362	37.466	43.109	48.147	53.854	58.907
$m13^T =$		1	2	3	4	5	6	7	8	9	10	
	1	6.834	13.133	18.141	24.084	29.024	34.883	39.836	45.622	50.649	56.366	
$m31^T =$		1	2	3	4	5	6	7	8	9	10	11
	1	1.017	8.062	12.996	18.929	23.801	29.679	34.571	40.405	45.333	51.119	56.096
$m23^T =$		1	2	3	4	5	6	7	8	9	10	11
	1	4.551	10.558	15.777	21.429	26.572	32.225	37.308	43.004	48.029	53.766	58.743
$m32^T =$		1	2	3	4	5	6	7	8	9	10	11
	1	4.864	10.446	15.889	21.377	26.636	32.195	37.351	42.984	48.060	53.752	58.765
$m33^T =$		1	2	3	4	5	6	7	8	9	10	11
	1	3.343	10.017	15.515	21.172	26.416	32.058	37.197	42.879	47.941	53.673	58.670

Построение решений однородных задач. Для примера рассмотрим задачу (1–2). В формулах (14) $\alpha_1 = -1, \alpha_2 = 0, \beta_1 = 0, \beta_2 = 1$ значения характеристических чисел уже вычислены. Фрагмент листа вычислений среды Mathcad:

ORIGIN:=1 x0:=0.3 x1:=0.7 Ka:=1.4 Kl:=1.5 Kl:=1.5 Bi:=3 Bi0:=2 $\mu := m12$ k12:=11
k:=1..k12

$$C1(\mu) := J0(\mu \cdot x1) \cdot (\beta2 \cdot \mu \cdot Y1(\mu) + \alpha2 \cdot Y0(\mu)) - Y0(\mu \cdot x1) \cdot (\beta2 \cdot \mu \cdot J1(\mu) + \alpha2 \cdot J0(\mu))$$

$$c2(\mu) := Y0\left(\frac{\mu}{Ka} \cdot x1\right) \cdot \left(\frac{\beta1}{Ka} \cdot \mu \cdot J1\left(\frac{\mu}{Ka} \cdot x0\right) - \alpha1 \cdot J0\left(\frac{\mu}{Ka} \cdot x0\right)\right)$$

$$C2(\mu) := J0\left(\frac{\mu}{Ka} \cdot x1\right) \cdot \left(\frac{\beta1}{Ka} \cdot \mu \cdot Y1\left(\frac{\mu}{Ka} \cdot x0\right) - \alpha1 \cdot Y0\left(\frac{\mu}{Ka} \cdot x0\right)\right) - c2(\mu)$$

$$A1(\mu) := \left(\frac{\beta1 \cdot \mu}{Ka} \cdot Y1\left(\frac{\mu}{Ka} \cdot x0\right) - \alpha1 \cdot Y0\left(\frac{\mu}{Ka} \cdot x0\right)\right) \cdot C1(\mu) \quad B1(\mu) := -\left(\frac{\beta1 \cdot \mu}{Ka} \cdot J1\left(\frac{\mu}{Ka} \cdot x0\right) - \alpha1 \cdot J0\left(\frac{\mu}{Ka} \cdot x0\right)\right) \cdot C1(\mu)$$

$$A2(\mu) := (\beta2 \cdot \mu \cdot Y1(\mu) + \alpha2 \cdot Y0(\mu)) \cdot C2(\mu) \quad B2(\mu) := -(\beta2 \cdot \mu \cdot J1(\mu) + \alpha2 \cdot J0(\mu)) \cdot C2(\mu)$$

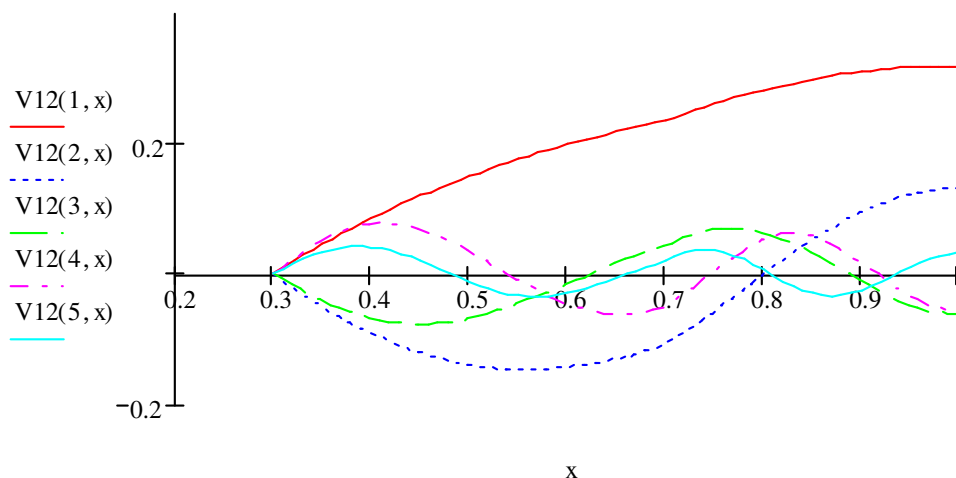
$$V12(k, x) := \text{if}\left(x \leq x1, A1(\mu_k) \cdot J0\left(\frac{\mu_k}{Ka} \cdot x\right) + B1(\mu_k) \cdot Y0\left(\frac{\mu_k}{Ka} \cdot x\right), A2(\mu_k) \cdot J0\left(\frac{\mu_k}{1} \cdot x\right) + B2(\mu_k) \cdot Y0\left(\frac{\mu_k}{1} \cdot x\right)\right)$$

$$U12(k, x) := \text{if}\left[x \leq x1, \left(A1(\mu_k) \cdot J0\left(\frac{\mu_k}{Ka} \cdot x\right) + B1(\mu_k) \cdot Y0\left(\frac{\mu_k}{Ka} \cdot x\right)\right)^2, \left(A2(\mu_k) \cdot J0\left(\frac{\mu_k}{1} \cdot x\right) + B2(\mu_k) \cdot Y0\left(\frac{\mu_k}{1} \cdot x\right)\right)^2\right]$$

Квадрат нормы функций $V12(x)$: $N12(k) = \int_{x0}^1 U12(k, x) dx$

Графики функций $V12(k, x)$:

$x := 0.3, 0.31.. 1$



Точно таким же образом находятся коэффициенты $A_{1,k}, A_{2,k}, B_{1,k}, B_{2,k}$ других задач. Для этого нужно лишь изменить значения коэффициентов $\alpha_1, \alpha_2, \beta_1, \beta_2$.

Решение неоднородных задач. Предварительно укажем для каждой задачи функцию, удовлетворяющую только граничным условиям. Выпишем для каждой задачи функцию $U(x, F)$, которую будем обозначать $U_{ij}(x, F)$ (по названию задачи):

$$U11(x, F) = \frac{x-1}{x0-1} \varphi_1(F) + \frac{x-x0}{1-x0} \varphi_2(F); \quad U12(x, F) = \varphi_1(F) + (x-x0)\varphi_2(F);$$

$$U21(x, F) = (x-1)\varphi_1(F) + \varphi_2(F); \quad U22(x, F) = \frac{0,5x^2 - x}{x0-1} \varphi_1(F) + \frac{0,5x^2 - x0 \cdot x}{1-x0} \varphi_2(F);$$

$$U13(x, F) = \frac{1+Bi-x \cdot Bi}{1+(1-x0) \cdot Bi} \varphi_1(F) + \frac{x-x0}{1+(1-x0) \cdot Bi} \varphi_2(F);$$

$$U31(x, F) = \frac{(1-x) \cdot \varphi_1(F)}{1+(1-x0)Bi0} + \left(\frac{Bi0 \cdot (x-1)}{1+(1-x0)Bi0} + 1 \right) \cdot \varphi_2(F); \quad U23(x, F) = \left(\frac{1}{Bi} - x \right) \varphi_1(F) + \left(\frac{1}{Bi} + 1 \right) \cdot \varphi_2(F);$$

$$U32(x, F) = \frac{\varphi_1(F)}{Bi0} + \left(\frac{1}{Bi0} - x0 \right) \varphi_2(F); \quad U33(x, F) = \frac{(-Bi \cdot x + 1 + Bi) \cdot \varphi_1(F) + (Bi0 \cdot x + 1 - Bi0 \cdot x0) \cdot \varphi_2(F)}{Bi + Bi0 + Bi \cdot Bi0 \cdot (1-x0)}.$$

Дальнейшие расчеты будут производиться с функциями $\varphi_1(F) = \frac{F}{F+1} \arctg(F)$, $\varphi_2(F) = \frac{F^2}{F+1} \cos(0.1 \cdot F)$ при условии, что источники тепла отсутствуют, т. е. в (1), (2) $W_1(x, F) = 0, W_2(x, F) = 0$.

Тогда решение каждой задачи можно получить в виде $T(x, F) = U(x, F) + TW(x, F) + Tf(x, F)$, где $U(x, F)$ – построенные выше функции, удовлетворяющие только граничным условиям. При подстановке этой суммы в уравнения (1), (2) возникает неоднородность $W(x, F) = \begin{cases} K_a^2 (U''_{xx} + x^{-1}U'_x) - U'_F & (x_0 \leq x \leq x_1) \\ U''_{xx} + x^{-1}U'_x - U'_F & (x_1 \leq x \leq 1) \end{cases}$ и $TW(x, F)$ – решение, учитывающее эту неоднородность; $Tf(x, F)$ – решение задачи, учитывающее начальное условие:

$$Tf(x, F) = \sum_{k=1}^{\infty} D_k V(k, x) \exp(-\mu_k^2 F), \quad D_k = N_k^{-1} \int_{x_0}^1 f(x) V(k, x) dx, \quad N_k = \int_{x_0}^1 V^2(k, x) dx;$$

$$TW(x, F) = \sum_{k=1}^{\infty} B_k(F)V(k, x), \quad B_k(F) = \int_0^F \exp(\mu_k^2(t-F))W_k(t)dt, \quad W(x, F) = \sum_{k=1}^{\infty} W_k(F)V(k, x),$$

т. е. $W_k(F)$ коэффициенты Фурье при разложении функции $W(x, F)$ по системе $\{V(k, x)\}_{k=1}^{\infty}$.

$$T0 := 2C \quad f(x) := T0 \quad K := 1..11 \quad \phi1(F) := \frac{F}{F+1} \cdot \text{atan}(F) \quad \phi2(F) := \frac{F^2}{F+1} \cdot \cos(0.1 \cdot F)$$

$$U(x, F) := \phi1(F) + (x - x0) \cdot \phi2(F) \quad \psi1(F) := \frac{1}{(F+1)^2} \cdot \text{atan}(F) + \frac{F}{F+1} \cdot \frac{1}{1+F^2}$$

$$\psi2(F) := \frac{\cos(0.1 \cdot F)}{(F+1)^2} \cdot (F^2 + 2 \cdot F) - \frac{F^2 \cdot 0.1}{F+1} \cdot \sin(0.1 \cdot F) \quad w1(x) := \text{if} \left(x \leq x1, \frac{Ka^2}{x}, \frac{1}{x} \right)$$

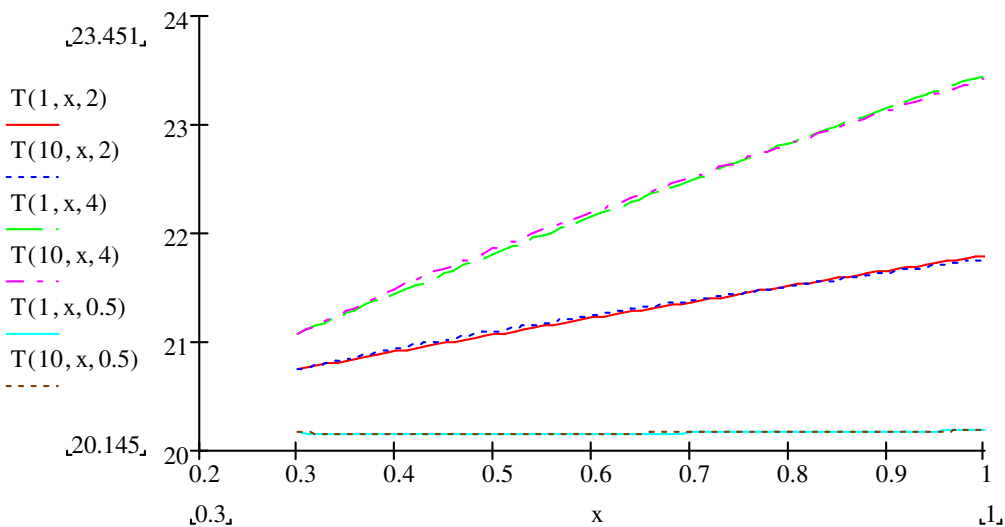
$$w2(x) := -1 \quad w3(x) := -(x - x0) \quad D1_k := \frac{1}{N12(k)} \cdot \int_{x0}^1 w1(x) \cdot V12(k, x) \cdot x dx$$

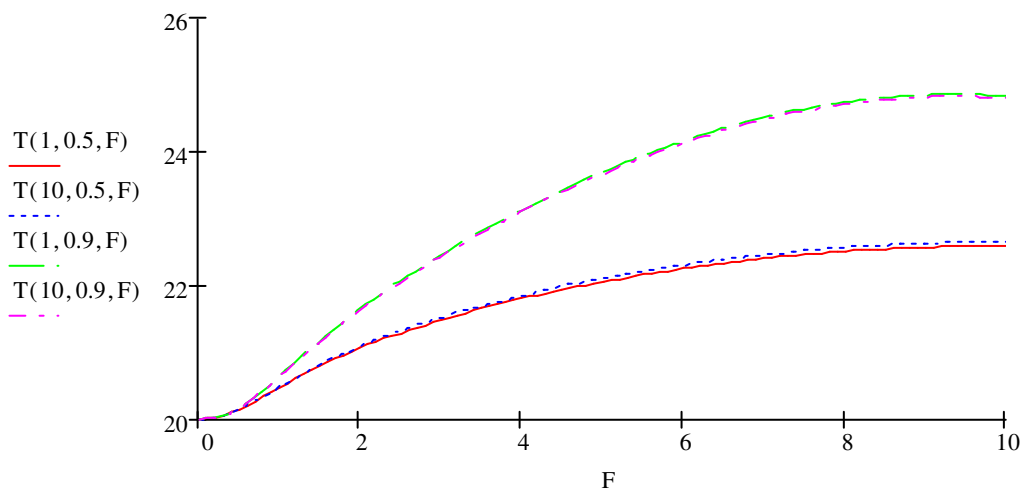
$$D2_k := \frac{1}{N12(k)} \cdot \int_{x0}^1 w2(x) \cdot V12(k, x) \cdot x dx \quad D3_k := \frac{1}{N12(k)} \cdot \int_{x0}^1 w3(x) \cdot V12(k, x) \cdot x dx$$

$$W(k, F) := D1_k \cdot \phi1(F) + D2_k \cdot \psi1(F) + D3_k \cdot \psi2(F) \quad B(k, F) := \int_0^F \exp(\mu_k^2 \cdot (t - F)) \cdot W(k, t) dt$$

$$TW(K, x, F) := \sum_{m=1}^K B(m, F) \cdot V12(m, x) \quad T(K, x, F) := T0 + TW(K, x, F) + U(x, F)$$

$x := 0.3, 0.31..1$



$F := 0, 0.1.. 10$ 

Как видно из графиков, значения температуры практически не зависят от того, сколько членов ряда взято для приближенного вычисления, т. е. для вычисления температурных полей с высокой степенью точности достаточно взять два–три первых члена ряда.

ЛИТЕРАТУРА

1. Сороговец, И.Б. Моделирование температурных полей двухслойного полого цилиндра методом разделения переменных / И.Б. Сороговец, О.Н. Медведева // Вестник Полоцкого государственного университета. Серия С, Фундаментальные науки. – 2017. – № 12. – С. 101–108.
2. Алексеев, Е.Р. Mathcad 12 / Е.Р. Алексеев, О.В. Чеснокова. – М. : NT Press, 2005. – 345 с.

УДК 004.5;004.921

ВИЗУАЛИЗАЦИЯ И ГРАФИКА В UNITY3D

И.А. КОРСАК

(Представлено: канд. техн. наук, доц. И.Б. БУРАЧЕНОК)

Рассматриваются основные методы экспорта и настройки 3D-моделей для приложения виртуальной реальности «Газорегуляторный пункт VR» на базе УП «Витебскоблгаз».

Введение. Существует множество популярных игровых движков, таких как: CryEngine, UnrealEngine, Urho3D, Unity3D. Однако именно Unity3D пользуется популярностью среди начинающих, так как он обладает низким «порогом вхождения».

Основная часть. Рассмотрим экспорт 3D-объекта из среды моделирования в Unity3D. Не смотря на поддержку в Unity множества форматов 3D-графики, экспорт в игровой движок необходимо делать в формате FBX. Этот формат специально разработан для обмена данными между 3D-редакторами, что обеспечит корректный перенос 3D-модели, материалов, карт развертки текстур и остальных данных.

Для корректного отображения материалов и текстур на 3D-модели в Unity, в редакторе 3D-графики необходимо создать карту развертки текстур (UVmap). На рисунке 1 показана установка карт развертки, на примере редактора Blender.

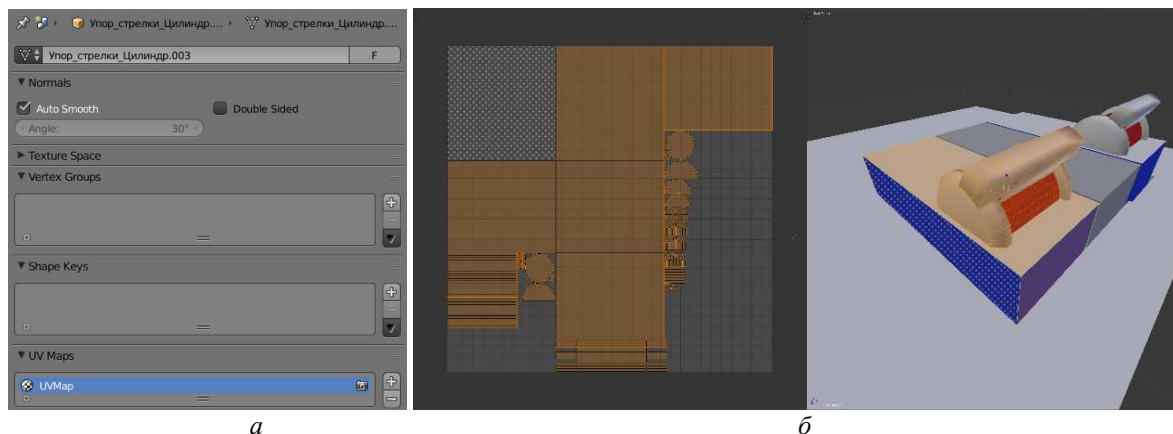


Рисунок 1. – Установка и настройка карт развертки

a – установка карт развертки;

б – настройка карт развертки для полигонов объекта

Это обеспечит соответствие между координатами на поверхности 3D-объекта и координатами на текстуре. Для более гибкого текстурирования цельного 3D-объекта можно создать несколько материалов и привязать их к необходимым полигонам. Пример настройки в редакторе Blender приведен на рисунке 2.

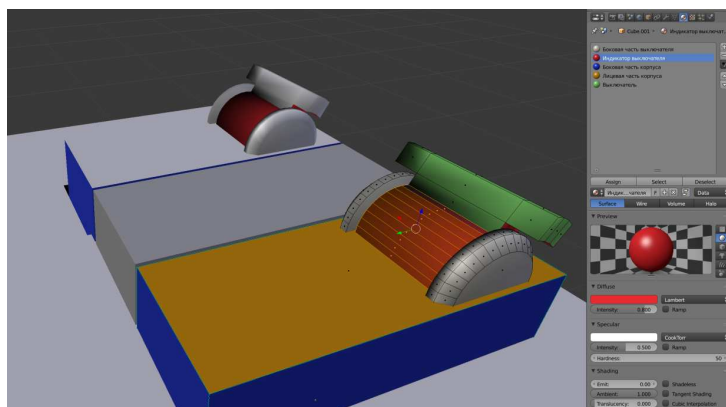


Рисунок 2. – Настройка материалов для группы полигонов

Для круглых и плавных поверхностей так же рекомендуется применять UV Smooth, это позволяет «спрятать» грубые переходы между полигонами 3D-модели. Это свойство следует применять к круглым или закругленным объектам. На рисунке 3 отображен результат работы UV Smooth.



Рисунок 3. – Настройка UV Smooth для модели регулятора давления

а – 3D-модель регулятора без сглаживания;
б – 3D-модель регулятора с свойством сглаживания

Однако, после переноса объекта из одного редактора в другой, материалы и текстуры могут быть не естественными или вовсе не отображаться. Поэтому необходимо создать новый материал и настроить его. Для этого необходимо нажать **Assets→Create→Material**.

Материал на основе стандартного шейдера Unity предлагает нижеперечисленные параметры. **Rendering mode** (режим отображения).

- 1 **Opaque** – стандартное отображение без прозрачных поверхностей.
 - 2 **Cutout** – позволяет создать эффект для объектов, которые должны быть местами прозрачными, и сплошными. Например, подходит для визуализации травы, листвы, волос, сетки забора.
 - 3 **Transparent** – прозрачный режим отображения. Подходит для стекол, пластика и других прозрачных или полупрозрачных элементов, так как оставляет засветы и отражения.
 - 4 **Fade** – позволяет делать отображаемый объект невидимым, убирая отражения и засветы. Не подходит для стекол или пластика, зато годится для, например, визуализации голограммы [1].
 - 5 **Albedo** – позволяет выбрать основной цвет или текстуру поверхности. Также позволяет установить прозрачность.
 - 6 **Specular** – позволяет установить эффект блеска, отражений. Принимает в качестве параметра текстуру или цвет. Годится для создания эффекта мокрого асфальта, кожаных поверхностей.
 - 7 **Metallic** – устанавливает параметры материала аналогично **Specular**. Подходит для воссоздания металлических и пластиковых деталей.
 - 8 **Smoothness**: позволяет установить эффект «микроповерхности». Предоставляя выбор между глянцевой и матовой поверхностью.
 - 9 **NormalMap** – позволяет установить карту поверхности, которая воссоздает эффект поверхности «играя» светом и отражениями. Это на много производительнее, чем воссоздавать поверхность в редакторе 3D моделей. Годится для визуализации царапин, мелких болтов, неровной или шероховатой поверхности.
 - 10 **Heightmap** – позволяет установить карту «высот», тем самым воссоздавая эффект выпуклости элемента. Например, годится для визуализации плитки или кирпича.
 - 11 **Occlusion** – Позволяет установить карту «освещения», благодаря этой карте, шейдер «знает» какую часть объекта лучше подсветить, а какую затемнить.
 - 12 **Emission** – Придает материалу свечение. Подходит для визуализации, например, экранов, ламп [1].
- На рисунке 4 представлены результаты применения **Rendering mode** на примере газорегуляторного пункта на базе УП «Витебскоблгаз». На рисунке 4, *а* – режим отображения у корпуса автомобиля **Opaque**; 4, *б* – режим отображения у листвы **Cutout**; 4, *в* – режим отображения окна **Transparent**; 4, *г* – режим отображения автомобиля **Fade**.

На рисунке 5 показаны примеры установки различных параметров стандартного шейдера. На рисунке 5, *а* демонстрируется, для создания эффекта металлической поверхности, использование режимов отображения **Specular** и **NormalMap**. Применение режима отображения **Heightmap**, для карт высот и нормалей на плоской поверхности, показано на рисунке 5, *б*. На рисунке 5, *в*, для реализации карт освещения

для улучшения цвета и тени, показан использование режима отображения Occlusion. На рисунке 5, *г* – использование режима отображения Emission для придания режима свечения лампы индикации.

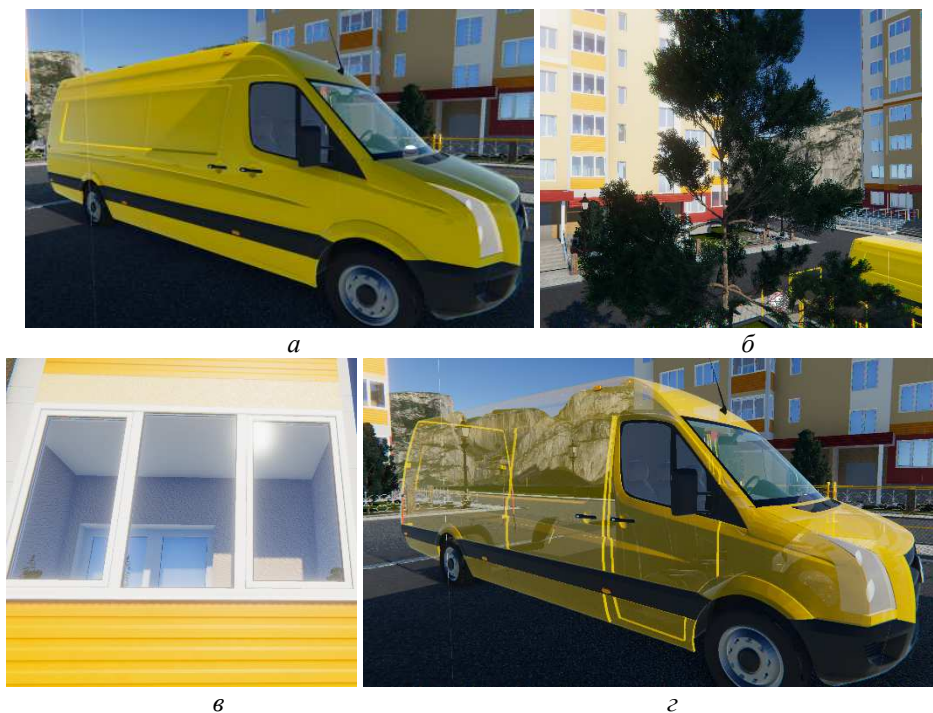


Рисунок 4 – Применение Rendering mode

а – режим отображения у корпуса автомобиля Opaque; *б* – режим отображения у листы Cutout; *в* – режим отображения окна Transparent; *г* – режим отображения автомобиля Fade

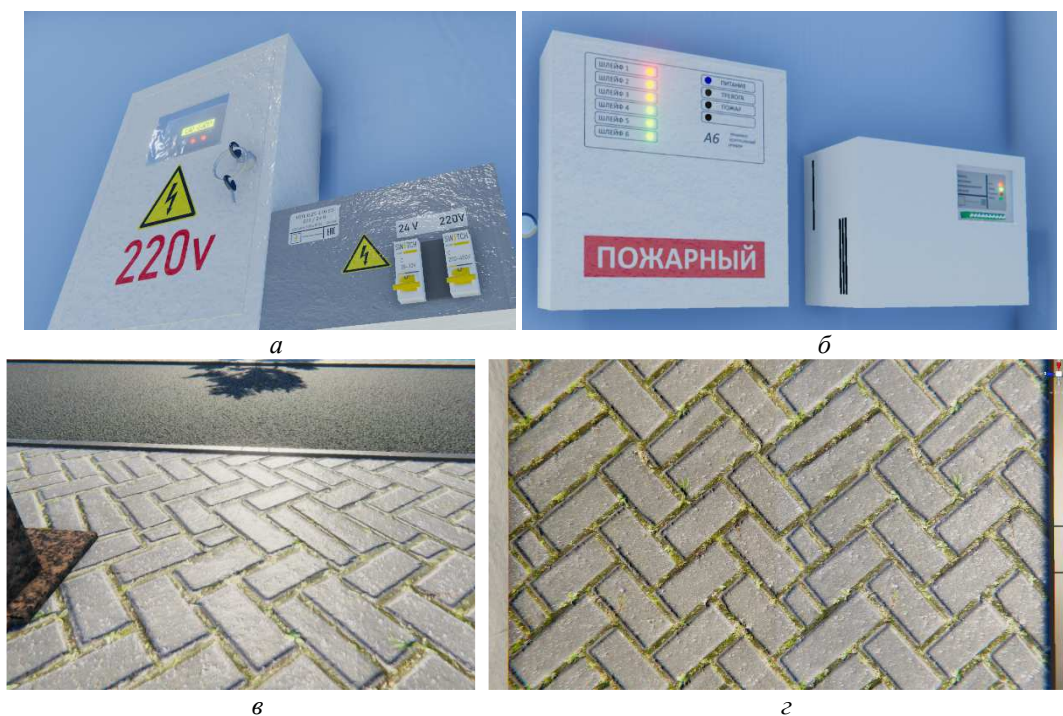


Рисунок 5 – Установка различных параметров стандартного шейдера

а – использование Specular и NormalMap – создание эффекта металлической поверхности;
б – использование Heightmap – создание карт высот и нормалей на плоской поверхности;
в – использование Occlusion – создание карт освещения для улучшения цвета и тени;
г – использование Emission – создание лампы индикации с отображением «светящегося» свойства материала

Заключение. Рассмотренные основные методы экспорта и визуализации 3D-моделей в Unity3D и применение наборов стандартных шейдеров в Unity3D позволяют добиться неплохих результатов при формировании объектов виртуальной реальности. Представленный в статье опыт разработки 3D-элементов в процессе реализации приложения виртуальной реальности на примере газорегуляторного пункта на базе УП «Витебскоблгаз» позволит начинающим разработчикам быстрее освоиться при работе с компьютерной графикой.

ЛИТЕРАТУРА

1. Unity User Manual [Электронный ресурс]. – Режим доступа <https://docs.unity3d.com/Manual/StandardShaderMaterialParameters.html> /. – Дата доступа: 20.09.2018.

УДК 004.5;004.921

**ТЕХНОЛОГИИ ВИРТУАЛЬНОЙ РЕАЛЬНОСТИ В ОБУЧЕНИИ ПЕРСОНАЛА.
УЧЕБНЫЙ КЛАСС ДЛЯ РАБОТЫ В ВИРТУАЛЬНОЙ РЕАЛЬНОСТИ****И.А. КОРСАК***(Представлено: канд. техн. наук, доц. И.Б. БУРАЧЕНОК)*

Рассматривается создание учебного класса для работы в виртуальной реальности на примере газорегуляторного пункта на базе УП «Витебскоблгаз».

Введение. Качественное обучение – залог успеха в практической деятельности. От уровня знаний конкретного сотрудника нередко зависит успех дела и эффективность работы компании. Но во многих сферах деятельности недостаточно получить знания, необходимо закрепить их на практике, проработав алгоритм действий в конкретных ситуациях.

Повышение уровня подготовки персонала технологических процессов – важнейший вызов современного производства, чему есть несколько основательных причин.

1. Производство постоянно усложняется: для снижения рисков строятся все более сложные системы управления и защиты, что, в свою очередь, повышает риски. Вырваться из этого замкнутого круга, охарактеризованного британской исследовательницей Л. Бэйнбридж как «ирония автоматизации», можно только воздействуя на ключевое звено цепочки – кардинально усилить обучение персонала.

2. Описанное выше происходит на фоне все более заметной дисквалификации персонала, вызванной демографическим кризисом, вынесением нового производства в регионы с невысоким уровнем подготовки операторов, снижением качества специального профессионального образования (особенно характерно стран СНГ), убыванием престижа операторской профессии.

3. Меняются общественные установки: работодатели ощущают большую ответственность не только за экономику, но и за жизнь, здоровье и социальное благополучие работников, состояние окружающей среды, что напрямую зависит от уровня подготовки операторов [1].

Для решения задач, связанных с практическим обучением специалистов газового хозяйства в УП «Витебскоблгаз» конструируются специальные полигоны, где можно отработать навыки эксплуатации газового оборудования, как при обычных, так и при аварийных ситуациях. Однако, из-за разнообразия оборудования и схем участков подачи газа, требуются огромные временные и денежные затраты по переоборудованию полигонов.

Одним из популярных направлений в обучении персонала работе со сложным оборудованием является симуляция реальных процессов, в том числе и при помощи технологии виртуальной реальности. Компьютерные тренажеры – незаменимое средство обучения, позволяющее развить навыки и умения персонала безопасно и эффективно управлять газовым оборудованием, а также поддерживать его высокую «боеготовность» к работе в нештатных и аварийных ситуациях. Важно при этом, что тренинг происходит в моделируемой среде без риска для людей, производственных активов и экологии [1].

Стоит отметить, что для успешного обучения специалистов с помощью технологий виртуальной реальности важно не только качественное оборудование и обучающее программное обеспечение, но и само помещение, в котором будут проходить занятия. В связи с этим помещение должно отвечать следующим требованиям:

– *достаточная площадь* – помещение должно обладать площадью для расположения зоны работы с гарнитурой виртуальной реальности, а также площадью для обучающегося и терминала управления;

– *оптимальный размер зоны взаимодействия* – зона работы с гарнитурой виртуальной реальности должна обладать оптимальными размерами для работы дачиков различных комплектов виртуальной реальности;

– *модульность* – помещение должно обладать средствами быстрой установки и внедрения новых гарнитур виртуальной реальности, а также средств дополненной и смешанной реальности;

– *централизованное управление освещением* – помещение должно обладать блоком управления освещением и подсветкой, для создания обратной связи от обучающего приложения;

– *дизайн* – помещение должно быть выполнено в минималистичном дизайне, ничего не должно мешать в процессе работы с гарнитурой виртуальной реальности.

Далее на примере приложения технологии виртуальной реальности по настройке и пуску газорегуляторного пункта (ГРП) на базе УП «Витебскоблгаз» рассмотрим особенности создания учебной комнаты. Разработанный макет комнаты виртуальной реальности представлен на рисунке 1.

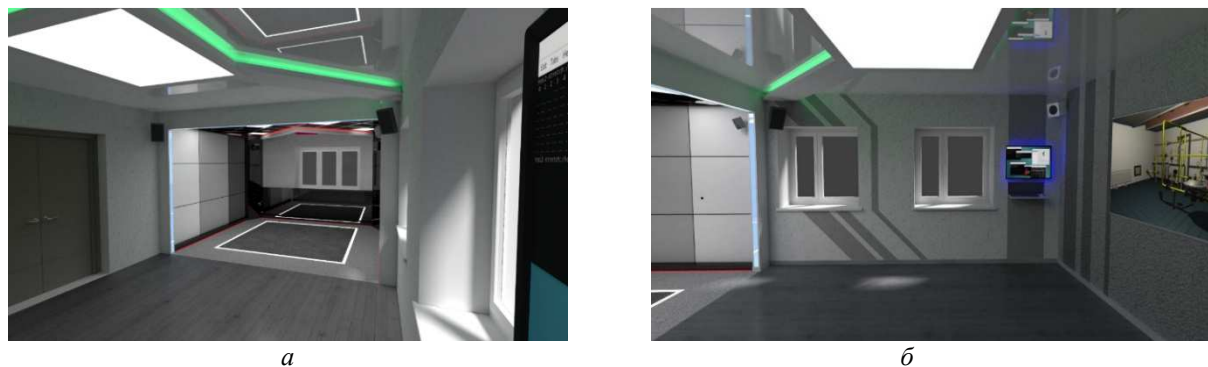


Рисунок 1. – Макет учебного класса виртуальной реальности
а – область взаимодействия с гарнитурой виртуальной реальности;
б – область оператора, проводящего обучение и ожидающих обучаемых

На рисунке 1, *а* можно увидеть область взаимодействия (отмечена белым квадратом). В угловых шкафах установлены блоки для зарядки джойстиков и беспроводного адаптера, компьютер, источник бесперебойного питания, блоки управления подсветкой, усилитель звуковой системы. Также продумана вентиляция и вывод горячего воздуха. За боковыми шкафами скрываются датчики Oculus Rift CV1, сетевой маршрутизатор, различные розетки (сетевые, звуковые, силовые), провода (USB, HDMI, сетевые, проводка для подсветки). На рисунке 1, *б* изображена область для оператора, проводящего обучение и ожидающих обучаемых. Здесь располагаются дисплей (на который транслируется картинка с гарнитуры виртуальной реальности), терминал для управления обучающим приложением и создания новых сценариев обучения.

Для обучающего приложения «Газорегуляторный пункт VR» на базе УП «Витебскоблгаз» разработаны программные модули (модуль взаимодействия гарнитуры с пользователем и сервер подсветки). Предусмотрена возможность работы в различных гарнитурах виртуальной реальности – Oculus Rift CV1 и HTC Vive. Так же имеется беспроводной комплект адаптеров позволяющий использовать гарнитуру HTC Vive без проводов и передвигаться спокойно по зоне работы с виртуальной реальностью.

Световая иллюминация является неотъемлемой частью проекта, цвет подсвечиваемой игровой зоны прямо зависит от действий пользователя, что дает четкое понимание происходящего оператору и наблюдающим. Световая иллюминация кабинета поделена на четыре секции. Секции подсветки указаны на рисунке 2.

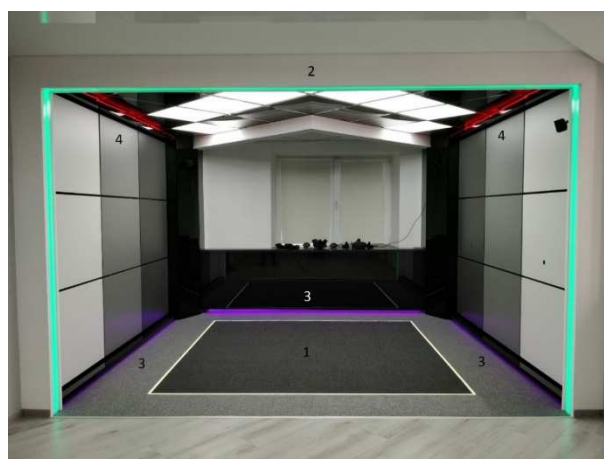


Рисунок 2. – Секции подсветки учебного кабинета

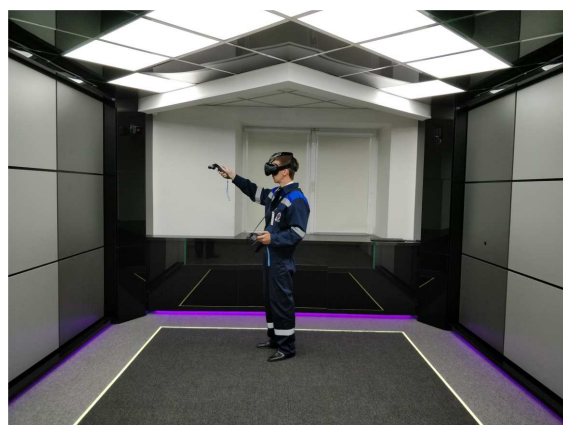
В качестве световых элементов использована светодиодная цветная лента SMD5050 с плотностью 60 светодиодных элементов на метр. В качестве контроллеров использованы устройства NooliteSD-180. Для связи контроллеров с компьютером используются USB – адаптер NooliteMTRF-64USB. Данный адаптер позволяет при помощи компьютера управлять различными устройствами системы nooLite и nooLite-F (как передатчик PC11xx) и одновременно принимать команды управления с передающих устройств (пультов, датчиков), выступая при этом как приемник RX2164. С точки зрения операционной

системы адаптер является переходником USB-COM, поэтому для общения с ним необходимо отправлять данные на COMпорт. Выбор этого аппаратного решения обусловлен открытым API и беспроводным соединением, работающей на частоте отличной от частоты работы гарнитур виртуальной реальности. На данный момент реализован сервер подсветки, позволяющий обучающему приложению выбрать сценарии подсветки, представленные в таблице.

Таблица. – Сценарии подсветки учебного кабинета

Действие пользователя/VR приложения	Цвет подсветки	Участок подсветки
VR приложение не запущено/ ошибка в сетевом соединении	Белый	Все участки
VR приложение запущено/ пользователь проходит экзамен	Бирюзово-синий	Участок №1 (игровая зона) Участок №3 (нижние панели)
	Зеленый	Участок №2 (арка)
Пользователь нарушил технологическую инструкцию	Красный	Участок №1 (игровая зона) Участок №2 (арка)
Пользователь выполнил задание	Зеленый	Участок №1 (игровая зона) Участок №2 (арка)

Сервер подсветки так же способен принимать команды по изменению цвета любых из участков. Теоретически доступно 16777215 цветов для каждого участка, если использовать качественные светодиоды. На рисунках 3, а и 3, б представлены фотографии обучения слесаря.



а



б

Рисунок 3. – Тренировка слесаря в обучающей программе по пуску ГРП

а – слесарь, проходящий обучение в виртуальной реальности;
б – визуализация происходящего в обучающей программе

Заключение. Данное программное обеспечение находится на стадии разработки, планируется добавить возможность управления стационарным освещением, индикацию выхода обучаемого из зоны взаимодействия, а также клиентское приложение дополненной реальности для управления освещением из Android смартфона.

Проект находится на последней стадии реализации, проводится тестирование специалистами газового хозяйства новой системы обучения, происходят завершающие этапы по техническому устройству помещения, усовершенствуется программное обеспечение. В качестве тренажера обеспечивающий «мягкий» переход от теории к практике, данный учебный кабинет зарекомендовал себя с положительной стороны.

ЛИТЕРАТУРА

1. Дозорцев, В.М. Технологии виртуальной реальности в обучении операторов технологических процессов // Автоматизация в промышленности. – 2018. – № 6. – С. 42.

УДК 004.5

**ПРИМЕНЕНИЕ ТЕХНОЛОГИЙ ВИРТУАЛЬНОЙ РЕАЛЬНОСТИ В ОБУЧЕНИИ
СПЕЦИАЛИСТОВ НА ПРИМЕРЕ ПРИЛОЖЕНИЯ ДЛЯ ГАЗОВЫХ СЛУЖБ****А.А. АНАНЕНКО***(Представлено: канд. техн. наук, доц. И.Б. БУРАЧЕНОК)*

Рассмотрены способы применения технологий виртуальной реальности в различных сферах жизнедеятельности человека. Описано приложение виртуальной реальности для сотрудников газовых служб. Обозначены достоинства использования технологий виртуальной реальности для обучения специалистов.

За последние пару лет своего развития технологии виртуальной реальности активно используются как в игровой индустрии, так и в различных сферах жизнедеятельности человека, например:

- архитектура – компьютерная визуализация будущих строений позволяет заказчикам и исполнителям путешествовать по этажам и помещениям еще до возведения фундамента;
- образование – школьники и студенты могут поработать в уникальных экспериментальных лабораториях, строить объемные диаграммы и проводить химические опыты;
- маркетинг и реклама – при покупке какого-либо изделия клиенту порой очень трудно составить представление об объекте продажи. С этой целью маркетологи используют виртуальную реальность. Это прекрасная возможность продемонстрировать продукт со всех сторон, включая сложные технические детали и конструктивы;
- промышленность – в автомобилестроении для проведения краш-тестов, компоновки узлов и агрегатов; инженеры-строители определяют оптимальную разводку бытовых коммуникаций, анализируют трудности монтажных работ в помещениях, виртуально размещают оборудование [1].

Но наиболее интересной областью для предприятий промышленного сектора является эксплуатация и обучение персонала – обширная сфера, где применение виртуальной реальности может стать просто незаменимо, т.к. ее использование открывает много новых возможностей в обучении, которые слишком сложны, затратны по времени или дороги при традиционных подходах.

Опираясь на это, было разработано специализированное приложение виртуальной реальности для обучения специалистов газовой промышленности. Настройка газового оборудования является сложной задачей и требует высокой точности. Ошибки, допущенные во время пуска газорегуляторного пункта (ГРП), могут привести к отключению газа у потребителей, что является не допустимым. Так же, в случае какой-либо аварии на участке газификации необходимо как можно быстрее и качественно ее предотвратить, при этом специалист, выполняющий работы на аварийном участке, имеет риск причинения вреда здоровью (работы с газом являются опасными).

Приложение представляет собой точную копию реально существующего ГРП, по которому, надев очки виртуальной реальности, можно походить и выполнить сценарий его пуска и продувки. Общий вид виртуального ГРП представлен на рисунке 1.



Рисунок 1. – Общий вид виртуального ГРП

Данный сценарий разработан в строгом соответствии с инструкцией и учитывает технологическую последовательность операций. В случае, если действия специалиста, отрабатывающего навыки на разработанном программном продукте, не соответствуют инструкции, он получает уведомление об ошибке и все действия придется повторить заново.

Кроме того, в приложении реализовано звуковое сопровождение, которое полностью соответствует реальным звукам газорегулировочного пункта, благодаря чему эффект присутствия в виртуальном ГРП значительно усиливается. Еще одним атрибутом, усиливающим данный эффект, является то, что все действия по взаимодействию с виртуальными объектами, аналогичны действиям в реальности. Это достигнуто благодаря воссозданию в виртуальном мире 3D-модели кисти руки и физики ее движения как показано на рисунке 2.

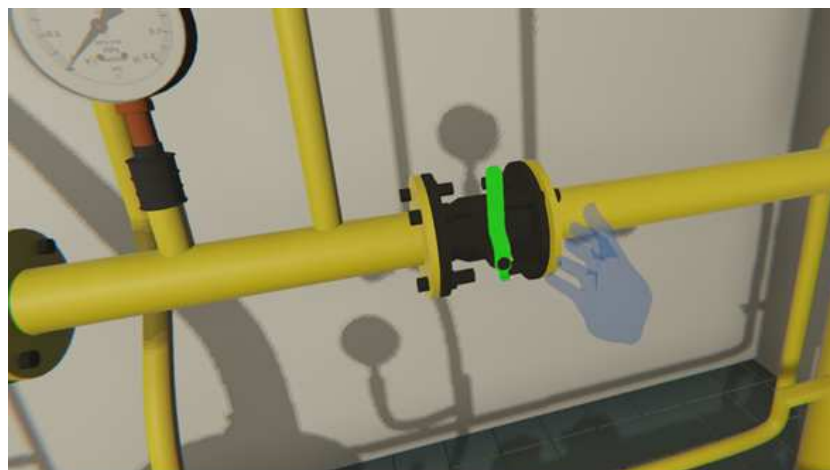


Рисунок 2. – Виртуальная рука

Стоит отметить, что приложение предусматривает два режима: экзамен и обучение. Во время режима обучения, пользователь выполняет сценарий пуска газа в ГРП в соответствии с подсказками, которые описывают последовательность действий, необходимых для успешного завершения работы. Суть режима экзамена заключается в самостоятельном выполнении сценария пуска газа. Для этого у экзаменуемого есть всего одна попытка.

На данный момент, ведется разработка всех возможных видов работ на ГРП: техническое обслуживание, текущий ремонт, проверка параметров срабатывания предохранительно-запорных и предохранительно-сбросных устройств. Это позволит специалистам, обслуживающим газорегуляторные пункты, практиковаться на обслуживании их виртуальных копий, а также ознакомиться с установленным на объектах оборудованием и его особенностями еще до выполнения реальных работ.

Приложение работает с такими системами отображения виртуальной реальности, как Oculus Rift CV1 и HTC Vive. Для обеспечения беспроводной передачи видеосигнала высокого разрешения используется TP Cast – съемный модуль для VR-шлема. Данное аппаратное решение позволяет полностью избавиться от проводов между пользователем и компьютером, что существенно повышает эргономику и позволяет пользователю беспрепятственно перемещаться по учебному классу.

Для создания 3D-моделей оборудования ГРП использовалась среда 3D-моделирования Blender – на данный момент самая удобная и понятная среда среди всех существующих бесплатных решений на рынке. Для разработки основного функционала приложения использовался игровой движок Unity3D.

Погружение в виртуальную реальность позволяет полностью сосредоточиться на обучающем материале, не отвлекаясь на внешние раздражители. Можно смоделировать различные ситуации, при отработке которых специалист получит определенный навык без малейшего вреда для здоровья.

Достоинства применения технологий виртуальной реальности в обучении: наглядность. Используя 3D-графику, можно не только дать сведения о самом явлении, но и продемонстрировать его с любой степенью детализации; безопасность. Отработка определенных ситуаций без малейших угроз для жизни; вовлечение. Виртуальный мир, который окружит зрителя со всех сторон на все 360 градусов, позволит целиком сосредоточиться на материале и не отвлекаться на внешние раздражители [2].

Многие крупные компании успешно используют виртуальную реальность в обучении производственным процессам. Так концерн Volkswagen внедряет специализированные приложения виртуальной реальности для осуществления операций логистики [3]. Siemens установили системы и программное обеспечение виртуальной реальности, направленные на разработку и производство приводов с регулиру-

емой частотой вращения для двигателей [4]. В московском аэропорту «Домодедово» при помощи данной технологии обучают сотрудников действиям в условиях экстренных ситуаций [5].

Исходя из всего вышперечисленного, можно сделать вывод о том, что виртуальная реальность сегодня находит свое применение абсолютно во всех областях жизнедеятельности человека и успешно используется в обучении специалистов различных сфер.

ЛИТЕРАТУРА

1. Next Space. [Электронный ресурс]. – Режим доступа: <https://next.space/>. – Дата доступа: 15.09.2018.
2. VR geek – новости виртуальной реальности. [Электронный ресурс]. – Режим доступа: <https://www.vrgeek.ru/>. – Дата доступа: 15.09.2018.
3. Автосалон Фольксваген – автоцентр Атлант-М [Электронный ресурс]. – Режим доступа: <https://www.atlantmotors.by/>. – Дата доступа: 16.09.2018.
4. Cybersense – VR-AR-MR-технологии [Электронный ресурс]. – Режим доступа: <https://www.cybersense.ru/>. – Дата доступа: 16.09.2018.
5. Домодедово – Московский аэропорт [Электронный ресурс]. – Режим доступа: <https://www.domodedovo.ru/>. – Дата доступа: 16.09.2018.

УДК 004.5;004.921

СОЗДАНИЕ ПРИЛОЖЕНИЯ ВИРТУАЛЬНОЙ РЕАЛЬНОСТИ «ГАЗОРЕГУЛЯТОРНЫЙ ПУНКТ VR: РАЗРАБОТКА 3D-МОДЕЛЕЙ И СКРИПТОВ ЛОГИКИ ПРИЛОЖЕНИЯ»

А.А. АНАНЕНКО

(Представлено: канд. техн. наук, доц. И.Б. БУРАЧЕНОК)

Рассматриваются результаты разработки основных 3D-элементов для приложения виртуальной реальности «Газорегуляторный пункт VR: разработка 3D-моделей и скриптов логики приложения. Описаны основные этапы разработки приложения и основное назначение разработанных 3D-моделей. Приведено описание и тексты листингов вспомогательных скриптов и скриптов логики приложения».

Введение. В последнее время широкое распространение в различных сферах деятельности человека получили VR–технологии (технологии виртуальной реальности). Они активно используются в игровой индустрии, в нефтедобыче, промышленности, образовании, медицине и во многих других отраслях.

Говоря о VR как о способе обучения можно сказать, что данный способ позволяет не только закрепить теоретические знания обучаемых, но и психологически подготовить их к определенным ситуациям. Несмотря на то, что реальность виртуальная, человек воспринимает все увиденное как действительность.

ДВ представленной статье приведем описание результатов разработки одного из модулей приложения виртуальной реальности на примере реального газорегуляторного пункта (ГРП) на базе УП «Витебскоблгаз». Приложение позволяет обучить и проконтролировать теоретические и практические навыки специалистов по обслуживанию и ремонту объектов газораспределительной системы и газопотребления, а также ознакомиться с установленным на объектах оборудованием и его особенностях прежде, чем допускать к реальным работам.

Создание 3D-моделей. Первым и одним из самых важных этапов при создании приложения «Газорегуляторный пункт VR: создание 3D-моделей и скриптов логики приложения», является этап создания точных 3D-моделей оборудования ГРП, а так как приложение является обучающим, то точность моделей газового оборудования должна быть максимальной.

Для разработки моделей была выбрана среда Blender – свободное и открытое программное обеспечение для создания трёхмерной компьютерной графики, включающий в себя средства моделирования, анимации, рендеринга, постобработки и монтажа видео со звуком, компоновки с помощью «узлов» (Node Compositing), а также для создания интерактивных игр. В настоящее время пользуется наибольшей популярностью среди бесплатных 3D-редакторов в связи с его быстрым и стабильным развитием, которому способствует профессиональная команда разработчиков [1].

В ходе разработки приложения были созданы следующие модели:

- модель предохранительно-сбросного клапана;
- модель инвентарной заглушки на входе и выходе ГРП;
- модель жидкостного манометра;
- модель вентиля.

Предназначение 3D-моделей. Предохранительно-сбросной клапан предназначен для изменения давления газа: его понижения. Это одно из самых важных устройств на ГРП.

Для перекрытия хода газа во время проведения работ на ГРП используются инвентарные заглушки на входе и выходе ГРП.

Жидкостной манометр предназначен для измерения давлений, разрежений и разности давлений неагрессивных газов.

Вентиль – запорное устройство в трубопроводах, предназначенное для перекрытия и регулирования потоков жидкости, пара или газа.

Результат разработки 3D-моделей показан на рисунке 1.

Создание вспомогательных скриптов логики приложения. Так как разработанное приложение является обучающим то, действия пользователя в приложении должны четко соответствовать определенной технологической инструкции по работе на ГРП, например, пуску и продувке ГРП. Также необходимо показать определенные моменты этого процесса для более детального представления

и лучшего понимания того, что происходит в данный момент, например, в каких участках находится газ и т.п. Для этого были написаны следующие скрипты:

- скрипт извлечения инвентарных заглушек на входе и выходе ГРП;
- скрипт настройки предохранительно-сбросного клапана;
- скрипт визуализации заполнения труб газом.

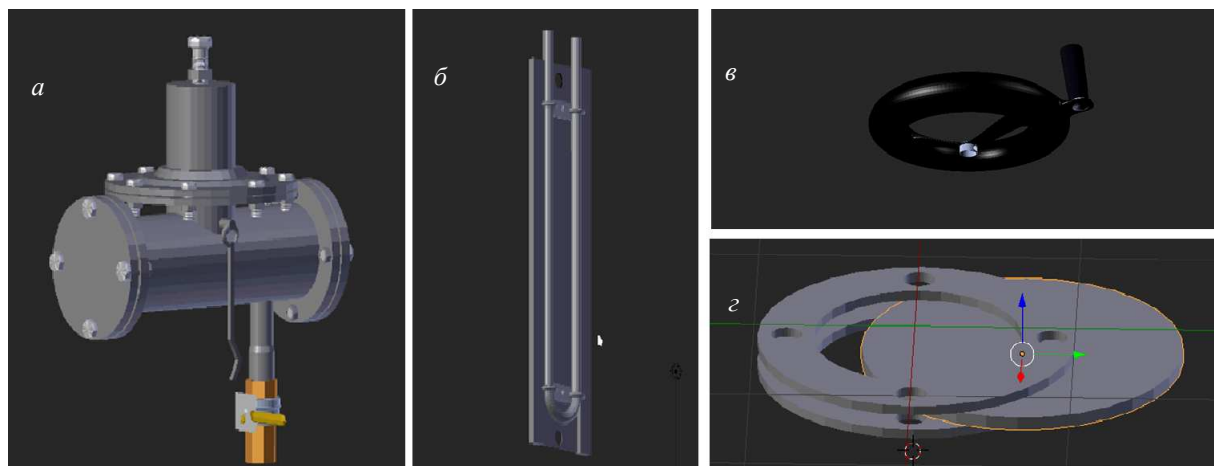


Рисунок 1. – Результат разработки 3D-моделей:

a – предохранительно-сбросной клапан; *б* – жидкостной манометр;
в – вентиль; *г* – инвентарные заглушки на входе и выходе ГРП

Так как аппаратное обеспечение программного продукта работает только под операционной системой Windows, то для объединения всех 3D-моделей в единую сцену и запуска ее в очках виртуальной реальности был выбран игровой движок Unity. Еще одним важным фактором в его пользу является то, что все SDK для работы с аппаратным обеспечением легко доступны для Unity, в отличие от других игровых движков, имеющих на рынке.

Скрипт извлечения инвентарных заглушек на входе и выходе ГРП

Согласно алгоритма пуска газа в ГРП одним из самых первых действий является извлечение заглушек на входе и выходе из ГРП. Для выполнения этого действия были поставлены триггеры в определенные места ГРП, при вхождении в которые пользователь получает визуальный сигнал (изменение цвета инвентарной заглушки на зеленый) о том, что можно начать извлечение заглушки. Само извлечение заглушек происходит посредством передачи имени объекта на сцене скрипту в инспекторе. Далее для начала извлечения необходимо сжать виртуальную руку в кулак, после чего объект (заглушка) следует за ней и как только рука выходит из зоны взаимодействия – заглушка исчезает. Удаление инвентарной заглушки подробно демонстрирует листинг 1.

Листинг 1 – Удаление инвентарной заглушки

```

1: void Update(){
2:   Renderer renderOgj = toRotate.transform.GetComponent<Renderer> ();
3:   if (onTriger == true){
4:     handOn = false;
5:     OVRInput.Update ();
6:     if ((OVRInput.Get (OVRInput.NearTouch.SecondaryThumbButtons)) && (OVRInput.Get (OVRInput.Axis1D.SecondaryIndexTrigger) > 0)
7:       && (OVRInput.Get (OVRInput.Axis1D.SecondaryHandTrigger) > 0) && !handOn) {
8:       handOn = true;}
9:     if (handOn){
10:      getRotation = true;
11:      forDelete = true;}
12:     else if (!handOn) {
13:      renderOgj.material.color = Color.green;
14:      getRotation = false;

```

```

15: stopVibration();}
16: if (getRotation) funcOfMove();}
17: if(onTriger == false && forDelete == true && handOn == true){
18: funcOfDessappear();
19: stopVibration();
20: deleteOfStub = true; //меняется состояние удаления заглушки: true - заглушка удалена
21: EventManager.TriggerEvent("DeleteSTUB" + toRotate.gameObject.name + deleteOfStub);
22: Debug.Log("DeleteSTUB" + toRotate.gameObject.name + deleteOfStub);} }

```

Скрипт настройки предохранительно-сбросного клапана. В реальной жизни настройка предохранительно-сбросного клапана (ПСК) является очень сложной задачей и для полноценной ее реализации, необходимо было бы написать отдельное приложение виртуальной реальности.

Так как данное действие является неотъемлемой частью алгоритма, было решено упростить его: для настройки ПСК следует навести на него руку, если он не настроен, он подсветится желтым цветом, далее следует сжать кулак, когда элемент будет активен. После этого цвет ПСК плавно поменяется на зеленый и пользователь услышит определенный звук, что означает завершение настройки ПСК. Процесс настройки предохранительно-сбросного клапана подробно демонстрирует листинг 2.

Листинг 2 – Настройка ПСК

```

1: void Update(){
2: Renderer renderOgj = toRotate.transform.GetComponent<Renderer> ();
3: if (onTriger == true){
4: handOn = false;
5: OVRInput.Update ();
6: if ((OVRInput.Get (OVRInput.NearTouch.SecondaryThumbButtons)) && (OVRInput.Get
(OVRInput.Axis1D.SecondaryIndexTrigger) > 0) && (OVRInput.Ge
7: (OVRInput.Axis1D.SecondaryHandTrigger) > 0) && !handOn) {
8: handOn = true;}
9: if (handOn) getRotation = true;
10: else if (!handOn) {
11: renderOgj.material.color = Color.yellow;
12: getRotation = false;
13: if(toVibrate)
14: OVRInput.SetControllerVibration(0f, 0f, OVRInput.Controller.RTouch);}
15: if (getRotation){
16: func(renderOgj);
17: activeStateOfPSK = true;}} }

```

Скрипт визуализации заполнения труб газом

Скрипт визуализации заполнения труб газом принимает объекты участка труб, в которые должен попасть газ, после поворота определенного запорного устройства. Если запорный клапан был повернут, не нарушая определенный порядок действий, то участок подсветится голубым цветом и начнет плавно мигать, в противном случае – подсветки участка не произойдет. Этот эффект реализован посредством скрипта, представленного в листинге 3.

Листинг 3 – Изменение цвета трубы

```

1: void changeColor(GameObject obj)
2: {
3: for(int i = 0; i < obj.transform.childCount; i++)
4: {
5: childTmp = obj.gameObject.transform.GetChild(i);
6: Renderer render = childTmp.GetComponent<Renderer>();
7: render.material.color = Color.Lerp(Color.yellow, Color.cyan, Mathf.PingPong(Time.time, 2f));
8: }
9: childTmp = null;
10: }

```

Результаты работы скриптов, представленных в листингах 1–3 показаны на рисунке 2.

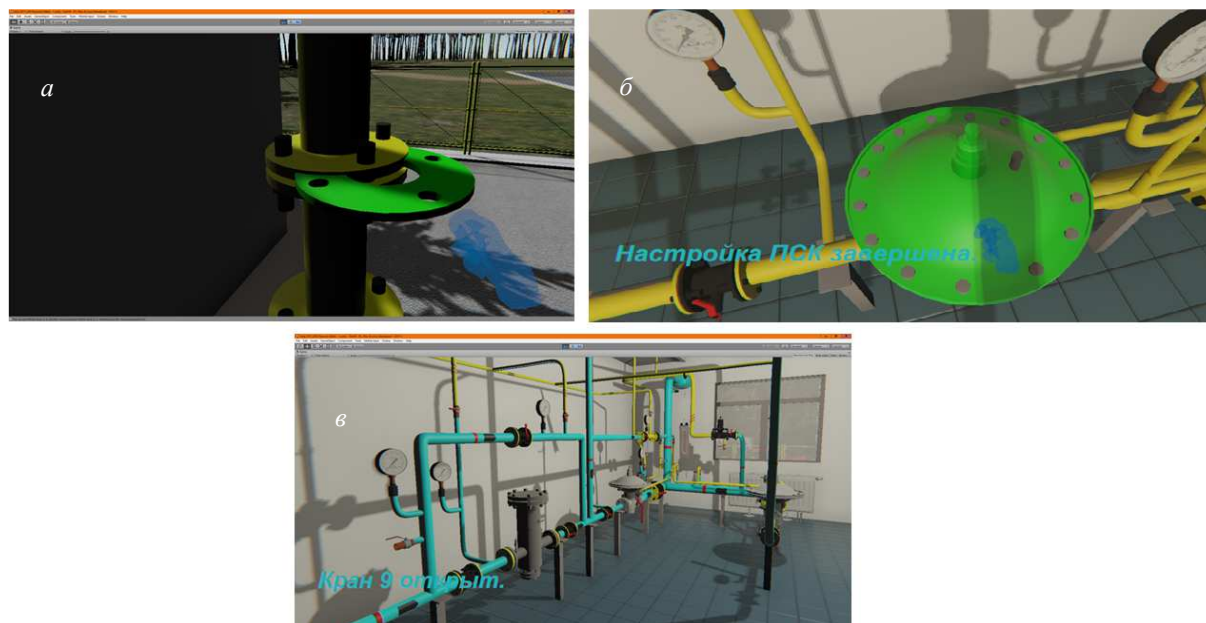


Рисунок 1. – Результат работы скриптов:

- a* – извлечение инвентарной заглушки;
- б* – настройка предохранительно-сбросного клапана;
- в* – визуализации заполнения труб газом

Заключение. Таким образом, представленный в статье опыт разработки 3D-элементов в процессе реализации приложения виртуальной реальности на примере настройки и пуска газорегуляторного пункта на базе УП «Витебскоблгаз» можно использовать и в других отраслях, что позволит осуществлять моделирование различных ситуаций, при отработке которых специалист получит определенный навык без малейшего вреда для здоровья и угроз для жизни. Соблюдение описанного в статье процесса создания приложения виртуальной реальности открывает новые возможности обучения персонала и позволяет исключить сложности его обучения, возникающие при традиционном подходе: дорогостоящем и затратном по времени.

ЛИТЕРАТУРА

1. Википедия. Свободная энциклопедия [Электронный ресурс]. – Режим доступа: <http://ru.wikipedia.org/wiki/Blender/>. – Дата доступа: 20.09.2018.

УДК 004.021

**ПРОЕКТИРОВАНИЕ ГРАФИЧЕСКОГО ИНТЕРФЕЙСА
ПРИЛОЖЕНИЯ ГИБРИДНАЯ КРИПТОСИСТЕМА****А.В. СУББОТИН***(Представлено: канд. физ.-мат. наук, доц. Ю.Ф. ПАСТУХОВ)*

Рассматривается проектирование графического интерфейса приложения для защиты пользовательских данных. Проведён анализ подходов проектирования графического интерфейса, наиболее подходящих для разработки данного приложения. Задача: реализация прототипа интерфейс.

Введение. Интерфейс пользователя - это та часть программы, которая находится у всех на виду. Интерфейс имеет важное значение для любой программной системы и является неотъемлемой ее составляющей, ориентированной на конечного пользователя. Именно через интерфейс пользователь судит о прикладной программе в целом; более того, часто решение об использовании прикладной программы пользователь принимает по тому, насколько ему удобен и понятен пользовательский интерфейс.

Программа должна помогать выполнить задачу, а не становиться этой задачей. Интерфейс должен быть легким для освоения и не создавать перед пользователем преграду, которую он должен будет преодолеть, чтобы приступить к работе.

Выбор технологий разработки интерфейса. Важной характеристикой в визуальной среде является дизайн пользовательского интерфейса, интуитивность и удобство навигации.

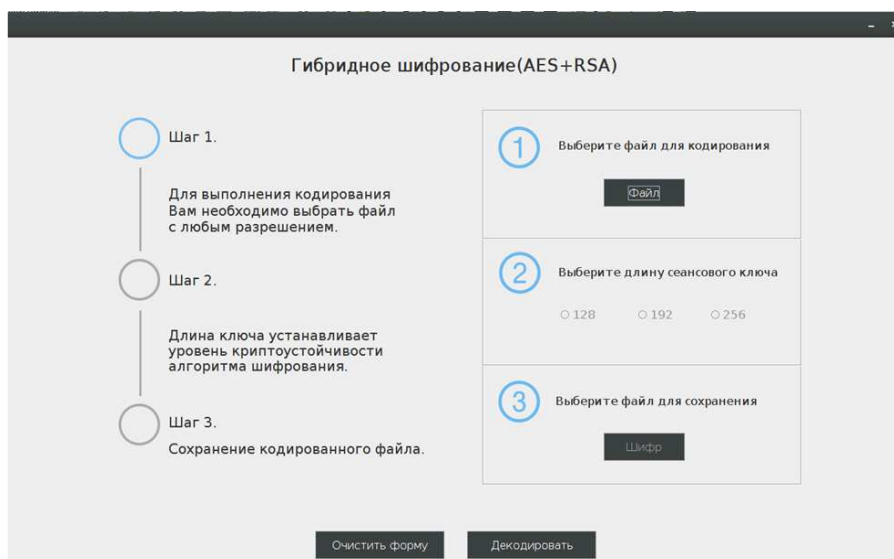
Для разработки программного средства организации и функционирования программы был выбран язык программирования Java.

Java – платформонезависимый язык программирования. Поддерживает объектно-ориентированный, императивный и функциональный стили.

Java – архитектурно-нейтральный и не имеющий зависимости от реализации аспектов спецификаций. Программирование на Java считается динамичным, так как он предназначен для адаптации к меняющимся условиям. Программы могут выполнять обширное количество операций во время обработки информации, которая может быть использована для проверки и разрешения доступа к объектам на время выполнения.

Кроме того, в данной работе, была использована библиотека Swing. Эта библиотека предоставляет набор готовых элементов пользовательского интерфейса для использования в Java программах и апплетах. Благодаря отдельной архитектуре модель-вид, встроенной поддержки HTML, богатому набору интерфейсных примитивов созданные с помощью библиотеки Swing приложения будут отлично работать и выглядеть на различных платформах.

Проектирование интерфейса. Интерфейс пользователя включает в себя следующие окна: окно выбора файла для кодирования, окно выбора файла для сохранения зашифрованной информации, окно выбора файла для декодирования (рис. 1).

**Рисунок 1. – Интерфейс программы**

Функциональными элементами являются выбор длины ключа и очистка формы.

Выбор длины сеансового ключа, а также выбор файла для записи закодированной информации неактивны до тех пор, пока не будет выбран файл для шифрования, что и отражено на рисунке 2.

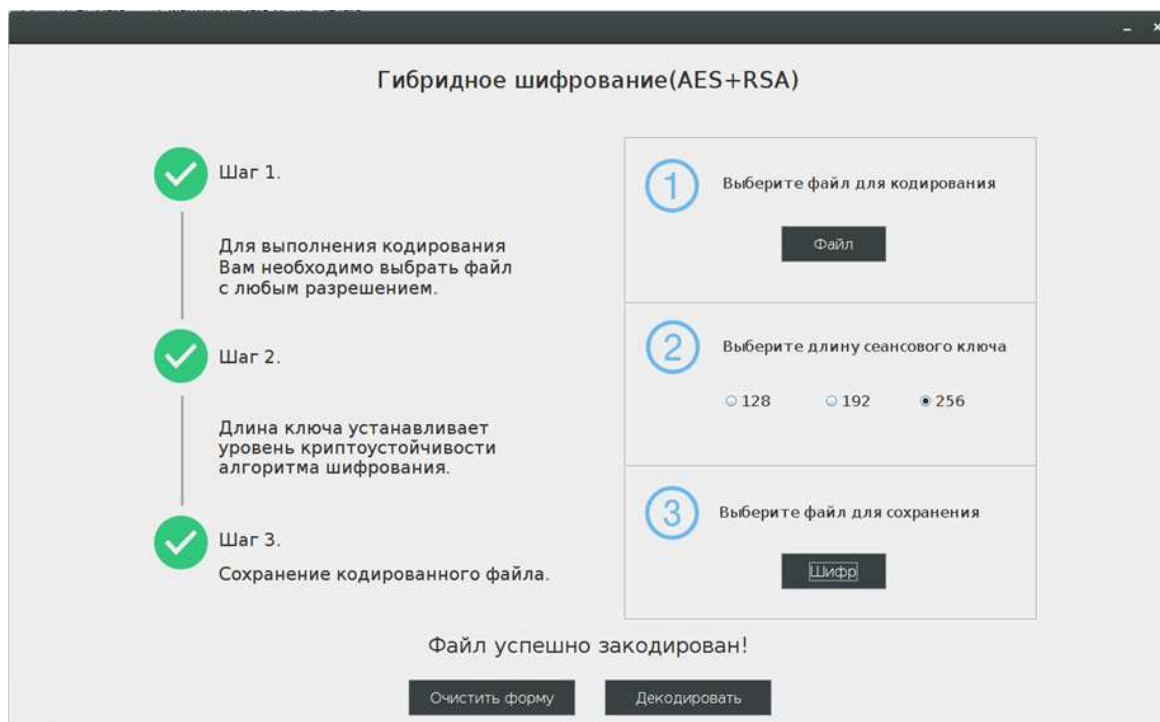


Рисунок 2. – Процесс шифрования данных

Для удобства декодирования в зашифрованный файл передается длина сеансового ключа (рис. 3). Необходимо отметить, что изменение случайного числа до момента дешифрования делает невозможным само дешифрование (рис. 4).



Рисунок 3. – Процесс декодирования данных



Рисунок 4. – Декодирование измененных данных

Заключение. Рассмотренные технологии Java и Swing, позволяют создать адаптивный интерфейс любого уровня сложности. В ходе данной работы был проведен анализ возможности использования данных технологий для проектирования графического интерфейса пользователя.

ЛИТЕРАТУРА

1. Материал из Википедии – свободной энциклопедии. Графический интерфейс пользователя [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/Графический_интерфейс_пользователя. – Дата доступа: 22.09.2018.
2. Материал из Википедии — свободной энциклопедии. Java [Электронный ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki/Java>. – Дата доступа: 22.09.2018.

УДК 004.021

**РАЗРАБОТКА ГИБРИДНОЙ КРИПТОСИСТЕМЫ
ДЛЯ ЗАЩИТЫ ПОЛЬЗОВАТЕЛЬСКИХ ДАННЫХ****А.В. СУББОТИН***(Представлено: канд. физ.-мат. наук, доц. Ю.Ф. ПАСТУХОВ)*

Рассматривается проектирование гибридной криптосистемы для защиты данных. Проведён анализ технологий, наиболее подходящих, для разработки данной схемы.

Введение. Важнейшим требованием к системе шифрования является стойкость данной системы. К сожалению, повышение стойкости при помощи любого метода приводит, как правило, к трудностям и при шифровании открытого текста и при его расшифровке. Одним из наиболее эффективных методов повышения стойкости шифротекста является метод комбинированного шифрования. Этот метод заключается в использовании и комбинировании нескольких простых способов шифрования.

Основная часть. Степень изученности методов шифрования достаточно высокая. Каждый год создается множество программ, литературы, посвящённые криптографическим системам. Криптографическая система – семейство преобразований шифра и совокупность ключей. Существуют симметричные и асимметричные криптосистемы.

Симметричные криптосистемы (с секретным ключом) – данные криптосистемы построены на основе сохранения в тайне ключа шифрования. Процессы шифрования и расшифровки используют один и тот же ключ. Секретность ключа является постулатом.

Асимметричные криптосистемы (системы открытого шифрования – с открытым ключом) – смысл данных криптосистем состоит в том, что для шифрования и расшифровки используются разные преобразования. Одно из них – шифрование – является абсолютно открытым для всех. Другое же – расшифровка – остается секретным [1].

На данный момент чаще используются достаточно стойкие системы, системы с достаточно сложным алгоритмом шифрования. Из-за необходимости различных объектов зашифровывать секретные данные и криптографические системы не стоят на месте и постоянно совершенствуются.

Выбор алгоритмов шифрования. На основании анализа наиболее криптоустойчивых алгоритмов были сделаны следующие выводы:

– шифрование информации с помощью симметричного алгоритма AES, так как, несмотря на недостатки, взломать защищенную с помощью этого алгоритма информацию практически нереально. Суть AES в том, что любая «лобовая атака» на защищенные данные – то есть подбор всех возможных паролей – в перспективе очень сильно растягивается. Если представить, что взломщик располагает огромными ресурсами, то есть целой коллекцией суперкомпьютеров, то при усердном старании доступ к зашифрованным данным он мог бы получить через десятки лет.

– шифрование сеансового ключа с помощью асимметричным алгоритмом RSA-OAEP, так как это не только возведение в степень по модулю большого числа. Это еще и добавление избыточных данных позволяющих реализовать дополнительную защиту вашей информации [2].

Структурная схема протокола обмена данными. Пусть два абонента договорились обмениваться данными. Схема, показанная на рисунке, предполагает наличие у каждого участника информационного обмена двух ключей: открытого PK и закрытого SK. Рассмотрим процесс пересылки некоего документа M. Отправитель (абонент А) вырабатывает секретный ключ – случайное число, используемое только один раз и поэтому называемое одноразовым или сеансовым ключом. Этот ключ используется для шифрования документа M при помощи симметричного криптоалгоритма. Сеансовый ключ зашифровывается на открытом ключе получателя (абонент В) и присоединяется к ранее зашифрованному документу. Сформированное сообщение отсылается получателю. Последний, получив сообщение, повторяет те же процедуры, но в обратном порядке. С помощью своего секретного ключа получатель восстанавливает сеансовый ключ, а затем с его помощью расшифровывает и сам документ.

Выбор длины сеансового ключа. Вторым шагом является выбор длины сеансового ключа. От размера ключа зависит число раундов шифрования:

- длина 128 бит – 10 раундов;
- длина 192 бита – 12 раундов;
- длина 256 бит – 14 раундов.

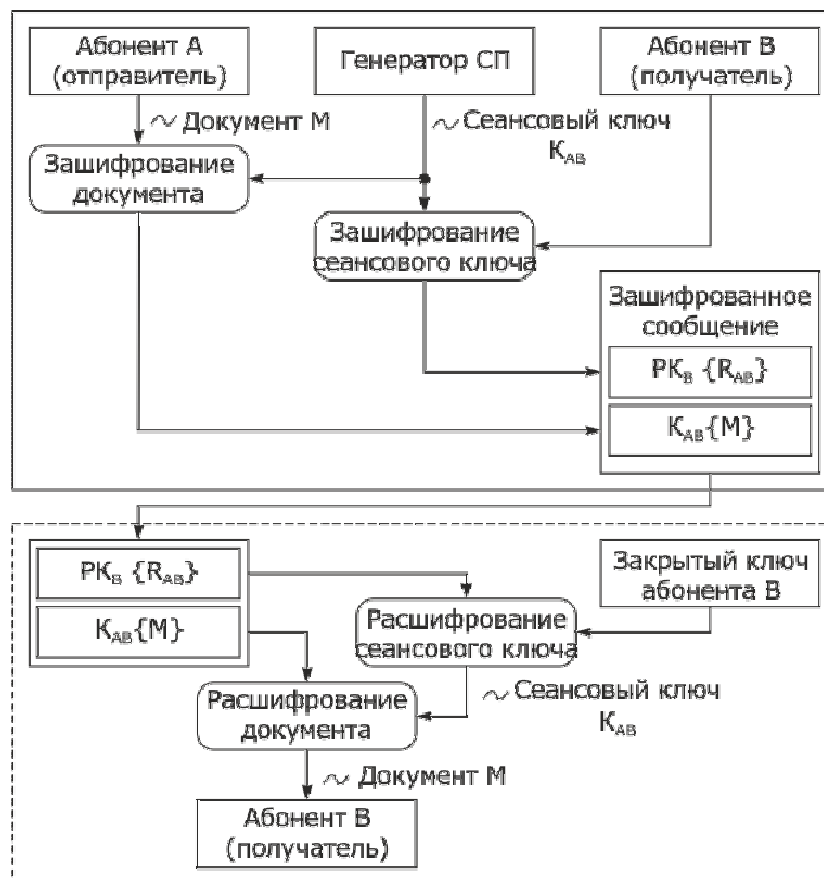


Рисунок. – Схема обмена двух ключей

В качестве примера примем длину ключа, равную 128 бит. Входными данными для операций шифрования есть массив из 16 байт. Перед началом шифрования байты этого массива размещаются последовательно в столбцы матрицы. Внутри алгоритма операции выполняются над матрицей байт, называемой матрицей состояний. Конечное значение матрицы состояния является выходом алгоритма и преобразуется в последовательность байтов шифротекста. Аналогично в столбцы исходной матрицы попадают и 16 байтов ключа шифра. Размерность всех матриц – 4×4 . Четыре байта в каждом столбце матрицы состояний или ключа можно рассматривать как одно 32-х битовое слово. Поэтому матрица состояний – это массив из 4 слов. Матрица, поступающая на вход каждого раунда, называется матрицей входящего состояния, а на выходе раунда образуется матрица выходящего состояния [3].

Заключение. В ходе данного исследования был спроектирована гибридная криптосистема для защиты пользовательских данных с использованием алгоритмов AES, RSA-OAEP. При этом разработанная схема оставляет возможность для доработки и введения дополнительных средств защиты.

ЛИТЕРАТУРА

1. Мао Венбо, Современная криптография: теория и практика : пер. с англ. / Мао Венбо. – М. : Издат. дом «Вильямс», 2005.
2. Pointcheval, D. HD-RSA: hybrid dependent RSA, a new public-key encryption scheme. Submission to IEEE P1363: A symmetric Encryption / D. Pointcheval. – 1999.
3. Петров, А.А. Компьютерная безопасность. Криптографические методы защиты ДМК / А.А. Петров. – 2000.

УДК 004.021

**ПРОЕКТИРОВАНИЕ ГРАФИЧЕСКОГО ИНТЕРФЕЙСА СИСТЕМЫ
ДЛЯ СКРЫТИЯ ИНФОРМАЦИИ НА ОСНОВЕ АЛГОРИТМА
ПОСЛЕДОВАТЕЛЬНОЙ СТЕГАНОГРАФИИ****И.С. АНДРЕЕВ***(Представлено: канд. физ.-мат. наук, доц. Ю.Ф. ПАСТУХОВ)*

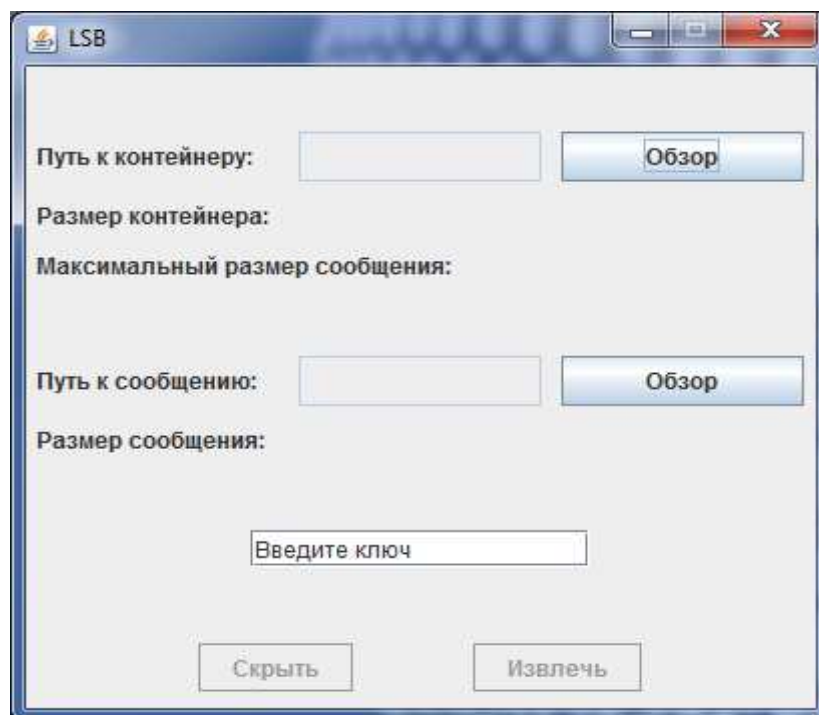
Рассматривается проектирование графического интерфейса системы для скрытия информации на основе алгоритма последовательной стеганографии. Проведён анализ степени пригодности контейнеров для модификации, моделирование атак, определение устойчивости к атакам и эффективности алгоритма.

Введение. Развитие человечества, обусловленное его стремлением к обеспеченной и безопасной жизни, всегда было в целом поступательным. Прогресс, расширение кругозора и углубление знаний постепенно улучшали качество жизни людей, их возможности, и увеличивало рост их потребностей и требований. По этой простой причине на сегодняшний момент в мире обеспечение и поддержание требуемого уровня безопасности является актуальным вопросом, и потому на его решение тратится немало средств, времени и сил. Одним из методов обеспечения безопасности хранимых и передаваемых файлов является преобразование их в стегоконтейнеры.

Приложение реализующее преобразование файлов в стегоконтейнеры должно обладать удобным и понятным графическим интерфейсом.

Основная часть. Любое приложение должно быть грамотно спроектировано и разделено на отдельные модули, которые должны быть относительно независимыми друг от друга. Подобное разделение значительно облегчает не только реализацию приложения, но и возможную его модификацию. В этом заключается принцип модульности объектно-ориентированного программирования.

Приложение «LSB» представляет собой программу для скрытия информации в изображениях. Для корректной работы программы необходимо выбрать контейнер (одного из следующих форматов: bmp, png, jpg, jpeg), выбрать скрываемый файл и ввести особый ключ(который будет необходим для извлечения).

**Рисунок 1. – Интерфейс программы**

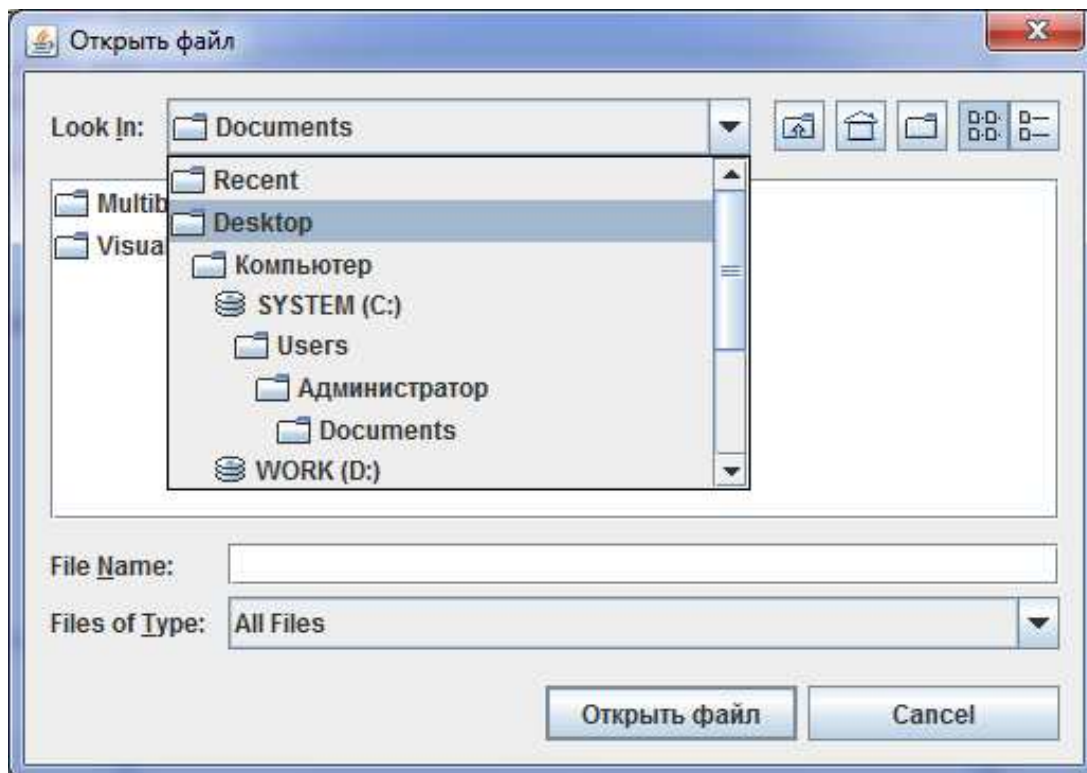


Рисунок 2. – Выбор файла для шифрования

На основании проведенного исследования, учитывая все факторы процесса разработки и работы готового программного продукта наиболее оптимальным языком программирования для реализации, был выбран язык программирования Java.

С целью обнаружения ошибок, проверки реализации функциональных и нефункциональных требований к программе, было проведено тестирование разработанной автоматизированной системы.

При проведении испытаний был задействован персональный компьютер, на котором установлен требуемый перечень программных продуктов, и отвечающий требованиям к составу аппаратных средств. На основании этого документа было проведено тестирование.

При атаках было выявлено, что алгоритм имеет устойчивость к большинству известных стеганоатак, в том числе к атаке сжатием, к аффинным преобразованиям, геометрическим атакам.

В ходе исследований было определено, что человеческий глаз более чувствителен к шуму при низкой интенсивности цвета.

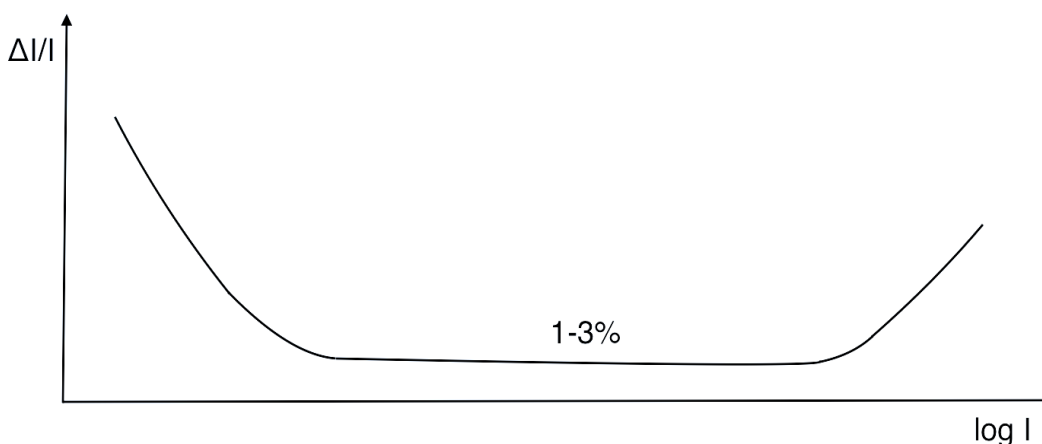


Рисунок 3. – Зависимость контраста от яркости

В ходе тестирования было выявлено что предложенный в метод позволяет произвести графическое сжатие до 10%.

Кроме того, было проведено подробное исследование для выбора среды разработки. Исходя из представленных фактов, для проектирования выбрана среда Eclipse, как оболочка для быстрой разработки качественного интерфейса.

Заключение. В данной статье рассмотрен способ построения графического интерфейса системы скрытия информации на основе алгоритма последовательной стеганографии, проведён анализ степени пригодности контейнера для модификации, моделирование атак и определение устойчивости к ним.

ЛИТЕРАТУРА

1. Стеганографические системы. Критерии и методическое обеспечение : учеб.-метод. пособие / под ред. В.Г. Грибунина [и др.]. – Саратов, 2016.
2. Коржик, В.И. Лекции по основам стеганографии / В.И. Коржик.

УДК 004.021

**РАЗРАБОТКА СИСТЕМЫ ДЛЯ СКРЫТИЯ ИНФОРМАЦИИ
НА ОСНОВЕ АЛГОРИТМА ПОСЛЕДОВАТЕЛЬНОЙ СТЕГАНОГРАФИИ****И.С. АНДРЕЕВ***(Представлено: канд. физ.-мат. наук, доц. Ю.Ф. ПАСТУХОВ)*

Рассматриваются: алгоритм скрытия информации в изображении на основе последовательной стеганографии, описание работы основной программы и основные функции разрабатываемого программного продукта.

Введение. Современный этап в развитии обмена информацией, который характеризуется интенсивным внедрением современных информационных технологий, широким распространением локальных, корпоративных и глобальных сетей во всех сферах жизни цивилизованного государства, создает новые возможности и качество информационного обмена. И поэтому проблемы информационной безопасности постоянно усугубляются процессами проникновения во все сферы общества технических средств обработки и передачи данных и, прежде всего, компьютерных системах. Для решений этих проблем и используют стеганографию, задача которой – передать данные так, чтобы противник не догадался о самом факте появления сообщения.

Основная часть. Цель работы – изучение стеганографического метода скрытия информации в изображениях как с точки зрения устойчивости к различным видам атак, так и с точки зрения сохранения приемлемого качества изображения.

Алгоритм последовательной стеганографии заключается в последовательном встраивании одного изображения в другое, что позволяет достичь большего коэффициента сжатия.

Основными функциями разрабатываемого программного продукта являются:

- шифрование информации;
- дешифрование информации;
- вывод промежуточной информации о выполненных операциях.

Рассматриваемая методика ляжет в основу разрабатываемого программного продукта «Скрытие информации на основе дискретных преобразований методом последовательной стеганографии», который реализует все задачи, в частности, главную – скрыть факт передачи информации.

Для скрытия информации методом последовательной стеганографии необходимо выполнить следующие пункты:

1. Изображение-контейнер разбивается на равные блоки.
2. Определяется размер встраиваемого сообщения и количество требуемых битов в контейнере для встраивания.
3. В каждом блоке происходит поиск наименее значащих бит.
4. Создается матрица встраивания (которая определяет в каких битах изображения будет происходить изменения).
5. С помощью алгоритма Блум-Блум-Шуба создается последовательность для матрицы встраивания.
6. Происходит замена наименее значащих бит в соответствии с матрицей встраивания.

Извлечение:

1. Изображение-контейнер разбивается на равные блоки.
2. С помощью специального кода из изображения извлекается информация о размере встроеного сообщения.
3. Создается матрица извлечения.
4. Происходит извлечение и сохранение изображения.

Обнаружение кодированного сообщения осуществляется по аномальным характеристикам распределения значений диапазона младших битов отсчетов цифрового сигнала.

Кодирование в программе реализовано следующим образом:

Листинг 1 – Описание функции, предназначенной для кодирования

1. `public void bitsToFile(ArrayList<Integer> bits, String filename, String ext, int byteCount) throws Exception {`
2. `System.out.println(bits.size()+" "+ filename+" "+ ext + " "+byteCount);`

```
3. byte[] bytes = new byte[byteCount];
4. int j = 0;
5. String str = "";
6. for (int i = bits.size() - byteCount * 8; i < bits.size(); i++) {
7.   str += bits.get(i);
8.   if (str.length() == 8) {
9.     int number = Integer.parseInt(str, 2);
10.    str = "";
11.    bytes[j] = (byte) number;
12.    j++;}
13. }
14. File fl = new File(filename);
15. String path = filename.replaceAll(fl.getName(), "");
16. path += "\\extractedMessage." + ext;
17. FileOutputStream fos = new FileOutputStream(new File(path));
18. fos.write(bytes);
19. fos.close();
20. result = new String[2];
21. result[0] = "true";
22. result[1] = path;
23. }
```

Декодирование в программе реализовано следующим образом:

Листинг 2 – Описание функции предназначенной для декодирования

```
1. public void loadMessage(String filename) throws Exception {
2.   FileInputStream fis = new FileInputStream(filename);
3.   byte[] fileBArray = new byte[fis.available()];
4.   fis.read(fileBArray);
5.   fis.close();
6.   String ex = filename.substring(filename.lastIndexOf(".") + 1);
7.   String str = ex + "." + fileBArray.length + "~";
8.   byte[] tmp = str.getBytes();
9.   int[] result = new int[fileBArray.length + tmp.length];
10.  int j = 0;
11.  for (int i = 0; i < tmp.length; i++) {
12.    if (tmp[i] < 0) {
13.      result[j] = 256 + tmp[i];
14.    } else {
15.      result[j] = tmp[i];
16.    }
17.    j++;
18.  }
19.  for (int i = 0; i < fileBArray.length; i++) {
20.    if (fileBArray[i] < 0) {
21.      result[j] = 256 + fileBArray[i];
22.    } else {
23.      result[j] = fileBArray[i];
24.    }
25.    j++;
26.  }
27.  message = result;
28. }
```

Для выбора порядка встраивания сообщения будет использоваться генератор ПСЧ «Алгоритм Блюм — Блюма — Шуба».

Заключение. В данной статье был рассмотрен алгоритм скрытия данных в изображении на основе последовательной стеганографии, описание работы основной программы и основные функции разрабатываемого программного продукта.

ЛИТЕРАТУРА

1. Blum, L. A Simple Unpredictable Pseudo-Random Number Generator / Lenore Blum, Manuel Blum, Michael Shub // *SIAM Journal on Computing*. – 1986. – Vol. 15. – P. 364–383.
2. Садов, В.С. Компьютерная стеганография (конспект лекций) / В.С. Садов. – Минск : Белорус. гос. ун-т, 2010.
3. Конахович, Г.Ф. Компьютерная стеганография. Теория и практика / Г.Ф. Конахович, А.Ю. Пузыренко. – К. : МК-Пресс, 2006. – 288.
4. Грибунин, В.Г. Цифровая стеганография / В.Г. Грибунин, И.Н. Оков, И.В. Туринцев. – М. : Солон-Пресс, 2002.

УДК 004.853

**ТЕНДЕНЦИИ РАЗВИТИЯ
СВЁРТОЧНЫХ НЕЙРОННЫХ СЕТЕЙ****А.А. СОЛОВЬЁВ***(Представлено: канд. физ.-мат. наук, доц. Ю.Ф. ПАСТУХОВ)*

Рассматриваются и анализируются основные идеи и новшества в реализации свёрточных нейронных сетей, а также их результаты на основании ежегодного международного проекта по широкомасштабному распознаванию образов ILSVRC.

В последние несколько лет область машинного обучения добилась огромного прогресса в решении задач, связанных с компьютерным зрением. В частности, было обнаружено, что свёрточные нейронные сети могут достигать разумной производительности по задачам визуального распознавания, сопоставимой или превышающей производительность человека в некоторых случаях.

Исследователи продемонстрировали устойчивый прогресс в компьютерном зрении, подтвердив свою работу с ImageNet — проектом по созданию и сопровождению массивной базы данных аннотированных изображений, предназначенная для отработки и тестирования методов распознавания образов и машинного зрения. С 2010 года ведётся проект ILSVRC (ImageNet Large Scale Visual Recognition Challenge), в рамках которого различные программные продукты ежегодно соревнуются в классификации и распознавания объектов и сцен в базе данных ImageNet [1]. Эту кампанию можно рассматривать как ежегодную Олимпиаду компьютерного зрения, в которой команды со всего мира конкурируют, чтобы увидеть, кто имеет лучшую модель компьютерного зрения для таких задач, как классификация, локализация, обнаружение и многое другое.

Основной метрикой ошибки на этой базе является так называемая Топ-5 ошибка. Это когда берутся первые пять предсказаний сети, в которых она наиболее уверена, и если правильный класс из разметки попал в эти пять предсказаний, то можно сказать, что сеть права. И ошибка — это когда в первые пять предсказаний не попал правильный класс. Чтобы сравнить модели, будет рассмотрено, как часто модель не может предсказать правильный ответ в качестве одной из своих пяти догадок.

В 2012 году Алексом Крижевским была представлена глубокая свёрточная нейронная сеть, которая стала победителем проекта ILSVRC. В статье [2] группа создателей обсудила архитектуру сети (которая называлась AlexNet). Они использовали относительно простую компоновку по сравнению с современными архитектурами. Сеть состояла из 5 свёрточных (convolutional) слоёв, слоёв с максимальным пулом (max pooling), слоёв отсева (dropout) и 3 полностью связанных (fully connected) слоёв. Созданная ими сеть использовалась для классификации 1000 различных классов.

Данная сеть обучалась на выборке, содержащей более 15 миллионов аннотированных изображений, более чем из 22 тысяч классов. Новаторством данной модели было использование активационной функции ReLU, что позволило значительно уменьшить время обучения сети, в отличие от ранее широко используемой функции гиперболического тангенса. Также в данной работе были использованы методы аугментации (augmentation) данных, что значительно увеличило объём обучающей выборки и улучшило обучение. Нововведением в борьбе с проблемой переобучения сети стали слои отсева нейронов.

Нейронная сеть AlexNet стала первой моделью СНС, которая показала отличные результаты на исторически сложном наборе данных ImageNet. Используя методы, которые всё ещё используются сегодня, такие как увеличение данных и отсева, данная модель действительно иллюстрирует преимущества СНС над другими методами классификации изображений.

После победы AlexNet модели на конкурсе ImageNet, СНС широко вошли в решения задач, связанных с компьютерным зрением, и число свёрточных нейронных сетей, представленных на конкурсе ILSVRC в 2013 году быстро возросло. Победителем конкурса в этом году была сеть ZF Net, которая достигла 11,2% ошибок.

В статье [3] создатели архитектуры ZF Net обсуждают идеи о том, что новый интерес к СНС обусловлен доступностью больших обучающих наборов и увеличением вычислительной мощности с использованием графических процессоров. Они также говорят о ограниченном знании, которое исследователи использовали во внутренних механизмах этих моделей, заявив, что без этого понимания разработка лучших моделей сводится к методу проб и ошибок. Основным вкладом в эту работу являются детали немного модифицированной модели AlexNet и очень интересный способ визуализации карт функций.

Архитектура ZF Net очень похожа на AlexNet, за исключением нескольких незначительных изменений. Основными отличиями было то, что ZF Net обучалась всего на 1,3 миллиона изображений. Вме-

сто использования фильтров размером 11x11 в первом слое (что и было реализовано в AlexNet) использовались фильтры размером 7x7 и уменьшенное значение шага. Обоснование этой модификации заключается в том, что меньший размер фильтра в первом свёрточном слое помогает сохранить много исходной информации о пикселях во входном томе. Фильтр размером 11x11 оказался плох тем, что пропускал много информации, что особенно плохо сказывалось на обучении, поскольку это первый слой СНС. Также в их работе была разработана технология визуализации с именем Deconvolutional Network, которая помогает исследовать различные активизации функций и их отношение к входному пространству.

ZF Net была не только победителем конкурса в 2013 году, но также продемонстрировала больше способов повышения производительности. Описанный ими подход к визуализации признаков помогает не только объяснить внутреннюю работу СНС, но также даёт представление об её улучшении.

Простота и глубина — это то, что модель, созданная в 2014 году, наилучшим образом демонстрировала с её частотой ошибок 7,3%. Карен Симонян и Эндрю Зиссерман из Оксфордского университета создали 19-слойную СНС, названную VGG Net [4], в которой использовались фильтры 3x3 с шагом и 1, что существенно отличается от фильтров 11x11 сети AlexNet в первом слое и фильтров 7x7 сети ZF Net. Авторы утверждают, что комбинация двух 3x3 свёрточных слоёв имеет эффективное восприимчивое поле 5x5. Это, в свою очередь, имитирует больший фильтр, сохраняя при этом преимущества меньшего размера фильтра. Одним из преимуществ их сети является уменьшение количества параметров. Кроме того, с двумя свёрточными слоями можно использовать два слоя ReLU вместо одного.

По мере уменьшения пространственного размера входных данных на каждом уровне глубина данных увеличивается из-за увеличения количества фильтров прохода по сети. Стоит заметить, что количество фильтров удваивается после каждого слоя максимального пулинга, что усиливает идею сокращения пространственных размеров и роста глубины сети.

VGG Net архитектура укрепила представление о том, что свёрточные нейронные сети должны иметь глубокую сеть слоёв, чтобы иерархическое представление визуальных данных работало.

Победителем в 2014 году с коэффициентом ошибок 6,7% стала 22-уровневая СНС сеть GoogLeNet [5]. Это была одна из первых архитектур СНС, которые действительно отклонялись от общего подхода простой укладки свёрточных слоёв и слоёв пулинга в последовательной структуре. В данной модели не использовались полносвязные слои, вместо этого они использовали средний пулинг, чтобы перейти от объёма 7x7x1024 к тому 1x1x1024, что сэкономило огромное количество параметров (в 12 раз меньше, чем в AlexNet).

Данная сеть была одной из первых моделей, которые представили идею о том, что слои СНС не всегда должны быть сложены последовательно. Авторы показали, что творческое структурирование слоев может привести к повышению производительности и эффективности вычислений.

Относительно новая 152-слойная сетевая архитектура ResNet выиграла ILSVRC 2015 с невероятной частотой ошибок 3,6% (в зависимости от их квалификации и опыта люди обычно колеблются с частотой ошибок 5-10%). Данная модель является по настоящему глубокой, однако авторы утверждают, что наивное увеличение уровней приводит к более высокой тренировочной и тестовой ошибкам.

Модель ResNet — это одна из лучших архитектур СНС, которую мы имеем в настоящее время.

Проанализировав рассмотренные СНС можно провести параллель между количеством слоёв и ошибкой сетей, представленных в таблице.

Таблица. — Показатели свёрточных нейронных сетей

Архитектура	Количество слоёв	Значение ошибки
AlexNet	8	15.4
ZF Net	8	11.2
VGG Net	19	7.3
GoogLeNet	22	6.7
ResNet	152	3.6

Проанализировав таблицу, можно сказать, что имеет место тенденция углубления архитектур СНС, что позволяет получать лучшие результаты по классификации изображений. Однако в новых статьях, опубликованных на научных конкурсах было описано использование моделей с намного большим количеством слоёв, которые показали результаты заметно хуже, чем та же модель ResNet, что говорит о том, что развитие СНС должно продолжаться не только в сторону увеличения глубины сети, но в улучшении старых и создании новых методов обучения.

ЛИТЕРАТУРА

1. ImageNet [Электронный ресурс]. — Режим доступа: <https://ru.wikipedia.org/wiki/ImageNet>. — Дата доступа: 24.09.2018г.
2. Krizhevsky, A. Imagenet classification with deep convolutional neural networks / Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton // NIPS. — 2012.
3. Zeiler, M. Visualizing and Understanding CNN / Matthew D. Zeiler, Rob Fergus // ECCV. — 2014.
4. Simonyan, K. Very Deep Convolutional Networks for Large-Scale Image Recognition / Karen Simonyan, Andrew Zisserman // arXiv technical report. — 2014.
5. Wei, L. Going deeper with convolutions / Christian Szegedy, Wei Liu, Pierre Sermanet, Scott Reed, Vincent Vanhoucke, Dragomir Anguelov, Andrew Rabinovich, Yangqing Jia, Dumitru Erhan // CVPR. — 2015.
6. Kaiming, H. Deep Residual Learning for Image Recognition / Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun // CVPR. — 2016.

УДК 004.853

БИБЛИОТЕКА МАШИННОГО ОБУЧЕНИЯ TENSORFLOW

А.А. СОЛОВЬЁВ

(Представлено: канд. физ.-мат. наук, доц. Ю.Ф. ПАСТУХОВ)

Рассматривается библиотека с открытым исходным кодом TensorFlow, предназначенная для решения задач построения и тренировки искусственных нейронных сетей, на примере классической задачи распознавания рукописных цифр.

TensorFlow — открытая программная библиотека для машинного обучения, разработанная компанией Google для решения задач построения и тренировки нейронной сети с целью автоматического нахождения и классификации образов, достигая качества человеческого восприятия [1, 2].

В качестве примера, на основе которого будут продемонстрированы некоторые возможности библиотеки, будет решаться классическая задача классификации рукописных цифр. В статье будут описаны основные шаги при взаимодействии с библиотекой, а также показаны результаты построенной и обученной нейронной сети.

В данной статье используется набор данных MNIST [3], который содержит 70 000 изображений рукописных цифр с низким разрешением (28 на 28 пикселей) в оттенках серого в 10 категориях (рис. 1). Этот набор данных относительно мал и используется для проверки того, что алгоритм работает так, как ожидалось. Он является хорошей отправной точкой для тестирования и отладки кода.

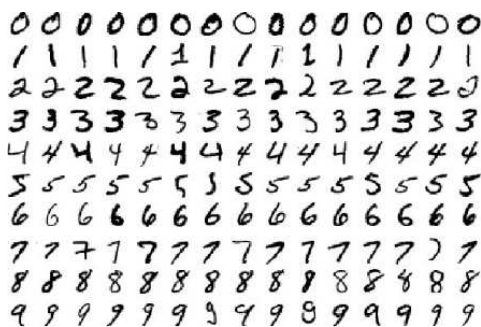


Рисунок 1. – База данных MNIST

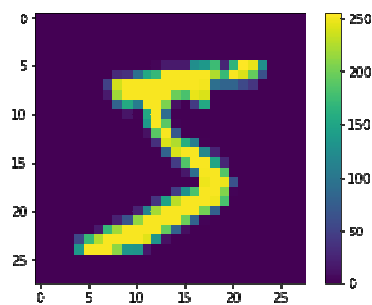


Рисунок 2. – График значений пикселей

Основными шагами, которые необходимо проделать для классификации изображений являются:

- Импорт набора данных MNIST.
- Предварительная обработка данных.
- Построение модели.
- Настройка слоев.
- Компиляция модели.
- Обучение модели.
- Оценка точности.
- Классификация.

В качестве примера будет использоваться 60 000 изображений для обучения сети и 10 000 изображений, для оценки того, насколько точно сеть научилась классифицировать изображения. Получить доступ к базе данных MNIST можно непосредственно из TensorFlow, просто импортировав и загрузив данные:

```
1: (train_images, train_labels), (test_images, test_labels) =
    keras.datasets.mnist.load_data()
```

Загрузка набора данных возвращает четыре массива NumPy [4]:

- Массивы train_images и train_labels — это наборы данных, которые использует модель для обучения.
- Модель тестируется на тестовом наборе, массивах test_images и test_labels.

Изображения представляют собой массивы 28x28 NumPy, значения пикселей которых находятся в диапазоне от 0 до 255. Метки представляют собой массив целых чисел от 0 до 9, которые соответствуют классу цифры. Каждое изображение отображается на одну метку.

Перед подготовкой сети данные должны быть предварительно обработаны. Если проверить первое изображение в наборе обучающих данных, то можно увидеть, что значения пикселей находятся в диапазоне от 0 до 255 (рисунок 2). Перед подачей изображений в модель нейронной сети необходимо значения всех пикселей выборки привести к диапазону от 0 до 1. Для этого значение каждого пикселя необходимо поделить на 255. Важно, чтобы наборы данных для обучения и тестов были предварительно обработаны таким же образом.

Построение нейронной сети требует настройки слоев модели, а затем компиляции модели.

Основным строительным блоком нейронной сети является слой. Слои извлекают представления из данных, подаваемых в них. Большая часть глубокого обучения состоит в объединении простых слоев. Большинство слоев, таких как `tf.keras.layers.Dense`, имеют параметры, которые изучаются во время обучения.

Первый слой нашей сети `tf.keras.layers.Flatten` преобразует формат изображений из 2d-массива (28 на 28 пикселей) в 1d-массив из $28 * 28 = 784$ пикселей. У этого слоя нет параметров для изучения; он только реформатирует данные.

После того, как пиксели сплюснены, сеть состоит из последовательности трёх слоёв `tf.keras.layers.Dense` и одного слоя `tf.keras.layers.Dropout`. Слой `Dense` — это плотно связанные или полностью связанные нейронные слои. Первый слой `Dense` содержит 256 узлов (или нейронов), а второй 128 узлов. Данные слои используют функции активации `ReLU`. После данных слоёв следует `Dropout` слой, который отсеивает нейроны для предотвращения переобучения нейронной сети. Последний полносвязный слой с 10-узлами содержит функцию активации `softmax`, который возвращает массив из десяти вероятностных оценок, сумма которых равна 1. Каждый узел содержит оценку, которая указывает вероятность того, что текущее изображение принадлежит одному из 10 классов.

```
1: model = keras.Sequential([
2:     keras.layers.Flatten(input_shape=(28, 28)),
3:     keras.layers.Dense(256, activation=tf.nn.relu),
4:     keras.layers.Dense(128, activation=tf.nn.relu),
5:     keras.layers.Dropout(0.2),
6:     keras.layers.Dense(10, activation=tf.nn.softmax)
7: ])
```

Прежде чем модель будет готова к обучению, ей потребуется еще несколько настроек. Они добавляются во время этапа компиляции модели:

- Оптимизатор влияет на то, как модель обновляется на основе данных, которые она видит.
- Функция потерь — это мера, которая указывает насколько будет точна модель во время обучения.
- Метрики — используются для контроля за этапами обучения и тестирования.

Чтобы начать обучение нейронной сети необходимо вызвать метод `model.fit`. Данный метод в качестве параметров принимает массив тренировочных изображений, их метки и количество тренировочных эпох.

Для оценки качества обученной нейронной сети предусмотрена функция `evaluate`, которая принимает массивы тестовых изображений и их меток. Данная функция возвращает значения, показывающие точность классификации сети на тестовых изображениях.

```
1: model.compile(optimizer=tf.train.AdamOptimizer(),
2:               loss='sparse_categorical_crossentropy',
3:               metrics=['accuracy'])
4:
5: model.fit(train_images, train_labels, epochs=5)
6:
7: test_loss, test_acc = model.evaluate(test_images, test_labels)
```

В нашем случае модель обучилась достаточно хорошо и имеет показатель точности 0,98.

Обученную модель можно использовать для прогнозирования изображений. Предсказание модели представляет собой массив из 10 чисел. Они описывают «уверенность» модели в том, что изображение

соответствует каждому из 10 разных цифр. Мы можем видеть, какая метка имеет наибольшее доверительное значение на рисунке 3.

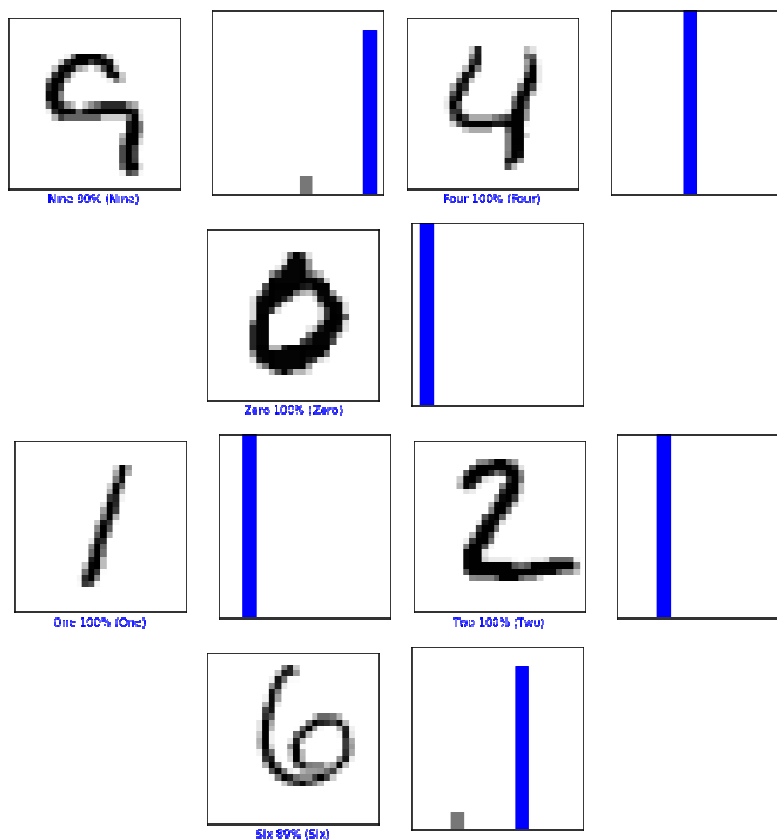


Рисунок 3. – Показатели уверенности обученной нейронной сети

На рисунке 3 можно увидеть, что модель уверенно определила рукописные цифры 4, 0, 1 и 2, и имела сомнения на примерах цифр 9 и 6. Данные сомнения сети возникли из-за не очевидного шрифта данных цифр.

В заключении статьи можно сказать, что рассмотренная библиотека TensorFlow обладает обширными методами и функциями построения нейронных сетей. Для построения и обучения моделей в TensorFlow используется `tf.keras` [6], API высокого уровня. В данной библиотеке реализованы функции для построения моделей любой сложности, а также содержащих различные слои с соответствующими параметрами. Вокруг библиотеки имеется огромное сообщество людей, которые делятся опытом создания нейронных сетей, а также помогают улучшать функционал самой библиотеки. TensorFlow является современным отличным инструментом в создании, обучении и использовании различных нейронных сетей.

ЛИТЕРАТУРА

1. TensorFlow [Электронный ресурс]. — Режим доступа: <https://ru.wikipedia.org/wiki/TensorFlow>. — Дата доступа: 24.09.2018.
2. Открытая программная библиотека TensorFlow [Электронный ресурс]. — Режим доступа: <https://www.tensorflow.org>. — Дата доступа: 24.09.2018.
3. База данных рукописных цифр MNIST [Электронный ресурс]. — Режим доступа: <http://yann.lecun.com/exdb/mnist/>. — Дата доступа: 24.09.2018.
4. Библиотека с открытым исходным кодом NumPy [Электронный ресурс]. — Режим доступа: <http://www.numpy.org>. — Дата доступа: 24.09.2018.
5. Открытая нейросетевая библиотека Keras [Электронный ресурс]. — Режим доступа: <https://keras.io>. — Дата доступа: 24.09.2018.
6. Графическая библиотека Matplotlib [Электронный ресурс]. — Режим доступа: <https://matplotlib.org>. — Дата доступа: 24.09.2018.

УДК 004.05

ПРОЕКТИРОВАНИЕ ИНТЕРНЕТ-МАГАЗИНА TELEGRAM-КАНАЛОВ**М.А. ПОТАПОВ***(Представлено: Т.С. РУДЬКОВА)*

Приводится обоснование необходимости создания веб-приложения, которое поможет приобрести Telegram-каналы различной тематики. Рассмотрены основные требования к проектированию, как функциональных возможностей, так и интерфейса веб-ресурса.

В современном мире, интернет-магазины являются удобным ресурсом для пользователей, так как позволяют быстро и комфортно заказывать товары и услуги прямо со своего компьютера, не выходя из дома. А это значит, что они перспективны и для самих владельцев данных магазинов.

Проанализировав существующие тенденции на современном рынке, можно сделать вывод, что создание интернет-магазина, который сможет осуществлять эффективные продажи Telegram-каналов весьма актуально, так как полных аналогов такого ресурса на данный момент не существует. Что подтверждает его уникальность в этом сегменте.

Так как интернет-магазин в первую очередь реализуют функцию витрины, которая представляет имеющийся товар или услугу, то необходимо разработать такое веб-приложение, которое позволяло бы размещать единицы товара, разделяя их как по категориям, так и ценам.

Поэтому на основании анализа предметной области определены основные требования к проектированию интернет-магазина Telegram-каналов:

- просмотр новостей на главной странице по тематике сайта;
- просмотр отзывов об интернет-магазине;
- просмотр видео-контента на главной странице сайта;
- выбор каналов по категориям;
- фильтрация каналов;
- покупка каналов;
- связь с администрацией сайта через раздел «О нас»;
- выбор каналов по тематике;
- поиск каналов на всех страницах сайта;
- администрирование сайта;
- возможность изменения контента с новостями на главной странице;
- изменение каталога товаров;
- анимированный дизайн;
- хлебные крошки;
- пагинация.

Функциональную структуру разрабатываемого ресурса можно представить в виде ряда подсистем (рис. 1):

- подсистема просмотра, редактирования и создания новостей (предоставляет возможность просмотра свежих новостей различной тематики на главной странице интернет-магазина: на главной странице может разместиться пять новостей; редактирование и создание новостей доступно только администратору интернет-магазина);
- подсистема просмотра, создания и редактирования отзывов об интернет-магазине (предоставляет возможность просмотра отзывов на главной странице интернет-магазина: можно разместить семь отзывов на главной странице; отзывы имеют текст, фотографию клиента и ссылку на социальную сеть, где был оставлен комментарий);
- подсистема просмотра, создания и редактирование категорий (у пользователя есть возможность выбора категории на свой вкус; у администратора есть возможность редактирование категорий; категория имеет название и свой уникальный URL);
- подсистема оформления заказа (каждый пользователь может приобрести товар через интернет-магазин, просто нажав под товаром кнопку «Купить» и заполнив свои данные).

Проектирование интерфейса интернет-магазина начинается с главной страницы (рис. 2), на которой необходимо разместить следующие элементы:

- выпадающее меню, которое позволит выбрать каталог категорий Telegram-каналов;
- слайдер новостей, позволит просматривать все новости без участия пользователя, смена новостей происходит автоматически, так же есть возможность смены новости с помощью мыши;

- строка поиска, представляет собой поиск каналов по названию (при поиске появляется подсказка с возможными совпадениями);
- слайдер отзывов, позволит клиентам сайта просматривать отзывы клиентов, которые уже воспользовались услугами сайта (перемещает отзывы автоматически, а также предусматривает функцию ручного просмотра отзывов).



Рисунок 1. – Функциональная структура интернет-магазина

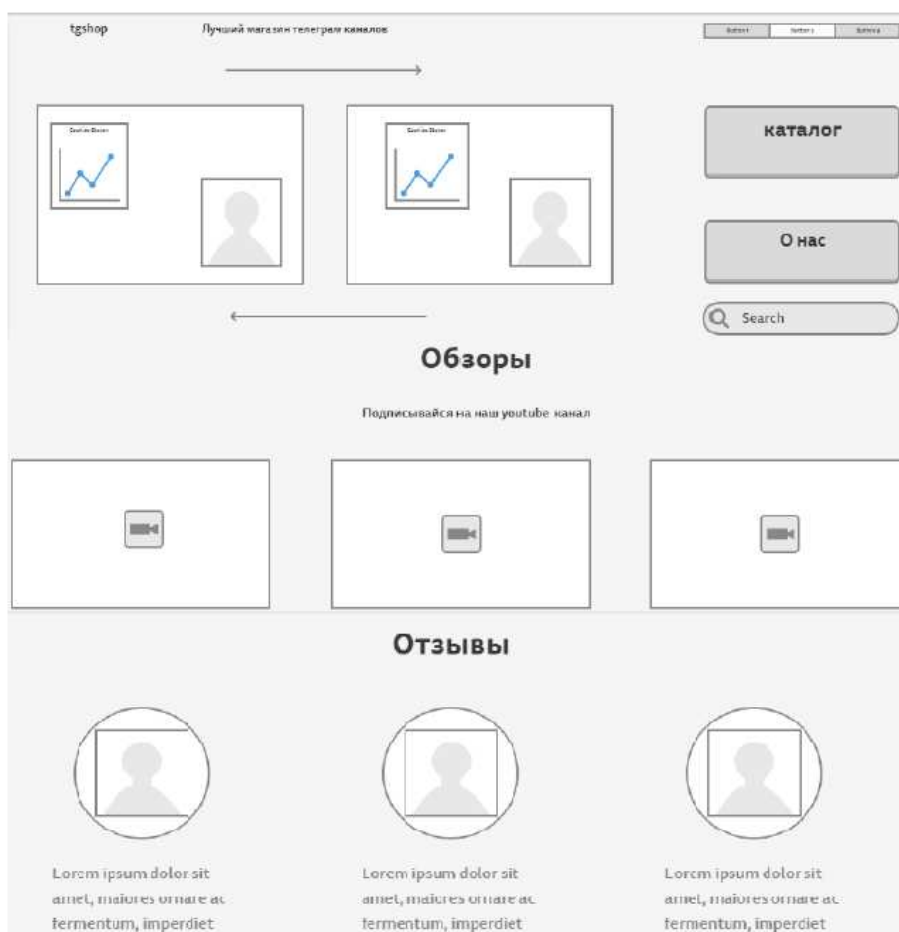


Рисунок 2. – Макет главной страница интернет-магазина

- На странице списка каналов (рис. 3) необходимо расположить элементы:
- выпадающее меню, которое позволит выбрать сортировку;
 - кнопку «Сортировать», которая позволит выполнить сортировку;
 - строка поиска, представляет собой поиск каналов по названиям и категориям (при поиске появляется подсказка с возможными совпадениями);
 - кнопка «Купить», позволяющая выполнить покупку, при нажатии на которую появится страница с формой;
 - элемент «Хлебные крошки», позволяющий переходить на предыдущие страницы;
 - логотип Telegram-канала;
 - логотип интернет-магазина, при нажатии на который, происходит перенаправление на главную страницу сайта;
 - цена Telegram-канала.

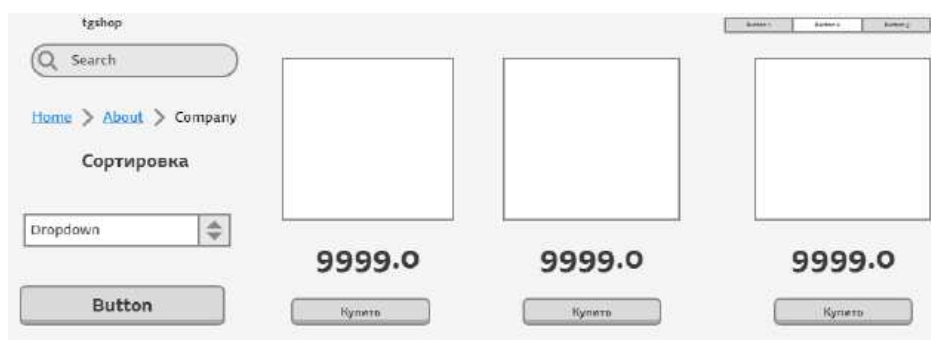


Рисунок 3. – Макет главной страницы списка каналов

На странице Telegram-бота (рис. 4) будут расположены все стандартные элементы мессенджера Telegram.

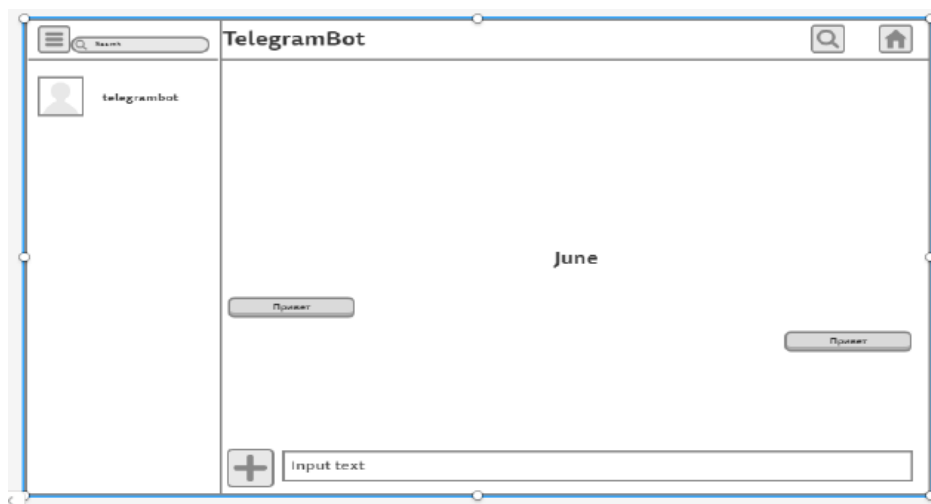


Рисунок 4. – Макет Telegram-бота

Таким образом, представленные макеты страниц, отражают проектирование основного функционала данного веб-ресурса. Такой интернет-магазин Telegram-каналов сможет заменить не только поиск каналов через сомнительные форумы и сайты, но и сможет обеспечить прозрачную и безопасную покупку.

ЛИТЕРАТУРА

1. Wikipedia [Электронный ресурс]. – Режим доступа [https://ru.wikipedia.org/wiki/ telegram](https://ru.wikipedia.org/wiki/telegram). – Дата обращения 18.09.2018.
2. ТСТ [Электронный ресурс]. – Режим доступа: <https://te-st.ru/2016/06/08/chatbots/>. – Дата доступа: 20.09.2018.

УДК 004.42

РАЗРАБОТКА БОТА-ПОМОЩНИКА ДЛЯ ИНТЕРНЕТ-МАГАЗИНА В TELEGRAM

М.А. ПОТАПОВ*(Представлено: Т.С. РУДЬКОВА)*

Отражены вопросы разработки бота-помощника, предназначенного для поиска клиентов в Telegram.

Сегодня чат-боты способны и повсеместно заменяют собой службы поддержки всевозможных сервисов. Заказать билеты, вызвать такси, сделать платеж или перевод, построить маршрут, скачать фильм или музыку тоже помогут боты. «Бот» (сокращение от «робот») – программа, имитирующая деятельность человека. Чат-бот, соответственно, имитирует собеседника в чате [1].

При разработке бота-помощника был использован язык программирования Python. Python – высокоуровневый язык программирования общего назначения, ориентированный на повышение производительности разработчика и читаемости кода. Синтаксис ядра Python минималистичен. В то же время стандартная библиотека включает большой объем полезных функций. Язык Python поддерживает несколько парадигм программирования, в том числе структурное, объектно-ориентированное, функциональное, императивное и аспектно-ориентированное. Основные архитектурные черты — динамическая типизация, автоматическое управление памятью, полная интроспекция, механизм обработки исключений, поддержка многопоточных вычислений и удобные высокоуровневые структуры данных. Код в Python организуется в функции и классы, которые могут объединяться в модули (они в свою очередь могут быть объединены в пакеты) [2].

PyCharm был выбран для реализации бота помощника в Telegram. PyCharm – интегрированная среда разработки для языка программирования Python. Предоставляет средства для анализа кода, графический отладчик, инструмент для запуска юнит-тестов и поддерживает веб-разработку на Django. PyCharm разработана компанией JetBrains на основе IntelliJ IDEA.

Возможности среды разработки:

- статический анализ кода, подсветка синтаксиса и ошибок;
- навигация по проекту и исходному коду: отображение файловой структуры проекта, быстрый переход между файлами, классами, методами и использованиями методов;
- рефакторинг: переименование, извлечение метода, введение переменной, введение константы, подъём и спуск метода;
- инструменты для веб-разработки с использованием фреймворка Django;
- встроенный отладчик для Python;
- встроенные инструменты для юнит-тестирования;
- разработка с использованием Google App Engine;
- поддержка систем контроля версий: общий пользовательский интерфейс для Mercurial, Git, Subversion, Perforce и CVS с поддержкой списков изменений и слияния [3].

На странице Telegram-бота расположены все стандартные элементы мессенджера Telegram. Доступ к боту-помощнику имеют абсолютно все зарегистрированные пользователи Telegram. Поиск бота осуществляется через стандартное поле поиска в Telegram.

Общение с ботом начинается после того как пользователь отправляет ему первое сообщение (рис.). Бот имеет определенный набор фраз для ответа случайному клиенту, данный словарь не имеет ограничений и его можно пополнять.

После того, как был написан функционал бота-помощника, данный блок был протестирован, чтобы удостовериться в том, что он соответствует указанным ранее требованиям. Тесты выявили несколько дефектов приложения, которые были незамедлительно устранены. Повторное тестирование дефектов не выявило.

Результатом проделанной работы стал бот-помощник для поиска клиентов в Telegram. Разработанный программный продукт может использоваться для поиска клиентов в Telegram, которые планируют приобрести Telegram-каналы в интернет-магазине для последующего получения прибыли от них.

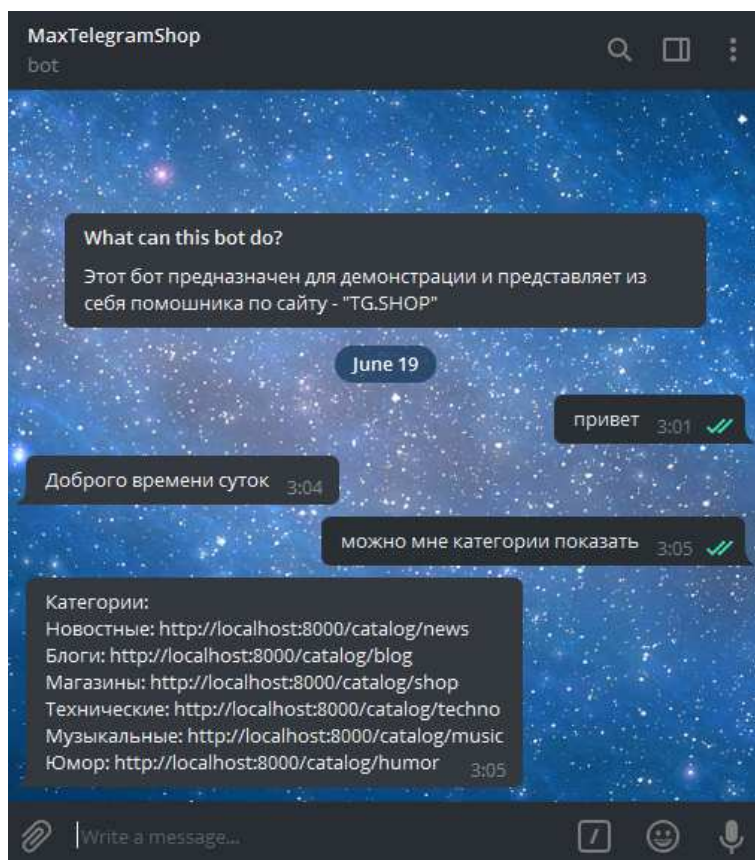


Рисунок. – Начало общения с ботом

ЛИТЕРАТУРА

1. ТСТ [Электронный ресурс] // Веб-ресурс. – Режим доступа: <https://te-st.ru/2016/06/08/chatbots/>. – Дата доступа: 20.09.2018.
2. Python [Электронный ресурс] // Википедия, свободная энциклопедия. – Режим доступа: <https://ru.wikipedia.org/wiki/Python>. – Дата доступа: 23.09.2018.
3. IntelliJ [Электронный ресурс] // Википедия, свободная энциклопедия. – Режим доступа: https://ru.wikipedia.org/wiki/IntelliJ_IDEA. – Дата доступа: 23.09.2018.

УДК 004.02

АНАЛИЗ ФУНКЦИОНАЛЬНЫХ ВОЗМОЖНОСТЕЙ ЭЛЕКТРОННОЙ ТОРГОВОЙ ПЛОЩАДКИ

П.В. ЗЯЗЮЛЯ

(Представлено: Т.С. РУДЬКОВА)

Представлено теоретическое обоснование работы электронной торговой площадки, сферы деятельности для применения данного функционала, а также анализ основных функциональных возможностей.

Введение. Всё больше людей хотят открыть своё дело. Причины могут быть разные: недовольство своей работой, начальством, условиями труда, зарплатой или просто хочется кардинальных перемен в жизни, которые возможны благодаря низкому размеру стартового капитала. Личные сбережения, гранты от государства, займы от друзей и родственников – этого бывает достаточно, чтобы открыть малый бизнес.

Малый бизнес – предпринимательство, опирающееся на деятельность небольших фирм, малых предприятий, формально не входящих в объединения [1].

Государство стремится поддерживать малые фирмы и проводит различные программы стимулирования – кредитование на льготных условиях, предоставление безвозмездной финансовой помощи, бесплатные консультации. Доля малого и среднего бизнеса в ВВП Беларуси к 2020 году вырастет до 40%. Об этом сообщил на открытии международной конференции «Современные инструменты финансирования малого предпринимательства» первый заместитель председателя правления Национального банка Тарас Надольный.

С развитием новых технологий открыть свой бизнес может любой желающий. Всё большее людей по всему миру пользуются веб-приложениями для поиска и покупки продуктов для открытия своего бизнеса. Причины популярности данных веб-приложений заключается в том, что в сети Интернет можно найти всё, что вздумается пользователю, в том числе и различные товары по оптовым ценам.

Электронная торговая площадка – это в первую очередь, сайт, ориентированный на поиск, покупку, заключения сделок и проведения аукционов между поставщиками и организатором. Кроме этого, пользователю предоставлен доступ к разделу «Каталог товаров» в котором он может заключать сделки на товары, не участвуя в аукционах, а покупая товары по фиксированной стоимости, указанной к каждому продукту.

Аукцион – публичная продажа, ценных бумаг, имущества предприятий, произведений искусства, и других объектов, которая производится по заранее установленным правилам аукциона. Общим для всех аукционов принципом является принцип состязательности между потенциальными покупателями. В процессе состязания между покупателями за право приобрести товар выявляется победитель аукциона. Победителем аукциона признаётся лицо, выигравшее аукцион в соответствии с его правилами. В этом случае объект приобретает у победителя аукциона [2].

Функциональная структура электронной торговой площадки. Функциональную структуру разрабатываемой системы можно представить в виде ряда подсистем, каждая из которых, в свою очередь, состоит из отдельных модулей либо представлена одним модулем. Поэтому в структуре электронной торговой площадки можно выделить следующие подсистемы:

- подсистема создания аукциона (отвечает за создание аукциона (рис. 1) в системе под учётной записью «Организатор»);
- подсистема редактирования аукциона (отвечает за редактирование существующих аукционов под учётной записью «Организатор»);
- подсистема каталог компании (хранит в себе данные о всех клиентах и компаниях, которые были зарегистрированы на торговой площадке);
- подсистема каталог товаров (хранит в себе список товаров, которые были созданы пользователями на торговой площадке, так же хранит информацию о заключённых и отклонённых сделках, создание и редактирование товаров);
- подсистема формы участия в аукционе (предоставляет возможность пользователям под учётной записью «Участник» присоединиться к доступному ему аукциону и осуществлять ставки);
- подсистема личный кабинет (предоставляет возможность настройки параметров системы под учётными записями «Организатор» и «Участник» по предпочтениям пользователей и по удобству работы с системой).

1 Параметры аукциона - Стандартные торги

Название
Новый аукцион

Категория
не указана

Начало аукциона
18.11.2014 16:34

Длительность
Дней 0 Часов 01:00

Тип аукциона
С продлением

Время продления
00:05

Повторные приглашения
01:00

Минимальный шаг ставки
Общий 0 руб. 1 коп.

Аукцион на повышение
 Запрещены равные ставки
 Запрещены ставки хуже рекомендованной цены
 Запрещены ставки хуже лидирующей
 Запрещены ставки до начала аукциона
 Разрешены ставки после завершения аукциона

Информация

Дополнительные файлы аукциона

Добавить файл

Имя файла.docx
Произвольное описание файла

Рисунок. – Макет формы «Создание аукциона – Стандартные торги»

Создав аукцион и добавив участников (поставщиков) организатор может быть уверен, что он приобретёт продукт по оптовым ценам, а поставщик в свою очередь, может найти себе новых партнёров для сбыта своего товара.

Пользователь может добавить себе в контакты партнёров и организаторов, чтобы включать их в новые аукционы. На сайте будет присутствовать раздел «Каталог товаров», в котором пользователи (организатор и участник) могут выкладывать товары и оформлять сделки на интересующие их товары, по оптовым ценам не принимая участие в аукционах.

Пользователю сайта будет предоставлена возможность ведения личного кабинета, где он может изменить свою личную информацию и вносить изменения в отображение какого-либо функционала на сайте (отключение колонок на форме аукциона, не создавать отчёты об аукционах, не получать уведомления от участников, которые не состоят с вами в партнёрских отношениях и т.д.).

Заключение. Малый бизнес в Республике Беларусь играет важную роль в социально-экономическом развитии страны. Он способствует формированию конкурентной среде, обеспечивает рост производства потребительских товаров, благодаря ему расширяется сфера услуг, а экономике придается дополнительная стабильность. Значениями малого предпринимательства определяется также и такими его особенностями, как способность быстро реагировать на спрос потребностей и оперативно его удовлетворять, быстро откликаться на изменения рыночной конъюнктуры и демонстрировать высокую маневренность. Помимо прочего, малый бизнес страны предоставляет много новых рабочих мест, что ведёт, в свою очередь, к сокращению численность нетрудоустроенного населения. На основе этого можно сделать вывод, что данная тема является актуальной, так как помогает пользователям за пару минут приобрести необходимые товары, что способствует развитию малого и среднего бизнеса.

ЛИТЕРАТУРА

1. Трудовой кодекс Республики Беларусь от 26.07.1999 г. № 296–з. – М. : Амалфея, 1999. – 99 с.
2. Wikipedia [Электронный ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki/Аукцион>. – Дата обращения: 17.09.2018.

УДК 004.42

РАЗРАБОТКА ЭЛЕКТРОННОЙ ТОРГОВОЙ ПЛОЩАДКИ

П.В. ЗЯЗЮЛЯ

(Представлено: Т.С. РУДЬКОВА)

Представлены основные функциональные возможности разрабатываемого веб-приложения, а также реализация электронной торговой площадки на примере модуля с аукционами и вкладкой с товарами.

В результате анализа предметной области и сущности проблемы, а также исходных данных, был выявлен ряд критериев и сформулированы требования к функциональности разрабатываемого веб-приложения «Электронная торговая площадка»:

- интуитивная в управлении панель настройки;
- дружелюбный интерфейс пользовательской части;
- фотографии/логотипы категорий и продуктов;
- редактирование для каждого продукта индивидуально;
- настройка порядка сортировки продуктов и категорий;
- предоставление интерфейса к БД продаваемых товаров;
- обеспечение безопасности личной информации участника и организатора;
- создание формы создания аукционов;
- создание формы участия в аукционах;
- создание формы редактирования аукционов.

Разработку системы было принято начать с формы для создания аукциона.

Для того чтобы создать новый аукцион (рис.) требуется нажать на кнопку «+ Создать» на вкладке «Мои аукционы». После нажатия на данную кнопку появится форма «Создание аукциона», на которой отображаются элементы для создания аукциона:

- кнопка «Отмена»;
- кнопка «Заполнить из шаблона»;
- поле для ввода «Название аукциона»;
- инпут «Категория»;
- дата-тайм пикер «Начало аукциона»;
- инпут и тайм-пикер «Длительность»;
- выпадающий список «Тип аукциона»;
- тайм-пикер «Время продления»;
- тайм-пикер «Повторные приглашения»;
- выпадающий список и инпут «Минимальный шаг ставки»;
- таблица «Список позиций»;
- таблица «Участники».

Создание аукциона Отмена

1 Также имеется возможность

1 Параметры аукциона *Создай свой аукцион в 3 простых шага*

Название Максимум 255 символов

Категория

Начало аукциона Длительность Дней Часов

Тип аукциона Время продления Повторные приглашения

Минимальный шаг ставки

Аукцион на повышение

Запрещены равные ставки

Запрещены ставки хуже рекомендованной цены

Запрещены ставки хуже лидирующей

Запрещены ставки до начала аукциона

Рисунок 1. – Форма «Создание аукциона – Стандартные торги»

После заполнения всех обязательных полей и нажатия на кнопку «Создать» в системе будет создан аукцион, который станет отображаться на вкладке «Мои аукционы» (рис. 2).

Мои аукционы + Создать



Название	Тип аукциона	Начало аукциона	Статус	Отчеты	Премодерация
Edit standart auction 1	Стандартные торги	31.05.2019 8:40:00	● Запланирован	 ▼	

Рис. 2. Отображение созданного аукциона

Созданный аукцион может иметь два статуса: «Запланирован» - аукцион будет начат в будущем, «Идёт» - аукцион доступен на данный момент и пользователем предоставлена возможность делать ставки и вносить изменения в параметры аукциона запрещено.

В системе «Электронная торговая площадка» также необходимо реализовать вкладку, в которой пользователям предоставлялась бы возможность покупать и продавать товары по оптовым ценам, не только в режиме аукционов, но и в режиме реальных сделок. Для этого, пользователь может перейти на вкладку «Каталог товаров», где будет доступно отображение имеющихся товаров, которые присутствуют в системе на данный момент (рис. 3).

СОРТИРОВАТЬ ПО: ЦЕНЕ ДАТЕ ПУБЛИКАЦИИ РЕЙТИНГУ ПРОДАВЦА




КЕФИР 100 ₺ от 1 л.

Дата публикации: 13.06.2018

Основные параметры:
 Жирность кефира: 1%
 Тип кефир: Обогащенный
 Наличие добавок: С добавками

Дополнительные параметры:
 Компания: Производитель
 Страна-производитель: Бельгия

Рейтинг продавца: ★★☆☆☆ 1.50



ЧАЙ 123 ₺ от 1 кг.

Дата публикации: 13.06.2018

Основные параметры:
 Тип чая: Ассамский
 Вид чая: Желтый

Дополнительные параметры:
 Компания: Производитель
 Страна-производитель: Аруба

Рейтинг продавца: ★★☆☆☆ 2.50

Рис. 3. Отображение списка товаров

Пользователю системы необходимо предоставить возможность просмотреть информацию о интересующем его продукте: изучить его основные параметры, просмотреть фото продукта, просмотреть дополнительные параметры по продукту, его рейтинг и прочитать дополнительную информацию о продукте и др.

Таким образом, разработанное веб-приложение «Электронная торговая площадка», предназначенное для создания, редактирования, создания шаблонов, редактирования шаблонов для аукционов, покупки и продажи товаров по оптовым ценам на торговой площадке, полностью удовлетворяет всех техническим и экономическим требованиям, и в дальнейшем может быть использовано в ведении малого бизнеса, среднего и крупного бизнеса. Данный программный продукт может найти широкое применение в Республике Беларусь.

ЛИТЕРАТУРА

1. Электронная торговая площадка [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/Электронная_торговая_площадка. – Дата обращения: 18.09.2018.
2. Преимущества создания собственной ЭТП [Электронный ресурс]. – Режим доступа: <http://www.zmsystems.ru/zakupki-po-223-fz/preimushhestva-sozdaniya-sobstvennoj-etp>. – Дата обращения: 18.09.2018.

УДК 004.056.5

ЗАЩИТА ДАННЫХ ПРИЛОЖЕНИЙ РАЗРАБОТАННЫХ НА UNITY3D

А.В. КАРНИЛОВИЧ

(Представлено: канд. физ.-мат. наук, доц. Д.Ф. ПАСТУХОВ)

Рассматриваются способы получения несанкционированного доступа к данным сохранений и ресурсов приложений, работающих на Unity3d, а также возможные способы решения этой проблемы.

Введение. Unity3D – кроссплатформенный игровой движок от компании Unity Technologies. Unity позволяет создавать приложения, работающие под более чем 20 различными операционными системами. Движок поддерживает множество популярных форматов моделей, звуков, материалов и текстур.

Проект в Unity делится на сцены (уровни) — отдельные файлы, содержащие свои игровые миры со своим набором объектов, сценариев, и настроек. Сцены могут содержать в себе как, собственно, объекты (модели), так и пустые игровые объекты — объекты, которые не имеют модели. Объекты, в свою очередь содержат наборы компонентов, с которыми и взаимодействуют скрипты.

Все эти данные необходимо защищать от копирования, нелегального использования, пиратского распространения, профессионального анализа и взлома.

Основная часть. При компиляции проекта Unity создается исполняемый файл игры, а в отдельной папке – данные игры (включая все игровые уровни и динамически подключаемые библиотеки).

Исходный код. Для Windows-сборок Unity компилирует и сохраняет исходный код всех игровых скриптов в директорию Managed. Находится код в библиотеках: Assembly-CSharp.dll, Assembly-CSharp-firstpass.dll и Assembly-UnityScript.dll. Для декомпиляции и просмотра managed-кода .NET библиотек существуют довольно удобные и при этом бесплатные утилиты: IISpy и dotPeek.

Данный подход особенно эффективен для наших целей: Unity очень скупко оптимизирует исходный код игровых скриптов, практически не изменяя его структуру, а также не скрывает названия переменных. Это позволяет с легкостью читать и понимать декомпилированный материал.

В таких случаях разработчикам приходится самим беспокоиться о безопасности своего кода. Для таких целей обычно используют обфускаторы.

Обфускация - это процесс, в результате которого код программы приобретает вид, трудный для анализа. Обфускация, осуществляется с целью защиты программного кода и алгоритмов, которые он реализует, от чужих глаз. Но в большинстве случаев обфускация имеет массу побочных эффектов. В особо неприятных обстоятельствах обфускатор может сделать программу совершенно непригодной к выполнению, в менее тяжёлых случаях в программе могут появиться новые ошибки. Поэтому обфускацию необходимо применять с максимальной осторожностью.

В AssetStore существует множество готовых решений, но большинство из них платные. Бесплатные версии, как правило, ограничены или имеют невысокую эффективность.

Ресурсы проекта. Большинство ресурсов Unity-проекта упаковываются в файлы проприетарного формата с расширениями .assets и .resources. Несмотря на закрытость форматов, существуют инструменты для распаковки таких файлов. Так, например, Unity Assets Explorer способен извлечь большинство текстур и шейдеров из игры. Полученные в результате текстуры будут иметь формат DDS, который можно открыть с помощью Windows Texture Viewer. Шейдеры извлекаются в уже скомпилированном виде и решений для их декомпиляции не существует. Тем не менее, это обстоятельство не мешает импортировать и использовать полученные шейдеры в другом Unity-проекте.

Трёхмерные модели в Unity-сборке расположены в различных ресурсах, а некоторые из них и вообще могут генерироваться во время игры. Получить такие данные можно напрямую из памяти графического ускорителя. Когда игра запущена, вся информация о текстурах и моделях, видимых на экране, находится в памяти видеокарты. С помощью утилиты 3D Ripper DX можно извлечь всю эту информацию и сохранить в формате, понятном 3D-редакторам.

Сохраняемые данные. PlayerPrefs — это класс из стандартной библиотеки Unity, который позволяет сохранять данные в долговременную память устройства. Представляет собой пару ключ - значение. Он часто используется разработчиками для хранения различных настроек, достижений, прогресса игрока и другой информации о состоянии игры. На ОС Windows эти данные сохраняются в системном реестре. В других операционных системах данные сохраняются на устройстве в локальной папке приложения в специальном файле. В большинстве случаев к ним можно легко получить доступ и модифицировать с помощью текстового редактора. Например, в Windows достаточно использовать встроенную утилиту

RegEdit чтобы модифицировать любые значения PlayerPrefs, изменяя тем самым конфигурацию и статус игры.

Простейшим способом противодействия является кодирование сохраняемых данных, например, base64. Данный способ не слишком эффективен, но может дать начальную защиту от просмотра. Проверить, изменялись ли данные без ведома помогут хеш-функции: сравнив контрольные суммы хранимых данных, мы сможем убедиться, что никто и ничто, кроме нашего кода эти данные не изменяло.

Для более надёжной защиты необходимо использовать шифрование или различные комбинации из перечисленных методов.

Также можно реализовать свой формат сохранений. Благодаря Mono Unity поддерживает работу с файловой системой. Таким образом можно сериализовать все необходимые данные, применить надёжное шифрование и сохранить в надёжном месте.

Оперативная память. Cheat Engine — широко известная программа для взлома памяти. Она находит ту область оперативной памяти, которая принадлежит процессу запущенного приложения и позволяет произвольно её изменять. За несколько итераций отсеивания можно с легкостью обнаружить расположение большинства переменных игры и произвольно их изменять.

Эта программа пользуется тем фактом, что разработчики очень редко защищают значения переменных. Защититься от таких программ довольно просто — нужно шифровать значения в памяти приложения. Шифровать каждый раз при записи и дешифровать каждый раз при чтении. И так как операция довольно частая, нужен очень быстрый алгоритм. Для этого, например, может подойти алгоритм шифрования на основе XOR и base64. Так как работа будет с данными в оперативной памяти, нет необходимости сохранять ключ, для этого можно использовать сессионный ключ, генерируемый непосредственно перед работой.

Но более эффективным методом будет сохранение особо важных значений сразу в долговременной памяти устройства. PlayerPrefs позволяет это делать довольно быстро и просто, но, опять же, необходимо позаботиться о сохранности этих данных.

Заключение. К сожалению, существует не так уж много способов защитить игру от взлома. Будучи установленной на пользовательское устройство, она фактически раскрывает все текстуры, модели и исходный код. Если кто-то захочет декомпилировать приложение и украсть ресурсы — это лишь вопрос времени.

Невзирая на это, существуют действенные методы, которые позволят серьёзно усложнить жизнь злоумышленникам.

ЛИТЕРАТУРА

1. Scarfone, K. Guide to Storage Encryption Technologies for End User Devices / Karen Scarfone, Murugiah Souppaya, Matt Sexton. – November, 2007. – 40 p.
2. Unity - Scripting API: PlayerPrefs [Электронный ресурс]. – Режим доступа: <https://docs.unity3d.com/ScriptReference/PlayerPrefs.html>. – Дата доступа: 22.09.2018.
3. A practical tutorial to hack (and protect) Unity games [Электронный ресурс]. – Режим доступа: <https://www.alanzucconi.com/2015/09/02/a-practical-tutorial-to-hack-and-protect-unity-games/>. – Дата доступа: 22.09.2018.

УДК 004.021

ПРОЕКТИРОВАНИЕ ГРАФИЧЕСКОГО ИНТЕРФЕЙСА ДЛЯ СИСТЕМЫ БЕЗОПАСНОГО ХРАНЕНИЯ ПЕРСОНАЛЬНОЙ ИНФОРМАЦИИ

*А.В. КАРНИЛОВИЧ**(Представлено: канд. физ.-мат. наук, доц. Д.Ф. ПАСТУХОВ)*

Рассматриваются принципы построения графического интерфейса пользователя для системы хранения персональной информации, разработанной на Unity3D.

Введение. Unity3D – кроссплатформенный игровой движок от компании Unity Technologies. Unity позволяет создавать приложения, работающие под более чем 20 различными операционными системами. Движок поддерживает множество популярных форматов моделей, звуков, материалов и текстур. Начиная с версии 4.6 была добавлена новая система интерфейса, позволяющая использовать встроенные средства движка для построения пользовательского интерфейса.

Основная часть. В основе понимания новой концепции лежат три компонента: Canvas, Rect Transform и Event Trigger.

Canvas – является контейнером для всех элементов UI и определяет режим рендера. Таких контейнеров на сцене может быть более одного.

Rect Transform – этот компонент позволяет задать положение и размер игрового объекта, используя удобные визуальные средства управления. Он вводит понятия ширины и высоты, точки опоры, а также поддерживает якоря.

Задание якорей – простой и мощный способ контролировать положение и размер ваших элементов пользовательского интерфейса относительно их родителей. Это особенно удобно, когда вы изменяете размер родительских элементов. При изменении размера родителя наш элемент будет стараться сохранить равномерное расстояние к точкам якоря, заставляя его перемещаться или изменять размер вслед за родителем.

Точка опоры является точкой, вокруг которой производятся все преобразования. Другими словами, определяя положение элемента, вы определяете его точку опоры. Если вы станете вращать элемент пользовательского интерфейса, то он будет вращаться вокруг этой точки. Точка опоры обозначается нормализованными координатами. То есть, ее значение изменяется от 0 до 1 для ширины и высоты, где (0,0) – нижний левый угол, а (1,1) – правый верхний угол.

Event System – доработанная система событий, которая включает компоненты визуального управления вызовом событий. Отвечает за обработку ввода, рейкастинг и события в сцене Unity.

При помощи перечисленных инструментов стало возможно разрабатывать сложные интерфейсы на основе иерархии простых компонентов без особых сложностей. При этом интерфейс будет адаптивным и автоматически подстраиваться под различные разрешения экрана.

Интерфейс приложения сделан используя современную концепцию построения внешнего вида, так называемый, плоский дизайн. Основные принципы плоского стиля – простота, лаконичность и минимализм. Основные отличительные черты – это те вещи, на которых можно акцентировать внимание – яркие необычные цвета, нестандартная типографика, графические примитивы. Так достигается простота и универсальность интерфейса.

Основной принцип flat-дизайна – это создание двумерного изображения, которое уже само по себе является плоским. При этом нужно избавиться от всех элементов, которые могут добавлять хоть какую-то глубину изображению: градиенты, плавные переходы, тени, скосы, объем, текстуры и так далее. Все элементы рисунка имеют жесткие края и также не выделяются, и не имеют глубины относительно других элементов изображения.

Во flat-дизайне вы не встретите элементы, которые пытаются выглядеть реалистично (скевоморфизм), 3D-анимацию, реалистичные пиктограммы и т.д. Однако при этом сам дизайн пытается оставаться аналогичным традиционному, но теперь на передний план выходят надписи, кнопки и навигация.

На первый взгляд сайт с применением Flat дизайна может показаться чересчур простым, хотя на самом деле этому стилю присуща четкая иерархия элементов, удобное размещение всех инструментов взаимодействия пользователя, а, следовательно, имеет высокий показатель юзабилити.

Плоский дизайн по своей натуре прост, и он хорошо сочетается с минималистичным подходом. В общем дизайне нужно избегать слишком больших наворотов. Конечно, простых цветов и текста может быть недостаточно. Поэтому, чтобы добавить визуальных эффектов, нужно выбирать простые фотографии.

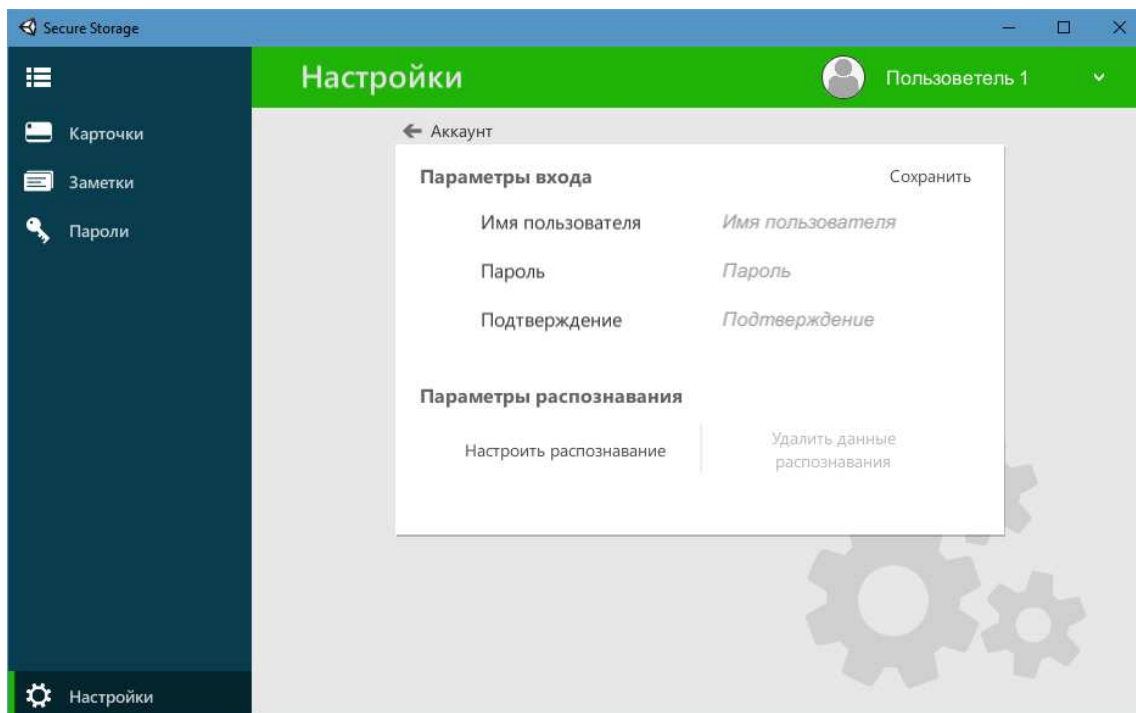


Рисунок. – Меню настроек приложения

Для создания плавных анимаций в Unity можно использовать сторонние плагины, например, DOTween. DOTween – это быстрый, эффективный, объектно-ориентированный движок анимации, оптимизированный для C#.

При помощи DOTween можно перемещать, изменять размер, поворачивать каждый объект на сцене независимо от других, при этом имея возможность настроить каждый аспект анимации по отдельности. Обратные вызовы, петли, различные функции интерполяции значений и многие другие варианты анимации.

Кроме того, он чрезвычайно точный. Это означает, что 1000 циклов по 1 секунде будут играть ровно столько же, сколько один цикл на 1000 секунд. Логичный и простой в использовании API, созданный для повышения эффективности, интуитивности и простоты использования.

DOTween может анимировать каждое числовое значение, а также некоторые нечисловые. Например, он может анимировать строки.

Плагин предоставляет полный контроль: Play, Pause, Rewind, Restart, Complete, Goto и множество других полезных методов для управления вашими анимациями. Анимации можно объединять в последовательности для создания сложных анимаций (которые не обязательно должны быть в последовательности, а также они могут перекрывать друг друга). Возможно изменение начальных / конечных значений и продолжительности воспроизведения движения в любой момент, даже во время воспроизведения.

DOTween построен с учетом расширяемой архитектуры, которая позволяет создавать собственные плагины для анимации в виде отдельных файлов. Поддерживаются вспомогательные виртуальные методы, чтобы делать такие вещи, как например вызов функции после определенной задержки.

Пример использования последовательности анимаций DOTween приведен в листинге 1.

Листинг 1 – Функция закрытия панели камеры

```
void HideCameraPanel()
{
    if (!isExpanded)
        return;

    Sequence itemSequence = DOTween.Sequence()
        .Append(cameraMask.rectTransform.DOSizeDelta(new Vector2(400, 0), 0.75f))
        .Append(cameraPanel.DOSizeDelta(new Vector2(0, 150), 0.25f))
        .Append(panel.DOSizeDelta(new Vector2(panel.sizeDelta.x, panel.sizeDelta.y - 400), 0.5f))
        .AppendInterval(1)
```

```
.Append(cameraPanel.DOScaleY(0, 0.5f))
.Join(panel.DOSizeDelta(new Vector2(panel.sizeDelta.x, panel.sizeDelta.y - 550), 0.5f))
.AppendCallback(() => cameraPanel.gameObject.SetActive(false));

isExpanded = false;
}
```

Таким образом, можно без труда настроить даже сложные последовательности анимаций без ущерба производительности.

Заключение. В данной статье были рассмотрены основные принципы и способы построения графического интерфейса в Unity для системы хранения персональной информации. Рассмотрены основные компоненты Unity UI system и способы построения анимации.

ЛИТЕРАТУРА

1. Unity - Scripting API: UI [Электронный ресурс]. – Режим доступа: <https://docs.unity3d.com/Manual/UISystem.html>. – Дата доступа: 24.09.2018.
2. Краткий обзор нового Unity UI с примерами организации интерфейса [Электронный ресурс]. – Режим доступа: <https://habr.com/post/246605/>. – Дата доступа: 24.09.2018.
3. Основные принципы плоского дизайна [Электронный ресурс]. – Режим доступа: <http://x-site.by/info/flat-design>. – Дата доступа: 24.09.2018.
4. DOTween [Электронный ресурс]. – Режим доступа: <http://dotween.demigiant.com/index.php>. – Дата доступа: 26.09.2018.

УДК 004.021

**ПРОЕКТИРОВАНИЕ ГРАФИЧЕСКОГО ИНТЕРФЕЙСА СИСТЕМЫ
ДЛЯ СКРЫТИЯ ИНФОРМАЦИИ НА ОСНОВЕ ДИСКРЕТНЫХ ПРЕОБРАЗОВАНИЙ
С ПОМОЩЬЮ АЛГОРИТМА КОХА И ЖАО****К.В. СТАНКЕВИЧ***(Представлено: канд. физ.-мат. наук, доц. Д.Ф. ПАСТУХОВ)*

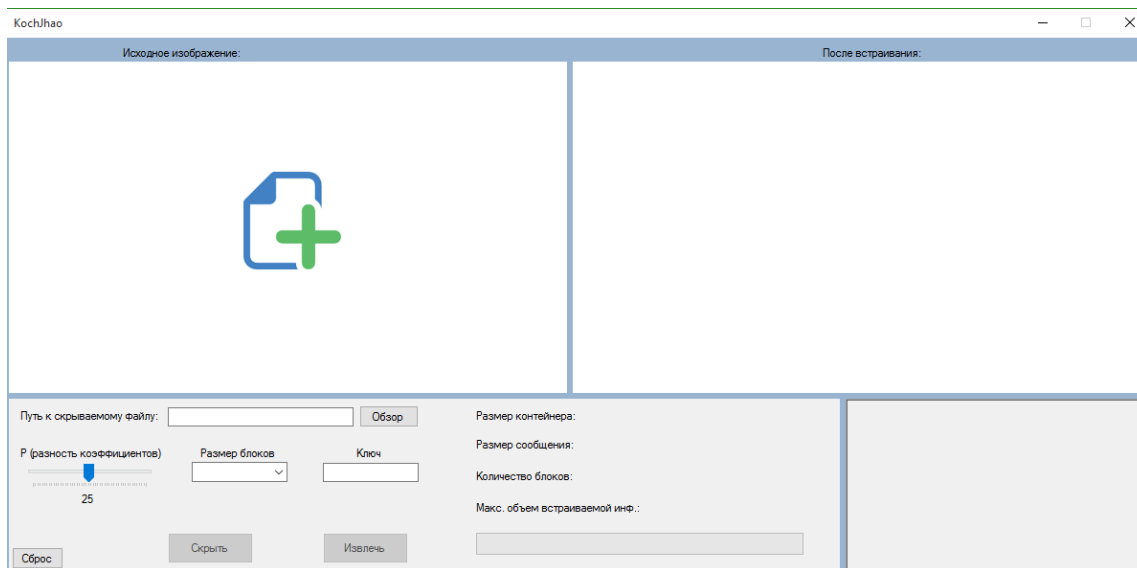
Рассматривается проектирование графического интерфейса системы для скрытия информации на основе дискретных преобразований с помощью алгоритма Коха и Жао. Проведён анализ степени пригодности контейнера для модификации, моделирование атак и определение устойчивости к ним.

Введение. Развитие средств вычислительной техники в последнее десятилетие дало новый толчок для развития компьютерной стеганографии. Появилось много новых областей применения. Сообщения встраивают теперь в цифровые данные, как правило, имеющие аналоговую природу.

Интерфейс программы должен обладать целым рядом свойств: естественность, согласованность, дружелюбность, принцип «обратной связи», простота, гибкость, эстетическая привлекательность.

Основная часть. Любое приложение должно быть грамотно спроектировано и разделено на отдельные модули, которые должны быть относительно независимыми друг от друга. Подобное разделение значительно облегчает не только реализацию приложения, но и возможную его модификацию. В этом заключается принцип модульности объектно-ориентированного программирования.

Приложение «KochZhao» представляет собой приложение для скрытия информации в изображениях. Для скрытия сообщения необходимо будет выбрать контейнер (поддерживаемые форматы: bmp и png), выбрать скрываемый файл и указать настройки: Р и размер сегментов (блоков). Для дальнейшего извлечения сообщения необходимо запомнить ключ, который будет выведен в соответствующем поле, и размер сегментов (блоков).

**Рисунок 1. – Интерфейс программы**

Для определения оптимального контейнера будем использовать различные изображения с преобладанием какой-то одной цветовой компоненты RGB.

В результате исследований было выявлено, что наиболее подходящими под контейнер являются изображения с преобладанием синей и зеленой компонентов.

Для определения устойчивости стеганосистемы проведены несколько видов атак: пассивного нарушителя, активного нарушителя.

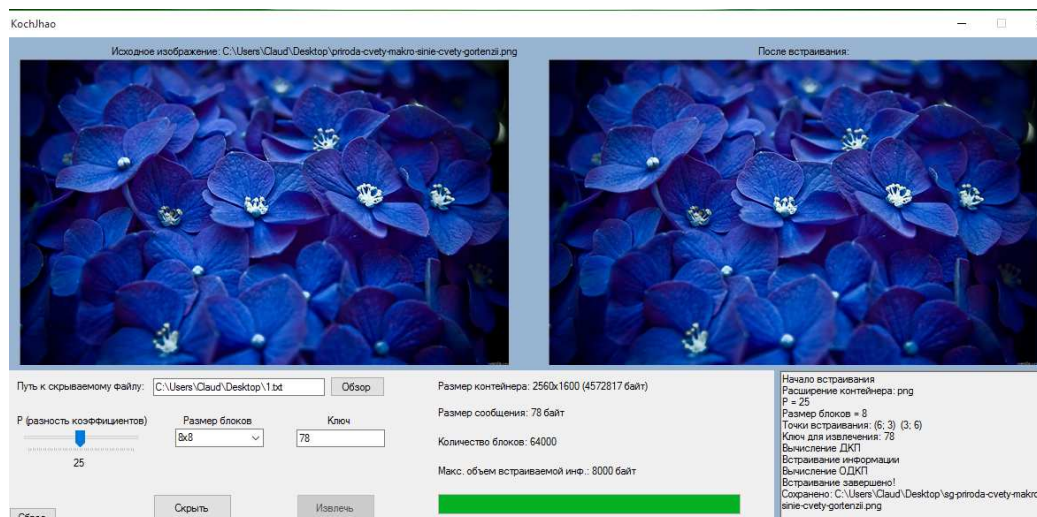


Рисунок 2. – Интерфейс шифрования

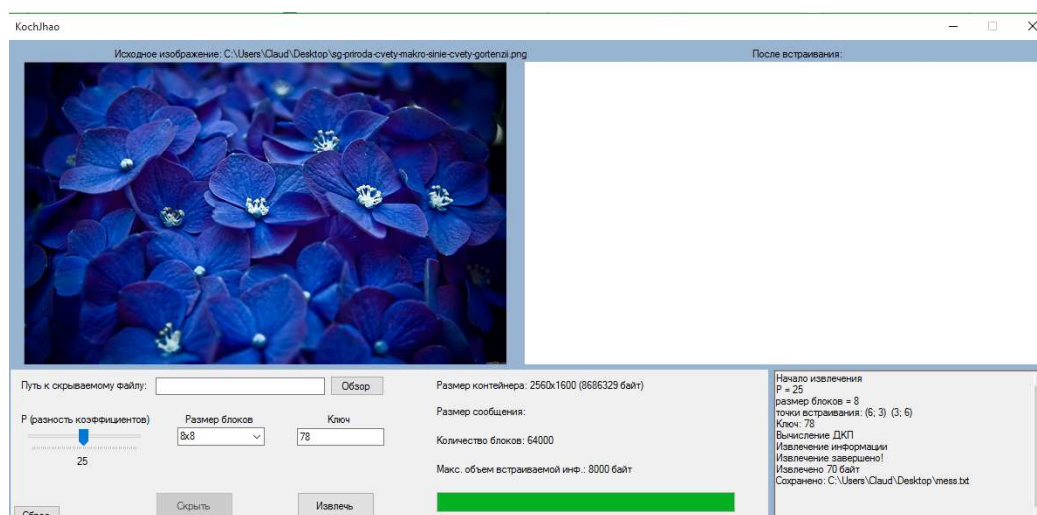


Рисунок 3. – Интерфейс дешифрования

Пассивной называется атака, при которой противник не имеет возможности модифицировать передаваемые сообщения и вставлять в информационный канал между отправителем и получателем свои сообщения. Целью пассивной атаки может быть только прослушивание передаваемых сообщений и анализ трафика.

Активной называется атака, при которой противник имеет возможность модифицировать передаваемые сообщения и вставлять свои сообщения. Различают следующие типы активных атак:

1. отказ в обслуживании – DoS-атака (Denial of Service);
2. модификация потока данных;
3. создание ложного потока (фальсификация);
4. Повторное использование.

При попытках определить наличие в контейнере скрытого сообщения, при помощи функции программы Adobe Photoshop уровни каналов, было заметно выделение синего канала у некоторых пикселей, что свидетельствует о наличии скрытого сообщения.

При атаках активного нарушителя было выявлено, что алгоритм имеет устойчивость к большинству известных стеганоатак, в том числе к атаке сжатием, к аффинным преобразованиям, геометрическим атакам.

В результате, проведенных тестов выяснила следующее:

- изменение цветовой модели перед ДКП значительно ухудшило результат отношения сигнал/шум;
- встраивание в высокочастотную область спектра незначительно улучшило результат;

- встраивание в низкочастотную область спектра значительно ухудшило результат, уменьшится устойчивость к компрессии;
- уменьшение блока до размерности 4x4 незначительно улучшило результат, уменьшится устойчивость к компрессии;
- встраивание 2 бит в блок в среднечастотную и высокочастотную область улучшило результат.

Заключение. В данной статье рассмотрен способ построения графического интерфейса системы скрытия информации на основе дискретных преобразований, проведён анализ степени пригодности контейнера для модификации, моделирование атак и определение устойчивости к ним.

ЛИТЕРАТУРА

1. Матюшик, В.Н. Методы и средства стеганографии для защиты графических образов / В.Н. Матюшик. – Минск : Белорусский государственный университет информатики и радиоэлектроники.
2. Грибунин, В.Г. Цифровая стеганография / В.Г. Грибунин, И.Н. Оков, И.В. Туринцев. – М. : СОЛОН-Пресс, 2002.
3. Конахович, Г.Ф. Компьютерная стеганография. Теория и практика / Г.Ф. Конахович, А.Ю. Пузыренко. – Киев : МК-Пресс, 2006.

УДК 004.021

РАЗРАБОТКА СИСТЕМЫ ДЛЯ СКРЫТИЯ ИНФОРМАЦИИ НА ОСНОВЕ ДИСКРЕТНЫХ ПРЕОБРАЗОВАНИЙ С ПОМОЩЬЮ АЛГОРИТМА КОХА И ЖАО

К.В. СТАНКЕВИЧ

(Представлено: канд. физ.-мат. наук, доц. Д.Ф. ПАСТУХОВ)

Рассматриваются алгоритм скрытия данных в изображении на основе дискретных преобразований, описание работы основной программы и основные функции разрабатываемого программного продукта.

Введение. Задача защиты информации от несанкционированного доступа решалась во все времена на протяжении истории человечества. Выделилось два основных направления решения этой задачи криптография и стеганография. Целью криптографии является скрытие содержимого сообщений за счет их шифрования. В отличие от этого, при стеганографии скрывается сам факт существования тайного сообщения. Скрытие информации возможно лишь благодаря тому, что противнику неизвестен метод скрытия. Система защиты информации должна обеспечивать свои функции даже при полной информированности противника о её структуре и алгоритмах функционирования

Основная часть. В статье будет рассмотрен стеганографический метод скрытия информации в изображениях, как с точки зрения устойчивости к различным видам атак, так и с точки зрения сохранения приемлимого качества изображения.

Алгоритм Коха-Жао для встраивания информации использует частотную область контейнера и заключается в относительной замене величин коэффициентов дискретного косинусного преобразования (ДКП).

Основными функциями разрабатываемого программного продукта являются:

- шифрование информации;
- дешифрование информации;
- вывод промежуточной информации о выполненных операциях.

Рассматриваемая методика ляжет в основу разрабатываемого программного продукта «Скрытие информации на основе дискретных преобразований методом Коха и Жао», который реализует все задачи, в частности, главную – скрыть факт передачи информации.

При проектировании приложения были выделены следующие классы (рис.):

- Koch – класс, реализующий алгоритм;
- Converter – класс для работы с двоичным представлением чисел;
- Form1 – пользовательский интерфейс.

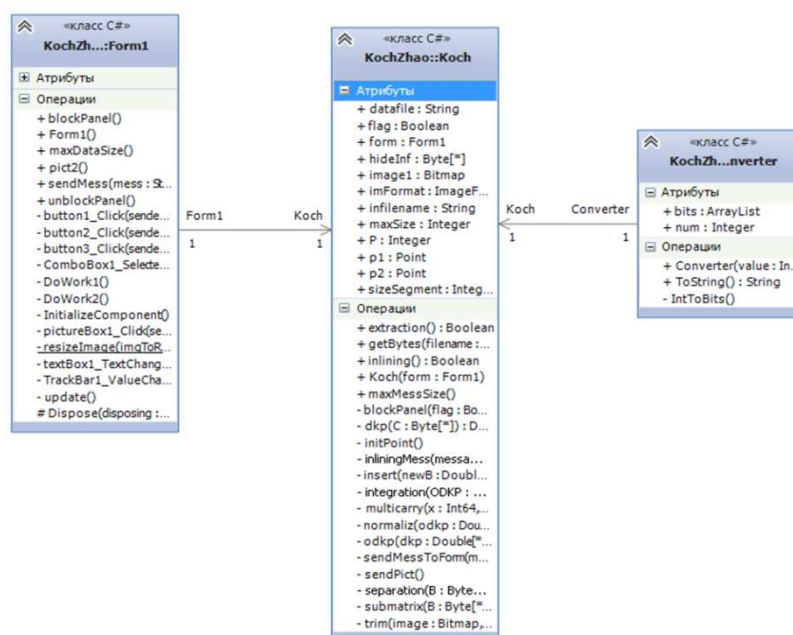


Рисунок. – Диаграмма классов

Для скрытия информации методом Коха и Жао необходимо выполнить следующие пункты:

1. Изображение-контейнер разбивается на блоки 8 на 8, 4 на 4 или 2 на 2 пикселей.
2. К каждому блоку применяется дискретное косинусное преобразование. В результате получаем матрицы 8 на 8 коэффициентов ДКП.
3. Выбирается любой блок (в каждый блок записывается 1 бит информации).
4. Выбираются два коэффициента ДКП в каждом блоке из высокочастотных коэффициентов.
5. Для передачи бита 0 необходимо, чтобы разница модулей коэффициентов ДКП превышала некоторую положительную величину (задается вручную); для передачи бита 1 разница должна быть меньше по сравнению с некоторой отрицательной величиной. Таким образом, при передаче 0 увеличиваем модуль первого коэффициента и уменьшаем модуль второго. При передаче 1 уменьшаем модуль первого коэффициента и увеличиваем модуль второго.
6. Выполняем пункты 3-5 до окончательной записи всего сообщения.
7. Для каждого блока выполняем обратное ДКП.
8. Блоки собираются обратно в изображение.

Извлечение:

1. Изображение-контейнер разбивается на блоки 8 на 8, 4 на 4 или 2 на 2 пикселей.
2. К каждому блоку применяется дискретное косинусное преобразование. В результате получаем матрицы 8 на 8 коэффициентов ДКП.
3. Выбирается любой блок (в каждый блок записывается 1 бит информации).
4. Считаем разность модулей двух коэффициентов ДКП.
5. В результате проверки разности модулей извлекаемому биту присваивается 0 или 1.

При обработке изображения будет использоваться двумерная версия дискретного косинусного преобразования:

$$\Omega(u, v) = \frac{\zeta(u) \cdot \zeta(v)}{\sqrt{2N}} \cdot \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} C(x, y) \cdot \cos \left[\frac{\pi u \cdot (2x+1)}{2N} \right] \cdot \cos \left[\frac{\pi v \cdot (2y+1)}{2N} \right], \quad (1)$$

$$S(x, y) = \frac{1}{\sqrt{2N}} \cdot \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} \zeta(u) \cdot \zeta(v) \cdot \Omega(u, v) \cdot \cos \left[\frac{\pi u \cdot (2x+1)}{2N} \right] \cdot \cos \left[\frac{\pi v \cdot (2y+1)}{2N} \right], \quad (2)$$

где $C(x, y)$ и $S(x, y)$ – соответственно, элементы исходного и восстановленного по коэффициентам ДКП изображения размерностью $N \times N$;

x, y – пространственные координаты пикселей изображения;

$\Omega(u, v)$ – массив коэффициентов ДКП;

u, v – координаты в частотной области;

$\zeta(u) = \frac{1}{\sqrt{2}}$, если $u = 0$, и $\zeta(u) = 1$, если $u > 0$; $\zeta(v) = \frac{1}{\sqrt{2}}$, если $v = 0$, и $\zeta(v) = 1$, если $v > 0$.

Программа прямого дискретного косинусного преобразования, соответствующая формуле (1) реализована так:

Листинг 1 – Прямое ДКП

```

1: private double[,] dkp(byte[,] C){
2:   int n = C.GetLength(0);
3:   double[,] result = new double[n, n];
4:   double U, V, temp = 0;
5:   for (int v = 0; v < n; v++){
6:     for (int u = 0; u < n; u++) {
7:       if (v == 0) V = 1.0 / Math.Sqrt(2);
8:       else V = 1;
9:       if (u == 0) U = 1.0 / Math.Sqrt(2);
10:      else U = 1;
11:      temp = 0;
12:      for (int i = 0; i < n; i++){
13:        for (int j = 0; j < n; j++){temp += C[i, j] * Math.Cos(Math.PI *
14: v * (2 * i + 1) / (2 * n)) * Math.Cos(Math.PI * u * (2 * j + 1) /
15: (2 * n));}}
16:      result[v, u] = U * V * temp / (Math.Sqrt(2 * n));}}
17:   return result;
18: }
```

Программа обратного дискретного косинусного преобразования, соответствующая формуле (2) была реализована так:

Листинг 2 – Обратное ДКП

```

1: private double[,] odkp(double[,] dkp) {
2:     int n = dkp.GetLength(0);
3:     double[,] result = new double[n, n];
4:     double U, V, temp = 0;
5:     for (int v = 0; v < n; v++){
6:         for (int u = 0; u < n; u++){
7:             temp = 0;
8:             for (int i = 0; i < n; i++){
9:                 for (int j = 0; j < n; j++){
10:                    if (i == 0) V = 1.0 / Math.Sqrt(2);
11:                    else V = 1;
12:                    if (j == 0) U = 1.0 / Math.Sqrt(2);
13:                    else U = 1;
14:                    temp += U * V * dkp[i, j] * Math.Cos(Math.PI * i * (2 * v + 1) /
15:                    (2 * n)) * Math.Cos(Math.PI * j * (2 * u + 1) / (2 * n));}}
16:                result[v, u] = temp / (Math.Sqrt(2 * n));}}
17:     return result;
18: }
```

В результате получаются матрицы 8×8 коэффициентов ДКП, которые зачастую обозначают $\Omega_b(u, v)$, где b – номер блока контейнера C , а (u, v) – позиция коэффициента в этом блоке. Каждый блок при этом предназначен для скрытия одного бита данных.

Для выбора каждого следующего блока для скрытия одного бита сообщения будет использоваться генератор ПСП «Marsaglia-Multicarry» основанный на методе умножения с переносом.

На следующем этапе выбираются два конкретных коэффициента ДКП из каждого блока, которые будут использоваться для скрытия данных. Для скрытия данных будут использоваться блоки из области высоких частот, что приведет к наименьшим искажениям контейнера. Данные коэффициенты задаются их координатами в массивах коэффициентов ДКП: $(u1, v1)$ и $(u2, v2)$.

Во время организации секретного канала абоненты должны предварительно договориться о двух конкретных коэффициентах ДКП из каждого блока.

Заключение. В данной статье был рассмотрен алгоритм скрытия данных в изображении на основе дискретных преобразований, описание работы основной программы и основные функции разрабатываемого программного продукта.

ЛИТЕРАТУРА

1. Садов, В.С. Компьютерная стеганография (конспект лекций) / В.С. Садов. – Минск : Беларус. гос. ун-т, 2010.
2. Грибунин, В.Г. Цифровая стеганография / В.Г. Грибунин, И.Н. Оков, И.В. Туринцев – М. : Солон-Пресс, 2002.

УДК 004.056.55

РАЗРАБОТКА АЛГОРИТМА ШИФРОВАНИЯ
НА ОСНОВЕ АЛГОРИТМОВ КУБИКА РУБИКА

И.Е. ИВАНЕНКО

(Представлено: канд. физ.-мат. наук, доц. Д.Ф. ПАСТУХОВ)

Рассматривается проектирование алгоритма шифрования на основе алгоритмов кубика Рубика. Приведён обобщённый алгоритм решения задачи.

Введение. Алгоритмы шифрования и дешифрования данных широко применяются в компьютерной технике в системах сокрытия конфиденциальной и коммерческой информации от злонамеренного использования сторонними лицами. Главным принципом в них является условие, что передатчик и приемник заранее знают алгоритм шифрования, а также ключ к сообщению, без которых информация представляет собой всего лишь набор символов, не имеющих смысла.

Алгоритм шифрования на основе кубика Рубика является шифром перестановки. Метод перестановки заключается в том, что символы шифруемого текста переставляются по определенным правилам внутри шифруемого блока символов, т.е. преобразования приводят к изменению только порядка следования символов исходного сообщения.

В 1991 г. В.М. Кузьмич предложил схему перестановки, основанной на кубике Рубика. Согласно этой схеме открытый текст записывается в ячейки граней куба по строкам. После осуществления заданного числа заданных поворотов слоев куба считывание шифртекста осуществляется по столбикам. Сложность дешифрования в этом случае определяется количеством ячеек на гранях куба и сложностью выполненных поворотов слоев. Перестановка, основанная на кубике Рубика, получила название объемной (многомерной) перестановки [1].

Данный алгоритм шифрования на основе алгоритма кубика Рубика был изменен для работы не с символами шифруемого текста, а с массивом байт, который получается путем преобразования шифруемого текста. Данный алгоритм является симметричным алгоритмом шифрования.

Алгоритм шифрования на основе алгоритма кубика Рубика работает с ключами шифрования различной длины. Для работы алгоритма могут быть использованы ключи 16 байт (128 бит), 32 байта (256 бит), 64 байта (512 бит), 128 байт (1024 бита) и 256 байт (2048 бит). Длина ключа определяет количество раундов, которое будет использовано для шифрования или дешифрования символов исходного сообщения.

Обобщенный алгоритм решения задачи.

Алгоритм шифрования включает в себя следующие этапы:

1. Для алгоритма генерируется или задается ключ шифрования необходимой длины.
2. Символов исходного сообщения преобразуются в массив байт.
3. Из полученного ключа генерируется массива раундовых ключей.
4. Массив данных разбивается на блоки по 6 байт (48 бит). Если данных недостаточно для формирования целого блока, то блок дополняется «0».
5. Каждый байт из блока данных заменяется на соответствующей ему из константной таблицы, представленной на рисунке 1.

		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
	1	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
	2	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
	3	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
	4	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
	5	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
	6	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
	7	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
	8	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
	9	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
	a	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
	b	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
	c	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
	d	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
	e	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
	f	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

Рисунок 1 – Константная таблица замен

- 6. Блок данных преобразуется в матрицу бит размером 6 на 8.
- 7. Полученная матрица перемешивается по алгоритму «кубика Рубика».
- 8. Полученная после перемешивания матрица складывается по модулю два с раундовым ключом.

9. Каждый блок сдвигается на 1 байт влево.

Раундовые ключи вырабатываются из ключа шифра К с помощью процедуры расширения ключа, в результате чего формируется массив раундовых ключей, из которого затем непосредственно выбирается необходимый раундовый ключ.

Каждый раундовый ключ имеет длину 128 бит (или 4 четырехбайтовых слова $w_i, w_{i+1}, w_{i+2}, w_{i+3}$, а длина в битах всех раундовых ключей равна 128 бит). Первые четыре слова $w_i, w_{i+1}, w_{i+2}, w_{i+3}$ в ключевом массиве заполнены ключом шифра, из остальных выработанных 44 слов выбираются по 4 слова для ключа раунда. Выбор слов прост: первые четыре слова (они совпадают с ключом шифра) являются ключом с номером 0, следующие четыре слова w_4, w_5, w_6, w_7 – раундовым ключом для первого полного раунда и т.д.

Новые слова $w_{i+4}, w_{i+5}, w_{i+6}, w_{i+7}$ следующего раундового ключа определяются из слов $w_i, w_{i+1}, w_{i+2}, w_{i+3}$ предыдущего ключа на основе уравнений:

$$\begin{aligned} - w_{i+5} &= w_{i+4} \oplus w_{i+1}; \\ - w_{i+6} &= w_{i+5} \oplus w_{i+2}; \\ - w_{i+7} &= w_{i+6} \oplus w_{i+3}. \end{aligned}$$

Первое слово w_{i+4} в каждом раундовом ключе изменяется по- другому:

$$- w_{i+4} = w_i \oplus g(w_{i+3}).$$

Здесь действие функции g сводится к последовательному выполнению трех шагов, отображающих слово в слово:

- 1 Циклический сдвиг четырехбайтового слова влево на один байт.
- 2 Замена каждого байта слова, полученного на шаге 10, в соответствии с таблицей константных замен, используемой при шифровании.
- 3 Суммирование по mod 2 байтов, полученных на шаге 2, с раундовой постоянной $R_{con}[i] = (RC[i], 0, 0, 0)$, несекретной и уникальной для каждого раундового ключа. Три самые правые байты этой константы – нулевые, а ненулевой левый байт меняется по известному закону рекурсии: $RC[1] = 1, RC[i] = 2 * RC[i-1], i = 1, 2, \dots, 11$.

Цель суммирования с раундовыми константами – разрушить любую симметрию, что может возникнуть на разных этапах разворачивания ключа и привести к появлению слабых ключей, как в алгоритме DES.

Работа алгоритма расширения ключа продемонстрирована на рисунке 2.

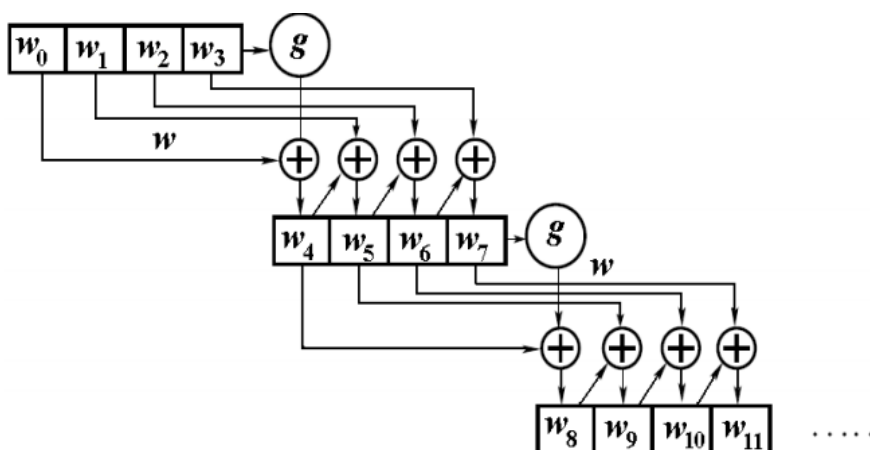


Рисунок 2. – Алгоритм расширения ключа

Заключение. В ходе данного исследования был спроектирован алгоритм шифрования на основе алгоритма кубика Рубика. При этом разработанный алгоритм оставляет возможность для доработки и введения дополнительных средств защиты.

ЛИТЕРАТУРА

1. Материал из StudFiles — файловый архив студентов. Шифры перестановки [Электронный ресурс]. – Режим доступа: <https://studfiles.net/preview/5470123/page:8/>. – Дата доступа: 20.09.2018.
2. Птицын, Н. Приложение теории детерминированного хаоса в криптографии / Н. Птицын. – М. : МГТУ им. Н.Э. Баумана, 2002. – 80 с.
3. Гатчин, Ю.А. Основы криптографических алгоритмов : учеб. пособие / Ю.А. Гатчин, А.Г. Коробейников. – СПб. : ГИТМО (ТУ), 2002. – 29 с.
4. Жданов, О.Н. Актуальные проблемы безопасности информационных технологий / О.Н. Жданов. – Красноярск : Сиб. гос. аэрокосмич. ун-т., 2009. – 144 с.

УДК 004.056.55

СРАВНИТЕЛЬНЫЙ АНАЛИЗ АЛГОРИТМА ШИФРОВАНИЯ НА ОСНОВЕ АЛГОРИТМА КУБИКА РУБИКА И AES

И.Е. ИВАНЕНКО*(Представлено: канд. физ.-мат. наук, доц. Д.Ф. ПАСТУХОВ)*

Рассматривается сравнение алгоритмов AES и алгоритма на основе алгоритмов кубика Рубика. Проведён анализ общих характеристик и работы алгоритмов.

Введение. AES – аббревиатура от Advanced Encryption Standard (перевод с англ. усовершенствованный стандарт шифрования).

После открытия в 1997 г. Национальным Институтом Стандартов и Технологии США (NIST) программы по разработке AES состоялись 3 этапа международного конкурса. Новый стандарт должен был иметь:

- стойкость не меньше, чем у 3DES.
- скорость шифрования больше скорости 3DES.
- прозрачную структуру.
- эффективную реализацию на платформе Pentium Pro.
- эффективную аппаратную реализацию.

Победителем конкурса стали бельгийцы Йоан Дамен и Винсент Реймен с алгоритмом RIJNDAEL (читается Рейн-дал от первых букв авторов). Стандарт начал действовать с 2002 г.

Алгоритм шифрования на основе кубика Рубика является шифром перестановки. Метод перестановки заключается в том, что символы шифруемого текста переставляются по определенным правилам внутри шифруемого блока символов, т.е. преобразования приводят к изменению только порядка следования символов исходного сообщения.

Данный алгоритм шифрования на основе алгоритма кубика Рубика был изменен для работы не с символами шифруемого текста, а с массивом байт, который получается путем преобразования шифруемого текста. Данный алгоритм является симметричным алгоритмом шифрования.

Сравнительный анализ алгоритмов.

Таблица 1. – Сравнение общих характеристик

Критерий сравнения	AES	Алгоритм шифрования на основе алгоритма кубика Рубика
1. Размер блока шифрования	AES зашифровывает и расшифровывает 128-битовые блоки данных	Данный алгоритм зашифровывает и расшифровывает 48-битовые блоки данных
2. Длина ключей шифрования	AES позволяет использовать три различных ключа длиной 128, 192 или 256 бит	Для работы алгоритма могут быть использованы ключи 128 бит, 256 бит, 512 бит, 1024 бита и 2048 бит
3. Число раундов	От размера ключа зависит число раундов шифрования: длина 128 бит – 10 раундов; длина 192 бита – 12 раундов; длина 256 бит – 14 раундов.	От размера ключа зависит число раундов шифрования: длина 128 бит – 12 раундов; длина 256 бит – 16 раундов; длина 512 бит – 20 раундов; длина 1024 бита – 24 раунда; длина 2048 бит – 32 раунда.

Исходя из данной таблицы 1, алгоритмы отличаются размером блока шифрования. Алгоритм шифрования на основе алгоритма кубика Рубика охватывает больший диапазон ключей, однако при одинаковом размере ключа имеет большее количество раундов, что может увеличить время работы алгоритма. Сравнение времени работы алгоритмов не имеет смысла без дальнейшей оптимизации алгоритма на основе алгоритма кубика Рубика, так как встроенные реализации алгоритма AES являются оптимизированными и выполняются в многопоточном режиме, тогда алгоритм на основе алгоритма кубика Рубика работает в однопоточном режиме и нуждается в дальнейшей оптимизации.

В таблице 2 представлены результаты работы AES и алгоритма на основе алгоритмов кубика Рубика на различных данных. Для работы алгоритмов был использован один и тот же ключе длиной 256 бит.

Таблица 2. – Результаты работы алгоритмов

Входной текст	Зашифрованный текст AES	Зашифрованный текст алгоритма на основе алгоритмов кубика Рубика
11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11	cc cf e 4 f 83 69 e2 a7 49 b9 2f d5 c0 6f d7 17 bc d1 ef dc 23 22 6c ac c1 49 c0 f7 e6 72 7c 63 34 26 d1 27 40 65 91 b7 ce 28 61 8b c da c7	a2 40 50 12 49 f2 12 f0 28 94 8b ef c8 ce c7 81 97 ab ae fe 65 c9 19 55 39 a7 42 84 72 40 85 78 f6 5 d 4b 99 c2 80 66 b1 a9 19 3b f 72 86 e
11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 10	49 3a 55 c4 df 86 8f 1f b 5b 90 40 d6 5 a9 49 a4 33 67 a 1f de ad a9 be f3 c6 ee 6a 1d 71 72 74 99 aa 5d 3 4a d1 a5 3d 4e f5 39 b2 24 da dd	6f 71 b9 1f c3 4b 54 ef 55 fc 39 c1 60 56 9a de a3 a7 b3 54 55 14 ed d9 2e 9f 4b 9a 8d 72 5 cf 4c 29 be 72 8f c3 46 2f f0 74 69 d5 a8 7b b3 a6
11 22 33 66 55 44 55 44 77 88 99 66 44 45 36 12	67 35 e5 1e 22 62 14 ca 7c 3e 57 3b c0 c7 59 7e e7 af 22 ed f4 b9 29 37 91 d2 17 e6 c6 23 0 c0 23 73 8 9c 8c c8 b9 98 5c f5 a3 ba e0 9c 7d	bd f8 5f f4 d7 49 6a 62 92 54 75 f2 3c e0 31 15 36 fc 3f ed 78 ce bf c8 a2 ad 25 4f 7e 57 2 cf 1c 60 e8 c9 bc e3 36 fc b8 ed c4 24 3d a 5f d5
11 22 33 66 55 44 55 44 77 88 99 66 44 45 36 11	5 fa c0 bb 6a 61 81 f3 2b 34 f6 2a dd 38 5 59 a8 b8 85 eb 2f ed 6d 96 44 38 25 d5 eb a2 d0 79 14 14 d6 b6 f8 38 73 70 57 21 99 b7 1e fe f1 5e	cd e4 96 2a 5 1e 4 11 1e c7 de 13 9d 7a 6d ed a8 38 5f 80 7a 6 3e 15 e6 a9 fc 44 43 54 f4 3d 9 11 16 91 f 35 f4 ed 2e 48 bd c5 6a 8c f0 cb
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	2 97 6c 34 50 fd 7a 19 5e a7 9d ae 98 19 f3 1b 83 ea 5f 6a 2f db 28 67 bb bc dd 11 11 f3 e8 c6 78 7a 88 2d b9 ff f5 69 94 3d fl 89 9c dd 5 5b	2e 24 28 fb 9a 5e 84 c0 7b 84 c2 74 68 d6 1f ce 81 fd e6 fa 3a 44 e0 79 db f7 23 23 f7 3 91 12 ae ec 92 42 23 8c 65 e0 7 e3 17 a8 1c eb ff 7b
10 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0 57 be 43 5e 52 6d 71 b1 bb 57 10 19 bc f8 e5 a7 b9 40 b 9e 8 28 82 fd 8c 49 a9 1a 5b ba 9d 24 55 d4 74 1e 71 4 68 9c ff 33 8 eb 6e 3b b4	6a 77 73 88 ae 6a 3 ad 3e 23 1f 70 37 3c 1b 56 75 e5 d 37 b9 63 64 9d 2e 89 6f a0 93 97 b6 bd e0 f9 a9 da d4 ff a0 39 f7 5f 73 59 8a 87 e4 c4

Исходя из данных таблицы 2 можно сделать вывод, что оба алгоритма обладают так называемым «лавиным эффектом» и зашифрованные данные обладают свойством нелинейности. «Лавинный эффект» означает, что изменения в одном бите входных данных должны распространяться по всем битам выходных данных. В свою очередь нелинейность означает невозможность подобрать линейную функцию, хорошо аппроксимирующую данное преобразование.

Заключение. В ходе данного исследования были получены результаты, исходя из которых можно сделать вывод, что алгоритм на основе алгоритмов кубика Рубика обладает некоторыми преимуществами, такими как больший диапазон ключей для шифрования по сравнению с алгоритмом AES. Так же исходя из результатов данный алгоритм обладает рядом существенных недостатков, такими как скорость работы алгоритма, по сравнению с алгоритмом AES, небольшой размер блока шифрования равный 48 бита, что негативно сказывается на скорости работы и при больших объемах данных может быть нарушен «лавиный эффект».

ЛИТЕРАТУРА

1. Материал из StudFiles — файловый архив студентов. Шифры перестановки [Электронный ресурс]. Режим доступа: <https://studfiles.net/preview/5470123/page:8/>. Дата доступа: 20.09.2018.
2. Н. Птицын. Приложение теории детерминированного хаоса в криптографии / Н. Птицын; доцент(к.н.) к.т.н. – Москва: МГТУ им. Н.Э. Баумана, 2002. 80 с.
3. Ю.А.Гатчин, А. Г. Коробейников. Основы криптографических алгоритмов. Учебное пособие / Ю.А.Гатчин; доктор технических наук, профессор кафедры проектирования и безопасности компьютерных систем. – СПб: ГИТМО (ТУ), 2002. 29 с.
4. Жданов О.Н., Актуальные проблемы безопасности информационных технологий/ Жданов О.Н. кандидат физико-математических наук, Профессор Российской Академии Естествознания - Сиб. гос. аэрокосмич. ун-т. – Красноярск, 2009. – 144 с.
5. Общее описание криптоалгоритма AES [Электронный ресурс]. – Режим доступа: <http://bit.nmu.org.ua/ua/student/metod/cryptology/лекция%209.pdf> – Дата доступа: 20.09.2018.

УДК 004.021

**РАЗРАБОТКА СИСТЕМЫ ДЛЯ СКРЫТИЯ ИНФОРМАЦИИ
С ИСПОЛЬЗОВАНИЕМ ГИПЕРБОЛИЧЕСКИХ ФУНКЦИЙ****А.И. СМОЛЯК***(Представлено: канд. физ.-мат. наук, доц. Д.Ф. ПАСТУХОВ)*

Рассматриваются: алгоритм скрытия данных в изображении на основе дискретных преобразований, описание работы основной программы и основные функции разрабатываемого программного продукта.

Введение. Первое появление гиперболических функций историки обнаружили в трудах английского математика Абрахама де Муавра. Современное определение и обстоятельное их исследование выполнил Винченцо Риккати в 1757 году, он же предложил их обозначения: sh, ch. Независимое открытие и дальнейшее исследование свойств гиперболических функций было проведено Иоганном Ламбертом, который установил широкий параллелизм формул обычной и гиперболической тригонометрии. Н. И. Лобачевский впоследствии использовал этот параллелизм, пытаясь доказать непротиворечивость неевклидовой геометрии, в которой круговая тригонометрия заменяется на гиперболическую.

Основная часть. Для разработки программного средства организации и функционирования программы необходимо выбрать среду разработки, с помощью которой будет производиться проектирование.

В основе систем быстрой разработки (RAD-систем, Rapid Application Development – среда быстрой разработки приложений) лежит технология визуального проектирования и событийного программирования, суть которой заключается в том, что среда разработки берет на себя большую часть рутинной работы, оставляя программисту работу по конструированию диалоговых окон и функций обработки событий. Производительность программиста при использовании RAD-систем является рекордно высокой.

Также можно рассмотреть такую среду разработки, как и Microsoft Developer Studio. Студия разработчика фирмы Microsoft (Microsoft Developer Studio) – это интегрированная среда для разработки, позволяющая функционировать различным средам разработки, одна из которых Visual C++, другая - Visual J++. В дальнейшем будет идти речь только о среде разработки Visual C++. Среда разработки Visual Studio позволяет создавать библиотеки dll на языке C++ и подключать основное ядро программы, написанное на C++ к интерфейсу, созданному на FORTRAN и динамически управляемым югославской библиотекой Xtfort.

Основными функциями разрабатываемого программного продукта являются:

- шифрование информации;
- дешифрование информации;

Рассматриваемая методика ляжет в основу разрабатываемого программного продукта «Скрытие информации с использованием гиперболических функций».

Основная идея алгоритма шифрования алгоритма шифрования гиперболическими функциями является шифрование каждого первого, второго и третьего символа сообщения отдельными ключами. Каждый из этих ключей имеют 3 параметра. В результате дешифрование сообщений только одним или двумя ключами с правильными параметрами ключей принципиально невозможно.

Подробнее опишем алгоритм работы программы.

1. Каждому английскому символу, цифрам и знакам клавиатуры соответствует некоторое целое число(номер) от 0 до 255, с помощью таблицы ASCII можно перевести любой символ клавиатуры в число, например, слово Polotsk шифруется 7 символами 80 111 108 111 116 - 115 107.

2. Всего основных символов в клавиатуре 256 с участием больших и малых букв английского шрифта, невидимые знаки (табуляция, начало и конец строки и т.д.) а также знаки типа «!»,»,№,(,) и т.д., цифры 0,1,2,3,4,5,6,7,8,9 пронумерованы от 0 до 255. Мы можем перевести каждый символ после применения команды ASCII в действительное число, заключенное от 0 до 1 по формуле $x = n / 255, 0 \leq x \leq 1$, где n -номер символа в ASCII.

3. Далее с единичным отрезком мы проводим линейное преобразование, растягивающее его в несколько раз и с применением параллельного переноса, смещающее левую точку по формуле: $res2[i] = a + (b - a)res1[i]$, где a, b левая и правая границы нового отрезка. В результате последнего преобразования все точки преобразованного отрезка имеют координаты $x_i \in [a, b]$. Сначала работает прямая функция с вызовом $res3[i]=c*f(res2[i],k)$. В которой по порядку следуют аргументы: число $x = res2[i], x \in [a, b]$, c- коэффициент подобия, k номер гиперболической функций целое k=1 для гипер-

болического синуса и $k = 1$ для гиперболического косинуса. Поскольку диапазон целых чисел значительно меньше диапазона действительных чисел с плавающей запятой двойной точности, то массив $res101[i]$ ($res101[i] = res3[i]$) заполняется действительными числами двойной точности, здесь применяется одна важная идея. В файл 1001.txt записываются сначала целые части действительных чисел шифра – все нечётные строки текстового файла $sad[2*i+1]$. Затем мантиссы шифра переводятся в целые числа (первые 9 знаков) и заполняются все чётные строки текстового файла $sad[2*i]$. Иначе при считывании чисел из файла 1001.txt может быть потеряна мантисса действительного числа. Это во много раз может снизить размерность пространства шифрующих ключей и надёжность дешифрования данных.

4. Открываем и копируем данные текстового файла 1001.txt в массив $sad[2*i]$, затем из каждой соседней пары целых чисел формируем целую часть и мантиссу шифра, создаём массив по формуле $res102[i] = double(sad[2*i]) + double(sad[2*i+1])*1e-9$

5. Далее столбцы кода до записи в текстовый файл и после чтения из него сравниваются $res3[i]$; и $res102[i]$ (должны совпадать).

6. Прочитанные данные записываются в файл $res102[i]$.

7. Проводим обратное масштабирование $res4[i] = res102[i]/c$.

8. Вызываем обратную функцию дешифрования: $res5[i] = \log(res4[i] + \sqrt{res4[i]^2 + L})$

$$x = \ln(z) = \ln\left(y + \sqrt{y^2 + L}\right), L = \begin{cases} 1, y(x) = sh(x) \\ -1, y(x) = ch(x) \end{cases} \Leftrightarrow y(x) = \frac{e^x - L e^{-x}}{2}, L = -1, 1.$$

9. Проводим обратное линейное преобразование и возвращаемся к переменной x по формуле: $res6[i] = 255.0 * (res5[i] - a) / (b - a)$.

10. Переводим действительный массив дешифрованного символа в целочисленный массив: $res7[i] = \text{int}(res6[i])$.

11. Делаем обратное преобразование ASCII. В результате исходная символьная фраза программой возвращается в эту же фразу. Таким образом, в программе использованы математические операции запись чисел с двойной точностью в текстовый файл, композиция из преобразования подобия, нелинейных гиперболических функций, однородного сжатия функции:

$$x_1 = \frac{n}{255} \cdot x_2 = a + (b - a)x_1, y_1 = \frac{\exp(x_2) - l \cdot \exp(-x_2)}{2}, y_2 = c y_1.$$

12. Обратные преобразования:

$$y_1 = \frac{y_2}{c}, x_2 = \ln\left(y_1 + \sqrt{y_1^2 + l}\right), x_1 = \frac{x_2 - a}{b - a} \cdot n = 255 x_1, (l = -1 \Leftrightarrow ch(x)).$$

Функция шифрования $f(x, k)$ содержит два аргумента. Первый аргумент x – действительное неотрицательное число, преобразованное шифруемым символом. Второй целочисленный аргумент принимает значение $k = 1$ для гиперболического синуса и $k = 2$ для гиперболического косинуса m - число символов в сообщении.

Листинг 1 – Задание параметров ключей

```

1: double f(double x, int k){
2:   if(k==1){ return (exp(x)-exp(-x))/2.0; }
3:   if(k==2){ return (exp(x)+exp(-x))/2.0; } }
4: int const m=20
5: main(){
6:   int i,aa,bb,k,l, res7[m];
7:   double x, res1[m], res2[m], res3[m], a1,b1,c1;
8:   double res4[m], res5[m], res6[m];
9:   double a2,b2,c2,a3,b3,c3,res101[m], res102[m];
10:  ]char str[m+1]="Smolak Igor 2018 FIT";
11:  k=2;l=1;
12:  for(i=1;i<=k+1;i++) {l=l*(-1);}
13:  a1=2.0;b1=3.0;c1=4.0;
14:  a2=7.0;b2=10.0;c2=5.0;
15:  a3=0.0;b3=3.0;c3=6.0;
```



```

16: for(i=0;i<=m-1;i++){
17: res1[i]=double(str[i])/255.0;

```

Листинг 2 – Шифрование

```

1:  if(i%3==0){
2:  res2[i]=a1+(b1-a1)*res1[i];
3:  res3[i]=c1*f(res2[i],k); }
4:  else if(i%3==1){
5:  res2[i]=a2+(b2-a2)*res1[i];
6:  res3[i]=c2*f(res2[i],k); }
7:  else if(i%3==2){
8:  res2[i]=a3+(b3-a3)*res1[i];
9:  res3[i]=c3*f(res2[i],k);}
10: printf("i=%d %c %d normal=%.16lf coder=%.16lf\n",i,str[i],
11: str[i],res1[i],res3[i]);}
12: for(i=0;i<=m-1;i++) {
13: printf("%.16lf\n", res3[i]);
14: res101[i]= res3[i]; }
15: FILE*file;
16: int sad0[2*m+1];
17: remove("1001.txt");
18: file=fopen("1001.txt","w");
19: for(i=0;i<=m-1;i++){
20: aa=int(res3[i]);
21: bb=int ((res3[i]-aa)*1e9);
22: printf("coder1(%d)=%.12lf\n",i,res3[i]);
23: fprintf(file,"%d\n", aa);
24: sad0[2*i]=aa;
25: fprintf(file,"%d\n", bb); )
26: sad0[2*i+1]=bb;}
27: fclose(file);
28: int sad[2*m+1];
29: char arr00[2*m+1];
30: file=fopen("1001.txt","r");
31: if(file==NULL){
32: printf(" not open file");}
33: else{
34: for(i=0; i<=2*m-1;i++){
35: fgets(arr00,2*m-1,file);
36: sad[i]=atoi(arr00);
37: printf(" coder2(%d)=%d coder1(%d)=%d\n", i, sad[i],i, sad0[i]);} }
38: fclose(file);
39: for(i=0;i<=m-1;i++){
40: res102[i] = double(sad[2*i]) + double(sad[2*i+1])*1e-9;
41: printf(" coder1(%d)=%.8lf coder2(%d)=%.8lf\n", i, res101[i], i,
42: res102[i] );}
43: for(i=0;i<=m-1;i++){
44: if(i%3==0){
45: res4[i]= res102[i]/c1;
46: res5[i]=log(res4[i]+sqrt(res4[i]*res4[i]+1));
47: res6[i]= 255.0*(res5[i]-a1)/(b1-a1);}
48: else if(i%3==1){
49: res4[i]= res102[i]/c2; // res4[i]= res3[i]/c2
50: res5[i]=log(res4[i]+sqrt(res4[i]*res4[i]+1)) ;// res5[i]= res2[i]
51: res6[i]= 255.0*(res5[i]-a2)/(b2-a2);} // res6[i]= res1[i]* 255.0
52: else if(i%3==2){
53: res4[i]= res102[i]/c3;
54: res5[i]=log(res4[i]+sqrt(res4[i]*res4[i]+1));
55: res6[i]= 255.0*(res5[i]-a3)/(b3-a3);}

```

```
56: res7[i]= int(res6[i])+1;  
57: printf("i=%d text(%d)=%c decoder(%d)=%c \n",i,i,str[i],i,res7[i]);}}
```

Заключение. В данной статье был рассмотрен алгоритм скрывания информации с использованием гиперболических функций, описание работы основной программы и основные функции разрабатываемого программного продукта.

ЛИТЕРАТУРА

1. Жданов, О.Н. Эллиптические кривые: Основы теории и криптографические приложения / О.Н. Жданов, В.А. Чалкин. – М. : Книжный дом ЛИБРИКОМ, 2013. – 200 с.
2. Бутакова, Н.Г. Криптографическая защита информации : учеб. пособие для вузов / Н.Г. Бутакова, В.А. Семенов, Н.В. Федоров. – М. : Изд-во МГИУ, 2011. – 316 с.

УДК 004.021

ПРОЕКТИРОВАНИЕ ГРАФИЧЕСКОГО ИНТЕРФЕЙСА СИСТЕМЫ ДЛЯ СКРЫТИЯ ИНФОРМАЦИИ С ИСПОЛЬЗОВАНИЕМ ГИПЕРБОЛИЧЕСКИХ ФУНКЦИЙ

*А.И. СМОЛЯК**(Представлено: канд. физ.-мат. наук, доц. Д.Ф. ПАСТУХОВ)*

Рассматривается проектирование графического интерфейса системы для скрытия информации с использованием гиперболических функций. Проведён анализ степени пригодности контейнера для модификации, моделирование атак и определение устойчивости к ним.

Введение. Необходимость в скрытии информации у человечества появилась очень давно. Вместе с появлением необходимости в скрытии информации, появилась и необходимость в взломе шифров. Так появилась криптология, наука о создании и взломе шифров.

Интерфейс программы должен обладать целым рядом свойств: естественность, согласованность, дружелюбность, простота, гибкость, эстетическая привлекательность.

Основная часть. Приложение «ExponentLog» представляет собой приложение для скрытия информации. В интерфейсе представлены 3 ключа шифрования, для каждого ключа есть 3 параметра, 2 функции на выбор: гиперболический косинус (1) и гиперболический синус (2).

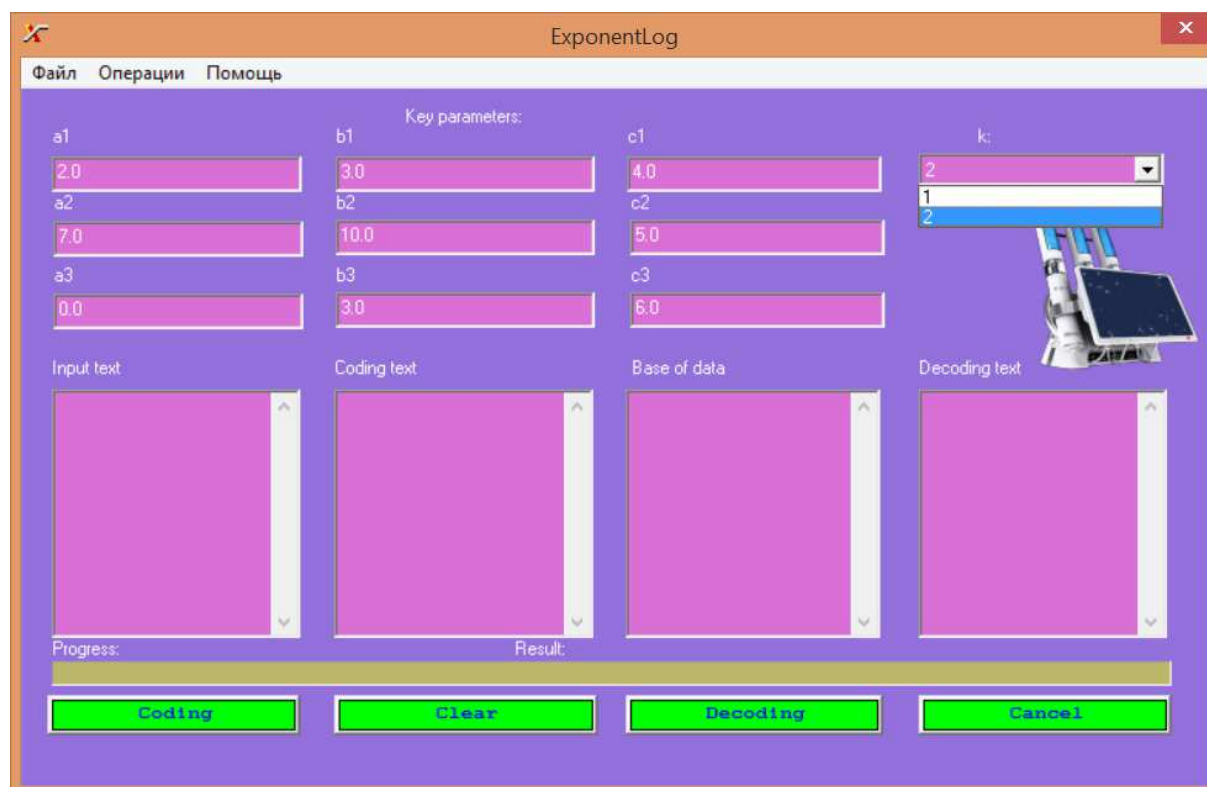


Рисунок 1. – Интерфейс программы

Для скрытия сообщения необходимо будет набрать текст для шифрования в поле Input text, выбрать 9 ключей шифрования и функцию k, нажать кнопку Coding. После нажатия произойдет заполнение строки состояния и текст будет зашифрован.



Рисунок 2. – Интерфейс шифрования

При нажатии кнопки Decoding происходит декодирование сообщения.



Рисунок 3. – Интерфейс дешифрования

Однако, если изменить хотя бы 1 из 9 ключей, то дешифрованное сообщение будет изменено.

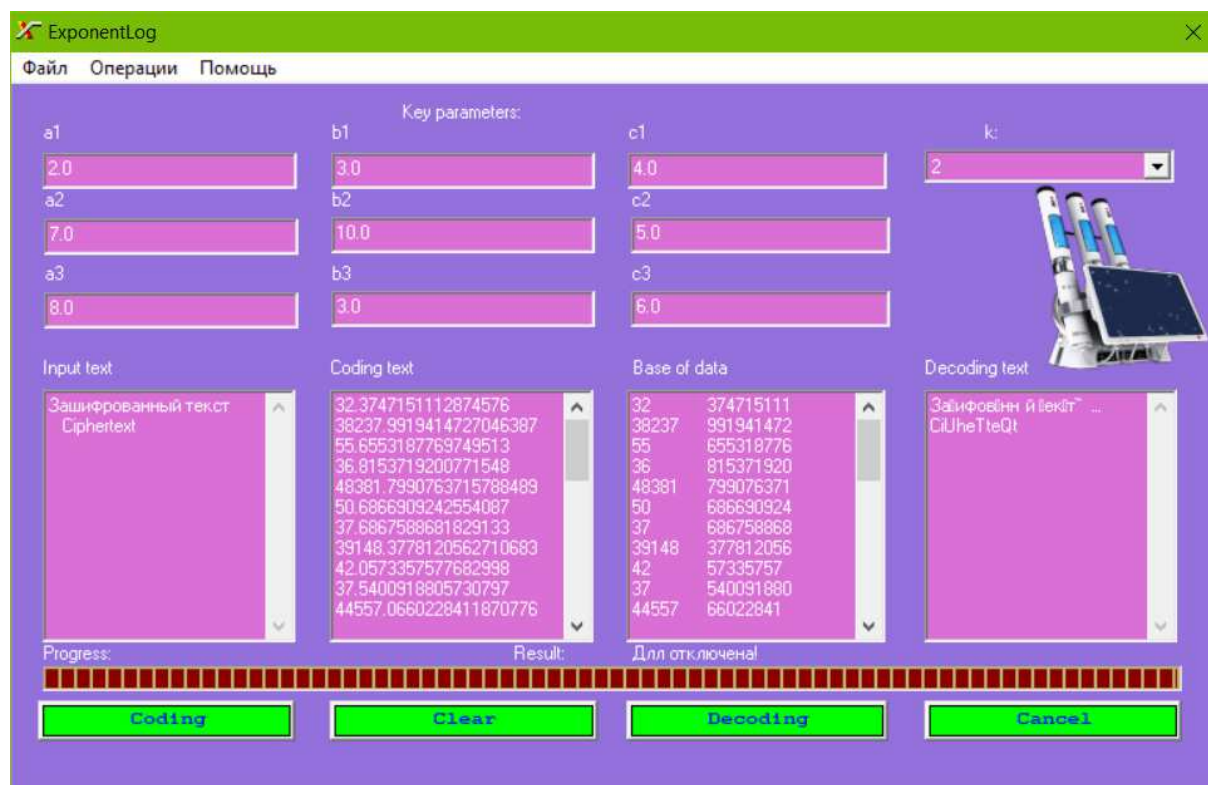


Рисунок 4. – Ошибка в дешифрованном тексте при изменении параметров ключа

Неимоверно большая чувствительность шифра относительно малых 10^{-10} – 10^{-12} изменений параметров ключа (*Left, right, a*), (*Left, right, n*), обеспечивает большую размерность пространства распределения ключей. Если бы суперкомпьютер криптоаналитика подбирал ключи к фиксированному шифру нелинейными функциями со скоростью $10^9 \frac{1}{сек}$, то понадобилось бы время взлома, превосходящее время существования Земли (более 5 млрд. лет). Такая большая размерность обеспечивается применением чисел двойной точности (double) для аргументов и нелинейных функций. Действительно, в единичный диапазон изменения параметра одного ключа можно разместить 10^{16} различных шифров. Криптостойкость шифрования нелинейными функциями обеспечивается двумя вескими причинами: большая размерность пространства ключей, большой набор нелинейных функций с «плавающей» областью определения – отрезком, позволяющей с одной стороны увеличить пространство ключей, а с другой стороны повысить криптостойкость шифрования.

В данном приложении применялись 2 класса нелинейных функций: гиперболический синус и гиперболический косинус. Применяемые алгоритмы в программе можно использовать для хранения в базе данных паролей длиной до нескольких сотен – тысяч символов. Добавление алгоритмов рандомизации к шифрованию нелинейными функциями делает применяемые алгоритмы неуязвимыми для криптоаналитика.

Заключение. В данной статье рассмотрен способ построения графического интерфейса системы скрытия информации на основе нелинейных функций, проведён анализ криптостойкости шифрования нелинейными функциями.

ЛИТЕРАТУРА

1. Матюшик, В.Н. Методы и средства стеганографии для защиты графических образов / В.Н. Матюшик. – Минск : Белорус. гос. ун-т информатики и радиоэлектроники.
2. Грибунин, В.Г. Цифровая стеганография / В.Г. Грибунин, И.Н. Оков, И.В. Туринцев. – М. : Солон-Пресс, 2002.
3. Конахович, Г.Ф. Компьютерная стеганография. Теория и практика / Г.Ф. Конахович, А.Ю. Пузыренко. – Киев : МК-Пресс, 2006.

УДК 004.021

**ПРОЕКТИРОВАНИЕ ГРАФИЧЕСКОГО ИНТЕРФЕЙСА ДЛЯ ВЕБ-ПРИЛОЖЕНИЯ,
РАЗРАБОТАННОГО ДЛЯ ОБМЕНА ИНФОРМАЦИЕЙ
И ФАЙЛАМИ В ОГРАНИЧЕННОМ КРУГУ ЛИЦ****П.В. МАХАНЬКО***(Представлено: Ю.Н. КРАВЧЕНКО)*

Рассматривается построение графического интерфейса пользователя для веб-приложения.

Введение. Создание графических интерфейсов в настоящее время базируется в основном на использовании:

- 1) языка гипертекстовой разметки HTML;
- 2) таблицы каскадных стилей CSS;
- 3) мультипарадигменного языка программирования Javascript

Разметка страницы осуществляется средствами языка HTML. При помощи этого языка, на странице создаются основные элементы, с которыми взаимодействует пользователь. Используя таблицу стилей CSS) можно достичь определенной стилистики страницы, а также описать анимацию и изменения страницы, в процессе ее использования. Основой служит язык программирования Javascript (Angular 5). Именно Javascript позволяет взаимодействовать страницам между собой, делать HTTP запросы к серверу и динамически изменять содержимое страницы.

Данная статья направлена на подробное описание технологий построения интерфейса в веб-приложении.

Основная часть. Как правило, приложения в Angular 5 строятся по шаблону MVC (Model-View-Controller или Модель-Вид-Контроллер). Идея данного шаблона проста – разделение обязанностей: задача контроллера – обработка действий пользователя (нажатия по кнопкам, обработка запросов к серверу и т.д.); модель предоставляет контроллеру данные, запрашиваемые пользователем; вид, в свою очередь, обеспечивает представление данных, полученных из модели.

Файл с расширением html – файл разметки, сформированный на языке HTML при помощи тегов. Тег - именованная метка или дескриптор, который содержит в себе информацию об элементе HTML - страницы и его расположении на данной странице.

Стилистические особенности прописываются в файлах с расширением css. Применение определенных стилей к определенным элементам разметки в файлах html производится при помощи классов и идентификаторов. Таблица стилей CSS позволяет инкапсулировать стили определенных тегов при помощи вложенности CSS селекторов и позволяет создавать функции, которые можно вызывать в разных селекторах, что значительно облегчает понимание кода.

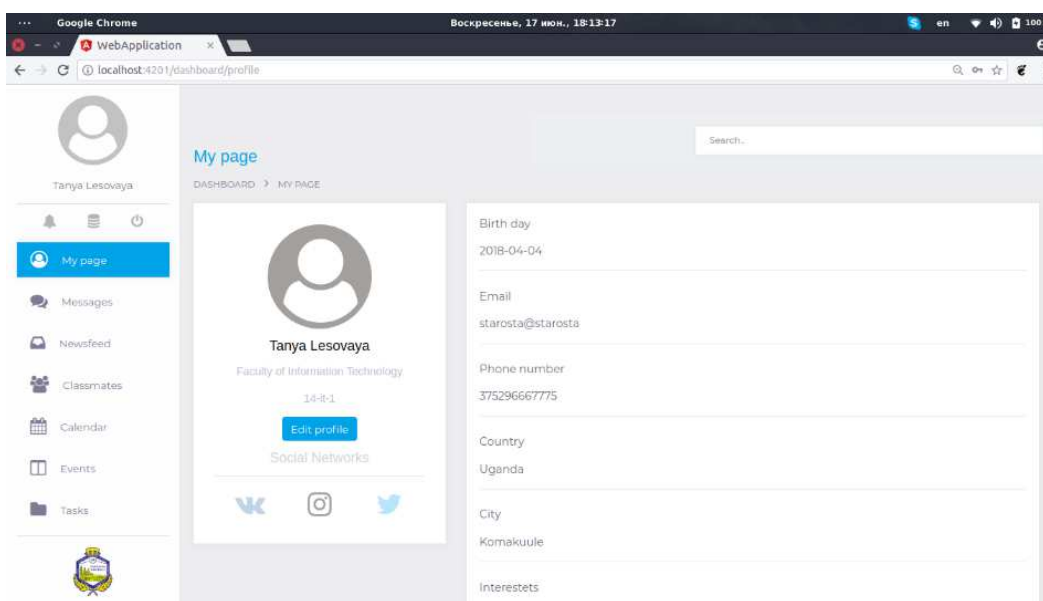


Рисунок 1. – Главная страница приложения

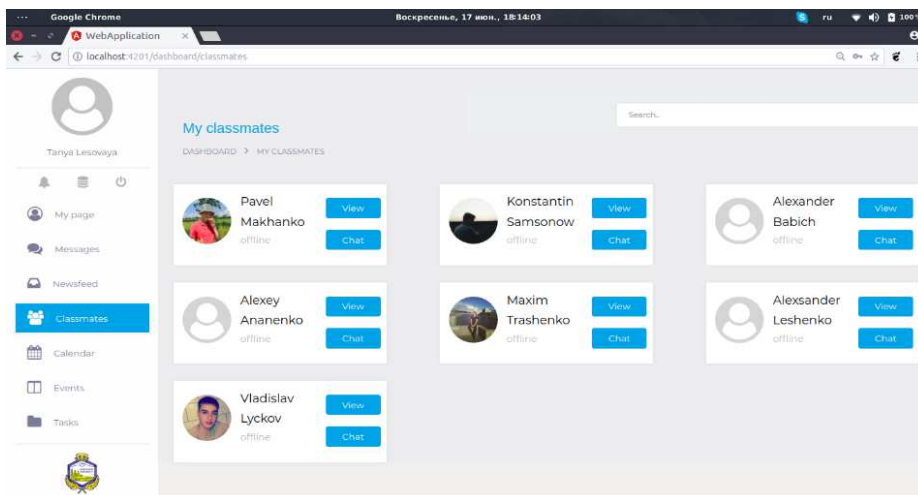


Рисунок 2. – Страница контактов.

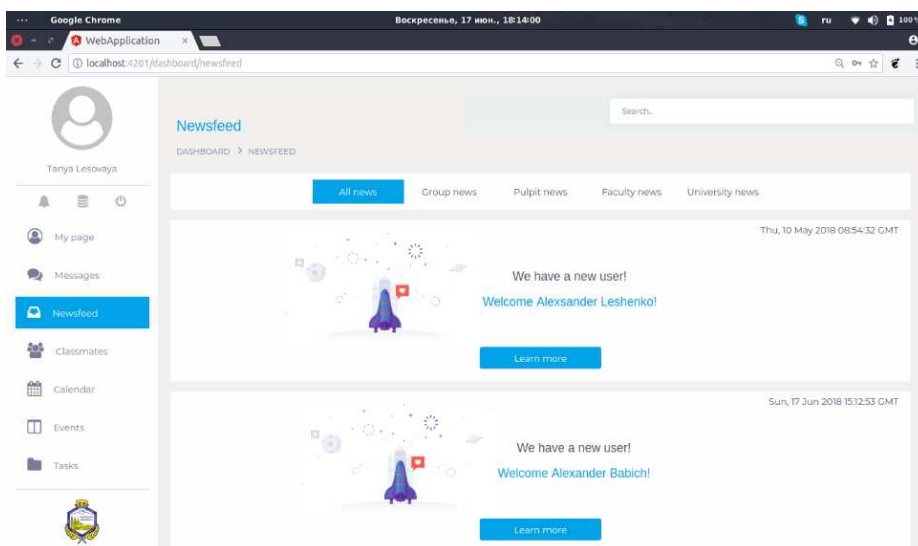


Рисунок 3. – Новостная лента

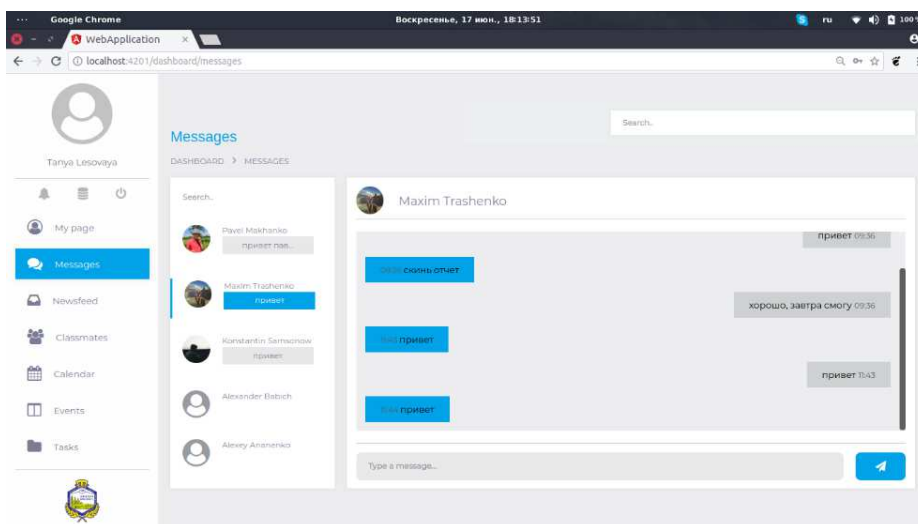


Рисунок 4. – Персональные сообщения

В качестве контроллеров используются файлы с расширением ts. Файлы ts пишутся на языке TypeScript, который расширяет возможности языка JavaScript. Одним из основных преимуществ TypeScript является статическая типизация (реализация ООП в классическом виде, более понятном для большинства программистов).

Взаимодействие html разметки и контроллеров происходит при помощи:

- атрибутивных директив;
- структурных директив;
- привязки данных.

В Angular есть четыре формы привязки данных:

- 1) привязка элемента DOM к значениям компонента;
- 2) привязка свойства элемента html к значению компонента;
- 3) привязка метода компонента к событию в DOM;
- 4) двусторонняя привязка, когда элемент DOM привязан к значению в компоненте, при этом изменения на одном конце привязки сразу приводят к изменениям на другом конце.

В качестве структурных директив используются такие, как ngIf, ngFor и ngSwitch (по аналогиям с конструкциями из программирования if, for и switch).

Атрибутивные директивы изменяют поведения элементов, изменяя его атрибут. Например, директива ngClass позволяет установить определенный класс, при определенных событиях.

Заключение. В данной статье рассмотрены основные элементы построения графического интерфейса для веб-приложения. Представлены примеры и результаты.

УДК 004.021

**ПРОЕКТИРОВАНИЕ API И БАЗЫ ДАННЫХ ДЛЯ ВЕБ-ПРИЛОЖЕНИЯ,
РАЗРАБОТАННОГО ДЛЯ ОБМЕНА ИНФОРМАЦИЕЙ И ФАЙЛАМИ
В ОГРАНИЧЕННОМ КРУГУ ЛИЦ****П.В. МАХАНЬКО***(Представлено: Ю.Н. КРАВЧЕНКО)**Рассматриваются цели и принципы моделирования API и базы данных для веб-приложения.*

Введение. В наше время трудно представить хорошее веб-приложение без разработанной базы данных и быстрого, оптимизированного API. База данных позволяет удобно хранить большие объемы информации не на устройстве пользователя, а на удаленном сервере, а API позволяет обрабатывать запросы веб-приложения, разграничивать права доступа и выдавать данные из базы данных, на любое устройство, где находится веб-приложение.

Данная статья направлена на описание базы данных и API для разработанного приложения.

Основная часть. Основной целью разработки API, является создание промежуточного звена между клиентской частью веб-приложения и базой данных. Это является отличным способом защиты пользовательских данных, а также обеспечение быстрого доступа к информации, не нагружая устройство пользователя.

В качестве защиты используется JSON Web token. Это открытый стандарт, создания токенов доступа основываясь на json. В момент авторизации пользователя, сервер генерирует хеш-ключи, которые отправляет клиентской части веб-приложения. В дальнейшем, каждый запрос к серверу, должен содержать в себе хеш-ключ для подтверждения прав доступа к ресурсам.

В качестве точек для запросов, сервер предоставляет веб-приложению конечные адреса, обращаясь к которым, веб-приложение получает определенные данные. Запросы проходят по стандартному HTTP протоколу, используя методы GET, POST. В качестве точек доступа, сервером представлены следующие адреса:

- Get /api/classmates - получение списка одноклассников;
- Get /api/classmate?classmate_id - получение информации об однокласснике по id;
- Get /api/user?user_id - получение персональной информации по id;
- Get /api/dialog?user_id получение диалогов по id;
- Get /api/new_messages?user_id получение новых сообщений с момента последнего визита в приложение;
- Get /api/last_visit_dialog?user_id - получение 5 последних диалогов с учетом новых сообщений;
- Get /api/newsfeed?user_id - получение новостной ленты определенного пользователя;
- Post /auth/login - авторизация пользователя;
- Post /auth/register - регистрация пользователя;
- Post /auth/register_complete завершение регистрации.

Чат и обмен файлами реализованы по технологии сокетов (протокол WebSocket). Сокет - протокол дуплексной связи между клиентом и сервером в режиме реального времени. В данном случае, веб-приложение подписывается на новые сообщения при успешном входе в приложение. Затем при получении нового сообщения оно мгновенно будет отправлено пользователю.

В качестве базы данных была выбрана документно-ориентированная база данных MongoDB. Синтаксис MongoDB схож с форматом json, а это в свою очередь не задает жестких рамок хранения данных, что позволяет удобно расширять хранимые данные. Вся база данных состоит из коллекций, а хранимые в ней данные - документы. Документ представляет из себя json-объект в виде пар ключ/значение, где в качестве значения могут храниться строки, числа, двоичные данные, массивы или другие вложенные объекты.

Реляционная модель представляет собой базу данных в виде множества взаимосвязанных отношений (таблиц), которые используются для хранения информации об объектах, представленных в базе данных.

В соответствии с требованиями программы, определены коллекции (набор) необходимых таблиц базы данных, представленный в таблице.

Таблица. – Перечень необходимых таблиц базы данных

№ п/п	Название коллекции	Описание
1	Users	Список всех пользователей
2	Groups	Список всех учебных групп
3	Pulpits	Список всех кафедр
4	Faculties	Список все факультетов
5	Messages	Сообщения пользователей
6	Newsfeeds	Новостные ленты пользователей

Важное достоинство баз данных – возможность хранения сгруппированных данных в разных коллекциях и в разных форматах.

Заключение. В данной статье были рассмотрены основные цели и принципы моделирования базы данных, построение быстрого, удобного API для веб-приложения. Были рассмотрены основные сущности базы данных. Спроектированы адреса для обращения клиентской части веб-приложения к серверной части.

УДК 004.021

РАССМОТРЕНИЕ ФРЕЙМВОРКОВ RНР В ОБЩЕМ ВИДЕ И ИЗУЧЕНИЕ УП В ЧАСТНОСТИ

В.С. РАДЧЕНКО

(Представлено: Д.В. ПЯТКИН)

Представлено теоретические данные по вопросу RНР-фреймворков и частное рассмотрение фреймворка УП.

1. ФРЕЙМВОРК ПРОГРАММНОЙ СИСТЕМЫ

Фреймворк (англ. framework, син. Каркас) - в информационных системах структура программной системы; программное обеспечение, облегчающее разработку и объединение разных компонентов большого программного проекта. В отличие от библиотек, которые объединяют набор подпрограмм близкой функциональности, фреймворк содержит в себе большое количество разных по назначению библиотек. Употребляется также слово каркас, а некоторые авторы используют его в качестве основного, в том числе не базируясь вообще на англоязычном аналоге. Можно также говорить о каркасном подходе как о подходе к построению программ, где любая конфигурация программы строится из двух частей: первая, постоянная часть - каркас, не меняющийся от конфигурации к конфигурации и несущий в себе гнезда, в которых размещается вторая, переменная часть - сменные модули (или точки расширения). Простыми словами: Фреймворк – это набор всевозможных библиотек (инструментов) для быстрой разработки повседневных задач.

2. КОНЦЕПЦИЯ RНР ФРЕЙМВОРКОВ

Фреймворки за последнее время набрали популярность, и стали базовой платформой для разработки веб-приложений. Другими словами, можно сказать, что они обеспечивают основную структуру приложения. Использование RНР-фреймворков, позволяет экономить большое количество времени, уменьшить нагрузку на процесс разработки, избавиться от проблемы повторяющегося кода, и быстро создавать приложения. Без использования RНР-фреймворков, становится гораздо сложнее создавать веб-приложения, сопровождать и модернизировать их. Между тем, использование RНР фреймворков делает процесс создания приложения гораздо более легким и функциональным.

Сейчас большинство RНР проектов построены при помощи архитектуры Model View Controller(MVC). MVC - это архитектурный шаблон проектирования, который используется в большинстве языков программирования и позволяет отделить бизнес-логику от пользовательского интерфейса, а также выделить область логики, которая производит обмен информации между базой данных и пользовательским интерфейсом. Таким образом можно изменить логику приложения, не затрагивая интерфейсной части, либо наоборот, что очень хорошо для дизайнеров и верстальщиков. Это позволяет избежать путаницы и упрощает весь процесс разработки. Когда говорится о MVC, то имеется в виду: Model - та часть архитектуры, которая взаимодействует с базой данных, View - представляет ту часть, которую непосредственно видит пользователь, то есть графический интерфейс, и Controller - это область логики, которая контролирует и управляет всеми ее составляющими и данными. Большинство современных фреймворков берут за основу именно архитектуру MVC. Так же в современном фреймворке используется шаблон проектирования Front Controller, который, в зависимости от запроса, перенаправляет его на нужный контроллер. Без Front Controller разработка с применением фреймворка не имела бы смысла.

3. СЛУЧАИ ИСПОЛЬЗОВАНИЯ RНР ФРЕЙМВОРКА

Для того чтобы воспользоваться всеми возможностями фреймворка, необходим немалый багаж знаний в разработке приложений. RНР-фреймворки могут помочь устранить очень частую ошибку при программировании приложений, а именно повторение кода, а также систематизировать процесс разработки. Фреймворки являются мощным инструментом для быстроразвивающегося языка программирования как RНР, которые помогут организовать ваш код.

Каждый человек имеет различные предпочтения и потребности. Для одного разработчика использование RНР-фреймворков может помочь в ускорении процесса программирования, а для другого это может показаться пустой тратой времени. В большинстве случаев это зависит от уровня профессионализма, но, в общем, RНР фреймворки предназначены, чтобы сэкономить время и абстрагироваться от рутинных задач.

В основном, RНР-фреймворки применяются для разработки проектов сложнее чем 2-3-х страничный сайт с текстовыми страницами.

4. Yii

Yii – это высокоэффективный основанный на компонентной структуре PHP-фреймворк для разработки масштабных веб-приложений. Он позволяет максимально применить концепцию повторного использования кода и может существенно ускорить процесс веб-разработки. Название Yii (произносится как Yee или [ji:]) означает простой (easy), эффективный (efficient) и расширяемый (extensible). Также автор фреймворка, Qiang Xue говорит, что название фреймворка изначально значило Yes It Is.

В Yii вложено много возможностей которые выделяют этот фреймворк на фоне остальных и делают разработку с его помощью очень быстрой и приятной.

Вот небольшой список, который стоит выделить:

1. Отличная поддержка. Для русскоязычных пользователей на официальном форуме фреймворка создан собственный раздел где вы всегда можете получить ответы на свои вопросы. Плюс к этому мною было создано русскоязычное сообщество <http://groups.google.com/group/yii-framework/>, которое постоянно пополняется новыми материалами.

2. Помощь от разработчика фреймворка. Большой плюс поддержки Yii - это возможность в любой день пообщаться и задать вопросы разработчику Yii на форуме. Было замечено что он практически каждый день просматривает и отвечает на вопросы на форуме, помогает советами и полезными "фишками". Вы всегда можете написать ему личное сообщение и быть уверенны, что получите самый правильный ответ по вашему вопросу. К сожалению, разработчик фреймворка плохо знает русский язык, поэтому сообщения просим излагать ему на английском или китайском.

3. Полная поддержка ООП. Фреймворк полностью заточен под пятую версию php что позволяет поддерживать весь функционал при объектно ориентированном программировании. Разработчики не пошли на поддержку php4 в ущерб гибкости и удобства ооп. В связи с этим фреймворк не будет работать на php4, но зато отлично покажет себя на php5! Если вы знаете что такое ООП и умеете его использовать в php - тогда Yii лучший выбор для вас.

4. Генератор кода. Yii предоставляет отличный встроенный генератор исходного кода. Указав лишь основные параметры – Yii генерирует для вас общую структуру приложения которая будет содержать все необходимые модели/контроллеры/отображения для старта. В Yii версии 1.1 обещают браузерный генератор кода с еще большим функционалом!

5. Темы. Благодаря встроенной поддержке тем вы можете изменять дизайн вашего приложения в пару нажатий клавиш.

6. Защита. Все стандартные классы Yii заточены под высокий уровень безопасности что при умелом обращении позволяет полностью обезопасить свой сайт от Sql-Inj, XSS, CSRF и других атак.

5. ЗАКЛЮЧЕНИЕ

Рассмотрев всю полученную информацию об Yii мы получили следующий список особенностей данного фреймворка

Особенности:

1. Высокая производительность;
2. Интерфейсы DAO и ActiveRecord для работы с базами данных (PDO);
3. Поддержка интернационализации;
4. Кэширование страниц и отдельных фрагментов;
5. Перехват и обработка ошибок;
6. Ввод и валидация форм;
7. Аутентификация и авторизация;
8. Использование AJAX и интеграция с jQuery;
9. Генерация базового PHP-кода для CRUD-операций (скаффолдинг);
10. Поддержка тем оформления для их легкой смены;
11. Возможность подключения сторонних библиотек;
12. Миграции базы данных;
13. Автоматическое тестирование;
14. Поддержка REST.
15. Активное русскоязычное сообщество.

ЛИТЕРАТУРА

1. Yii [Электронный ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki/Yii>. – Дата обращения: 17.09.2018.
2. Официальный сайт PhpStorm. – Режим доступа: <https://jetbrains.ru/products/phpstorm/>. – Дата обращения: 17.09.2018.
3. Руководство по Php Manual. – Режим доступа: <http://php.net/manual/ru/index.php>. – Дата обращения: 17.09.2018.

УДК 004.021

СОЗДАНИЕ ВЕБ-ПРИЛОЖЕНИЯ С ИСПОЛЬЗОВАНИЕМ PHP-ФРЕЙМВОРКА Yii**В.С. РАДЧЕНКО***(Представлено: Д.В. ПЯТКИН)**Представлен путь, пройденный при разработке ознакомительного приложения при изучении Yii.***РАЗРАБОТКА ПРОЕКТА**

В качестве проекта, для ознакомления с возможностями фреймворков, была выбрана разработка сайта-каталога одежды и обуви, который будет обладать следующими особенностями:

- 1) товары разделены по категориям, с возможностью создания подкатегорий;
- 2) удобная административная панель;
- 3) многоязычность, поддержка русского, румынского и английского языков;
- 4) поддержка высоких нагрузок(кэширование).

Для разработки такого проекта был выбран MVC-фреймворк yii, который обладает относительно низким «порогом вхождения» и большим русскоязычным сообществом.

Работу над созданием любого сайта можно разделить на следующие этапы:

- 1) выбор и установка необходимых инструментов;
- 2) разработка дизайна проекта и верстки;
- 3) проектирование и создание базы данных;
- 4) создание основы приложения и конфигурация;
- 5) установка верстки и доработка каркаса;
- 6) настройка кэширования, многоязычности и «красивых» ссылок.

1. Приложения, использованные при разработке проекта

Для оптимизации разработки проекта, а также удобства были использованы следующие приложения:

- 1) пакет «denwer»;
- 2) графический интерфейс «HeidiSQL» для управления СУБД MySQL;
- 3) редактор программного кода PhpStorm.

Каждое из этих приложений обладает рядом уникальных, незаменимых свойств, облегчающих разработку и тестирование web-приложения.

1.1. Система управления СУБД MySQL «HeidiSQL»

HeidiSQL - бесплатный клиент с открытым исходным кодом, а так же графический интерфейс для управления и администрирования СУБД MySQL. Проект находится в активном развитии и поддерживает множество возможностей:

- 1) поддержка множества одновременно открытых подключений с помощью TCP/IP, именованных каналов или SSH-туннелирования, с возможностью сохранения авторизационных данных.
- 2) управление пользователями и их правами на сервере в рамках базы данных или глобально.
- 3) поддержка управления серверными переменными;
- 4) просмотр серверной статистики и управление запущенными процессами с возможностью проанализировать выполняемые SQL-запросы и прервать «плохие».
- 5) поддержка экспорта баз данных в SQL файл или на другой сервер, с возможностью последующего импорта;
- 6) просмотр и управление базами данных, таблицами, отображениями, триггерами и хранимыми процедурами.

2. Проектирование и создание базы данных

Во время проектирования база данных были выделены следующие таблицы:

1. таблица категорий «Category», которая хранит в себе название категорий на 3-х языках в древовидном виде Adjacency List: titleRu, titleEn, titleRo и parent;
2. таблица товаров «Price» с названиями и описаниями на 3-х языках, цену, размеры, наличие товара, а также дату добавления и поле для связи с категорией: titleRu, titleRo, titleEn, descrRu, descrRo, descrEn, idCategory, cost, time, present, sizes;
3. таблица для хранения пути к фотографиям «Photo» с полем для связи с товаром: itemId, path;
4. таблица для хранения текстовых страниц «Text» с названиями и описаниями на 3-х языках, а также красивым адресом для ссылок: titleRu, titleRo, titleEn, descrRu, descrRo, descrEn, name.

Помимо перечисленных выше полей каждая таблица обладает первичным ключом id для связи с другими таблицами и удобства управления записями.

3. Доработка правил валидации моделей и провайдеров данных

Так как модель отвечает за выборку и валидацию (проверку) данных, то в ней описываются условия выборки и проверки данных.

Правила валидации описываются в методе `rules` модели. Этот метод должен вернуть массив с описанием правил валидации, например:

```
public function rules()
{
    array(('titleRu','titleEn','titleRo','length','max'=>255,'min'=>4),('titleRu','titleEn','titleRo','required'),('parent',
'numerical', 'integerOnly'=>true,'allowEmpty'=>false),
);
}

```

Формат описания правила выглядит следующим образом:

`array(<имя св-ва/св-в модели (полей в таблице)>,<псевдоним валидатора>,<имя доп. параметра валидатора 1>=><значение параметра 1>, ...)`

Каждое имя валидатора является псевдонимом класса. Это позволяет создавать и использовать свои классы валидации. Основные валидаторы:

- 1) `default` (`CDefaultValueValidator`) - присваивает значение по умолчанию выбранным атрибутам;
- 2) `in` (`CRangeValidator`) - проверяет, содержится ли значение атрибута в указанном наборе значений;
- 3) `length` (`CStringValidator`) - проверяет, находится ли длина строкового значения атрибута в в указанном интервале;
- 4) `numerical` (`CNumberValidator`) - проверяет, является ли значение атрибута числом;
- 5) `required` (`CRequiredValidator`) - проверяет, не является ли значение атрибута пустым;
- 6) `match`: псевдоним класса `CRegularExpressionValidator`, проверяющего значение атрибута на соответствие регулярному выражению;
- 7) `unique`: псевдоним класса `CUniqueValidator`, который проверяет, является ли значение атрибута уникальными в пределах столбца таблицы базы данных;

Дополнительные параметры каждого из валидаторов подробно описаны в руководстве по yii.

После того как правила валидации были настроены, в модель были добавлены статические методы для выборки списка товаров. Каждой такой метод использует провайдер данных и выглядит примерно так:

```
public static function searchByCategory($category,$pageSize)
{
    // создание условия выборки
    $criteria=new CDbCriteria();
    $criteria->select='Id, title'.ucfirst(Yii::app()->language).' as title,cost';
    $criteria->addInCondition('idCategory',Category::getAllChildren($category));
    // создание провайдера данных new CActiveDataProvider(__CLASS__,array(
    // условие выборки
    'criteria'=>$criteria,
    // настройка постраничного вывода
    'pagination'=>array(
    'pageSize'=>$pageSize
    ),
    // поддержка сортировки
    'sort'=>array(
    'defaultOrder'=>'title'.ucfirst(Yii::app()->language),
    'attributes'=>array(
    'title'.ucfirst(Yii::app()->language),
    'time',
    'cost'
    )
    )
    ));
}

```

Как видно из приведенного выше кода провайдер данных является экземпляром класса `CActiveDataProvider` (а так же потомком `CDataProvider`) и очень гибок в настройке.

4. Настройка кэширования, многоязычности и «красивых» ссылок

После того как все основные действия по доработке каркаса выполнены, осталось добавить поддержку больших нагрузок (кэширования), настроить многоязычность и красивые ссылки.

Для активации кэширования в файле конфигурации было добавлено описание компонента приложения cache с классом CFileCache, отвечающим за хранение кэша в локальных файлах.

Для настройки кэширования в контроллерах использовался компонент COutputCache, описанный в методе filters контроллера:

```
array(
    // Компонент отключен для администраторских действий
    'COutputCache -create, update, upload',
    // Срок годности
    'duration'=>24*3600*365,
    // зависимости
    'dependency'=>array(
        'class'=>'CChainedCacheDependency',
        'dependencies'=>array(
            // по глобальному изменению pm_time
            new CGlobalStateCacheDependency('pm_time'),
            // по глобальному изменению cu_time CGlobalStateCacheDependency('cu_time'),
        ),
    ),
    // данные в кэше отличаются по категории, сортировке, страницам, языку
    'varyByParam'=>array('category','sort','page','lang'),
    // и по статусу авторизованности пользователя
    'varyByExpression'=>'Yii::app()->user->isGuest',
    // кэш используется только для GET-запросов
    'requestTypes'=>array('GET'),
),
```

где cu_time - последнее время обновления категорий, а ru_time - товаров и меняются по событию сохранения модели.

Так как, по-умолчанию, приложение уже поддерживает многоязычность с помощью класса CPhpMessageSource, который хранит переводы в php-файлах перевода, то в проекте использовался именно этот способ хранения.

Каждое сообщение перевода относится к какой-либо категории. Сообщения переводов хранятся по следующему пути: protected/messages/<код языка>/<имя категории>.php. Файлы переводов содержат в себе ассоциативный массив, где ключом является фраза на исходном (английский) языке, а значение - переведенное для данного кода (русский, например).

Для перевода сообщения, фреймворк yii предоставляет статический метод Yii::t(<имя категории>,<сообщение>), а для хранения текущего языка в ссылке в файл конфигурации было подключено стороннее расширение в качестве компонента приложения urlManager: LangUrlManager.

Для активации «красивых», человекопонятных ссылок компонент приложения urlManager был настроен следующим образом:

```
'urlManager'=>array(
    'class'=>'application.extensions.urlManager.CLangUrlManager',
    //формировать ссылку в формате /route/p1/v1/p2/v2/...
    'urlFormat'=>'path',
    'rules'=>array(
        // код языка всегда расположен впереди ссылки
        '<lang:(en|ru|ro)>'=>'/price',
        '<lang:(en|ru|ro)>/<_c>'=>'/<_c>',
        '<lang:(en|ru|ro)>/<_c>/<_a>'=>'/<_c>/<_a>',
        '<lang:(en|ru|ro)>/text/index/<page:.*>'=>'text/index',
    ),
    // не выводить index.php в ссылке
    'showScriptName'=>false,
),
```

а в корневой папке сайта (www) был добавлен файл .htaccess с активацией и настройкой mod_rewrite в web-сервере apache:

```
Options +FollowSymLinks*/*on%{REQUEST_FILENAME} !-f%{REQUEST_FILENAME} !-d. index.php
```

ЗАКЛЮЧЕНИЕ

При создании данного приложения были рассмотрены некоторые из большого количества возможностей фреймворка Yii и изучены на примере созданной программы. В данной статье мыф ещё раз убедились в обширных возможностях фреймворка Yii.

ЛИТЕРАТУРА

1. Русскоязычное сообщество Yii. – Режим доступа: <http://yiiframework.ru/>. – Дата обращения: 17.09.2018.
2. Официальный сайт Yii Framework [Электронный ресурс]. – Режим доступа: <http://yiiframework.com/>. – Дата обращения: 18.09.2018.
3. Официальный сайт PhpStorm. – Режим доступа: <https://jetbrains.ru/products/phpstorm/>. – Дата обращения: 17.09.2018.
4. Сайт PhpStorm. – Режим доступа: https://geekbrains.ru/posts/php_frameworks. – Дата обращения: 17.09.2018.

УДК 004.056.2

**УПРАВЛЕНИЕ ТРАНЗАКЦИЯМИ НАД ФАЙЛОВЫМИ СИСТЕМАМИ
В JAVA-ПРИЛОЖЕНИЯХ****В.А. МАКАРЫЧЕВА, М.Ю. МАКАРЫЧЕВ**
(Представлено: канд. техн. наук, доц. А.Ф. ОСЬКИН)

Рассматривается пример использования системы XADisk, которая позволяет работать с файловой системой, обеспечивая поддержку транзакций.

Введение. Большинство программ хранят важные данные в файловых системах, базах данных и других хранилищах. Практически любая база данных имеет механизм транзакции. При взаимодействии с файловой системой, механизм транзакции зачастую не применяется, потому как могут возникнуть проблемы, например, с целостностью данных. Во избежание этой и других проблем, связанных с хранением данных на файловых системах, предлагается использовать XADisk.

Основная часть. XADisk – это комплексная транзакционная система с открытым исходным кодом, написанная на языке программирования Java, предназначенная для возможности обеспечить транзакционный доступ к существующим файловым системам для всех видов Java-приложений, работающих на серверах Web-приложений, приложений JavaEE и т.д. [1].

При помощи нескольких простых API вы можете организовать XADisk поверх любой виртуальной машины Java (JVM) без необходимости установки на операционную систему (ОС). После запуска приложения будут вызывать API XADisk для выполнения различных операций ввода-вывода с файлами в рамках транзакций.

Сама система XADisk представляет собой промежуточную ступень между Java-приложениями и файловыми системами. Продукт XADisk не является реализацией файловой системы; он может работать с самыми разнообразными файловыми системами (например, NTFS, FAT, ext3), обеспечивая возможность транзакционного доступа приложений к этим файловым системам. Приложения вызывают API XADisk для выполнения различных операций ввода-вывода и могут применять команды commit (фиксация) или rollback (откат) для всех этих операций в рамках единой транзакции [2].

Для примера применения XADisk можно рассмотреть некое Java-приложение. Этому приложению нужно создать новый файл с именем file1, записать определенные данные в этот файл из другого файла с именем file2, а затем удалить файл file2. Эти три операции могут быть выполнены в рамках одной транзакции с использованием XADisk, при этом соблюдаются такие свойства, как:

- 1) целостность;
- 2) согласованность;
- 3) изолированность;
- 4) долговечность.

XADisk будет предоставлять эти гарантии даже в том случае, если во время процесса произойдет аварийное завершение работы приложения.

XADisk обладает следующими операциями:

1. с файлами:
 - 1.1. create (создание);
 - 1.2. delete (удаление);
 - 1.3. move (перемещение);
 - 1.4. copy (копирование);
 - 1.5. truncate (усечение);
 - 1.6. read (чтение);
 - 1.7. write (запись);
 - 1.8. exists (проверка существования);
 - 1.9. getLength (возврат длины);
2. с каталогами:
 - 2.1. create (создание);
 - 2.2. delete (удаление);
 - 2.3. move (перемещение);
 - 2.4. list children (список потомков);
 - 2.5. exists (проверка существования).

При эксплуатации XADisk не требуется, чтобы приложения и файловая система находились на одной JVM. Поэтому приложение может удаленно вызвать API XADisk, как и сам XADisk, расположенный на той же JVM.

Перед тем, как вызвать операции ввода/вывода через XADisk, необходимо загрузить экземпляр XADisk на JVM. На одной виртуальной машине может располагаться не один экземпляр XADisk, но при этом, каждый экземпляр XADisk должен иметь собственное изолированное состояние.

После того, как экземпляр XADisk загружен и готов к работе, можно выполнять транзакционных операций с файлами и каталогами. Так как данные операции будут осуществляться внутри транзакции, то необходимо создать сеанс для транзакции. Только после установки сеанса можно переходить к выполнению операций ввода/вывода.

После завершения всех операций можно выполнить фиксацию (или откат) транзакции с помощью простого вызова сеансового объекта. Для новой транзакции необходимо создать новый сеансовый объект. Работу экземпляра XADisk можно завершить из любого места в JVM.

Примера использования XADisk представлен в листинге 1.

Листинг 1 – код работы XADisk

```

1: String XADiskSystemDirectory = "/home/systems/XADiskSystem1";
2: StandaloneFileSystemConfiguration configuration =
3: new StandaloneFileSystemConfiguration(XADiskSystemDirectory, "instance-1");
4: XAFileSystem xaf = XAFileSystemProxy.bootNativeXAFileSystem(configuration);
5: xaf.waitForBootup(10000L);
6: XAFileSystem xaf = XAFileSystemProxy.getNativeXAFileSystemReference("instance-1");
7: XAFileSystem xaf = XAFileSystemProxy.getRemoteXAFileSystemReference("10.30.9.200", 5151);
8: Session session = xaf.createSessionForLocalTransaction();
9: File f = new File("/testAPIs/test.txt");
10: if(session.fileExists(f)) {
11:     XAFileInputStream xis = session.createXAFileInputStream(f);
12:     for (int i = 0; i < 100; i++) {
13:         byte a = (byte) xis.read();
14:         if( a== -1) {
15:             break;
16:         }
17:         System.out.print(a);
18:     }
19:     xis.close();
20:     session.moveFile(f, new File("/testAPIs/test.txt____" + System.currentTimeMillis()));
21: }
22: else {
23:     session.createFile(f, false);
24:     XAFileOutputStream xafos = session.createXAFileOutputStream(f, false);
25:     byte[] buffer = new byte[100];
26:     for (int i = 0; i < 100; i++) {
27:         buffer[i] = i*i;
28:     }
29:     xafos.write(buffer);
30:     xafos.close();
31: }
32: //фиксация транзакции
33: session.commit();
34: //или откат транзакции
35: session.rollback();

```

Заключение. Реализация доступа к файловым системам с использованием механизма транзакций может предложить множество преимуществ для приложений, которые хранят свои данные в файловых системах, а именно целостность: согласованность; изолированность; долговечность.

ЛИТЕРАТУРА

1. stackoverflow [Электронный ресурс]. – XADisk. – Режим доступа: <https://stackoverflow.com/tags/xadisk/info>. – Дата доступа: 20.09.18.
2. IBM [Электронный ресурс] // Транзакционный доступ к файловым системам с использованием XADisk. – Режим доступа: <https://www.ibm.com/developerworks/ru/library/os-xadisk/>. – Дата доступа: 20.09.18.

УДК 004.632

**СИСТЕМА БЕЗОПАСНОГО ВЫПОЛНЕНИЯ
ДИНАМИЧЕСКИ ГЕНЕРИРУЕМОГО JAVA-КОДА****В.А. МАКАРЫЧЕВА, М.Ю. МАКАРЫЧЕВ**
(Представлено: канд. техн. наук, доц. А.Ф. ОСЬКИН)

Представлен и подробно рассмотрен способ компиляции и безопасного исполнения исходного Java-кода во время работы программы.

Введение. В процессе разработки программного обеспечения иногда возникают проблемы, связанные с неопределённостью структуры программы на момент её компиляции. Другими словами, программа в некоторых местах имеет доступ к исходному коду, который требуется выполнить, только во время её работы. Технология JSP (Java Server Pages) является примером динамической генерации, компиляции и выполнения Java-кода. Транслятор JSP преобразует .jsp файлы в Java-сервлеты (файлы исходного кода), которые компилируются в момент выполнения и загружаются контейнером сервлетов.

Основная часть. Для решения задач этого класса Java предоставляет пакет `javax.tools`, который имеет следующие преимущества:

- Входит в состав JDK, представляет собой стандартный API, разработанный в рамках Java Community Process (JSR 199).
- Используется исходный код Java, а не байт-код. Это позволяет избежать запуска кода, который имеет ошибки компиляции, и предотвращает от некоторых уязвимостей JVM.
- Используется один механизм для генерации и загрузки кода, не ограничивая программиста использованием исходного кода на базе файлов.
- Используется проверенная версия компилятора.
- Поддерживается переносимость между реализациями JDK версии 1.6 и выше от любых производителей.
- В отличие от систем, основанных на интерпретации, загруженные классы могут пользоваться всеми оптимизациями, выполняемыми JRE во время выполнения [1].

Пакет `javax.tools` предоставляет настолько общие классы и интерфейсы, что позволяет не только загружать исходный код из файлов, но и получать его из различных объектов-источников.

Компиляция исходного кода требует следующих компонентов:

– Объект `classpath`, через который компилятор находит классы библиотек. Он состоит из списка каталогов файловой системы и архивных файлов (JAR или ZIP), которые содержат скомпилированные .class-файлы.

– `javac options` – параметры компилятора `javac`.

– `Source files` – один или несколько исходных .java-файлов. Объект `JavaFileManager` предоставляет абстрактную файловую систему, которая привязывает имена исходных и получившихся в результате файлов к экземплярам объектов `JavaFileObject`. В данном случае файл означает связь между уникальным именем и последовательностью байтов. Клиенту не обязательно использовать настоящую файловую систему. Например, `JavaFileManager` может управлять привязками между именами классов и объектами `CharSequence`, которые содержат исходный Java-код для компиляции.

– `Output directories` – каталоги, в которые компилятор записывает .class-файлы, содержащие байт-код.

– `compiler` (компилятор). Класс `JavaCompiler` создаёт объекты `JavaCompiler.CompilationTask`, которые компилируют исходный код объектов `JavaFileObject SOURCE` в объекте `JavaFileManager`, создавая новые выходные `JavaFileObject CLASS` файлы и объекты типа `Diagnostic` (предупреждения и ошибки). Статический метод `ToolProvider.getSystemJavaCompiler()` возвращает экземпляр компилятора.

– `Compiler warnings and errors` – сообщения от компилятора с предупреждениями и ошибками, которые реализуются классами `Diagnostic` и `DiagnosticListener`.

Далее подробнее рассматривается компиляция исходного кода, который содержится в объектах `CharSequence` (ими могут быть `String`, `StringBuffer` и `StringBuilder`).

Чтобы скомпилировать код из объекта (или объектов) `CharSequence`, необходимо сконструировать свой класс, к примеру, `CharSequenceJavaCompiler`, который будет иметь следующий API:

– Конструктор с параметрами `ClassLoader` и `Iterable<String>`. Первый параметр представляет собой загрузчик классов, который необходимо передать компилятору, чтобы тот мог находить зависимые классы. Второй параметр представляет собой список опций компилятора.

– Метод компиляции `compile` с параметрами `String` и `CharSequence`. Первый параметр – имя класса, второй – исходный код этого класса. Если требуется скомпилировать сразу несколько классов в разных объектах `CharSequence`, этот метод должен быть перегружен. В этом случае он может принимать параметр `Map<String, CharSequence>`, ключи которого – имена классов, а значения – исходный код. Метод должен вернуть объект типа `Map<String, Class<T>>`, ключи которого – имена классов, а значения – объекты типа `Class`, которые были загружены. Также этот метод должен заполнить список объектов типа `Diagnostic`.

– Метод `getClassLoader`, который возвращает загрузчик классов, созданный компилятором во время генерации `.class`-файлов, чтобы из него можно было загружать другие классы или ресурсы.

– Метод `loadClass`, который принимает `String` – имя класса и возвращает объект `Class`. Метод `compile`, используя метод `loadClass`, может загрузить несколько классов, включая вложенные, с одного `CharSequence` объекта.

Для работы всех методов класса `CharSequenceJavaCompiler` необходимо реализовать интерфейсы `JavaFileObject` и `JavaFileManager`. Реализация первого интерфейса `JavaFileObjectImpl` будет хранить объекты `CharSequence` с исходным кодом и выходные `CLASS`-объекты, которые создаст компилятор. Ключевой метод этого класса должен предоставлять текст исходного кода. Реализация второго интерфейса `JavaFileManagerImpl` связывает имена с экземплярами `JavaFileObjectImpl` для управления последовательностями исходного кода и байт-кода, созданного компилятором. На самом деле `JavaFileManagerImpl` может не реализовывать интерфейс напрямую, вместо этого он может расширить класс, то есть стать наследником `ForwardingJavaFileManager<JavaFileManager>`.

Теперь с этими вспомогательными классами можно определить класс `CharSequenceJavaCompiler`. Он создается с объектом `ClassLoader` (загрузчиком классов времени исполнения) и параметрами компилятора. В нем используется метод `ToolProvider.getSystemJavaCompiler()` для получения экземпляра `JavaCompiler`, затем создается `JavaFileManagerImpl`, который перенаправляется в стандартный менеджер файлов компилятора.

Метод `compile()` проходит по поданной на вход таблице ключ-значение, создавая объекты `JavaFileObjectImpl` из каждой пары имя/объект `CharSequence` и добавляя их в `JavaFileManager`, чтобы `JavaCompiler` нашел их при вызове метода `getFileForInput()` менеджера файлов. Метод `compile()` затем создает объект `JavaCompiler.Task` и запускает его. Сбои приводят к выдаче исключительной ситуации `CharSequenceJavaCompilerException`. Далее для каждого элемента исходного кода, переданного в метод `compile()`, загружается полученный объект `Class` и помещается в выходную коллекцию типа `Map`.

Загрузчик классов, связанный с `CharSequenceJavaCompiler` – это объект типа `ClassLoaderImpl`, который ищет байт-код для класса в объекте типа `JavaFileManagerImpl`, возвращая `.class` файлы, созданные компилятором [2].

Заключение. Использование `java.tools` API полностью является безопасным. Например, класс `JavaFileManager` может отказать в записи любого непредвиденного `.class`-файла, бросая `SecurityException`. Также `JavaFileManager` может допускать только сгенерированные имена классов и пакетов, которые пользователь не сможет предугадать или подменить. Кроме этого рекомендуется использовать специальные объекты `SecurityManager` или `ClassLoader`, предотвращающие небезопасное поведение, загрузку анонимных классов или других классов, которые не контролируются непосредственно разработчиком.

ЛИТЕРАТУРА

1. Oracle [Электронный ресурс]. – Package `java.tools`. – Режим доступа: <https://www.ibm.com/developerworks/ru/library/j-jcomp/index.html>. – Дата доступа: 20.09.18.
2. IBM [Электронный ресурс] // Создание динамических приложений с помощью `java.tools`. – Режим доступа: <https://www.ibm.com/developerworks/ru/library/j-jcomp/index.html>. – Дата доступа: 20.09.18.

УДК 004.223

**ПРОЕКТИРОВАНИЕ МОБИЛЬНОГО ИГРОВОГО ПРИЛОЖЕНИЯ «CUBE»
ПОД УПРАВЛЕНИЕМ ОПЕРАЦИОННОЙ СИСТЕМЫ ANDROID****И.В. ЛАРИОНОВ***(Представлено: Ю.Н. КРАВЧЕНКО)*

Рассматриваются способы хранения данных для мобильного игрового приложения под управлением операционной системы Android, а также способы их защиты.

Введение. Unity3D – кроссплатформенный игровой движок от компании Unity Technologies. Unity позволяет создавать приложения, работающие под более чем 20 различными операционными системами. Движок поддерживает множество популярных форматов моделей, звуков, материалов и текстур.

Проект в Unity делится на сцены (уровни) — отдельные файлы, содержащие свои игровые миры со своим набором объектов, сценариев, и настроек. Сцены могут содержать в себе как, собственно, объекты (модели), так и пустые игровые объекты — объекты, которые не имеют модели. Объекты, в свою очередь содержат наборы компонентов, с которыми и взаимодействуют скрипты.

Все эти данные необходимо защищать от копирования, нелегального использования, пиратского распространения, профессионального анализа и взлома.

Основная часть. При компиляции проекта Unity создается исполняемый файл игры, а в отдельной папке – данные игры (включая все игровые уровни и динамически подключаемые библиотеки).

Способы хранения данных. Информационные объекты, взаимосвязи между ними, а также способы влияния их на систему должны быть представлены в конечном программном продукте с помощью модели данных. Отсюда следует, что модель данных - есть совокупность структур данных и операций по их обработке.

Разрабатываемое приложение будет включить большое количество заранее подготовленной для пользователя информации, и необходимо подобрать удобный и эффективный способ ее хранения и обработки.

Отметим тезисно основные категории данных, которые необходимо хранить для дальнейшей интерпретации игрой:

- информация об игровых уровнях;
- информация о ценах во внутри игровом магазине;
- заранее заданные настройки.

Для хранения приведенных выше данных подходят 2 варианта: хранение в формате XML и бинарная сериализация.

Идеальным вариантом будет не выбирать среди предложенных технологий, а использовать оба варианта, взяв лучшее из каждой и применив в идеально подходящей для текущей задачи ситуации.

Большим плюсом бинарной сериализации является небольшой размер выходных файлов. Данные очень компактно складываются в файлы и сжимаются самыми эффективными алгоритмами, но существует один недостаток: невозможность чтения данных без специального программного обеспечения.

Бинарная сериализация отлично подходит для хранения самых разнообразных настроек, цен и прочих конфигурационных параметров.

Способ хранения уровней требует прозрачности и простого доступа к данным. Данные должны с легкостью читаться при помощи выбранного языка программирования и при этом быть очевидными при простом просмотре документа. Именно поэтому для хранения игровых уровней было выбрано использование документов в формате XML.

Защита данных. Для Windows-сборок Unity компилирует и сохраняет исходный код всех игровых скриптов в директорию Managed. Находится код в библиотеках: Assembly-CSharp.dll, Assembly-CSharp-firstpass.dll и Assembly-UnityScript.dll. Для декомпиляции и просмотра managed-кода .NET библиотек существуют довольно удобные и при этом бесплатные утилиты: IISpy и dotPeek.

Данный подход особенно эффективен для наших целей: Unity очень скупо оптимизирует исходный код игровых скриптов, практически не изменяя его структуру, а также не скрывает названия переменных. Это позволяет с легкостью читать и понимать декомпилированный материал.

В таких случаях разработчикам приходится самим беспокоиться о безопасности своего кода. Для таких целей обычно используют обфускаторы.

Обфускация - это процесс, в результате которого код программы приобретает вид, трудный для анализа. Обфускация, осуществляется с целью защиты программного кода и алгоритмов, которые он реализует, от чужих глаз. Но в большинстве случаев обфускация имеет массу побочных эффектов. В особо

неприятных обстоятельствах обфускатор может сделать программу совершенно непригодной к выполнению, в менее тяжёлых случаях в программе могут появиться новые ошибки. Поэтому обфускацию необходимо применять с максимальной осторожностью.

В AssetStore существует множество готовых решений, но большинство из них платные. Бесплатные версии, как правило, ограничены или имеют невысокую эффективность.

Большинство ресурсов Unity-проекта упаковываются в файлы проприетарного формата с расширениями .assets и .resources. Несмотря на закрытость форматов, существуют инструменты для распаковки таких файлов. Так, например, Unity Assets Explorer способен извлечь большинство текстур и шейдеров из игры. Полученные в результате текстуры будут иметь формат DDS, который можно открыть с помощью Windows Texture Viewer. Шейдеры извлекаются в уже скомпилированном виде и решений для их декомпиляции не существует. Тем не менее, это обстоятельство не мешает импортировать и использовать полученные шейдеры в другом Unity-проекте.

Трёхмерные модели в Unity-сборке расположены в различных ресурсах, а некоторые из них и вовсе могут генерироваться во время игры. Получить такие данные можно напрямую из памяти графического ускорителя. Когда игра запущена, вся информация о текстурах и моделях, видимых на экране, находится в памяти видеокарты. С помощью утилиты 3D Ripper DX можно извлечь всю эту информацию и сохранить в формате, понятном 3D-редакторам.

PlayerPrefs — это класс из стандартной библиотеки Unity, который позволяет сохранять данные в долговременную память устройства. Представляет собой пару ключ - значение. Он часто используется разработчиками для хранения различных настроек, достижений, прогресса игрока и другой информации о состоянии игры. На ОС Windows эти данные сохраняются в системном реестре. В других операционных системах данные сохраняются на устройстве в локальной папке приложения в специальном файле. В большинстве случаев к ним можно легко получить доступ и модифицировать с помощью текстового редактора. Например, в Windows достаточно использовать встроенную утилиту RegEdit чтобы модифицировать любые значения PlayerPrefs, изменяя тем самым конфигурацию и статус игры.

Простейшим способом противодействия является кодирование сохраняемых данных, например, base64. Данный способ не слишком эффективен, но может дать начальную защиту от просмотра. Проверить, изменялись ли данные без ведома помогут хеш-функции: сравнив контрольные суммы хранимых данных, мы сможем убедиться, что никто и ничто, кроме нашего кода эти данные не изменяло.

Для более надёжной защиты необходимо использовать шифрование или различные комбинации из перечисленных методов.

Также можно реализовать свой формат сохранений. Благодаря Mono Unity поддерживает работу с файловой системой. Таким образом можно сериализовать все необходимые данные, применить надёжное шифрование и сохранить в надёжном месте.

Заключение. В данной статье были рассмотрены способы хранения данных для игрового приложения под управлением операционной системы Android, а также способы их защиты.

К сожалению, существует не так уж много способов защитить игру от взлома. Будучи установленной на пользовательское устройство, она фактически раскрывает все текстуры, модели и исходный код. Если кто-то захочет декомпилировать приложение и украсть ресурсы — это лишь вопрос времени.

Невзирая на это, существуют действенные методы, которые позволят серьёзно усложнить жизнь злоумышленникам.

ЛИТЕРАТУРА

1. Guide to Storage Encryption Technologies for End User Devices / Karen Scarfone, Murugiah Souppaya, Matt Sexton. – November 2007. – 40 p.
2. Базы данных и модели данных [Электронный ресурс]. Режим доступа: <http://edu.tltsu.ru>. Дата доступа: 22.09.2018.
3. Unity - Scripting API: PlayerPrefs [Электронный ресурс]. Режим доступа: <https://docs.unity3d.com/ScriptReference/PlayerPrefs.html>. Дата доступа: 22.09.2018.
4. A practical tutorial to hack (and protect) Unity games [Электронный ресурс]. Режим доступа: <https://www.alanzucconi.com/2015/09/02/a-practical-tutorial-to-hack-and-protect-unity-games/>. Дата доступа: 22.09.2018.

УДК 004.021

**РАЗРАБОТКА МОБИЛЬНОГО ИГРОВОГО ПРИЛОЖЕНИЯ «CUBE»
ПОД УПРАВЛЕНИЕМ ОПЕРАЦИОННОЙ СИСТЕМЫ ANDROID****И.В. ЛАРИОНОВ***(Представлено: Ю.Н. КРАВЧЕНКО)*

Рассматриваются принципы построения графического интерфейса пользователя для мобильного игрового приложения под управлением операционной системы Android.

Введение. Unity3D — кроссплатформенный игровой движок от компании Unity Technologies. Unity позволяет создавать приложения, работающие под более чем 20 различными операционными системами. Движок поддерживает множество популярных форматов моделей, звуков, материалов и текстур. Начиная с версии 4.6 была добавлена новая система интерфейса, позволяющая использовать встроенные средства движка для построения пользовательского интерфейса.

Основная часть. В основе понимания новой концепции лежат три компонента: Canvas, Rect Transform и Event Trigger.

Canvas — является контейнером для всех элементов UI и определяет режим рендера. Таких контейнеров на сцене может быть более одного.

Rect Transform — этот компонент позволяет задать положение и размер игрового объекта, используя удобные визуальные средства управления. Он вводит понятия ширины и высоты, точки опоры, а также поддерживает якоря.

Задание якорей — простой и мощный способ контролировать положение и размер ваших элементов пользовательского интерфейса относительно их родителей. Это особенно удобно, когда вы изменяете размер родительских элементов. При изменении размера родителя наш элемент будет стараться сохранить равномерное расстояние к точкам якоря, заставляя его перемещаться или изменять размер вслед за родителем.

Точка опоры является точкой, вокруг которой производятся все преобразования. Другими словами, определяя положение элемента, вы определяете его точку опоры. Если вы станете вращать элемент пользовательского интерфейса, то он будет вращаться вокруг этой точки. Точка опоры обозначается нормализованными координатами. То есть, ее значение изменяется от 0 до 1 для ширины и высоты, где (0,0) — нижний левый угол, а (1,1) — правый верхний угол.

Event System — доработанная система событий, которая включает компоненты визуального управления вызовом событий. Отвечает за обработку ввода, рейкастинг и события в сцене Unity.

При помощи перечисленных инструментов стало возможно разрабатывать сложные интерфейсы на основе иерархии простых компонентов без особых сложностей. При этом интерфейс будет адаптивным и автоматически подстраиваться под различные разрешения экрана.

Интерфейс приложения сделан используя современную концепцию построения внешнего вида, так называемый, плоский дизайн. Основные принципы плоского стиля — простота, лаконичность и минимализм. Основные отличительные черты — это те вещи, на которых можно акцентировать внимание — яркие необычные цвета, нестандартная типографика, графические примитивы. Так достигается простота и универсальность интерфейса.

Основной принцип flat-дизайна — это создание двумерного изображения, которое уже само по себе является плоским. При этом нужно избавиться от всех элементов, которые могут добавлять хоть какую-то глубину изображению: градиенты, плавные переходы, тени, скосы, объем, текстуры и так далее. Все элементы рисунка имеют жесткие края и также не выделяются, и не имеют глубины относительно других элементов изображения.

Во flat-дизайне вы не встретите элементы, которые пытаются выглядеть реалистично (скевоморфизм), 3D-анимацию, реалистичные пиктограммы и т.д. Однако при этом сам дизайн пытается оставаться аналогичным традиционному, но теперь на передний план выходят надписи, кнопки и навигация.

На первый взгляд приложение с применением Flat дизайна может показаться чересчур простым, хотя на самом деле этому стилю присуща четкая иерархия элементов, удобное размещение всех инструментов взаимодействия пользователя, а, следовательно, имеет высокий показатель юзабилити.

Плоский дизайн по своей натуре прост, и он хорошо сочетается с минималистичным подходом. В общем дизайне нужно избегать слишком больших наворотов. Конечно, простых цветов и текста может быть недостаточно. Поэтому, чтобы добавить визуальных эффектов, нужно выбирать простые фотографии.

Для создания плавных анимаций в Unity можно использовать сторонние плагины, например, DOTween. DOTween – это быстрый, эффективный, объектно-ориентированный движок анимации, оптимизированный для C#.

При помощи DOTween можно перемещать, изменять размер, поворачивать каждый объект на сцене независимо от других, при этом имея возможность настроить каждый аспект анимации по отдельности. Обратные вызовы, петли, различные функции интерполяции значений и многие другие варианты анимации.

Кроме того, он чрезвычайно точный. Это означает, что 1000 циклов по 1 секунде будут играть ровно столько же, сколько один цикл на 1000 секунд. Логичный и простой в использовании API, созданный для повышения эффективности, интуитивности и простоты использования.

DOTween может анимировать каждое числовое значение, а также некоторые нечисловые. Например, он может анимировать строки.

Плагин предоставляет полный контроль: Play, Pause, Rewind, Restart, Complete, Goto и множество других полезных методов для управления вашими анимациями. Анимации можно объединять в последовательности для создания сложных анимаций (которые не обязательно должны быть в последовательности, а также они могут перекрывать друг друга). Возможно изменение начальных / конечных значений и продолжительности воспроизведения движения в любой момент, даже во время воспроизведения.

DOTween построен с учетом расширяемой архитектуры, которая позволяет создавать собственные плагины для анимации в виде отдельных файлов. Поддерживаются вспомогательные виртуальные методы, чтобы делать такие вещи, как например вызов функции после определенной задержки.

Таким образом можно без труда настроить даже сложные последовательности анимаций без ущерба производительности.

Заключение. В данной статье были рассмотрены основные принципы и способы построения графического интерфейса в Unity для мобильного игрового приложения «Cube» под управлением операционной системы Android. Рассмотрены основные компоненты Unity UI system и способы построения анимации.

ЛИТЕРАТУРА

1. Unity - Scripting API: UI [Электронный ресурс]. – Режим доступа: <https://docs.unity3d.com/Manual/UISystem.html>. – Дата доступа: 24.09.2018.
2. Краткий обзор нового Unity UI с примерами организации интерфейса [Электронный ресурс]. – Режим доступа: <https://habr.com/post/246605/>. – Дата доступа: 24.09.2018.
3. Основные принципы плоского дизайна [Электронный ресурс]. – Режим доступа: <http://x-site.by/info/flat-design>. – Дата доступа: 24.09.2018.
4. DOTween [Электронный ресурс]. – Режим доступа: <http://dotween.demigiant.com/index.php>. – Дата доступа: 26.09.2018.

УДК 004.223.2

**РАЗРАБОТКА ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА
ДЛЯ АВТОМАТИЗИРОВАННОГО РАБОЧЕГО МЕСТА ПО УЧЕТУ АВТОТРАНСПОРТА****Д.Н. ПАЦАНКОВ****(Представлено: Ю.Н. КРАВЧЕНКО)**

Рассматриваются принципы построения графического интерфейса пользователя приложения.

Немаловажной задачей является разработка графического интерфейса пользователя. Согласно техническому заданию, программное обеспечение разрабатывается под операционную систему семейства Windows, следовательно, необходимо руководствоваться спецификациями, разработанными фирмой Microsoft для разработки программного обеспечения.

Под графическим интерфейсом пользователя (Graphical User Interface - GUI) подразумевается тип экранного представления, при котором пользователь может выбирать команды, запускать задачи и просматривать списки файлов, указывая на пиктограммы или пункты в списках меню, показанных на экране.

Пользовательский интерфейс – это совокупность информационной модели проблемной области, средств и способов взаимодействия пользователя с информационной моделью, а также компонентов, обеспечивающих формирование информационной модели в процессе работы программной системы

Основное достоинство хорошего интерфейса пользователя заключается в том, что пользователь всегда чувствует, что он управляет программным обеспечением, а не программное обеспечение управляет им.

Для создания у пользователя такого ощущения «внутренней свободы» интерфейс должен обладать целым рядом свойств.

- естественность интерфейса;
- согласованность интерфейса;
- дружелюбность интерфейса (Принцип «прощения пользователя»);
- принцип «обратной связи»;
- простота интерфейса;
- гибкость интерфейса;
- эстетическая привлекательность [1].

Любое приложение под операционную систему Windows – представляет собой окно, которое создается компонентом Form. Форма – это важнейший визуальный компонент. Формы представляют собой видимые окна Windows и являются основной частью практически любого приложения.

Приложение может иметь несколько форм, одна из которых считается главной и при запуске программы отображается первой. При закрытии главной формы приложения, прекращается работа всего приложения, при этом также закрываются все открытые окна приложения.

Принимая во внимание рекомендации, описанные выше, графический интерфейс на главной форме целесообразно представить кнопками быстрого вызова, которые будут отвечать за определенные функции программы:

- хранение информации о работе автотранспорта предприятия;
- хранение информации о водителях;
- систематизирование данные по путевым листам;
- систематизирование данные по автомобилям;
- систематизирование данные по водителям;
- формирования данных в .xml формат.

Что касается основных возможностей разрабатываемого интерфейса, можно отметить следующее. Выбрав меню «Файл/Открыть» и загрузив БД, можно работать с полно функционирующей программой. После открытие файла БД все пункты меню становятся активными.

Пункт меню «Справочники» позволяет просмотреть, добавить, отредактировать или удалить данные. Чтобы просмотреть необходимые данные, требуется выбрать один из пунктов меню.

При нажатии клавиши «Конвертировать», данные сохраняются в выбранный пользователем каталог и переводятся в XML формат.

При нажатии кнопки «Добавить» выполняется переход на новую форму, на которой будет производиться добавление данных. При добавлении новой записи учитываются проверка на корректный ввод, в случае неправильного ввода данных на экран выведется сообщение.

При выборе одного из пунктов меню «Путевые листы» появляется дочерняя форма, на которой в правом верхнем углу расположены кнопки поиска данных по заданному критерию, а в левом верхнем углу кнопки возможных операций над данными – добавление, редактирование, удаление и конвертирование.

Выводы. В данной статье рассмотрены способы построения графического интерфейса приложений под операционную систему Windows. Представлены примеры работы приложения. Основное достоинство хорошего интерфейса пользователя заключается в том, что пользователь всегда чувствует, что он управляет программным обеспечением, а не программное обеспечение управляет им.

ЛИТЕРАТУРА

1. Studfiles [Электронный ресурс]. – Режим доступа: <https://studfiles.net/preview/423781/page:2/>. – Дата доступа: 27.09.18.

УДК 004.42

**ПРОЕКТИРОВАНИЕ АВТОМАТИЗИРОВАННОГО РАБОЧЕГО МЕСТА
ПО УЧЕТУ РАБОТЫ АВТОТРАНСПОРТА****Д.Н. ПАЦАНКОВ***(Представлено: Ю.Н. КРАВЧЕНКО)*

Рассматриваются цели и принципы моделирования базы данных для приложения «Автоматизированное рабочее место по учету автотранспорта».

Трудно представить хорошее приложение АРМ без разработанной базы данных. Оно позволяет удобно хранить большие объемы информации. Приложение, которое может эти данные предоставить по запросу. Данная статья направлена на описание базы данных для разработанного приложения и методов ее проектирования.

«Автоматизированное рабочее место по учету работы автотранспорта» предназначено для автоматизации учета расхода топлива собственным транспортом, учета годового, ежемесячного пробега автомобилей, поиска необходимой информации по категориям автотранспорта, списку водителей, видам ГСМ.

Можно выделить следующие функции разрабатываемой системы:

- хранение информации;
- обработка информации;
- конвертирование данных;
- работа с формами.

На данном этапе разработки базы данных необходимо собрать всю необходимую информацию, которая будет храниться в будущей базе. Будем выделять сущности, их атрибуты и связи между ними. В последствии эту информацию мы будем преобразовывать, и использовать для непосредственной реализации базы данных в специализированной среде.

В ходе проведения анализа предметной области будут постепенно выявляться необходимые сущности.

Для облегчения процесса анализа и выявления сущностей выделим следующие блоки:

- водители;
- путевые листы;
- транспортное средство;
- списание.

Связь – это ассоциирование двух или более сущностей. Эта ассоциация всегда является бинарной и может существовать между двумя разными сущностями или между сущностью и ей же самой (рекурсивная связь). В любой связи выделяются два конца (в соответствии с существующей парой связываемых сущностей), на каждом из которых указывается имя конца связи, степень конца связи (сколько экземпляров данной сущности связывается), обязательность связи (т.е. любой ли экземпляр данной сущности должен участвовать в данной связи) [2].

Построение набора необходимых отношений базы данных

Для построения схемы реляционной базы данных необходимо определить совокупность отношений, составляющих данную базу данных. Эта совокупность отношений будет содержать всю информацию, которая должна храниться в базе данных.

Ранее нами была описана инфологическая концептуальная модель базы данных «Автоматизированное рабочее место по учету автотранспорта». Эта модель содержит всю информацию, хранящуюся в данной базе данных. На основе полученной концептуальной модели можно определить набор необходимых отношений базы данных. Под отношением понимается совокупность двумерных таблиц (имя атрибута, имя домена или тип данных) особого вида. На рисунке представлен набор отношений для базы данных «Автоматизированное рабочее место по учету автотранспорта».

Важное достоинство баз данных – возможность хранить сгруппированные данные в разных таблицах и устанавливать связи между этими таблицами посредством ключей и последующее объединение этих данных в единую базу. Это позволяет снизить избыточность данных, которые хранятся, упростить организацию запросов для их получения.

Первичный ключ служит ограничителем целостности в рамках одной таблицы для идентификации, т.к. поле первичного ключа не может повторяться или быть пустым. Важное условие – каждая таблица может содержать только один первичный ключ.

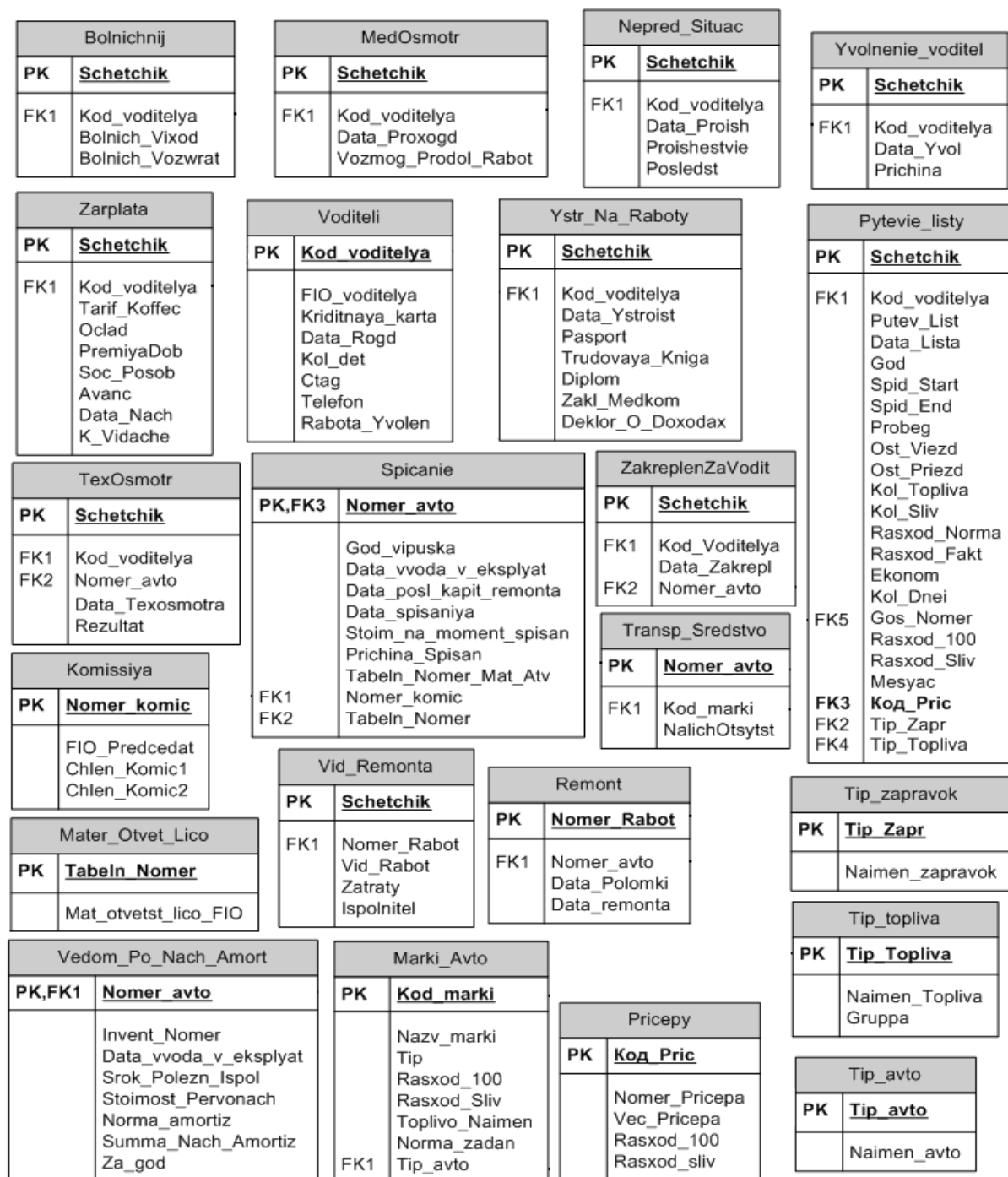


Рисунок. – Концептуальная схема базы данных

Для организации связей между двумя и более таблицами, используются вторичные ключи. Они служат ограничителями целостности связей нескольких таблиц, т.к. подчиненная таблица не может ссылаться на несуществующие записи главной таблицы (что позволяет строить целостные модели данных).

Существует три вида связей между таблицами:

- «один-ко-многим»;
- «один-к-одному»;
- «многие-ко-многим».

Связь «один-ко-многим» – самая распространенная для реляционных баз данных. Для обеспечения ссылочной целостности в дочерней таблице создается внешний ключ, через который организовывается связь с дочерней таблицей. Обязательное условие состоит в том, что внешний ключ по составу полей должен совпадать с первичным ключом родительской таблицы.

При создании диаграмм вариантов использования нужно выделить два вида сущностей – актеры и прецеденты. Актер – это множество логически связанных ролей, исполняемых при взаимодействии с прецедентами. Прецедент – это описание множества последовательных событий, включая варианты, выполняемых системой, которые приводят к наблюдаемому актером результату.

Для «Автоматизированное рабочее место по учету автотранспорта» можно выделить одного актера – это пользователь. Пользователь сможет выбирать различные пункты меню, просматривать информацию по определенному критерию, добавлять, редактировать, удалять, конвертировать данные.

Выводы. В данной статье были рассмотрены основные цели и принципы проектирования для приложения «Автоматизированное рабочее место по учету автотранспорта». Также были рассмотрены основные сущности спроектированной базы данных.

ЛИТЕРАТУРА

1. Help4Students [Электронный ресурс] / Методология моделирования данных ИС. – Режим доступа: <https://hlp4asp.wordpress.com/2011/10/13/методология-моделирования-данных-ис>. – Дата доступа: 27.09.2018.
2. CitForum [Электронный ресурс] / Семантическое моделирование данных, ER-диаграммы. – Режим доступа: http://citforum.ru/database/osbd/glava_24.shtml. – Дата доступа: 27.09.2018.

УДК 004.223.2

**ВОЗМОЖНЫЕ ВАРИАНТЫ ХРАНЕНИЯ ИНФОРМАЦИИ, АНАЛИЗ
И ВЫБОР ОПТИМАЛЬНОГО ХРАНИЛИЩА ДЛЯ ДАЛЬНЕЙШЕЙ РЕАЛИЗАЦИИ****Е.С. ДЕНИСОВА***(Представлено: канд. физ.-мат. наук, доц. Д.Ф. ПАСТУХОВ)*

Рассматриваются различные варианты хранения информации. В ходе анализа был выбран оптимальный вариант хранения информации. Проектирование системы для хранения и возможности передачи информации, также вопросы защиты этих данных. Проведены исследования по актуальности разработки данной системы.

В современном мире пользователи персонального компьютера, смартфона и другого устройства с доступом к Всемирной Сети имеют большое количество информации (фотографии, видеозаписи, музыка, различные документы и т.п.), которую необходимо где-то хранить. Для хранения информации существует большое количество различных ресурсов. Рассмотрим возможные варианты хранения информации:

1. Жесткие диски – на сегодняшний день используются в большинстве настольных ПК, а также нашли применение в качестве портативных хранилищ данных. Обычно, такой носитель исправно работает в течении 3-10 лет и срок его службы зависит от множества внешних факторов и самого качества изготовления;

2. Флешки и SSD накопители – такие устройства, в среднем, исправно работают около пяти лет. Многие флешки могут ломаться даже намного раньше, ведь они могут не перенести скачок напряжения или статический разряд, в момент подключения к ПК;

3. Оптические диски – это всем известные CD, DVD и Blu-Ray. Пожалуй, это одни, из самых долговременных способов сохранить информацию, в некоторых случаях такой диск будет надежно хранить все записанные данные более чем 100 лет, но оптические диски могут занимать большое количество физического пространства, что не очень удобно будет пользователю;

4. Облачные хранилища – это модель облачных вычислений, предусматривающая хранение данных в Интернете с помощью поставщика облачных вычислительных ресурсов, который предоставляет хранилище данных как сервис и обеспечивает управление им. Использование облачных носителей очень популярно в наше время. Очень удобный вариант предоставлять информацию любому человеку, будучи в любой точке планеты, имея при себе любое мультимедийное устройство и доступ к Интернету.

Исходя из вышперечисленных вариантов можно сделать вывод, что в век современных технологий и доступности Интернета, пользователи все чаще «доверяют» свои данные облачным сервисам, что достаточно удобно, потому что сервис не занимает у пользователя никакого физического пространства в отличии от других возможных вариантов хранения информации.

Актуальность разработки своего облачного хранилища:

Основу разрабатываемой системы, составляет возможность хранить, накапливать и обрабатывать данные. Также возможность делиться файлами со сторонними людьми. При разработке собственного облачного хранилища у нас есть возможность сделать шифрование «под себя», то есть выбрать алгоритмы, которые нам нужны и при необходимости видоизменить их. На данном этапе реализована двухключевая математическая модель криптосистемы на основе двух типов шифрований (AES+RSA) таким образом, что в любой момент можно изменить/добавить алгоритм шифрования. Что делает нашу систему гибкой для разработки и более защищенной от умышленного хищения информации.

Принцип защиты информации.

Основываясь на том, что данная система предназначена для хранения конфиденциальных данных пользователей, необходимо разработать систему аутентификации, и криптографической защиты, данных.

Для защищённой передачи данных по сети интернет предназначены следующие виды технологии:

1. Протокол HTTPS - стандарт передачи данных между различными машинами, который определяет, что должно выступать в качестве сигнала начала передачи, как обозначаются данные и т. д. При этом данные шифруются по протоколу SSL, что делает проблематичным не только перехват, но и получение конфиденциальной информации [5].

2. Протокол SRTP – это профиль расширения RTP (Real-Time Transport Protocol, транспортный протокол в реальном времени), который добавляет дополнительные функции безопасности, такие как аутентификация сообщений, конфиденциальность и защита от прослушивания. Данный протокол будет использован при передаче видео сообщений [6].

3. Шифр AES - симметричный алгоритм блочного шифрования. Данный вид шифрования будет использовать один и тот же симметричный ключ. Для разрабатываемой системы, это является оптимальным вариантом, так как подобные системы также защищены юридическими соглашениями, о неразглашении третьим лицам. Используемый метод шифрования, будет использован для шифрации конфиденциальных данных, на уровне базы данных.

4. Шифр RSA – алгоритм шифрования с открытым ключом, базирующийся на факторизации простых чисел.

Заключение. В ходе данного исследования и анализа были сделаны выводы о выборе оптимального хранилища информации для пользователя и обоснование актуальности создания своего облачного хранилища.

ЛИТЕРАТУРА

1. WebRTC API [Электронный ресурс]. – Режим доступа: https://developer.mozilla.org/ru/docs/Web/API/WebRTC_API. – Дата доступа: 23.09.2018.
2. Протокол HTTPS - что такое? [Электронный ресурс]. – Режим доступа: <http://fb.ru/article/221368/protokol-https---chto-takoe>. – Дата доступа: 23.09.2018.
3. SRTP — что такое безопасный протокол передачи данных в реальном времени? [Электронный ресурс]. – Режим доступа: <https://www.3cx.ru/webrtc/srtp/>. – Дата доступа: 23.09.2018.

УДК 004.223.2

**ПОДХОДЫ К РАЗРАБОТКЕ ИНТЕРФЕЙСА ДЛЯ ВЕБ-ПРИЛОЖЕНИЯ
«РАЗРАБОТКА И РЕАЛИЗАЦИЯ МАТЕМАТИЧЕСКОЙ
ДУВХКЛЮЧЕВОЙ КРИПТОСИСТЕМЫ ДЛЯ ОБЛАЧНОГО ХРАНИЛИЩА»****Е.С. ДЕНИСОВА***(Представлено: канд. физ.-мат. наук, доц. Д.Ф. ПАСТУХОВ)*

Представлен практический способ создания интерфейса для работы с облачным хранилищем. Проведён анализ технологий, наиболее подходящих, для разработки данной системы. Задача, реализация прототипа интерфейса на основе веб-технологий.

Введение. На сегодняшний день, любое веб-приложение должно быть основано на обще поддерживаемых стандартах разработки, а также обладать интуитивно понятным для пользователя интерфейсом. Разрабатываемое веб-приложение «Математическая модель двухключевой криптосистемы в облачном хранилище», должно позволять производить работу с данными пользователя, а также удовлетворять требованиям к предоставляемым данным, по видам хранимой информации и её представлению.

В данном приложении процесс создания интерфейса приложения служит для прямого взаимодействия приложения с базой данных, сочетающей в себе всю необходимую информацию для вывода непосредственно в приложение. Интерфейс создается путем добавления необходимых компонентов и последующей их настройки. В интерфейсе любого приложения есть компоненты, обеспечивающие взаимодействие программы с пользователем.

Выбор технологий разработки интерфейса. Данное веб-приложение планируется разрабатывать с помощью языка программирования Java и фреймворка веб-разработки Spring Framework. Отсюда следует, что технологии разработки веб-интерфейса должны быть совместимы с данными технологиями, при этом давать все возможности современных интерфейсов.

Разрабатываемое приложение, по своей структуре и основной потребности, должно давать возможности по работе и представлению большого количества данных, которые бы могли динамически меняться, при действиях пользователя. При анализе инструментов для создания веб-интерфейса, было выявлено несколько технологий хорошо подходящих под данные цели:

JSP (JavaServer Pages) — технология, позволяющая веб-разработчикам создавать содержимое, которое имеет как статические, так и динамические компоненты. Страница JSP содержит текст двух типов: статические исходные данные, которые могут быть оформлены в одном из текстовых форматов HTML, SVG, WML, или XML, и JSP-элементы, которые конструируют динамическое содержимое. Кроме этого могут использоваться библиотеки JSP-тегов, а также EL (Expression Language), для внедрения Java-кода в статичное содержимое JSP-страниц [1].

JSF (JavaServer Faces) — это фреймворк для веб-приложений, написанный на Java. Он служит для того, чтобы облегчать разработку пользовательских интерфейсов для Java EE-приложений. В отличие от прочих MVC-фреймворков, которые управляются запросами, подход JSF основывается на использовании компонентов. Состояние компонентов пользовательского интерфейса сохраняется, когда пользователь запрашивает новую страницу и затем восстанавливается, если запрос повторяется [2].

Обе технологии поддерживают возможности работы с динамическими данными, поэтому был отдан выбор технологии JavaServer Pages, так как её поддержка ведётся наиболее интенсивным образом, а также данная технология имеет большую совместимость с Spring Framework благодаря работе через Servlet.

Также было решено добавить возможность адаптивности интерфейса для разрабатываемого приложения. Для этого был выбран Bootstrap.

Bootstrap — свободный набор инструментов для создания сайтов и веб-приложений. Включает в себя HTML- и CSS-шаблоны оформления для типографии, веб-форм, кнопок, меток, блоков навигации и прочих компонентов веб-интерфейса, включая JavaScript-расширения.

Bootstrap использует современные наработки в области CSS и HTML, поэтому необходимо быть внимательным при поддержке старых браузеров [3].

Проектирование интерфейса. При разработке интерфейса, для удобства использования была применена вкладочная структура – информация логически разделена по вкладкам, что упрощает ее поиск, не загромождает главную страницу и не приводит к необходимости создавать дополнительные формы, а также уменьшает время на загрузку других страниц (рис. 1).

Информация о документах хранится в таблицах, над которыми, имеет возможность производить действия, после нажатия на ссылку непосредственно рядом с файлом. Данная структура, является наиболее простой и понятной для пользователей, с различным уровнем владения персональным компьютером. Использование стандартов элементов управления, позволяет максимально снизить время обучения работе с приложением, а стилизация их с помощью шаблонов Bootstrap, позволяет улучшить внешний вид

и привлекательность данного приложения. Также на странице есть поиск пользователей, с помощью которого просто найти пользователя для того, чтобы поделиться файлами с ним внутри системы (рис. 2).

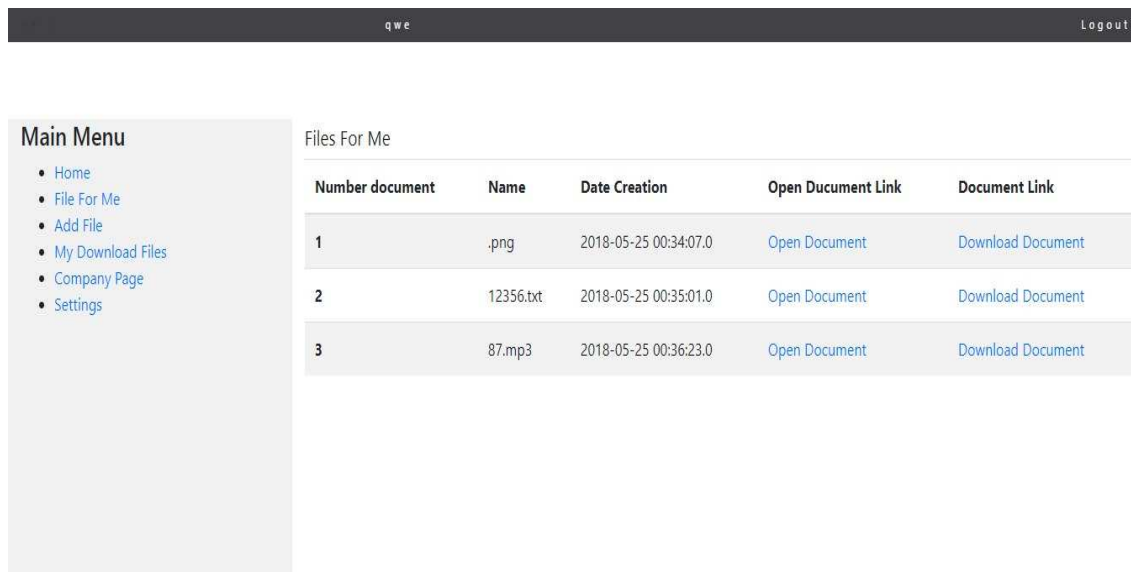


Рисунок 1. – Прототип главной страницы для пользователя

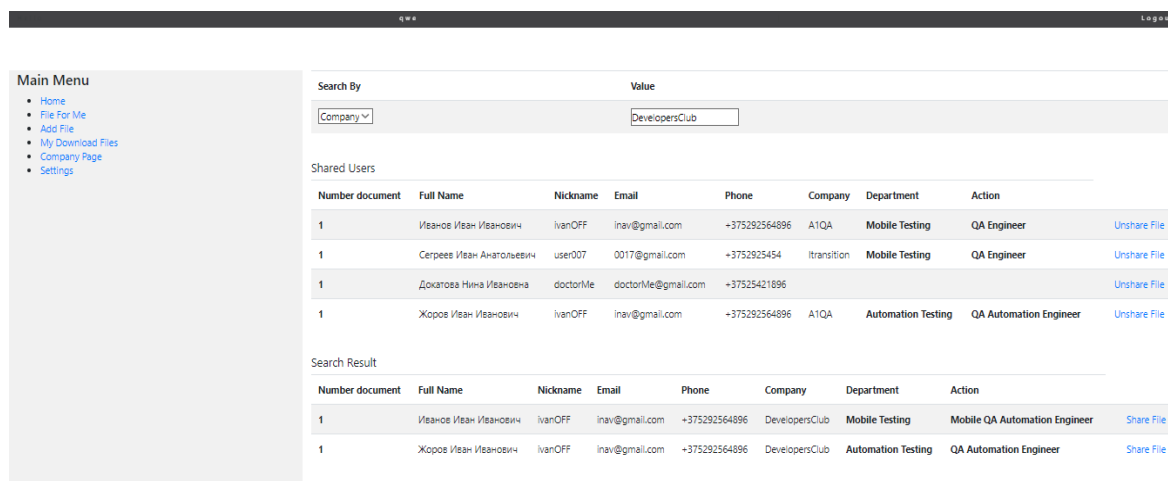


Рисунок 2. – Прототип страницы добавления пользователя, с которым возможно поделиться файлами

Закключение. Правильным подходом при разработке веб-интерфейса является использование современных технологий, которые позволяют решать свои задачи во всех браузерах. Это, в первую очередь, экономит время при разработке интерфейса, а также ресурсы на обработку данных. Это является очень актуальным, при большом количестве информации, которая подлежит постоянному изменению.

Рассмотрены технологии JSP и Bootstrap, при взаимодействии, позволяют создать адаптивный интерфейс, любого уровня сложности, при этом сохраняя его функциональные возможности и привлекательный внешний вид. По итогам данной работы, была описана возможность их применения к построению интерфейса и динамической обработке запросов без перезагрузки страницы.

ЛИТЕРАТУРА

1. Материал из Википедии — свободной энциклопедии. [JavaServerPages](https://ru.wikipedia.org/wiki/JSP) [Электронный ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki/JSP>. – Дата доступа: 25.09.2018.
2. Материал из Википедии — свободной энциклопедии. [Bootstrap](https://ru.wikipedia.org/wiki/Bootstrap_(фреймворк)) [Электронный ресурс]. – Режим доступа: [https://ru.wikipedia.org/wiki/Bootstrap_\(фреймворк\)](https://ru.wikipedia.org/wiki/Bootstrap_(фреймворк)). – Дата доступа: 25.09.2018.
3. Материал из Википедии — свободной энциклопедии. [JavaServer_Faces](https://ru.wikipedia.org/wiki/JavaServer_Faces) [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/JavaServer_Faces. – Дата доступа: 25.09.2018.

УДК 004.223.2

**СОЗДАНИЕ СХЕМЫ ЗАЩИЩЁННОЙ ПЕРЕДАЧИ СООБЩЕНИЙ
МЕЖДУ ПОЛЬЗОВАТЕЛЯМИ****В.В. ПЕТЮКЕВИЧ***(Представлено: канд. физ.-мат. наук, доц. Д.Ф. ПАСТУХОВ)*

Рассматривается проектирование схемы защищённой передачи данных между пользователями по схеме peer-to-peer, также вопросы защиты этих данных. Проведён анализ технологий, наиболее подходящих, для разработки данной схемы.

Введение. Некоторые средства передачи информации между пользователями, такие как Viber и Telegram, используют шифрование сообщений, однако осуществляют передачу сообщений через собственные сервера. Таким образом получается, что все сообщения пользователей могут храниться на сервере и, впоследствии, быть переданными кому-либо.

Эту проблему можно решить, используя peer-to-peer соединение.

Peer-to-peer сеть – оверлейная компьютерная сеть, основанная на равноправии участников. Часто в такой сети отсутствуют выделенные серверы, а каждый узел (peer) является как клиентом, так и выполняет функции сервера. В отличие от архитектуры клиент-сервера, такая организация позволяет сохранять работоспособность сети при любом количестве и любом сочетании доступных узлов. Участниками сети являются пиры [1].

Преимущества, получаемые при использовании peer-to-peer:

1. Защита от утечки данных со стороны сервера;
2. Снижение нагрузки на сервер приложения, т. к. сервер перестанет участвовать в процессе пересылки сообщений.

Существующие решения.

Ни одно из популярных средств обмена сообщениями не использует защиту сообщений на максимально возможном уровне. Viber и Telegram используют end-to-end шифрование, но их общая проблема – это контроль всего процесса передачи сообщений серверами этих сервисов.

End-to-end шифрование – способ передачи данных, в котором только пользователи, участвующие в общении, имеют доступ к сообщениям. Таким образом, использование сквозного шифрования не позволяет получить доступ к криптографическим ключам со стороны третьих лиц [2].

Тема совмещения end-to-end шифрования и peer-to-peer соединений раскрыта недостаточно хорошо. В интернете была найдена только одна статья на английском языке. Но в данной статье были рассмотрены только теоретические вопросы без рассмотрения вариантов используемых технологий.

Средства решения задачи.

В данной статье рассмотрен только случай передачи текстовых данных. Однако, используемые в данной схеме технологии можно применить и для передачи данных других видов.

End-to-end шифрование в peer-to-peer сети позволит избежать проблему перехвата пересылаемых данных. Так же, данное приложение должно работать в браузере, чтобы пользователю не приходилось ничего устанавливать.

В связи с требованиями, для установки peer-to-peer соединения между клиентами используется WebRTC:

WebRTC – проект с открытым исходным кодом, предназначенный для организации передачи потоковых данных между браузерами или другими поддерживающими его приложениями по технологии peer-to-peer [3].

На уровне API технология стандартизируется консорциумом W3C, а на протокольном уровне — сообществом IETF. Его включение в рекомендации W3C поддерживается Google Chrome (и других на его основе), Mozilla и Opera.

Преимущества технологии:

1. Проведение конференции в браузере значительно упрощает процесс проведения конференции – пользователю не нужно устанавливать для этого отдельные приложения;
2. Используемые кодеки обеспечивают хорошее качество связи;
3. Возможность реализации любых элементов интерфейса средствами HTML5 и JavaScript;
4. Открытый исходный код даёт больше возможностей для использования.

Недостатки технологии:

Технология определяет только общий стандарт передачи данных (видео и звука), но отдельные решения разных браузеров относительно адресации абонентов и прочих управляющих процессов не сов-

местимы между собой. Поэтому даже звонки между парой различных браузеров представляют отдельную сложность.

Для end-to-end шифрования применяется протокол Диффи–Хеллмана:

Протокол Диффи–Хеллмана – криптографический протокол, позволяющий двум и более сторонам получить общий секретный ключ, используя незащищенный от прослушивания канал связи. Полученный ключ используется для шифрования дальнейшего обмена с помощью алгоритмов симметричного шифрования [4].

Описание схемы передачи сообщения:

1. При авторизации клиенты генерируют, по протоколу Диффи–Хеллмана, свою пару открытый/закрытый ключ и отправляют открытый ключ на сервер;
2. Перед отправкой сообщений, клиенты через сервер получают открытые ключи друг друга и обмениваются сообщениями для установки peer-to-peer соединения через WebRTC. Для снижения нагрузки на сервер, сообщения установки peer-to-peer соединения передаются через протокол WebSocket;
3. В результате получения открытых ключей друг друга, клиенты могут сгенерировать общий ключ по протоколу Диффи–Хеллмана;
4. После установки peer-to-peer соединения, клиенты могут обмениваться зашифрованными общим ключом сообщениями.

Алгоритмы, использующиеся в данной схеме:

1. Для создания пары открытый/закрытый ключ используется протокол Диффи–Хеллмана;
2. Для генерации закрытого ключа используется алгоритм HMAC-SHA256 с размером блока 64 бит. Алгоритмом HMAC-SHA256 хэшируется имя пользователя и случайное число. Сделано это для того, чтобы при каждой новой сессии ключ получался разным;
3. Для шифрования сообщений используется алгоритм AES с длиной ключа 256 бит и размером блока 64 бит. В качестве ключа для шифрования/дешифрования сообщений используется общий ключ двух пользователей, который был получен по протоколу Диффи–Хеллмана.

Установка Peer-to-Peer соединения между двумя клиентами:

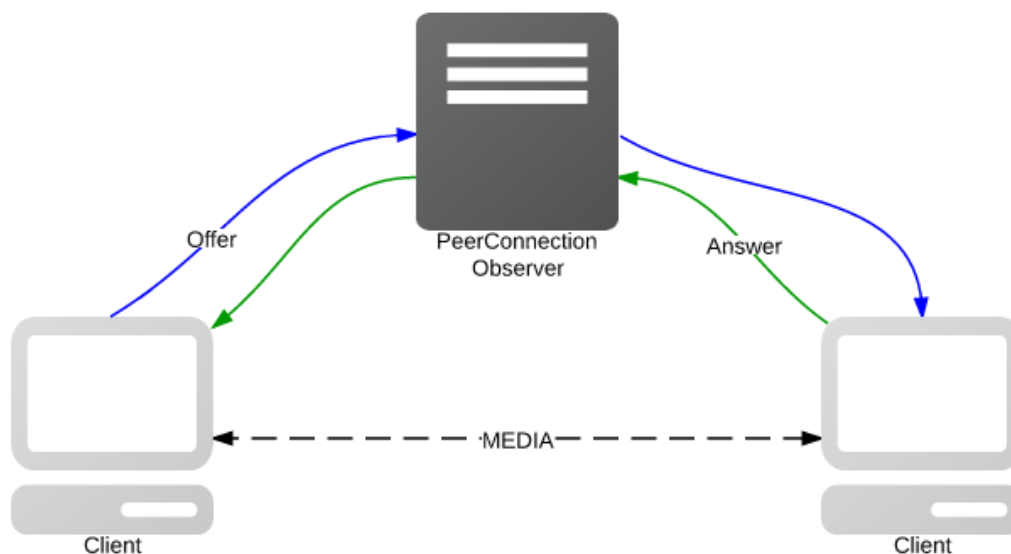


Рисунок. – Схема peer-to-peer соединения между двумя клиентами

Упрощенная схема соединения между двумя клиентами:

1. Первый клиент отправляет так называемый Offer второму клиенту через сервер;
2. Второй клиент отправляет через сервер ответ первому клиенту;
3. Устанавливается peer-to-peer соединение между клиентами.

Таблица 1. – Замеры скорости шифрования реализации AES на языке JavaScript

Количество символов	Время, с
200	0,3
500	0,6
1000	1

Для быстрой передачи сообщений на сервер используется протокол WebSocket. WebSocket – протокол полнодуплексной связи поверх TCP-соединения, предназначенный для обмена сообщениями между браузером и веб-сервером в режиме реального времени [5]. В сравнении с HTTP, WebSocket передаёт намного меньше служебной информации с каждым запросом.

Заключение. В ходе данного исследования был спроектирована схема защищённой передачи данных между пользователями с использованием end-to-end шифрования в peer-to-peer сети. End-to-end шифрование было реализовано через протокол Диффи-Хеллмана с использованием алгоритмов HMAC-SHA256 и AES. Для установки peer-to-peer соединения в браузере был использован WebRTC. Следует отметить, что разработанная схема оставляет возможность для доработки и введения дополнительных средств защиты.

ЛИТЕРАТУРА

1. Материал из Википедии – свободной энциклопедии [Электронный ресурс] // Одноранговая сеть. – Режим доступа: https://ru.wikipedia.org/wiki/Одноранговая_сеть. – Дата доступа: 20.09.2018.
2. Бутакова, Н.Г. Криптографическая защита информации : учеб. пособие для вузов / Н.Г. Бутакова, В.А. Семенов, Н.В. Федоров. – М. : Изд-во МГИУ, 2011. – С 91–102.
3. Материал из Википедии – свободной энциклопедии [Электронный ресурс] // WebRTC. – Режим доступа: <https://ru.wikipedia.org/wiki/WebRTC>. – Дата доступа: 20.09.2018.
4. Материал из Википедии – свободной энциклопедии [Электронный ресурс] // Протокол Диффи-Хеллмана. – Режим доступа: https://ru.wikipedia.org/wiki/Протокол_Диффи_Хеллмана. – Дата доступа: 20.09.2018.
5. Материал из Википедии – свободной энциклопедии [Электронный ресурс] // WebSocket. – Режим доступа: <https://ru.wikipedia.org/wiki/WebSocket>. – Дата доступа: 20.09.2018.

УДК 004.223.2

ПРОЕКТИРОВАНИЕ ГРАФИЧЕСКОГО ИНТЕРФЕЙСА КЛИЕНТСКОЙ ЧАСТИ ВЕБ-ПРИЛОЖЕНИЯ ЧАТ

В.В. ПЕТЮКЕВИЧ*(Представлено: канд. физ.-мат. наук, доц. Д.Ф. ПАСТУХОВ)*

Рассматривается проектирование графического интерфейса веб-приложения для обмена сообщениями между пользователями. Проведён анализ подходов проектирования графического интерфейса, наиболее подходящих для разработки данного веб-приложения. Задача: реализация прототипа интерфейса на основе веб-технологий.

Введение. На сегодняшний день, любое веб-приложение должно быть основано на обще поддерживаемых стандартах разработки, а также обладать понятным для пользователя интерфейсом.

Графический интерфейс пользователя является частью пользовательского интерфейса и определяет взаимодействие с пользователем на уровне визуализированной информации [1].

Интерфейс должен быть интуитивно понятным. Таким, чтобы пользователю не требовалось объяснять, как им пользоваться. Разрабатываемое веб-приложение для общения пользователей в сети чат, позволяет производить обмен сообщениями между пользователями чата.

Выбор технологий разработки интерфейса.

Чат – средство обмена сообщениями по компьютерной сети в режиме реального времени, а также программное обеспечение, позволяющее организовывать такое общение. Характерной особенностью является коммуникация именно в реальном времени или близкая к этому, что отличает чат от форумов и других «медленных» средств. То есть, если на форуме можно написать вопрос и ждать, пока кто-нибудь посчитает нужным на него ответить (в то же время, можно получить и несколько ответов сразу от разных пользователей), то в чате общение происходит только с теми, кто присутствует в нём в настоящий момент, а результаты обмена сообщениями могут и не сохраняться.

На основании анализа сущности проблемы и исходных данных были сформулированы основные функции разрабатываемого программного продукта: наличие дружественного интерфейса, не требующего специальной подготовки пользователей перед использованием; должен отображаться список подключенных пользователей.

Из современных языков программирования для написания веб-приложений используется JavaScript.

JavaScript – мультипарадигменный язык программирования. Поддерживает объектно-ориентированный, императивный и функциональный стили. Является реализацией языка ECMAScript (стандарт ECMA-262) [2].

JavaScript обычно используется как встраиваемый язык для программного доступа к объектам приложений. Наиболее широкое применение находит в браузерах как язык сценариев для придания интерактивности веб-страницам.

Основные архитектурные черты: динамическая типизация, слабая типизация, автоматическое управление памятью, прототипное программирование, функции как объекты первого класса.

Для обеспечения скорости работы пользовательского интерфейса и снижению излишней сложности кода клиентского сайта, была использована ReactJS. ReactJS может использоваться при разработке одностраничных приложений и мобильных приложений. Он направлен, прежде всего, на обеспечение скорости, простоты и масштабируемости. ReactJS компонентно-ориентированный, он позволяет создавать инкапсулированные компоненты, которые управляют своим состоянием, а затем использовать их для создания сложных пользовательских интерфейсов [3].

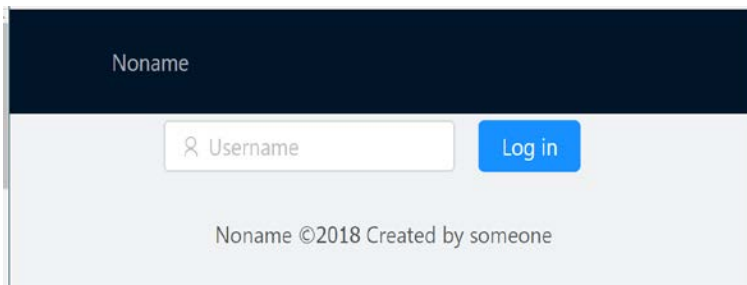
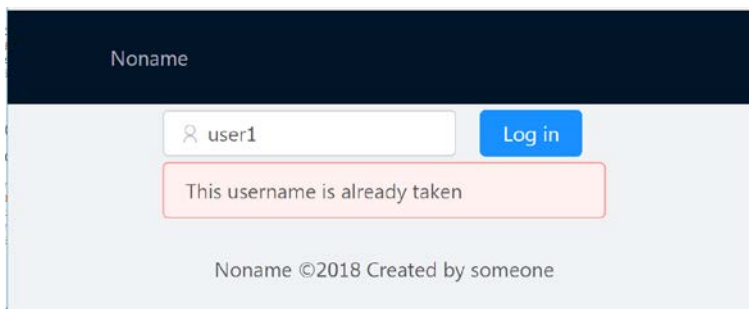
Так же, в данной работе, была использована библиотека Ant Design. Эта библиотека предоставляет набор готовых элементов пользовательского интерфейса для использования в фреймворке ReactJS. Использование данной библиотеки помогло уменьшить временные затраты на проектирование графического интерфейса и улучшить его качество.

Ant Design использует современные наработки в области CSS и HTML, поэтому необходимо быть внимательным при поддержке старых браузеров.

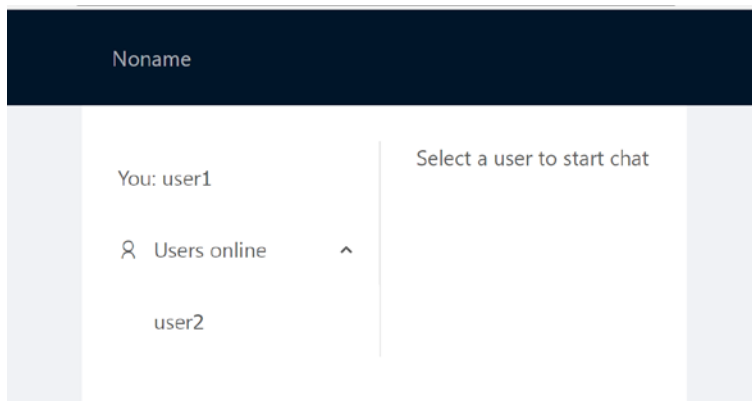
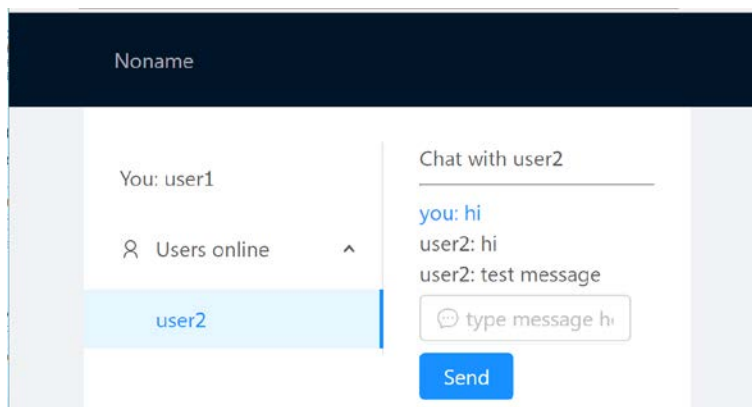
Проектирование интерфейса.

Весь интерфейс приложения состоит из двух экранов: экран входа и основной экран. Сами элементы графического интерфейса подобраны так, чтобы быть максимально интуитивно понятными и знакомыми большинству пользователей.

При посещении клиентского сайта, пользователь изначально направляется на страницу авторизации. На странице авторизации пользователю предлагается ввести свой логин, который будет использоваться для идентификации пользователя.

**Рисунок 1. – Страница ввода логина****Рисунок 2. – Сообщение при вводе уже используемого логина**

Далее пользователь попадает на основную страницу чата. На странице чата пользователь может выбрать собеседника и начать отправлять ему сообщения.

**Рисунок 3. – Приглашение выбора собеседника****Рисунок 4. – Страница чата**

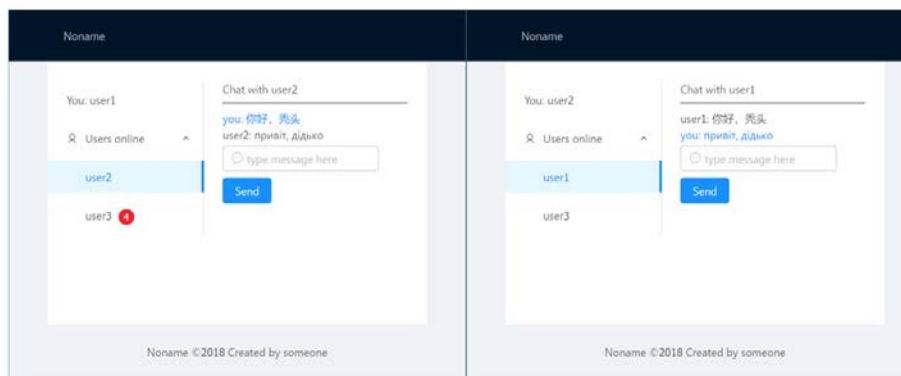


Рисунок 5. – Передача сообщений на разных языках

Заключение. Правильным подходом при разработке веб-интерфейса является использование современных технологий, которые позволяют решать свои задачи во всех браузерах. Это, в первую очередь, экономит время при разработке интерфейса, а также ресурсы на обработку данных.

Рассмотренные технологии React.JS и Ant Design, при взаимодействии, позволяют создать адаптивный интерфейс, любого уровня сложности. По итогам данной работы, была описана возможность их применения к проектированию графического интерфейса пользователя.

ЛИТЕРАТУРА

6. Материал из Википедии – свободной энциклопедии [Электронный ресурс] // Графический интерфейс пользователя. Режим доступа: https://ru.wikipedia.org/wiki/Графический_интерфейс_пользователя. – Дата доступа: 20.09.2018.
7. Материал из Википедии – свободной энциклопедии [Электронный ресурс] // JavaScript. – Режим доступа: <https://ru.wikipedia.org/wiki/JavaScript>. – Дата доступа: 20.09.2018.
8. Design Patterns – Elements of Reusable Object-Oriented Software / Erich Gamma [et al.] – Library of Congress Cataloging-in-Publication Data, 2009. – P. 15–20.

УДК 004.432.2

**ПРОБЛЕМА ИСПОЛЬЗОВАНИЯ ЧИСТЫХ ФУНКЦИЙ
В ЯЗЫКЕ ПРОГРАММИРОВАНИЯ JAVASCRIPT****К.И. ЛАПКОВСКИЙ***(Представлено: ст. преп. Д.В. ПЯТКИН)*

Статья посвящена использованию чистых функций в языке программирования, который не разрабатывался для написания программ в функциональном стиле. Рассмотрим применимость языка JavaScript в функциональном программировании на примере главной концепции функционального программирования – использовании чистых функций. Проанализировав примеры различных функций в языке JavaScript, сформулируем правила, которым необходимо следовать при написании программы в функциональном стиле.

Введение. Функциональное программирование с каждым годом приобретает все большую популярность. Одна из концепций функционального программирования – использование чистых функций. Написание программы, опираясь эту на парадигму и используя специальные языки программирования (Scala, Haskell), позволяет создать приложение, которое одинаково легко справляется с любым объемом данных, работая при этом очень быстро. Однако насколько возможно использование не специализированных языков программирования для такой цели? Рассмотрим применение чистых функций в популярном языке программирования JavaScript.

Основной раздел

JavaScript – это язык программирования с слабой динамической типизация, реализующий прототипную модель наследования [1].

Прототипное наследование

Прототип – хранилище методов [2]. Если в объекте вызывается метод, то этот метод сначала ищется в самом объекте, потом в прототипе этого объекта. Прототипы могут формировать цепочки, создавая при этом сколь угодно сложную иерархию. Технически в прототипе можно хранить не только методы, но и свойства, но этой возможностью пользуются редко. Использование прототипов помогает в переиспользовании программный код [3]. Классическое наследование решает ту же задачу.

Чистые функции

Чистыми называют функции, которые не имеют побочных эффектов ввода-вывода и памяти (они зависят только от своих параметров и возвращают только свой результат). Чистые функции обладают несколькими полезными свойствами, многие из которых можно использовать для оптимизации кода:

– если результат чистой функции не используется, её вызов может быть удален без вреда для других выражений;

– результат вызова чистой функции может быть мемоизирован, то есть сохранен в таблице значений вместе с аргументами вызова. Если в дальнейшем функция вызывается с этими же аргументами, её результат может быть взят прямо из таблицы, не вычисляясь (иногда это называется принципом прозрачности ссылок). Мемоизация, ценой небольшого расхода памяти, позволяет существенно увеличить производительность и уменьшить порядок роста некоторых рекурсивных алгоритмов;

– если нет никакой зависимости по данным между двумя чистыми функциями, то порядок их вычисления можно поменять или распараллелить (говоря иначе вычисление чистых функций удовлетворяет принципам thread-safe);

– если весь язык не допускает побочных эффектов, то можно использовать любую политику вычисления. Это предоставляет свободу компилятору комбинировать и реорганизовывать вычисление выражений в программе (например, исключить дублирующие структуры) [1].

Примеры функций в JavaScript

В листинге 1 представлены четыре функции. Проанализируем являются ли они чистыми.

Листинг 1 – функции JavaScript

```
// А: Простое умножение
function doubleA(n) {
    return n * 2
}
// В: С переменной
var two = 2
```



```
function doubleB(n) {
  return n * two
}
// C: С вспомогательной функцией
function getTwo() {
  return 2
}
function doubleC(n) {
  return n * getTwo()
}
// D: Преобразование массива
function doubleD(arr) {
  return arr.map(n => n * 2)
}
```

Функция В не является чистой, так как читает переменную из внешней области видимости, тем самым становится зависимой от этой переменной. Если в функцию В два раза передать одинаковые параметры, но перед вторым вызовом функции изменить переменную `two`, возвращаемые значения после первого и второго вызова функции `doubleB` будут отличаться.

Функция `doubleC` так же нельзя назвать чистой. Функция `doubleC` зависит от функции `getTwo`, которая может быть переопределена. Если `getTwo` присвоить значение отличное от функции, то `doubleC` выбросит исключение, так как нельзя вызвать не функцию.

Рассмотрим функцию `doubleD`. Внутри функции вызывается метод `map`, который хранится в прототипе. Так как все экземпляры классов ссылаются на один и тот же прототип, то при изменении прототипа поведение всех экземпляров класса тоже изменится. Помимо этого, функцию можно “перекрыть”, то есть определить функцию с таким же названием, как и в прототипе. Функция `doubleD` не чистая.

Чтобы понять является ли `doubleA` чистой функцией необходимо знать механизм, который использует JavaScript для преобразования объектов в примитивы. Когда интерпритатору необходимо преобразовать объект в примитив, для объекта вызывается скрытый метод `toPrimitive` [4]. `ToPrimitive` вызывает метод `valueOf`. Если `valueOf` возвращает примитив, то этот примитив и используется как результат преобразования. Если `valueOf` вернул объект, то вызывается метод `toString`, возвращаемое значение метода `toString` и будет результатом преобразования. Методы `toString`, `toPrimitive` и `valueOf` можно переопределить или перекрыть, как и обычные методы прототипа.

Заключение

Таким образом возможность изменять прототип и перекрывать функции мешает использовать чистые функции в JavaScript.

Однако можно обезопасить свою программу от перекрытия функций, если явно вызывать методы прототипа, и не доверять поиск метода интерпритатору JavaScript. То есть для вызова метода `map` рекомендуется использовать синтаксис, представленный в листинге 2.

Листинг 2 – вызов метода прототипа

```
Array.prototype.map.call(array, ()=>{ })
```

ЛИТЕРАТУРА

1. Wikipedia [Электронный ресурс] // Функциональное программирование. – Режим доступа: https://ru.wikipedia.org/wiki/Функциональное_программирование#Чистые_функции. – Дата доступа: 24.09.2018.
2. Современный учебник JavaScript [Электронный ресурс] // Прототип объекта. – Режим доступа: <https://learn.javascript.ru/prototype>. Дата доступа: 24.09.2018.
3. MDN web docs [Электронный ресурс] // Object.prototype. – Режим доступа: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object/prototype. – Дата доступа: 24.09.2018.
4. You don't know JS [Электронный ресурс] // Coercion. – Режим доступа: <https://github.com/getify/You-Dont-Know-JS/blob/master/types%20%26%20grammar/ch4.md>. – Дата доступа: 25.09.2018.

УДК 004.432.2

**ПЕРЕОПРЕДЕЛЕНИЕ КОНТЕКСТА ВЫЗОВА ПО УМОЛЧАНИЮ
В ЯЗЫКЕ ПРОГРАММИРОВАНИЯ JAVASCRIPT****К.И. ЛАПКОВСКИЙ***(Представлено: ст. преп. Д.В. ПЯТКИН)*

Статья посвящена переопределению контекста вызова функций в JavaScript. Рассмотрим все возможные случаи задания контекста функций. Проанализируем насколько эффективны уже существующие механизмы. В результате будет сформулирован альтернативный способ задания контекста функций.

Введение. JavaScript – мультипарадигменный язык программирования. Такая универсальность языка влечет за собой некоторые проблемы. Например, если рассматривать JavaScript как объектно-ориентированный язык программирования, можно заметить, что на уровне языка не реализованы такие механизмы как абстрактный класс (хотя программист сам без труда может запрограммировать класс, экземпляр которого нельзя создать), а прототипное наследование усложняет написание программы в функциональном стиле.

Неправильное задание контекста вызова скорее приведет к ошибкам во время выполнения программы, чем к невозможности реализовать какие-либо принципы разных парадигм программирования. Однако если программа не была должным образом протестирована, исключительные ситуации могут появляться редко, воспроизводиться сложно, при этом сильно влиять на работу программы.

Основной раздел. Контекст вызова – это значение, в контексте которого была вызвана функция. Любая функция в JavaScript может обращаться к ключевому слову `this` внутри себя [1]. Ключевое слово `this` ссылается на контекст вызова функции. Функция в момент вызова связывается с объектом, и `this` внутри этой функции получает соответствующее значение. Рассмотрим четыре возможных случая определения контекста в порядке их приоритета:

– Связывание с помощью оператора `new`. Если функция была вызвана с оператором `new`, то `this` получит ссылку на только что созданный, пустой объект. Пример представлен в листинге 1.

Листинг 1 – определение контекста для функции `foo` с помощью оператора `new`:

```
var bar = new foo()
```

– Явное связывание. Если функция вызвана с помощью методов `call` или `apply`, или к функции привязан контекст с помощью функции `bind`, тогда контекстом вызова будет первый аргумент, переданный в эти функции. Пример представлен в листинге 2.

Листинг 2 – определение контекста функции методом `call`:

```
var bar = foo.call(obj2)
```

– Неявное связывание. Если функция вызвана в контексте какого-то объекта, так же известного как объект-контейнера. То в `this` попадет ссылка на этот объект. Пример в листинге 3.

Листинг 3 – неявное задание контекста для функции:

```
var bar = obj1.foo()
```

– Связывание по умолчанию. Если ни один из вышеописанных случаев не применим к вызову функции, то в `this` попадет значение по умолчанию. В строгом режиме – это `undefined`, в не строгом – это глобальный объект [2]. Пример представлен в листинге 4.

Листинг 4 – задание контекста функции по умолчанию:

```
var bar = foo()
```

Проблема жесткого связывания и связывания по умолчанию.

Контекст функции теряется, если использовать функцию как переменную [3]. Обычно для привязки контекста используется жесткую привязку, или стрелочную функцию, которая получает контекст из замыкания.

Жесткая привязка контекста вызова к функции (через метод `bind`), уменьшает гибкость функции, не позволяя переопределить контекст с помощью неявного или даже явного задания контекста вызова.

Привязка функции к контексту по умолчанию редко является намеренной, так как в случае с привязкой к `undefined`, любая попытка считать свойство у `this`, или вызов метода, приведет к ошибке во время выполнения программы. Если же использовать не строгий режим, то в контекст вызова попадет гло-

бальный объект, и изменение любого свойства или метода объекта приведет к изменению глобального объекта. Сценарий с глобальным объектом в качестве контекста вызова часто приводит к сложно уловимым ошибкам.

Реализация переопределения контекста по умолчанию

В качестве решения проблемы жесткого связывания и связывания по умолчанию можно использовать переопределение контекста вызова. Реализация функции представлена в листинге 5.

Листинг 5 – реализация переопределения контекста по умолчанию

```
if (!Function.prototype.softBind) {
  Function.prototype.softBind = function(obj) {
    var fn = this,
        curried = [].slice.call( arguments, 1 ),
        bound = function bound() {
          return fn.apply(
            (!this ||
             (typeof window !== "undefined" &&
              this === window) ||
             (typeof global !== "undefined" &&
              this === global)
            ) ? obj : this,
            curried.concat.apply( curried, arguments )
          );
        };
    bound.prototype = Object.create( fn.prototype );
    return bound;
  };
}
```

Заключение. В статье рассмотрены различные способы привязки контекста, и проблемы их использования. Разработана функция, задающая контекст по умолчанию, позволяющая решить проблему потери контекста, при этом сохраняя возможность его переопределения далее в программе.

ЛИТЕРАТУРА

1. JavaScript.info [Электронный ресурс] // Object methods, this. – Режим доступа: <https://javascript.info/object-methods>. – Дата доступа: 24.09.2018.
2. You don't know JS [Электронный ресурс] // this & Objects Prototypes. – Режим доступа: <https://github.com/getify/You-Dont-Know-JS/blob/master/this%20%26%20object%20prototypes/ch2.md>. – Дата доступа: 24.09.2018.
3. Dmitri Pavlutin Blog [Электронный ресурс] // Gentle explanation of 'this' keyword in JavaScript. – Режим доступа: <https://dmitripavlutin.com/gentle-explanation-of-this-in-javascript/>. – Дата доступа: 24.09.2018.
4. MDN web docs [Электронный ресурс] // this. – Режим доступа: <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/this>. – Дата доступа: 24.09.2018

УДК 004.021

ПРОЕКТИРОВАНИЕ ГРАФИЧЕСКОГО ИНТЕРФЕЙСА ДЛЯ ВЕБ-ПРИЛОЖЕНИЯ, РАЗРАБОТАННОГО ДЛЯ ОТСЛЕЖИВАНИЯ СТАТУСА ВЫПОЛНЯЕМЫХ ИТЕРАЦИЙ И СЛЕЖЕНИЕ ВРЕМЕНИ ИХ ВЫПОЛНЕНИЯ

К.С. САМСОНОВ

(Представлено: ст. преп. Ю.Н. КРАВЧЕНКО)

Рассматривается построение графического интерфейса пользователя для веб-приложения.

Введение. Одним из самых известных языков программирования в разработке веб-приложений является JavaScript. Он далеко не прост в изучении, но предоставляет гарантию создания детально продуманного и многофункционального решения. Это язык объектно-ориентированного программирования, который дает возможность реализовать сложную механику веб-страниц. Пользователь может наблюдать разного рода 2D и 3D анимацию, прокрутку видео и обновление контента, одним словом – придает динамичность и не только. Все это реализуется с помощью JavaScript.

Главные плюсы JavaScript, за которые он и получил такую распространенность:

- Полная интеграция с HTML/CSS.
- По умолчанию поддерживается всеми распространенными браузерами и включен в них.
- Простая реализация простых задач.

Основной раздел. Angular 5 предоставляет такую функциональность, как двустороннее связывание, позволяющее динамически изменять данные в одном месте интерфейса при изменении данных модели в другом, шаблоны, маршрутизация. Одной из ключевых особенностей Angular является то, что он использует в качестве языка программирования JavaScript. Основная цель данного фреймворка предназначена для разработки одностраничных приложений. Его цель — расширение браузерных приложений на основе MVC-шаблона, а также упрощение тестирования и разработки.

Фреймворк работает с HTML, содержащим дополнительные пользовательские атрибуты, которые описываются директивами, и связывает ввод или вывод области страницы с моделью, представляющей собой обычные переменные JavaScript. Значения этих переменных задаются вручную или извлекаются из статических или динамических JSON-данных.

Веб-приложение состоит из множества файлов: текстового содержимого, кода, таблиц стилей, медиа-контента, и так далее. Когда вы создаете веб-сайт, вы должны собрать эти файлы в разумную структуру и убедиться, что они могут взаимодействовать друг с другом.

Язык гипертекстовой разметки (Hypertext Markup Language, HTML) - это код, который вы используете для структурирования веб-содержимого и придания ему смысла и цели. Каскадные таблицы стилей (Cascading Stylesheets, CSS) - это код, который вы используете для стилизации своего веб-сайта. JavaScript – это язык программирования, который вы используете для добавления интерактивных функций для вашего веб-сайта, например, игр, событий, которые происходят при нажатии кнопок или ввода данных в формы, динамических эффектов стилизации, анимации и многого другого.

Description:

Release Date:

Schedule:

The new version will be added chronologically after the selected version. It will appear above the selected version on the list. Select 'Before First Version' to add before any other version (will appear at the bottom of list).

Name	Description	Release Date	Schedule
next bugfix			↓ ↻
2.0.0			↻ ↑ ↓ ↻
2.0.0-m1			↻ ↑ ↓ ↻
1.1.x			↻ ↑ ↓ ↻
1.1.0			↻ ↑ ↓ ↻
1.1.0-rc1			↻ ↑ ↓ ↻
1.1.0-M3			↻ ↑ ↓ ↻
1.1.0-M2			↻ ↑ ↓ ↻
1.1.0-M1			↻ ↑

Рисунок. – Макет главной страницы приложения

Компоненты представляют основные строительные блоки приложения Angular 2. Каждое приложение Angular имеет как минимум один компонент. Поэтому создадим в папке `src/app` новый файл, который назовем `app.component.ts` и в котором определим следующий код компонента

Далее собственно идет функция-декоратор `@Component`, которая ассоциирует метаданные с классом компонента `AppComponent`. В этой функции, во-первых, определяется параметр `selector` или селектор `css` для HTML-элемента, который будет представлять компонент. Во-вторых, здесь определяется параметр `template` или шаблон, который указывает, как надо визуализировать компонент. В этом шаблоне задана двусторонняя привязка с помощью выражений `[(ngModel)]="name"` и `{{name}}` к некоторой модели `name`.

И в конце собственно экспортируется класс компонента `AppComponent`, в котором как раз определяется модель `name` - в данном случае это пустая строка.

Приложение Angular состоит из модулей. Модульная структура позволяет легко подгружать и задействовать только те модули, которые непосредственно необходимы. И каждое приложение имеет как минимум один корневой модуль. Поэтому создадим в папке `src/app` новый файл, который назовем `app.module.ts` со следующим содержанием:

Этот модуль, который в данном случае называется `AppModule`, будет входной точкой в приложение.

С помощью директив `import` здесь импортируется ряд нужных нам модулей. Прежде всего, это модуль `NgModule`. Для работы с браузером также требуется модуль `BrowserModule`. Так как наш компонент использует элемент `input` или элемент формы, то также подключаем модуль `FormsModule`. И далее импортируется созданный ранее компонент.

Теперь нам надо указать Angular, как запускать наше приложение. Для этого создадим в папке `src` (на уровень выше, чем расположены файлы `app.component.ts` и `app.module.ts`) файл `main.ts` со следующим содержанием

Этот код инициализирует платформу, которая запускает приложение, и затем использует эту платформу для загрузки модуля `AppModule`.

Данный файл определяет полифилы - инструменты, которые необходимы для поддержки приложения на Angular старыми браузерами. В частности, первая большая группа полифилов предназначена для работы в IE9-11. Однако если мы не планируем поддерживать этот браузер, то данную группу импорта пакетов можно не включать. Но в любом случае файл должен содержать импорт `zone: import 'zone.js/dist/zone'`.

Заключение. В данной статье рассмотрены основные элементы построения графического интерфейса для веб-приложения. Представлены примеры и результаты.

УДК 004.021

**ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ ДЛЯ ВЕБ-ПРИЛОЖЕНИЯ,
ПРЕДНАЗНАЧЕННОГО ДЛЯ ОТСЛЕЖИВАНИЯ СТАДИЙ ВЫПОЛНЕНИЯ
РАЗРАБАТЫВАЕМЫХ ПРОЕКТОВ****К.С. САМСОНОВ***(Представлено: ст. преп. Ю.Н. КРАВЧЕНКО)**Рассматриваются принципы моделирования базы данных для веб-приложения*

Введение. Разрабатываемое веб-приложение предназначено для организации процесса контроля запросов и задач, обладающих функциональностью, частично присущей большим проектам.

Пользователь может организовать контроль разработки проектов, раздав задачи исполнителям. Он же может определить свой собственный метод движения заданий - от создания к исполнению и контролю результатов, сконфигурировать правила уведомления о событиях всех участников процесса, управлять правами доступа пользователей и делать многое другое.

Основной раздел

Пользователем веб-приложения считается каждый, кто имеет полный доступ ко всем встроенным функциям и функциональным возможностям приложения, созданным специально для команд разработчиков. Это подразумевает доступ к scrum- и kanban-доскам, десяткам отчетов, глубокую интеграцию с инструментами для разработки.

Веб-приложение призвано помочь пользователям назначать и устанавливать приоритеты для своей работы. Оно позволяет управлять всем процессом разработки приложений, гарантируя, что ничего не будет потеряно, от концепции до запуска. Её простой интуитивно понятный интерфейс позволяет сотрудничать с товарищами по команде и позволяет эффективно выполнять свою работу

Пользователь в приложении может использовать его главный функционал как для планирования разработки программного обеспечения, так и для отслеживания и выпуска эффективного программного обеспечения. Пользователь можете создавать истории проблемы, планировать спринты и назначать задания членам вашей команды. Приоритеты и обсуждение задач вашей команды полностью доступны и удобны для анализа. Кроме того, веб-приложение предлагает отчеты в реальном времени, визуальные данные для повышения производительности команды.

Данное веб-приложение написано на языке программирования JavaScript с использованием Angular-5 Framework, а также современных средств для работы с веб, такими как HTML5, CSS3, JavaScript.

Вся информация на сервере храниться в базе данных MongoDB. MongoDB – документоориентированная система управления базами данных с открытым исходным кодом, не требующая описания схемы таблиц. Классифицирована как язык запросов NoSQL, использует JSON-подобные документы. Пример данных в MongoDB в виде JSON-подобных документов представлен в листинге 1.

```
_id: "58b97f69a20c1217adfd9a4f",  
Type_task: "bug",  
reported: "kosta",  
assigned: "alex",  
summary: "deffect 001"  
description: "fix logo"
```

Заключение. В данной статье были рассмотрены основные сущности и принципы моделирования базы данных, для разрабатываемого веб-приложения.

УДК 621.396

ОПТИЧЕСКИЙ ГЕНЕРАТОР ДЛЯ ПЕРЕДАЧИ ВИДЕОСИГНАЛОВ ПО ОПТИЧЕСКОМУ КАНАЛУ НА ПОДНЕСУЩИХ ЧАСТОТАХ

Е.В. СМЕТАНА

(Представлено: канд. техн. наук, доц. В.Ф. ЯНУШКЕВИЧ)

Проведен анализ основных параметров оптических генераторов, управляемых током накачки частотного модулятора на поднесущей частоте для повышения линейности амплитудных характеристик источника излучения и снижения уровня гармонических искажений.

Способ передачи видеосигналов по оптическому каналу с использованием поднесущих частот обладает следующими преимуществами: большая информационная емкость канала связи, высокая помехоустойчивость канала связи. Важное значение для тракта передачи видеосигналов имеет уровень нелинейных искажений. Для получения небольших нелинейных искажений необходимо: уменьшить m (глубина модуляции), что приведет к уменьшению отношения сигнал/шум на приемной стороне, либо можно осуществить компенсацию вносимых нелинейных искажений [1–3].

Используют следующие методы: метод фазовой модуляции, введение предискажений, метод полупредискажений, метод отрицательной ОС [4].

Для уменьшения нелинейных искажений оптического генератора была разработана схема оптического излучателя на основе структурной схемы, изображенной на рисунке 1:

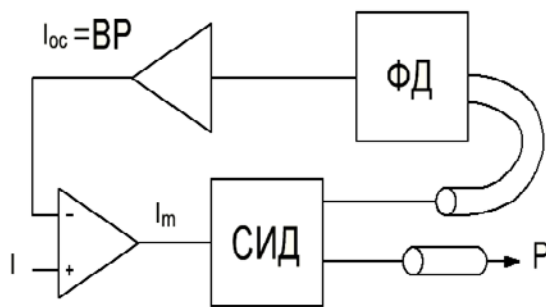


Рисунок 1. – Метод ОС:

СИД – светоизлучающий диод; ФД – фотодетектор;
 А1 – усилитель в цепи обратной связи; А2 – усилитель входного сигнала

На светоизлучающем диоде собран оптический генератор, сигнал с выхода которого проходит через цепь обратной связи, образованную фотодетектором и усилителем А1, обратно на вход генератора.

Применение такой схемы позволяет уменьшить нелинейные искажения в $(1+\beta\alpha)^2$ раз.

Схема разработанного оптического генератора представлена на рисунке 2:

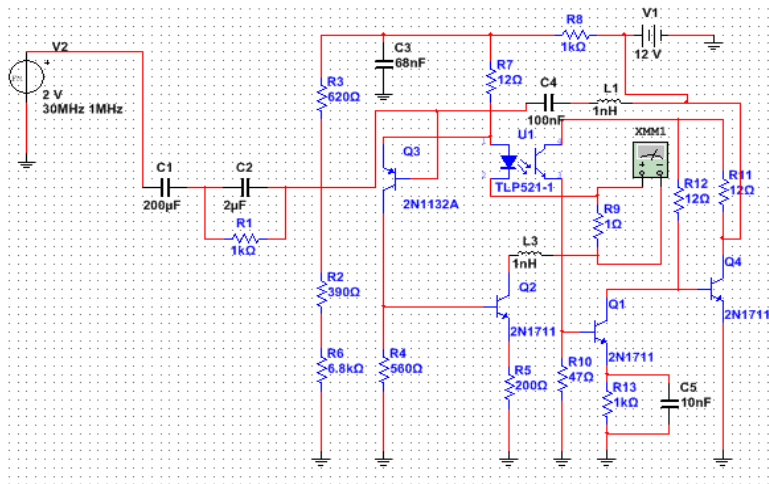


Рисунок 2. – Схема оптического генератора

На рисунке 3 отображен график зависимости тока накачки от входного напряжения.

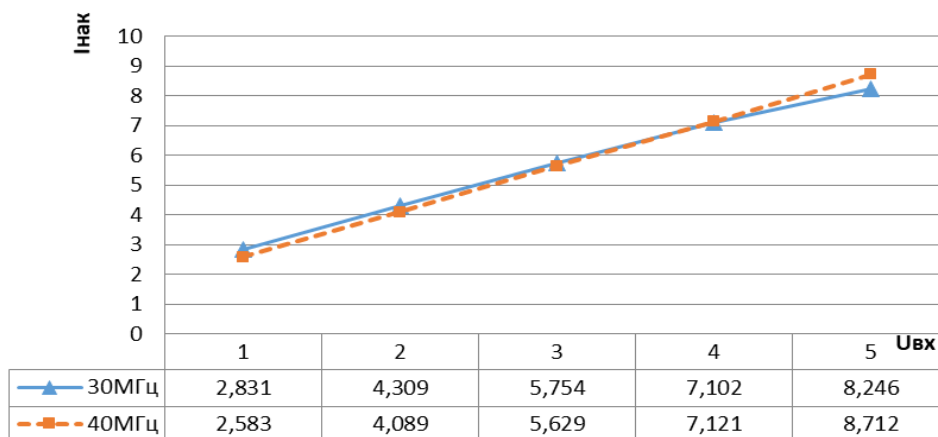


Рисунок 3. – Зависимость $I_{нак}$ от $U_{вх}$ при $f_0 = 30-40$ МГц

В качестве источника сигнала используется блок FM_VOLTAGE из стандартной библиотеки программного пакета Multisim. Источником оптического служит светодиод оптопары U1. В качестве фотодетектора выступает фотодиод данной оптопары. На транзисторах Q2, Q3 собран усилитель входного сигнала. Сигнал по цепи обратной связи усиливается транзисторами Q1, Q4 и через корректирующую цепочку L1, C4 подается обратно на вход усилителя на транзисторе Q2. Цепь обратной связи корректирует коэффициент усиления каскада Q2, Q3 и тем самым регулирует величину нелинейных искажений.

Заключение. Результаты исследования могут быть использованы в системах кабельного телевидения, где требуются высокоэффективные системы обработки информации. В связи с развитием радиоэлектронных устройств и систем кабельного телевидения появилась необходимость в качественном улучшении технических характеристик. Данный метод способствует увеличению отношения сигнал / шум. Результаты исследований свидетельствуют о хорошей линейности разработанного оптического генератора.

ЛИТЕРАТУРА

1. Янушкевич, В.Ф. Устройства оптической обработки сигналов : метод. пособие / В.Ф. Янушкевич. – Новополоцк, ПГУ, 2004.
2. Щербак, Ю.М. Устройства оптической обработки сигналов / Ю.М. Щербак. – Минск : БГУИР, 1997.
3. Оптическая обработка радиосигналов в реальном времени / под ред. С.В. Кулакова. – М. : Радио и Связь, 1989.
4. Мустель, Е.Р. Методы модуляции и сканирования света / Е.Р. Мустель, В.К. Парыгин. – М. : Наука, 1970.
5. Гауэр, Дж. Оптические линии и связи : пер. с англ. / под ред. А.И. Ларкина. – М. : Радио и связь, 1989.

УДК 621.396

ЧАСТОТНЫЙ МОДУЛЯТОР ДЛЯ ПЕРЕДАЧИ ВИДЕОСИГНАЛОВ ПО ОПТИЧЕСКОМУ КАНАЛУ НА ПОДНЕСУЩИХ ЧАСТОТАХ

Е.В. СМЕТАНА

(Представлено: канд. техн. наук, доц. В.Ф. ЯНУШКЕВИЧ)

Результаты исследования могут быть использованы в системах кабельного телевидения, где требуются высокоэффективные системы обработки информации с улучшенными техническими характеристиками. В статье представлены результаты исследований частотного модулятора. В связи с развитием радиоэлектронных устройств и применением их в системах оптического диапазона появилась необходимость в качественном улучшении технических характеристик. Использование эффективных систем обработки информации в радиотехнических системах позволяет повысить возможности систем телевидения, так как количество информации, которая передается в единицу времени прямо пропорционально полосе частот сигнала, позволяет также обеспечить высокую помехозащищенность каналов связи, повысить точность при оценке качества приема.

В системах кабельного телевидения широко используется метод передачи сигналов на поднесущих частотах. При данном способе используется двойная модуляция. Информационный электрический сигнал поступает на модулятор, где происходит первая операция: модуляция поднесущей частоты либо по амплитуде, либо по частоте или фазе, а затем промодулированная ω_n , поступает на оптический излучатель, где происходит вторая операция: модуляция интенсивности оптического излучения. На приемной стороне демодуляция происходит в обратном порядке. Метод использования поднесущей увеличивает отношение сигнал/шум и уменьшает нелинейные искажения при условии хорошей линейности характеристик модулятора и демодулятора [1–2].

Моделирование проводилось на основе экспериментальной установки, изображенной на рисунке 1.

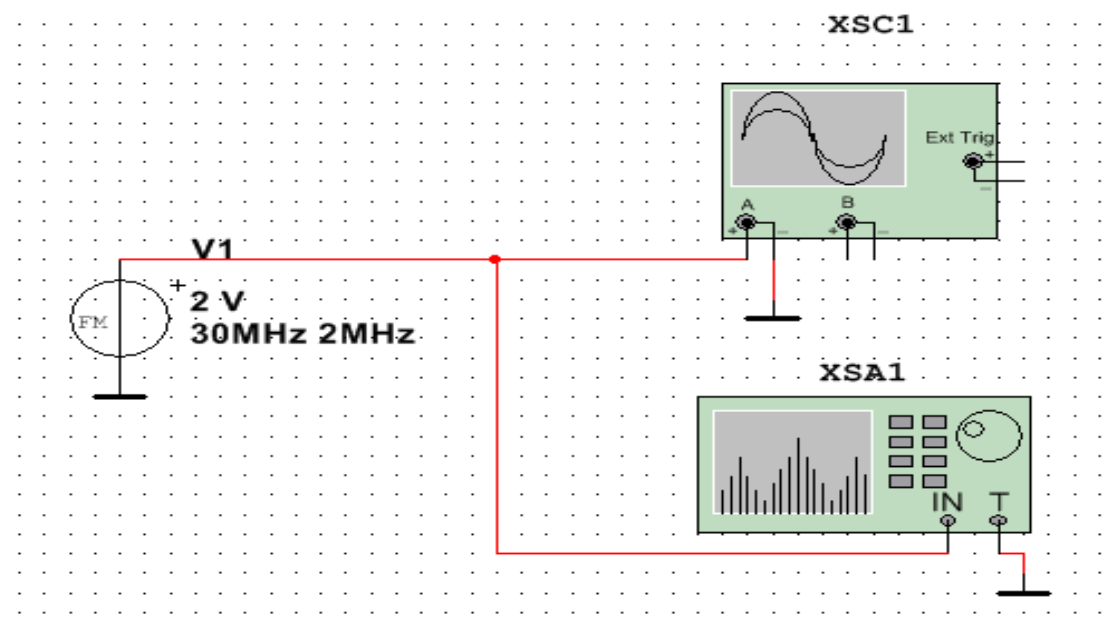


Рисунок 1. – Схема ЧМ модулятора:

V1 – частотный модулятор, XSA1 – анализатор спектра, XSC1 – осциллограф

В качестве источника ЧМ сигнала выбирается стандартный блок FM_VOLTAGE из библиотеки программы Multisim позволяющий измерять следующие характеристики сигналов:

- Амплитуду несущего колебания;
- Частоту несущего колебания;
- Индекс частотной модуляции;
- Частоту модуляции.

Для измерения характеристик исследуемых сигналов используется осциллограф XSC1 и анализатор спектра XSA1.

На рисунке 2 отображен график зависимости выходного напряжения $U_{\text{вых}}$ от несущей частоты f_0 .

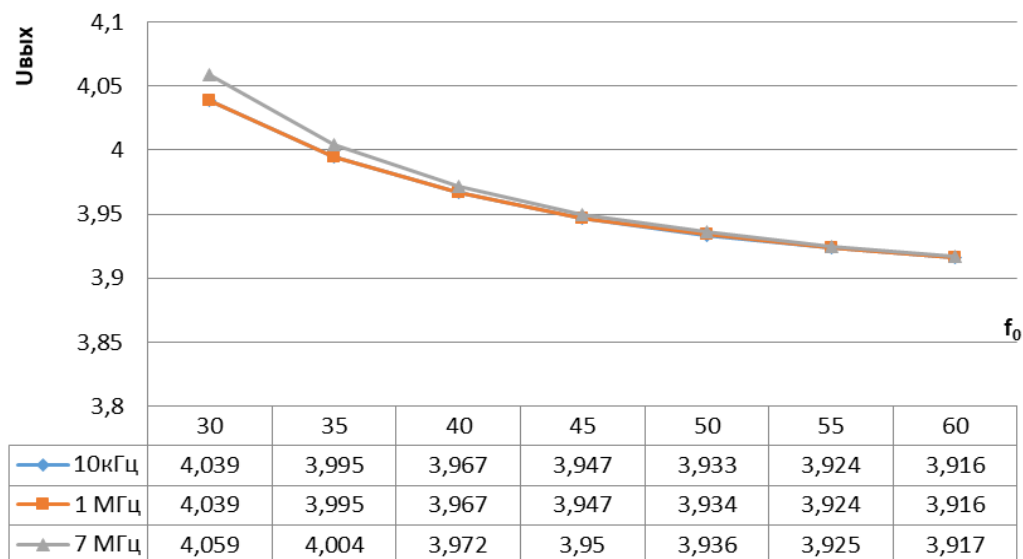


Рисунок 2. – Зависимости $U_{\text{вых}}$ от несущей частоты

Из представленных зависимостей видно, что уровень выходного сигнала уменьшается с ростом несущей частоты. Данные изменения не значительны, хотя при передаче видеосигналов на более высоких несущих частотах это будет отражаться на уровне передаваемого сигнала, особенно при большой протяженности оптического волокна. При измерении частоты модуляции уровень выходного сигнала изменяется незначительно. Установлено, что при частоте модуляции $F = 7\text{МГц}$ наблюдается некоторое увеличение сигнала по сравнению с частотами 10 кГц и 1 МГц [3].

Заключение. В статье представлены результаты моделирования параметров частотного модулятора. Результаты исследований свидетельствуют о повышении требований к каналу передачи данных и к средствам обработки информации.

ЛИТЕРАТУРА

1. Янушкевич, В.Ф. Устройства оптической обработки сигналов : метод. указания / В.Ф. Янушкевич. Новополоцк : ПГУ, 2002.
2. Оптические кабели [Электронный ресурс]. – Электронные данные. – Режим доступа: [http://ru.wikipedia.org/wiki/Оптические кабели](http://ru.wikipedia.org/wiki/Оптические_кабели). – Дата доступа: 25.08.2018.
3. Оптические кабели (ОК) [Электронный ресурс]. – Электронные данные. – Режим доступа: <http://victel.by/OK/>. – Дата доступа: 26.09.2018.

УДК 621.37:681.586

STM32CUBEMX В НАСТРОЙКЕ ПЕРИФЕРИИ МИКРОКОНТРОЛЛЕРОВ КОМПАНИИ ST

С.Ю. ЗМИТРОВИЧ

(Представлено: канд. физ.-мат. наук, доц. С.А. Вабищевич)

Рассматриваются основные аспекты программной среды, а также настройки возможной периферии микроконтроллеров компании ST [1] без написания программного блока на примере разработанного измерительного устройства с использованием микроконтроллера STM32F103C8T6.

Введение. Цель работы заключалась в проектировании микроконтроллерного устройства с программным обеспечением [2] для измерения аналоговых величин физических параметров с возможностью последующего импорта данных в другие программные продукты. Структурная схема проекта представлена на рисунке 1.

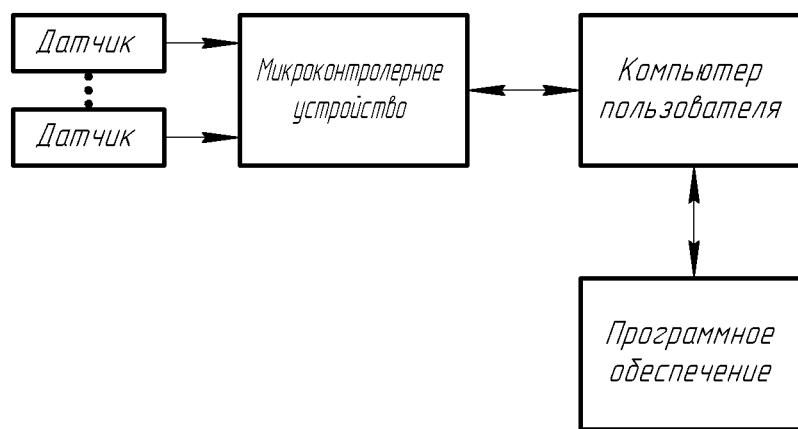


Рисунок 1. – Структурная схема разработанного проекта

Актуальность работы заключается в автоматизации, что позволяет повысить достоверность экспериментальных данных за счет снижения величины случайной погрешности, сократить время измерений, повысить эффективность использования датчиковой аппаратуры.

Одним из основных моментов при программировании микроконтроллерной системы является настройка периферийных блоков. Данный процесс, зачастую, занимает продолжительное время, особенно у начинающих программистов. STM32CubeMX [3] – программная среда, позволяющая при помощи соответствующих блоков и клавиатуры настраивать всю периферию микроконтроллера.

На рисунке 2 представлена среда разработки STM32CubeMX.

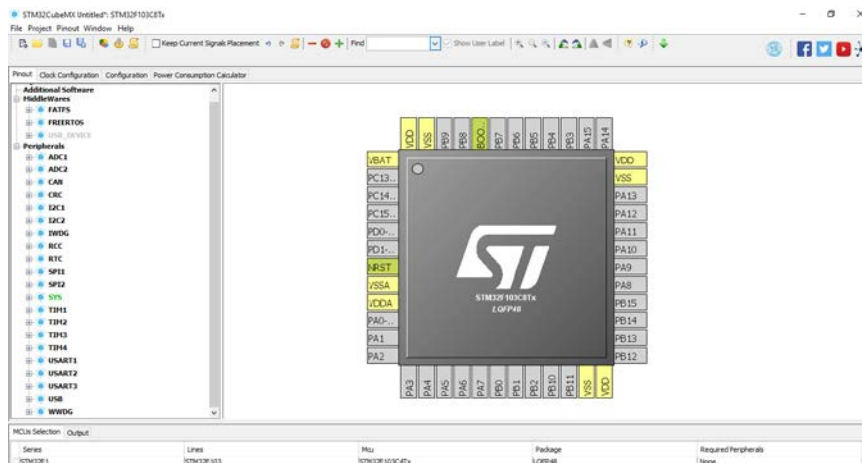
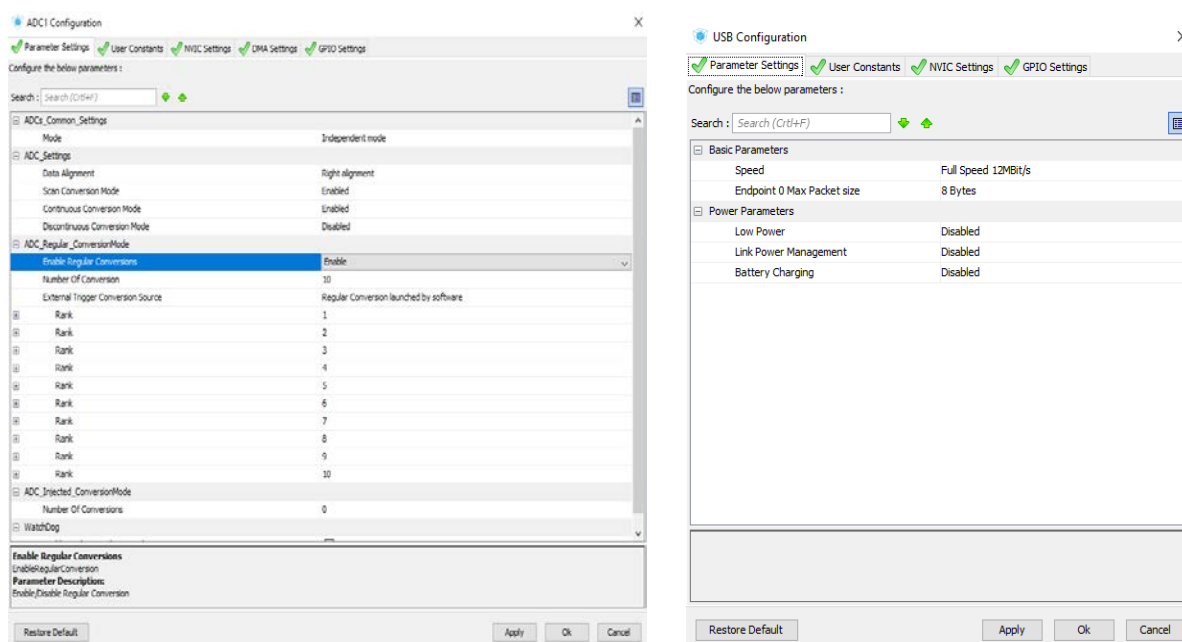


Рисунок 2. – Программная среда разработки STM32CubeMX

Настройка системы. В разработанном измерительном устройстве использованы аналого-цифровой преобразователь (АЦП) и USB модули микроконтроллера. Устройство использует все выводы АЦП модуля для проведения измерений. АЦП модуль работает в режиме DMA, то есть адресация к памяти происходит, минуя процессор микроконтроллера, что позволяет сократить время измерений. На рисунке 3а представлены настройки АЦП модуля микроконтроллера.

Аналого-цифровой преобразователь настроен в режиме «Continuous Conversion Mode». Данный режим позволяет производить измерения входных величин непрерывно, однако, в порядке очереди, которая организуется выставлением значения измерений в поле «Number Of Conversion» после выставления «Enable» в «Enable Regular Conversion». В появившемся поле «Rank» указана очередь, а также номер АЦП порта и скорость измерения входной величины в циклах. Уменьшение скорости измерения позволяет увеличивать выходной импеданс датчикового оборудования.

USB модуль микроконтроллера используется в режиме «Communication Device Class (Virtual Port COM)», данный режим позволяет создавать соединение с компьютером пользователя, избегая для этого написания специального программного обеспечения. Настройки USB модуля представлены на рисунке 3 б.



а)

б)

**Рисунок 3. – Настройки АЦП модуля (а),
настройки USB модуля микроконтроллера STM32F103C8T6 (б)**

Для обеспечения работоспособности системы необходимо настроить частоты тактирования микроконтроллера, что можно осуществить в окне «Clock Configuration», рисунок 4. Так как в настоящей системе используется внешний кварцевый резонатор, то предварительно в «Pinout»->«RCC» было выставлено использование внешнего высокоскоростного кварцевого резонатора «High Speed Clock (HSE)»->«Crystal/Ceramic Resonator». STM32CubeMX позволяет производить настройки тактирования на интуитивном уровне, так как всё представлено в графическом виде. Тактовая частота равная 8 МГц поступает на микроконтроллер с внешнего кварцевого резонатора (HSE), после чего в окошке «PLL» умножается на коэффициент равный 6 – это необходимо, так как частота работы USB составляет 48 МГц. После чего полученная частота разбивается коэффициентами деления на частоты необходимые для нормальной работы периферии микроконтроллера. Для АЦП модуля частота тактирования, в нашем случае, составляет 12 МГц.

После совершения всей конфигурации микроконтроллера можно начать генерировать исходный код программы. Данная возможность осуществляется путём нажатия «Project»->«Generate Code». Однако перед генерированием программного кода рекомендуется выбрать ядро в меню «Project»->«Settings», которое в дальнейшем будет использоваться для написания кода программы.

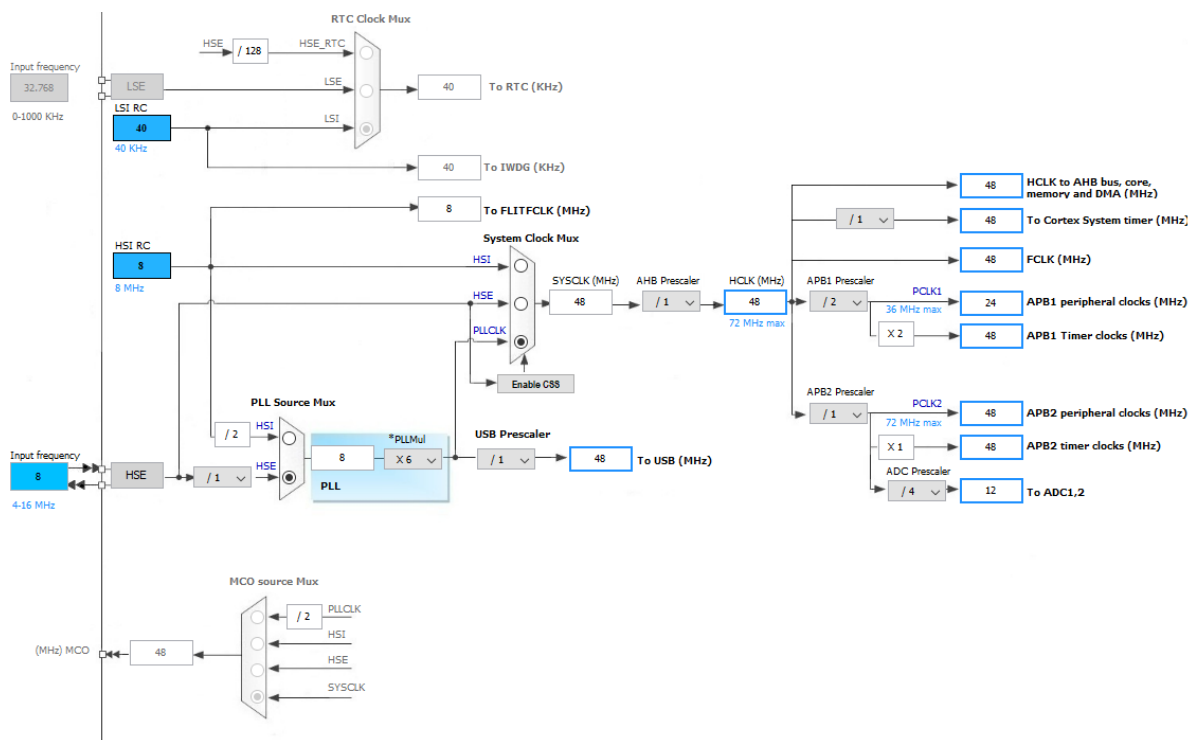


Рисунок 4. – Окно настройки тактирования микроконтроллера

Заключение. В данной статье рассмотрено программное обеспечение графической конфигурации STMCubeMX, а также его использование для конфигурирования периферии микроконтроллера на примере измерительного устройства, использующего АЦП и USB модули.

ЛИТЕРАТУРА

1. STMicroelectronics [Электронный ресурс] // About ST – Режим доступа: https://www.st.com/content/st_com/en/about/st_company_information/who-we-are.html.
2. Змитрович, С.Ю. Обработка аналоговых сигналов датчиковой аппаратуры. Информационно-коммуникационные технологии: достижения, проблемы, инновации (ИКТ-2018) [Электронный ресурс] : электронный сборник статей I международной научно-практической конференции, посвященной 50-летию Полоцкого государственного университета, Новополоцк, 14–15 июня 2018 г. /, С.Ю. Змитрович, С.А. Вабищевич ; Полоцкий государственный университет. – Новополоцк, 2018. – 1 электрон. опт. диск (CD-ROM). – С. 272–275.
3. STMicroelectronics [Электронный ресурс] // STM32CubeMX for STM32 configuration and initialization C code generation. – Режим доступа: <https://clk.ru/ERtM3>.

УДК 004.622:004.514

**ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ АНАЛИЗА АНАЛОГОВЫХ СИГНАЛОВ
ФИЗИЧЕСКИХ ВЕЛИЧИН****С.Ю. ЗМИТРОВИЧ***(Представлено: канд. физ.-мат. наук, доц. С.А. ВАБИЩЕВИЧ)*

Рассматривается использование спроектированного программного обеспечения (ПО) для снятия и анализа поступающих на входы микроконтроллерного устройства, сигналов с последующей их конвертацией в другие ПО для дальнейшей обработки экспериментальных результатов измерений.

Введение. Целью разработки программного обеспечения для измерительного устройства аналоговых величин являлось предоставление наглядности изменения величин от времени или других параметров. При этом основной задачей являлись разработка ПО с возможностью обмена данными с микроконтроллерной системой, а также построение графиков на основе проделанных измерений с последующим импортом данных в другие программные среды.

Актуальность проектирования данного приложения заключается в автоматизации производимых измерений и построений сопутствующих графиков изменений величин.

Измерение параметров физических величин может происходить с использованием различных приборов, включая датчиковую аппаратуру [1]. Информация, поступающая от датчиков, зачастую, имеет аналоговую форму, что в электронике представлено в виде напряжения, которое в свою очередь обрабатывается микроконтроллерным устройством и передаётся на компьютер пользователя (ПК).

Программное обеспечение. На рисунке 1 представлено разработанное программное обеспечение для осуществления замеров физических параметров.

Основным полем данного ПО является поле графиков, коих количество может достигать 10. Данное поле используется для отображения всех произведённых измерений.

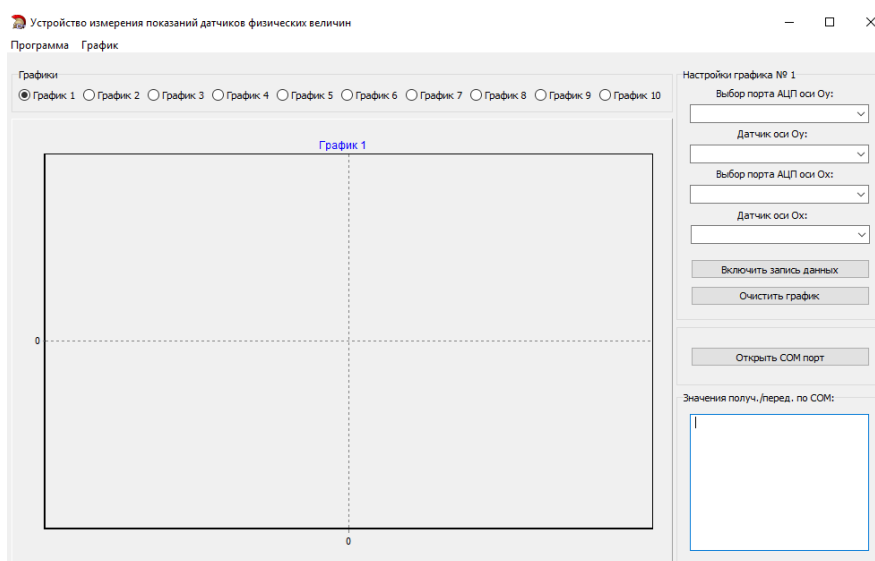


Рисунок 1. – Основное окно программного обеспечения

В правом углу приложения расположено поле «Настройки графика №*», в котором можно произвести его конфигурацию. Выпадающие списки «Выбор порта АЦП оси O*» соответствующих осей графиков предназначены для определения порта АЦП модуля микроконтроллерного устройства с которого будут происходить измерения. Списки «Датчик по оси O*» необходимы для определения измеряемых аналоговых величин. Кнопкой «Включить запись данных» пользователь указывает приложению, что данный график готов к отображению измеряемых параметров. Кнопка «Открыть СОМ Порт» позволяет ПО начать обмен данными с измеряющим устройством с последующим их отображением в поле графиков.

Для каждого из графиков существует возможность изменения параметров: дискретности отображения точек; использования двух сенсоров; цвета линии; ширины линии; логарифмического вида. Произвести данные изменения можно пройдя по пути «Программа-Параметры», рисунок 2а.

Приложение позволяет выводить измеренную информацию в трёх форматах: *txt*; *bmp*; *excel*. На рисунках 3 и 4 представлены графики зависимости сопротивления от температуры термистора B57164K 222-J в форматах *bmp* и *excel* соответственно.

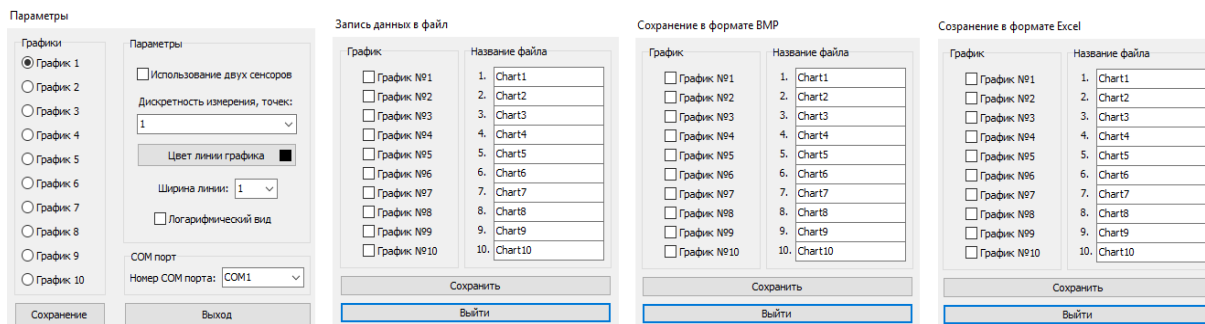


Рисунок 2 – а) окно параметров графиков; б) окно сохранения в *txt* формате; в) окно сохранения в *bmp* формате; г) окно сохранения в *excel* формате

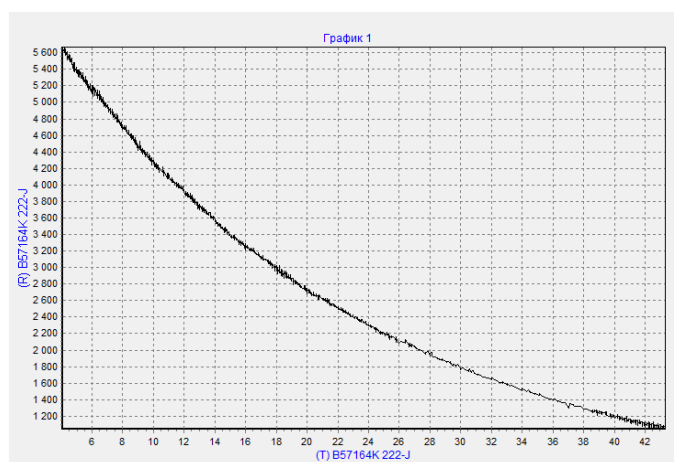


Рисунок 3 – График зависимости сопротивления от температуры термистора B57164K 222-J, представленный в формате *bmp*

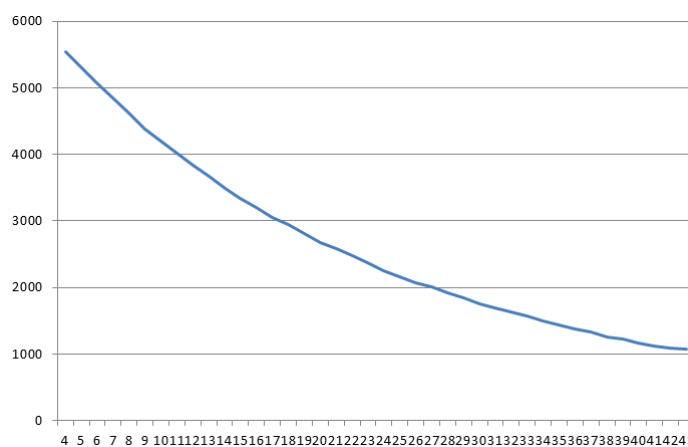


Рисунок 4 – график зависимости сопротивления от температуры термистора B57164K 222-J, представленный в формате *excel*

Заключение. Разработанное программное обеспечение позволяет совместно с измеряющим микроконтроллерным устройством производить замеры параметров физических величин, с последующим построением соответствующих графиков. Приложение позволяет выводить полученные данные в трёх форматах: *bmp*, *excel*, *txt*. Данное программное обеспечение возможно использовать как в образовательной, так и в научно-исследовательской деятельности.

ЛИТЕРАТУРА

1. Змитрович, С.Ю. Обработка аналоговых сигналов датчиковой аппаратуры. Информационно-коммуникационные технологии: достижения, проблемы, инновации (ИКТ-2018) [Электронный ресурс] : электронный сборник статей I международной научно-практической конференции, посвященной 50-летию Полоцкого государственного университета, Новополоцк, 14–15 июня 2018 г. /, С.Ю., Змитрович, С.А. Вабишевич ; Полоцкий государственный университет. – Новополоцк, 2018. – 1 электрон. опт. диск (CD-ROM). – С. 272–275.

УДК 621.371.39

ФИЛЬТРАЦИЯ СПЕКЛ-ШУМА В СИСТЕМАХ ДИСТАНЦИОННОГО ЗОНДИРОВАНИЯ ЗЕМЛИ

А.О. СТАНОВОЙ

(Представлено: канд. техн. наук, доц. В.Ф. ЯНУШКЕВИЧ)

Представлены результаты фильтрации спекл-шума. Наличие спекл – шума приводит к ухудшению дешифрируемости радиолокационного изображения, повышению требований к каналу передачи данных, и средствах отображения информации, а так же радиометрического разрешения и точности измерения компонентов радиолокационного портрета объекта.

Радиолокационные системы землеобзора космического базирования являются эффективным средством получения оперативной и долговременной информации о состоянии и динамике объектов и районов земного шара в глобальных и региональных масштабах независимо от метеорологических условий и времени суток.

Научные и инженерные основы для реализации таких систем заложены в середине прошлого века. Они были подготовлены достижениями в области самолетного приборостроения, в том числе разработкой в НИИ – 17 панорамной РЛС «Кобальт» для бортового комплекса прицельного оборудования «Рубидий» бомбардировщика Ту – 4 (1949). С помощью этой аппаратуры решались задачи навигации и прицеливания по крупным площадным объектам в отсутствии оптической видимости [1].

Совершенствование аппаратуры (внедрение перестройки частоты, цветной индикации, автоматического обнаружения объектов на земной поверхности и др.), а главное, переход от панорамных РЛС к радиолокаторам бокового обзора (РБО) с улучшенным угловым разрешением по азимуту (до 9...15 угл. мин), способствовало развитию нового применения радиолокации – землеобзора.

Интенсивные исследования по совершенствованию радиолокаторов землеобзора привели к созданию принципиально новых средств радиолокационного наблюдения – когерентных радиолокаторов с синтезированной апертурой антенны (РСА), способные решать задачи радиовидения с пространственным разрешением в единицы и доли метра [2].

Исторически сложилось, что космические РСА первого поколения имели разрешение по горизонтальной дальности около $\rho_r \approx 30$ м, более грубое, чем предельное разрешение $\rho_x \approx 5$ м, обусловленное горизонтальным размером используемых антенн ($D_{\text{ант}} \approx 10$ м). Для синтеза РЛИ применяли оптические устройства обработки сигналов с выбором апертуры синтеза так, чтобы иметь одинаковое разрешение по обеим координатам. Имеющий запас длительности сигнала использовали для некогерентного накопления.

Метод, при котором в одном элементе разрешения синтезированной ДНА усреднялись РЛИ, полученные на разных доплеровских частотах (внутриэлементное некогерентное накопление), получил в зарубежной литературе название «Multilook». На рисунке 1 представлена структурная схема внутриэлементного некогерентного накопления. При цифровом синтезе РЛИ реализация внутриэлементного накопления обеспечивается путем разделения сигнала на субапертуры, когерентной обработки (синтез парциальных КРЛИ, полученных на разных доплеровских частотах с разными локальными углами скоса), детектирования и последующего суммирования парциальных РЛИ. Вместо временного разделения сигнала можно применить разделение азимутального спектра доплеровских частот на субспектры с последующим их синтезом, детектированием и суммированием парциальных РЛИ. Такую же операцию некогерентного накопления можно выполнять и по дальности. Принципиально число наблюдений не обязательно должно быть целым, например, при оптическом накоплении или при перекрывающихся субапертурах (субспектрах) [3–5].



Рисунок 1. – Структурная схема внутриэлементного некогерентного накопления

В качестве шума принимаем сигнал, показанный на рисунке 2:

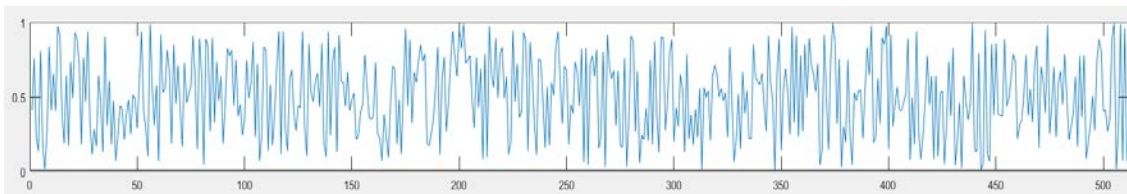


Рисунок 2. – Шум

В качестве входного сигнала будет 1-ая строка матрицы размером 512×512 показанная на рисунке 3:

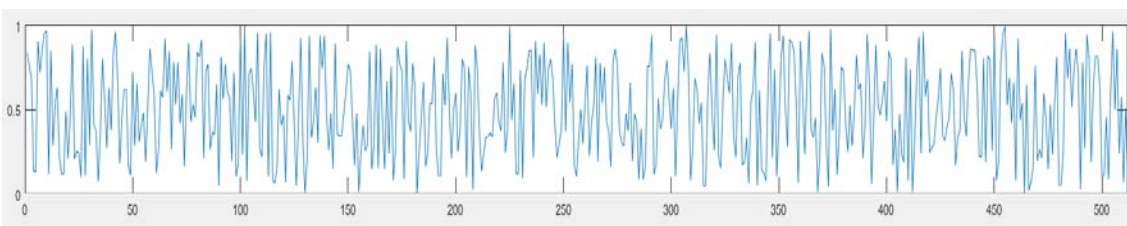


Рисунок 3. – 1-ая строка матрицы

Результатом перемножения строки матрицы на шум показан на рисунке 4:

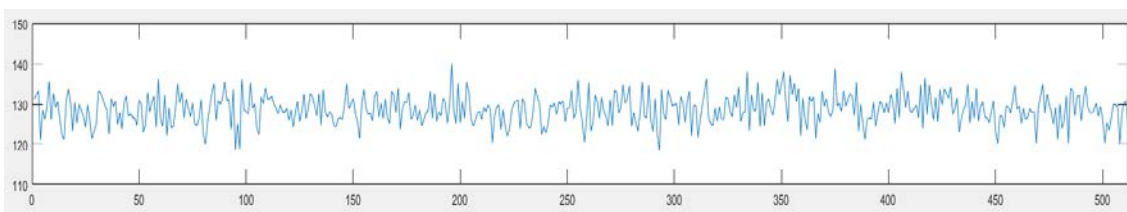


Рисунок 4. – Результат перемножения

В качестве фильтра у нас будет алгоритм скользящего среднего. Результаты представлены на рисунке 5:

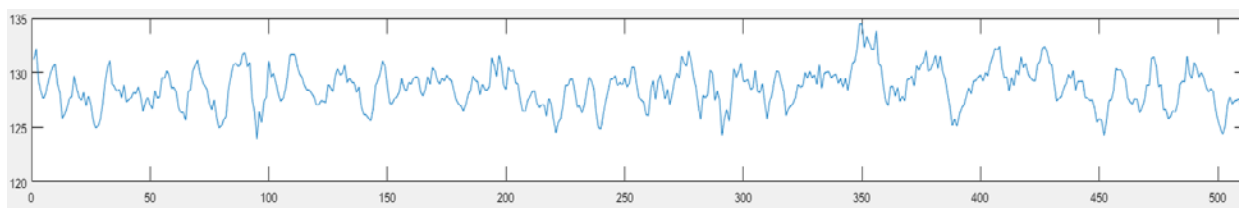


Рисунок 5. – Результат фильтрации

Проведенный анализ показал, что применение фильтрации спекл-шумов ведет к качественному улучшению технических характеристик систем обработки информации.

ЛИТЕРАТУРА

1. Радиолокационные системы землеобзора космического базирования / под ред. В.С. Вербы. – М. : Радиотехника, 2010. – 680 с. : ил.
2. Арманд, Н.А. Современные спутниковые РСА системы для дистанционного зондирования земли: Достижения и перспективы / Н.А. Арманд, А.И. Захаров // Труды конференции по ДЗЗ. – Муром, 2003.
3. Белоруков, А.А. Методы сглаживания спекл – шума на радиолокационных изображениях земной поверхности / А.А. Белоруков // Зарубежная радиоэлектроника. – 1990. – № 6. – С. 26–35.

4. Радиолокационные станции с цифровым синтезированием апертуры антенны / под ред. В. Т. Горянова. – М. : Радио и связь, 1988. – 304 с.
5. Неронский, Л.Б. Перспективы совершенствования космических средств радиолокационного наблюдения земной поверхности // Научно-технические технологии. – № 8–9. – С. 66–87.
6. «Венера – 15», «Венера – 16»: Радиолокатор с синтезированной апертурой на орбите ИСВ // Изв. ВУЗов. Сер. «Радиофизика». 1985, том XXVIII. № 3. С.259 – 274.
7. Буренин, Н.И. Радиолокационные станции с синтезированной антенной. – М. : Сов. Радио, 1972. – 160 с.
8. Гарбук, С.В. Космические системы дистанционного зондирования Земли. – М. : Сканэкс, 1997. – 296 с.
9. Елизаветин, И.В. Оценка влияния некогерентного накопления при обработке данных с космического аппарата // Исследование Земли из космоса. – 1993. – № 1. – С. 32–35.

УДК 621.371.39

АНАЛИЗ БАЗОВЫХ АЛГОРИТМОВ ФИЛЬТРАЦИИ СПЕКЛ – ШУМА

А.О. СТАНОВОЙ

(Представлено: канд. техн. наук, доц. В.Ф. ЯНУШКЕВИЧ)

Представлен анализ фильтрации фильтрации спекл-шума. В ходе многолетней эксплуатации космических РСА и практического использования материалов радиолокационной съемки подтверждена правильность подходов к проблеме фильтрации спекл – шума, получена сравнительная оценка этих алгоритмов и заложена база для разработки новых алгоритмов с использованием современных вычислительных методов обработки радиолокационной информации РСА нового поколения.

Широкое применение алгоритмов фильтрации спекл – шума относится к 1980 – 1990 гг. Основная цель, решаемая при фильтрации спекл – шума – повышение точности измерения удельная эффективная площадь рассеяния пространственно распределительных объектов с сохранением информации, содержащейся в текстурных признаках, характеризующих отраженный сигнал, гистограммах его распределения, включая возможность поляриметрической обработки. В публикации последних лет рассмотрены пути применения методов вейвлет-фильтрации, фрактального анализа, нейронных сетей [1].

Разработка математического аппарата адаптивной фильтрации спекл – шума основана на моделях, аппроксимирующих законы распределения отраженного сигнала от наблюдаемой поверхности в зависимости от ее характера: распределения Рэлея Вейбулла, К-распределение. В ряде случаев используют непараметрический подход, не требующий точного знания закона распределения [2].

Наиболее часто для аналитического описания исходного РЛИ рассматривают ряд моделей взаимодействия сигнала и шума. Самая простая из них – модель мультипликативного шума

$$U(x, y) = S(x, y)n(x, y) \quad (1)$$

где x, y – координаты пикселя на РЛИ; $S(x, y)$ – стационарная составляющая (текстура), значение которой может быть найдено при усреднении сигналов по ансамблю; $n(x, y)$ – случайная составляющая, не зависящая от сигнала и определяемая мгновенным распределением фаз элементарных отражателей в элементе разрешения при конкретном ракурсе и углу места наблюдения.

Следующая по сложности – это модель аддитивного шума, модулируемого сигналом

$$U(x, y) = S(x, y) + f\{S(x, y)\}n(x, y) \quad (2)$$

где $f\{S(x, y)\}$ в общем случае, нелинейная функция.

Модификация модели (1) для описания нестационарных РЛИ – мультипликативная модель с аддитивным средним

$$U(x, y) = S(x, y) + [q + r(x, y)] \quad (3)$$

где q – среднее значение помехи; $r(x, y)$ – стационарный эргодический процесс с нулевым средним.

Как показывается анализ материалов радиолокационной съемки с получением РЛИ, значительно отличающихся по пространственному разрешению – от единиц метров, во многих случаях достаточно применение простейшей модели (1).

Локальные адаптивные фильтры основаны на формировании локальных статистик относительно центрального пикселя в окне обработки и принятии решения о выборе формулы для расчета центрального элемента. Алгоритм ЛИ исходит из мультипликативной модели спекл – шума (1). Выходной отчет ЭРЛИ в центральном пикселе окна W вычисляются по формуле

$$E_N = E_K W + E_W (1 - W) \quad (4)$$

где E_K – отсчет ЭРЛИ в центральном пикселе окна $k = (m, n)$; $E_W = \text{mean}\{E_K\}_W$ – среднее значение отсчетов по окну с центром в k ; $W = 1 - ENL/N$ – весовая функция.

В результате имеем

$$E_N = E_K \left(1 - \frac{ENL}{N}\right) + E_W \frac{ENL}{N} \quad (5)$$

Таким образом, когда попадающий в окно участок местности однороден $ENL \geq N$, фильтр ЛИ работает как простой сглаживающий фильтр. В противном случае он вычисляет взвешенную сумму ре-

ального значение и среднего значения в соответствии с (4). Условие $ENL \geq N$ является критерием однородной поверхности.

Фильтр Куана предполагает преобразование мультипликативного шума к аддитивной модели, зависящей от сигнала $U(x,y)$ (2). К этой модели применяют критерий минимума среднеквадратичной ошибки. Формула для вычисления выходных отсчетов совпадает с алгоритмом ЛИ (4), но весовая функция имеет отличие

$$W = 1 - ENL/(N + 1) \quad (6)$$

В результате, для однородной поверхности, когда $ENL \approx N$, в каждом пикселе выходного РЛИ имеем полусумму отсчетов исходного РЛИ в данном пикселе и усредненного значения по окну. Если неравномерность отсчетов по окну велика, то в выходные РЛИ записывают значения исходного РЛИ [3].

ЛИТЕРАТУРА

1. Радиолокационные системы землеобзора космического базирования / под ред. В.С. Вербы. – М. : Радиотехника, 2010. – 680 с. : ил.
2. Белокуров, А.А. Методы сглаживания спекл – шума на радиолокационных изображениях земной поверхности / А.А. Белокуров // Зарубежная радиоэлектроника. – 1990. – №6. – С. 26–35.
3. Арманд, Н.А. Современные спутниковые РСА системы для дистанционного зондирования земли: Достижения и перспективы / Н.А. Арманд, А.И. Захаров // Труды конференции по ДЗЗ. – Муром, 2003.

УДК 537.86

ПЕРЧАТКА С ДАТЧИКАМИ ИЗГИБА

Е.С. ЦАРЬКОВА

(Представлено: канд. техн. наук, доц. О.Н. ПЕТРОВИЧ)

Представлено применение датчика изгиба в помощи немым людям. Показан метод самостоятельного изготовления датчика изгиба, отображены преимущества и недостатки разных видов датчиков, также приведён примерный расчет стоимости изделия. Показан алгоритм работы перчатки, коды.

Введение. Количество глухих и слабослышащих людей, говорящих на жестовых языках, оценивается в десятки миллионов человек. Общение со слышащими людьми представляет для них большую проблему.

Немота – утрата способности говорить, обусловленная главным образом врожденной или возникшей в раннем детстве глухотой. При сохранном слухе связана с поражением речевого аппарата или центральной нервной системы [1].

Язык немых – совсем не то же самое, что жестикуляция, которой пытаются объяснить люди с иностранцами. В этом случае жестовая речь бедна и узко функциональна, в то время как лишенным возможности говорить нужен богатый лексикон, способный передать и художественные образы, и математические термины. Первый язык жестов возник еще в 18 веке: Германия и Франция открыли сурдопедагогические центры. Базой невербальной речи стали естественные жесты, самопроизвольно возникшие в местечковых сообществах глухих. В России первый центр был создан в 1806 году, в городе Павловске. В нем использовался опыт французских сурдопедагогов; московская школа, открытая на полвека позже, ориентировалась на наработки немцев. В результате современная российская сурдопедагогика представляет собой симбиоз этих двух школ [2].

Перчатка-сурдопереводчик. Существует несколько путей для помощи коммуникации людей с немотой:

1. Переводчик человек

Преимущества: воспринимает язык жестов и может отвечать на языке жестов.

Недостатки: постоянно нужен человек, со знанием языка жестов.

2. Чтение по губам

Преимущества: не нужен человек.

Недостатки: собеседник не всегда находится в зоне видимости, возможно только воспринимать речь, нет обратной связи.

3. Проблему обратной связи может решить перчатка-сурдопереводчик.

На каждый палец перчатки прикреплен датчик изгиба. Они присоединены к контроллеру Arduino UNO (рис. 1). Перчатка преобразует жесты в текст.

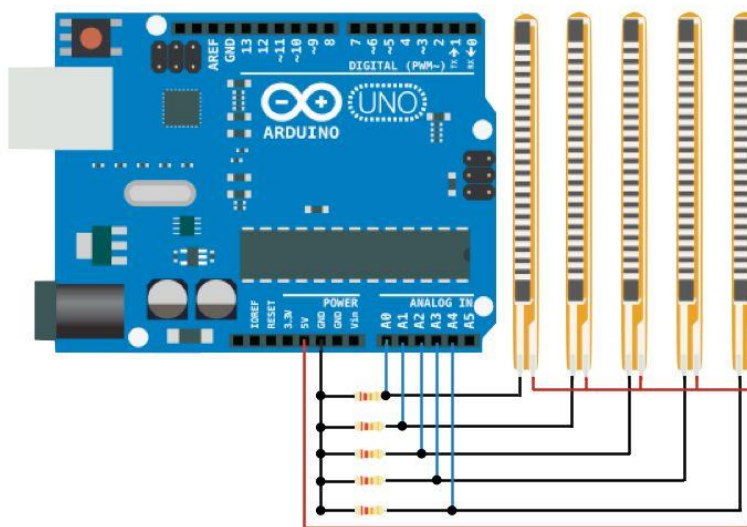


Рисунок 1. – Схема сборки перчатки

Датчик изгиба – тонкая длинная резистивная полоска. Такой датчик меняет свое сопротивление в зависимости от величины изгиба. То есть преобразует изменение механической структуры в электрическое сопротивление при этом чем больше изгиб, тем больше значение сопротивления. Такие датчики являются аналоговыми резисторами, которые работают в составе аналоговых делителей напряжения.

Для изготовления перчатки можно использовать следующие виды датчиков:

1. Готовый датчик

Плюсы – точность.

Минусы – цена.

2. Датчик, изготовленный самостоятельно при помощи датчиков изгиба.

Плюсы: точность, цена.

Минусы: сложность конструкции.

Алгоритм работы перчатки:

1. Перевод языка жестов в цифровой код.

2. Перевод кода в текст.

3. Отображение текста.

Существует несколько способов отображения текста:

1. Телефон – телефон. Но в данном способе есть существенный недостаток: низкая скорость общения.

2. Дисплей. Данный способ информации возможен при применении перчатки.

Процесс разработки

1. Собрать схему (рисунок 1).

2. Определить значения датчиков. Для этого нужно:

1) Загрузить скетч 1 (приложение 1) в контроллер Arduino UNO.

2) Открыть монитор последовательного порта.

3) Узнать значения датчиков для каждой буквы (из русской дактильной азбуки).

3. Добавить значение датчиков для каждой отдельной буквы в скетч 2 (приложение 2).

4. Таким образом мы получаем исходный код (приложение 3).

Необходимые детали

- Перчатка
- Контроллер Arduino UNO
- 5 резисторов на 22 кОм
- 5 датчиков изгиба (можно изготовить самостоятельно)
- Соединительные провода
- USB провод для подключения к компьютеру
- Макетная плата

Изготовление датчика изгиба

Стоимость датчика сопротивления в среднем составляет 12\$, т.к. для данного проекта необходимо 5 датчиков, то итоговая стоимость датчиков составит 60\$. Но такой датчик можно изготовить самостоятельно. Наименование деталей и средняя стоимость представлены в таблице 1. Принцип его работы основан на фоторезисторе и светодиоде. Свет от светодиода будет поступать на фоторезистор по силиконовой трубке, а при её изгибе свет будет падать в меньшем количестве, а значит и сопротивление будет меняться на выходе у фоторезистора.

Таблица 1. – Расчёт стоимости датчиков

Название детали	Количество	Цена, \$	Итоговая стоимость\$
Фоторезисторы	5	0,356	1,78
Светодиоды	5	0,436	2,18
Силиконовая трубка	1	0,09	0,09
Резистор на 10 кОм	5	0,03	0,15
Резистор на 220 Ом	5	0,03	0,15
Изолента	1	0,825	0,825
Итого		1,767	5,175

Таким образом экономия составляет 54,825\$

Сборка датчика

1) Отрезать кусок силиконовой трубки необходимой длины.

2) С одной стороны вставить фоторезистор, а с другой светодиод.

- 3) Обмотать изолянтной (желательно черной) трубку.
- 4) Припаять к светодиоиду токоограничивающий резистор (220 Ом).
- 5) К фоторезистору припаять подтягивающий резистор (1 кОм).
- 6) Таким образом изготовить ещё 4 датчика.

Преимущества:

- Низкая стоимость
- Простая сборка
- Доступность расходных материалов
- Возможность изготовления датчиков конкретного размера

Недостатки:

- Слабая механическая стойкость
- Неидеальная линейность показаний датчика

Приложение 1

```
#define F1 A0 // значение на первом пальце (F-finger)
#define F2 A1
#define F3 A2
#define F4 A3
#define F5 A4
void setup()
{
  pinMode(F1, INPUT);
  pinMode(F2, INPUT);
  pinMode(F3, INPUT);
  pinMode(F4, INPUT);
  pinMode(F5, INPUT);
  Serial.begin(9600);
}
void loop()
{
  int F1_value, F2_value, F3_value, F4_value, F5_value;
  F1_value = analogRead(F1);
  F2_value = analogRead(F2);
  F3_value = analogRead(F3);
  F4_value = analogRead(F4);
  F5_value = analogRead(F5);
  Serial.print(" F1:");
  Serial.print("\t");
  Serial.print(F1_value);
  Serial.print(" F2:");
  Serial.print("\t");
  Serial.print(F2_value);
  Serial.print(" F3:");
  Serial.print("\t");
  Serial.print(F3_value);
  Serial.print(" F4:");
  Serial.print("\t");
  Serial.print(F4_value);
  Serial.print(" F5:");
  Serial.print("\t");
  Serial.print(F5_value);
  Serial.println("");

  delay(10000);
}
```

Приложение 2

```
#define F1 A0 // значение на первом пальце (F-finger)
#define F2 A1
#define F3 A2
#define F4 A3
```



```
#define F5 A4
void setup()
{
  pinMode(F1, INPUT);
  pinMode(F2, INPUT);
  pinMode(F3, INPUT);
  pinMode(F4, INPUT);
  pinMode(F5, INPUT);
  Serial.begin(9600);
}
void loop()
{
  int F1_value, F2_value, F3_value, F4_value, F5_value;
  F1_value = analogRead(F1);
  F2_value = analogRead(F2);
  F3_value = analogRead(F3);
  F4_value = analogRead(F4);
  F5_value = analogRead(F5);
  //s
  if(F1_value>900 & F1_value<950){
  if(F2_value>26 & F2_value<89){
  if(F3_value>250 & F3_value<420){
  if(F4_value>268 & F3_value<480){
  if(F5_value>295 & F5_value<362){
  Serial.print("s");
  }
  }
  }
  }
  }

  delay(2000);
}
```

Вывод. В данной статье изучены методы коммуникации людей с немотой, предложены разнообразные методы изготовления перчатки с датчиками изгиба. Рассчитана приблизительная стоимость изготовления датчиков, необходимое для изготовления перчатки. Представлена программная часть перчатки

ЛИТЕРАТУРА

1. Современная энциклопедия – Аванта+, 2000 – 640 с.
2. Немые люди: причины безмолвия. Язык немых [Электронный ресурс]. – Режим доступа: <http://fb.ru/article/254410/nemyie-lyudi-prichinyi-bezmolviya-yazyik-nemyih>. – Дата доступа: 29.06.2016.

УДК 621.311.4-52:

**ПРИМЕНЕНИЕ МИКРОПРОЦЕССОРОВ
ДЛЯ СИСТЕМ РЕЛЕЙНОЙ ЗАЩИТЫ И АВТОМАТИКИ****В.В. ЛОВКИС***(Представлено: канд. техн. наук, доц. В.Е. ПИТОЛИН)*

Рассмотрены принципы построения систем релейной защиты и автоматики для автоматизированных подстанций на основе микропроцессорных устройств, а также недостатки и преимущества этих систем.

Для повышения эффективности управления системой распределения электрической энергии используются современные измерительные системы, каналы связи, средства передачи, обработки и отображения информации, объединенные в автоматизированную систему диспетчерского управления (АСДУ) [1].

Электрические сети (ЭС) являются важным звеном в системе производства и потребления электроэнергии. Устройства автоматизации с использованием микропроцессорной техники с каждым годом все больше внедряются в различных отраслях промышленности, в том числе в энергетике. Это позволяет существенно повысить качество регулирования производственных процессов, снизить стоимость оборудования систем автоматики.

Автоматизация энергетических систем, выполняющая информационное, алгоритмическое и машинное обеспечение включает и такую область, как релейная защита. Релейная защита – это основной вид электротехнической автоматики, поддерживающей процесс контроля состояния всех элементов электроэнергетической системы. Правильная настройка релейной защиты и противоаварийной автоматики играет важную роль в обеспечении надежной работы электрических сетей.

В прошлом столетии релейная защита строилась на базе электромеханических реле. Современные системы автоматизации и средства релейной защиты могут выполняться с использованием программируемых микроконтроллеров. Обычно микроконтроллер строится на базе выбранного микропроцессора. Использование микропроцессоров в устройствах релейной защиты в настоящее время находит все большее распространение. Можно утверждать, что существование электроэнергетики в будущем невозможно без микропроцессорных устройств релейной защиты. Действительно, никаких новых функций защита на базе микропроцессоров не принесли, не изменилась логика работы защиты и принципы передачи электроэнергии. Однако развитие электромеханических реле было признано экстенсивным и остановлено около тридцати. Все усилия разработчиков были сосредоточены на создании электронных, а позднее и микропроцессорных защит. Это объяснялось меньшими затратами на производство микропроцессорных защит благодаря возможности автоматизации их производства (по сравнению с ручной сборкой электромеханических реле), учитывая то, что стоимость продажи микропроцессорных защит не стала ниже из-за сложности их программирования и наладки.

К средствам релейной защиты и автоматики линий электропередачи (ЛЭП) с напряжением 110–750 кВ предъявляются следующие требования:

- она должна быть быстродействующей, так как во время коротких замыканий (КЗ) интенсивны переходные процессы на длинных линиях высокого напряжения;
- все виды повреждений на защищаемой высоковольтной линии, как в полнофазном режиме ее работы, так и в режиме работы двух фаз в цикле однофазного автоматического повторного включения (ОАПВ) должны быть отключены максимально быстро [2].

Правильная настройка релейной защиты и противоаварийной автоматики играет ответственную роль в обеспечении надежной работы электрических сетей. В последние годы большое распространение получили микропроцессорные устройства релейной защиты. Так, на многих подстанциях электроэнергетической системы Белоруссии вследствие реконструкции электромеханические реле были заменены на микропроцессорные.

Особенность воздушных линий (ВЛ) электропередачи заключается в том, что они имеют огромную протяженность, поэтому в большей степени подвержены повреждениям, чем другое электрическое оборудование. Более того, ВЛ подвержены внешним повреждениям от грозных ударов, образования гололеда, сильного ветра, налипания грязи на изоляторах и т.п.

Для защиты линий с односторонним питанием могут быть применены: токовая отсечка; максимальная токовая защита; направленная токовая поперечная защита; дифференциальная защита параллельных линий; токовая поперечная дифференциальная защита параллельных линий. Для линий с дву-

сторонним питанием, кроме уже перечисленных, применяются следующие защиты: направленная отсечка; максимальная направленная защита; продольная дифференциальная защита; дистанционная защита; высокочастотная защита.

В основе работы микропроцессорных систем лежат принципы цифровых измерений. Они позволяют достичь более высокой точности вычислений, надежности и термостабильности. Благодаря этому можно использовать более сложные адаптивные алгоритмы, которые не осуществимы аналоговой техникой в механическом исполнении. В результате микропроцессорные устройства защиты дают возможность выполнить функциональный и тестовый контроль, интегрированное самонаблюдение (мониторинг), охватывающее аналоговые входы, микропроцессорную систему и выходные реле, дружественный человеко-машинный интерфейс и интеграцию функций защиты, автоматики и АСУ ТП любых энергообъектов.

Алгоритмы работы, программы и сам процесс регулировки записываются в память цифрового устройства. Также компьютерные программы используются для дистанционной настройки и обслуживания микропроцессорной системы.

Микропроцессорные реле имеют также свои недостатки: малую помехоустойчивость и маломощный выходной сигнал. Последнее приводит к необходимости применения усилителей и силовых реле.

Несмотря на то, что цифровые устройства защиты имеют различные назначения, они все имеют общую структурную схему. Структурная схема таких устройств представлена на рисунке 1.

Микропроцессорная система является центральным узлом, который с помощью устройств обменивается информацией с периферийными узлами. Эти дополнительные узлы осуществляют взаимодействие микропроцессора с внешней средой, т.е. объектом управления, оператором, датчиками исходной информации и т.д.

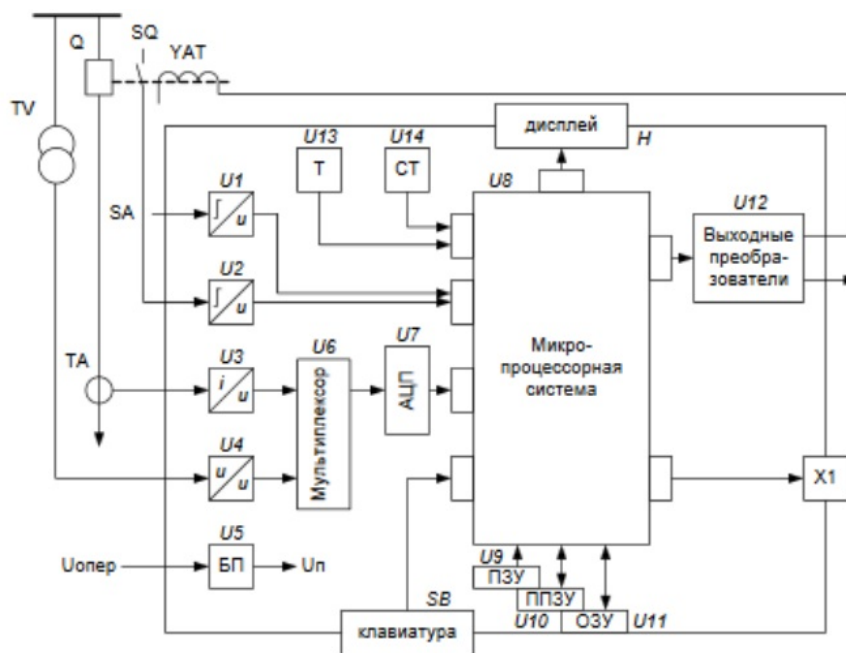


Рисунок 1. – Общая структурная схема цифровых устройств защиты

Основные узлы цифрового устройства защиты, входящие в эту схему следующие:

- входные и выходные преобразователи сигналов;
- тракт аналого-цифрового преобразования;
- микропроцессорный блок;
- функциональная клавиатура управления;
- дисплей;
- блок питания;
- интерфейс обслуживания;
- подсистема памяти;
- интервальный таймер-часы реального времени;
- сторожевой таймер.

Из-за существенного различия технических основ и способов обработки информации алгоритмы и принципы построения микропроцессорной релейной защиты могут отличаться от применяемых в электромеханических и электронных релейных защитах.

Информация со входа, получаемая цифровой релейной защитой, в общем случае содержит следующие составляющие: аналоговые сигналы; входная дискретная информация, включая сигналы от коммутационных аппаратов, других устройств релейной защиты и от обслуживающего персонала; цифровая информация от других устройств релейной защиты, управление настройками и параметрами цифровой защиты. Информация на выходе цифровой релейной защиты представлена следующими пунктами: выходная дискретная информация; цифровая информация к другим устройствам, сообщения различных видов, включая логические выходные сигналы и цифровые данные и др.

Современная концепция построения микропроцессорных устройств релейной защиты требует, чтобы все функции, являющиеся жизненно важными для предотвращения разрушения электрооборудования, обеспечения устойчивости и надежности работы энергосистем, должны выполняться децентрализованными (разобщенными) [3].

В случае, когда защита присоединения состоит из нескольких взаиморезервируемых систем защиты, эти системы должны быть полностью независимы друг от друга. Это необходимо для того, чтобы никакой отказ в одной системе не увеличивал времени отключения или не вызывал отказ в других системах. Также рекомендуется выполнять независимые системы защиты с разными принципами действия.

В цифровой системе релейной защиты и автоматики надежность системы энергоснабжения достигается за счет многократного резервирования и постоянным контролем исправности устройств оператором и дежурным персоналом. Повреждения локализуются автоматически и дистанционно, что в свою очередь сводит к минимуму ущерб от перерывов энергоснабжения.

Заключение. Применение микропроцессорных устройств в системах релейной защиты электроэнергетических систем является перспективным направлением развития средств автоматизации производственных процессов. Это позволяет не только снизить стоимость систем РЗА, но и усовершенствовать алгоритмы их работы. Однако совершенствование алгоритмов управления требует дальнейшего исследования процессов регулирования, построения более адекватных расчетных моделей, используемых для создания этих алгоритмов.

ЛИТЕРАТУРА

1. Гуревич, В.И. Еще раз о надежности микропроцессорных устройств релейной защиты / В.И. Гуревич // Электротехнический рынок. – 2009. – № 3 (29). – С. 40–45.
2. Дони, Н.А. Модернизация Микропроцессорных ДФЗ ВЛ 110–220 кВ / Н.А. Дони, Э.Г. Галеев, В.М. Лопухов // Релейная защита и автоматизация. – 2012. – № 4. – С 48–50.
3. Сарры, С.В. Программно-аппаратные средства микропроцессорных устройств релейной защиты электроэнергетических систем : учеб. пособие / С.В. Сарры ; Юж.-Рос. гос. техн. ун-т (НПИ). – Новочеркасск : ЮРГТУ, 2009. – 137 с.

МАШИНОСТРОЕНИЕ

УДК 681.625.9

ОСОБЕННОСТИ ТЕХНОЛОГИЧЕСКОГО ОБЕСПЕЧЕНИЯ ПЕРЕРАБОТКИ ОФИСНОЙ МАКУЛАТУРЫ

Д.А. ГАВРИЛОВЕЦ

(Представлено: А.С. КИРИЕНКО)

Исследуется проблема накопления офисной макулатуры и ее решение с помощью персонализированной технологии переработки офисной макулатуры непосредственно в офисах. Раскрывается технология сухого разрыва бумажного волокна и дается экономическое обоснование целесообразности использования персональной станции переработки офисной макулатуры, организациями, имеющими большие обороты бумаги.

Актуальность. В настоящее время мировые компании повсеместно работают над улучшением и внедрением технологий переработки макулатуры [1] непосредственно вместе ее потребления [2]. На мировом рынке присутствует уже запущенная в производство станция переработки офисной макулатуры от подразделения PaperLab компании Epson. Данная станция перерабатывает 3,6 кг/ч макулатурного сырья, что является огромным количеством для любого офиса и равно 720 листам формата А4. Такая станция снижает затраты на пункты хранения, выброса, заключение договоров со спецавтобазами, предприятиями по переработке отходов, или с третьими лицами сбора макулатуры, которые осуществляют ее вывоз. Также такая станция решает проблемы экологии, связанные с вредными реагентами (С1) используемыми для отбеливания бумажного сырья на предприятиях.

Соответственно реализация технологического обеспечения на основе использования персонального и компактного оборудования для экологически чистой переработки макулатуры, непосредственно на территории организации-пользователя является актуальной задачей, решение которой позволяет снизить или полностью исключить вышеуказанные расходы [3]. Исследования в данной области соответствует приоритетному направлению фундаментальных и прикладных научных исследований Республики Беларусь на 2016–2020 годы (Перечень утвержден постановлением Совета Министров Республики Беларусь от 12.03.2015 № 190): многофункциональные материалы и технологии, а также направлению обеспечения ускоренного технологического развития отечественных отраслей экономики на основе создания и внедрения новых и высоких технологий для производства наукоемкой конкурентоспособной продукции Государственной программы «Наукоемкие технологии и техника» на 2016 –2020 годы (Постановление Совета Министров Республики Беларусь от 21.04.2016 № 327).

Цель исследования. Целью исследования, является достижение научно-технического решения задачи по технологическому обеспечению переработки офисной макулатуры непосредственно в самом офисе, в котором должна быть установлена перерабатывающая мини-станция, решающая проблему накопления макулатуры.

Анализ проблемы переработки бумаги офисной макулатуры. Современные аналоги станций переработки офисной макулатуры имеют ряд недостатков таких как:

- высокую стоимость оборудования и технического обслуживания;
- отсутствие опреснительных камер для ликвидации влажного нагретого воздуха из зоны каландрирования, поскольку обильное выделение пара в этой зоне заставляет конденсироваться воду на стенках камеры формования, что негативно сказывается на ее работоспособности;
- отсутствие контроля однородности толщины получаемого бумажного волокна после камеры формования.

Предлагаются следующие направления решения этих проблем:

- упрощение процесса переработки для сокращения времени переработки;
- создание дополнительных систем опреснения воздуха [4] в установке и повторного использования полученной воды;
- развитие конструкции контроля и регулировки напыления и каландрирования бумажного волокна;
- применение новых недорогих и качественных материалов которые позволят сократить затраты на обслуживание;

- уменьшение размера узлов что повлияет на себестоимость станции;
- создание производственных линий на территории Республики Беларусь по производству таких установок.

В этой связи работа в данной области является весьма актуальной, значительной для экономики страны, способствующей высвобождению производственных мощностей по складированию, хранению и переработки макулатуры.

Описание предлагаемого решения. Предлагаемое решение реализовано в разработанной конструкции станции, перерабатывающей бумагу, которая занимает малую площадь помещения в сравнении с огромными и грязными цехами по переработке макулатуры [5]. Для данной установки требуется подвод воды и электричества. Предлагаемая конструкция [6] сможет перерабатывать листы формата А4 с чернильными или красящими загрязнениями и формовать новые листы из переработанной массы макулатурного сырья. В процессе переработки в бумагу можно вводить различные материалы, изменяющие ее характеристики такие как цвет, прочность, плотность и огнестойкость.

Предлагаемое решение технологического обеспечения переработки офисной макулатуры имеет блочную последовательную систему, объединённую в одной компактной конструкции, которая собирается по блочно-модульному принципу для повышения эффективности усовершенствования узлов-модулей, конструирования и производства новых блоков, разборки и сборки, замены, ремонта и последующего технологического обслуживания, замены расходных материалов.

Схема персональной станции переработки макулатуры показана на рисунке.

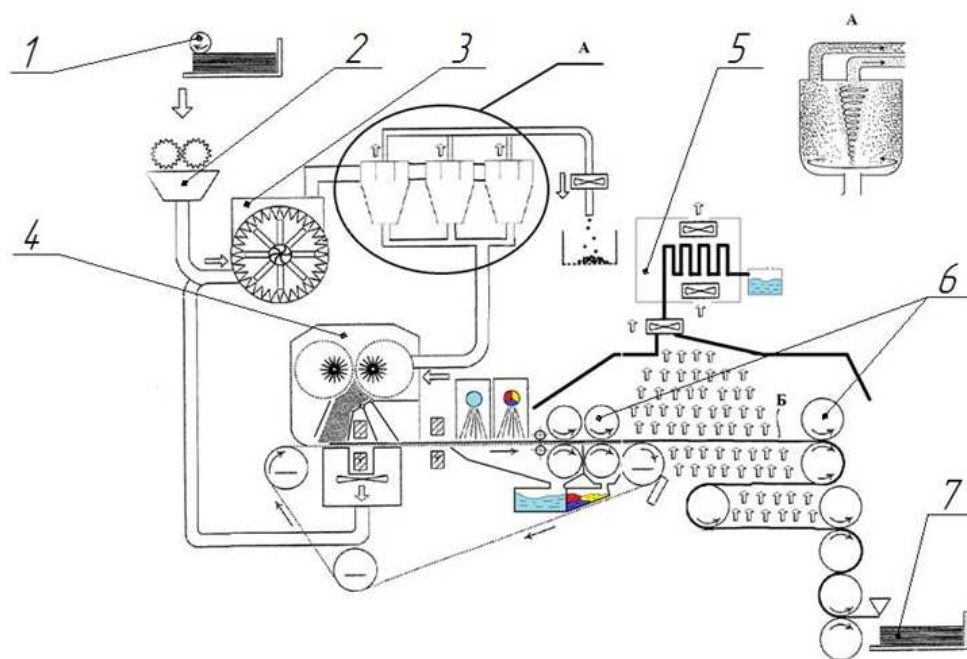


Рисунок. – Схема персональной станции переработки офисной макулатуры:

А – блок очистки бумажного волокна от загрязнений;

1 – блок приема бумаги, в котором осуществляется сбор офисной макулатуры; 2 – блок разрезания и измельчения бумаги до нужного размера при дальнейшей переработке; 3 – блок механического разрыва бумажной массы; 4 – блок формования для придания формы бумажному листу; 5 – блок сушки и опреснения влаги; 6 – блок каландрирования, т.е. создания полимерных связей; 7 – блок резки и вывода бумажного листа

Макулатура поступает в приемник 1 после чего разрезается в шредере 2 до мелких частиц нужного размера. Измельченная бумага по трубопроводу поступает в разрывную машину 3 и разрывается на волокна, которые фильтруются в центрифуге. Полученное и отфильтрованное бумажное волокно попадает в формовочную камеру 4 где смачивается и каландрируется 6 бумажный лист, который в последующем нарезается на нужный размер потребителю 7.

Конкурентные преимущества. Все существующие конкурентные технологии переработки макулатуры являются экологически грязными и трудоемкими, наносящими вред окружающей среде и экологии. В настоящее время конкурент по данной технологии является компания Epson Paper Lab. Поэтому в качестве аналога к предлагаемому решению использована запатентованная схема, компании Epson [3].

Основываясь на технологии компании Erpson и других известных решениях по переработке макулатуры Китайских и Японских компаний нами разработана технология и оборудование, для переработки офисной макулатуры которое по расчетным и экспериментальным данным увеличивает производительность и качество переработки в 2 раза.

Аналогичное производство на территории Республики Беларусь отсутствует, что является одним из положительных факторов развития использования данной инновации.

Вывод. В настоящее время в практическую организационную деятельность компаний все чаще внедряются электронные источники информации, организуются электронные базы документов и информационные офисы, но полный переход на них еще не близок. Поскольку до полного отказа от бумажных носителей информации еще далеко, предлагаемое нами решение вопроса экологически чистой персональной переработки офисной макулатуры на спроектированной станции является актуальной и экономически обоснованной задачей. Особенно это очевидно при огромных оборотах офисной макулатуры.

ЛИТЕРАТУРА

1. Фляте, Д.М. Технология бумаги : учебник для вузов / Д.М. Фляте. – М: лесн. Промсть, 1988 – 440с.
2. Гавриловец, Д.А. Персональная станция переработки офисной макулатуры / Д.А. Гавриловец // Труды молодых специалистов Полоцкого государственного университета. – 2017. – № 90, Машиностроение. – С. 302–303.
3. Иванов, С.Н. Технология бумаги / С.Н. Иванов. – М. : Лес. пром-сть, 1960. – 712 с.
4. Гавриловец, Д.А. Совершенствование эффективности блока формования плотности листа персональной станции переработки офисной макулатуры / Д.А. Гавриловец // Труды молодых специалистов Полоцкого государственного университета. – 2017. – № 90, Машиностроение. – С. 304–305.
5. Способ получения суспензии целлюлозной массы бумажно-содержащего материала : пат. публикации № EP20020004578, 06.11.2002 / Патент заявки № EP1254986 A1. 02.05.2001 / Jürgen Dockal-Bauer, Lothar Dr. Pfalzer, Harald Selder. – Режим доступа: <https://www.google.com/patents/EP1254986A1?cl=ru>. – Дата доступа: 27.09.2018.
6. Гавриловец, Д.А. Разработка технологии и оборудования для переработки офисной макулатуры / Д.А. Гавриловец, А.С. Кириенко // Первый шаг в науку – 2017 : сб. материалов Междунар. форума студенческой и учащейся молодежи. – Минск : Четыре четверти, 2017. – С. 82–84.

УДК 681.625.9

**СОВЕРШЕНСТВОВАНИЕ ЭФФЕКТИВНОСТИ
БЛОКА МЕХАНИЧЕСКОГО РАЗРЫВА БУМАЖНОЙ МАССЫ
ПЕРСОНАЛЬНОЙ СТАНЦИИ ПЕРЕРАБОТКИ ОФИСНОЙ МАКУЛАТУРЫ****Д.А. ГАВРИЛОВЕЦ***(Представлено: А.С. КИРИЕНКО)*

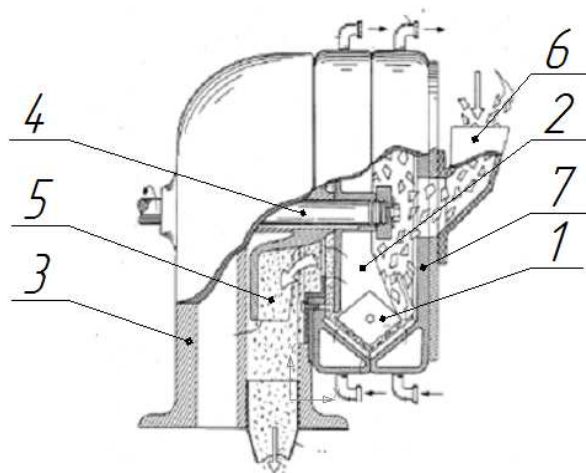
Исследуются пути повышения эффективности блока механического разрыва бумажной массы, применяющегося в персональной станции переработки офисной макулатуры, что позволит увеличить качество и максимально быстрое создание бумажного листа, сформированного в персональной станции. Рассмотрены как положительные, так и отрицательные стороны данного блока механического разрыва бумажной массы, его устройство и технологии совершенствования эффективности его функционирования.

Актуальность. Персональная станция по переработке макулатуры [1] имеет ряд возможностей для усовершенствования, что повышает ее эффективность. В аналоге [2] блок механического разрыва не имеет дополнительного турбинного колеса, что является недостатком при транспортировке больших масс макулатуры. Конструкция и расположение разрывных ножей не позволяет перерабатывать большие количества бумажного сырья без увеличения скорости вращения ротора. Такая система не имеет возможности изменения скорости потока, что сказывается на увеличении времени нахождения макулатурной массы в разрывной машине. Предлагаемое усовершенствование имеет один дополнительный нож на каждой лопасти ротора, измененную конструкцию ротора для лучшего подвода макулатурной массы и дополнительное турбинное колесо с электромуфтой, позволяющей регулировать скорость подачи материала. Улучшение процесса разрыва позволит увеличить объем перерабатываемого сырья в 2 раза.

Цель исследования. Целью исследования является достижение решения научно-технической задачи по усовершенствованию блока механического разрыва бумажного волокна станции переработки офисной макулатуры, в котором эффективно реализован технологический процесс разрыва макулатурной массы заданного размера.

Описание предлагаемого решения. Блок механического разрыва бумажной массы состоит из разрывной машины (рис. 1) (где 1 – нож, 2 – ротор, 3 – корпус, 4 – вал, 5 – ротор всасывания, 6 – приемник, 7 – статор) для постепенного разрыва макулатуры и отделения ее на волокно.

В описании патента [2], показанного на рисунке, принцип работы состоит в том, что разрывная машина загружается разрезанной бумажной массой (заданного размера) в приемник 6 транспортной линии. Масса, поступающая в разрывную машину, засасывается за счет разрежения, создаваемого ротором 2 и 5 в приемнике.

**Рисунок 1. – Разрывная машина**

Ножи 1, установленные на роторе 2, вращаются, тем самым измельчая макулатуру. На статор 7 устанавливают дополнительную пару ножей для совместного измельчения с ножами 1. Масса бумаги вместе с потоком воздуха попадает между ножами и измельчается. Полученная измельченная масса проходит через сито и выводится из разрывной машины. Масса, не измельчившаяся до нужного размера, задерживается в машине за счёт установленного в ней сита, со временем частицы измельчаются до нужной формы и размера и выводятся вместе с остальным волокном.

С целью увеличения производительности бумаги [3] предлагается дополнительно увеличить количество ножей 3. При увеличении количества ножей 3, в соответствии с рисунком 2 (где 1 – турбинное колесо, 2 – заглушка, 3 – ножи), время измельчения бумажного волокна увеличится, следовательно, бумажная масса уменьшится в размерах, чтобы избежать чрезмерного измельчения требуется увеличить скорость потока [1], решением этого будет установка на валу дополнительного турбинного колеса 1 с электромuftой. Электромuftа позволит регулировать включение и выключение принудительного потока воздуха.

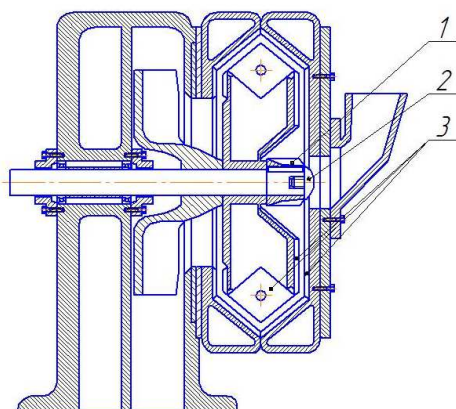


Рисунок 2. – Модифицированная разрывная установка

Конкурентные преимущества. Разработанный усовершенствованный блок [4] механического разрыва бумажной массы наиболее эффективен по сравнению с аналогом компании Epson. Система разрыва макулатурной массы получила ряд устойчивых изменений, таких как установка дополнительных ножей и турбинного колеса с электромuftой, что позволит увеличить производительность в 2 раза по сравнению с аналогами. Так же контроль потока, а именно электромuftа позволяет регулировать поток воздуха, регулируя время нахождения макулатурной массы в разрывной машине.

Вывод. Поскольку мировая тенденция складывается к уменьшению затрат на производство и увеличению прибыли, то наиболее эффективна технология разрыва макулатурной массы без водной массы, обеспечивающая уменьшение затрат на переработку макулатуры, а следовательно, ее использование целесообразно, как для производителя так и для потребителя станций по переработке офисной макулатуры с такой модернизацией.

ЛИТЕРАТУРА

1. Гавриловец, Д.А. Персональная станция переработки офисной макулатуры / Д.А. Гавриловец // Труды молодых специалистов Полоцкого государственного университета. – 2017. – № 90, Машиностроение. – С. 302–303.
2. Способ получения суспензии целлюлозной массы бумажно-содержащего материала : пат. публикации № EP20020004578, 06.11.2002 / Патент заявки № EP1254986 A1. 02.05.2001 / Jürgen Dockal-Bauer, Lothar Dr. Pfalzer, Harald Selder. – Режим доступа: <https://www.google.com/patents/EP1254986A1?cl=ru>. – Дата доступа: 27.09.2018.
3. Гавриловец, Д.А. Совершенствование эффективности блока формирования плотности листа персональной станции переработки офисной макулатуры / Д.А. Гавриловец // Труды молодых специалистов Полоцкого государственного университета. – 2017. – № 90, Машиностроение. – С. 304–305.
4. Гавриловец, Д.А. Разработка технологии и оборудования для переработки офисной макулатуры / Д.А. Гавриловец, А.С. Кириенко // Первый шаг в науку – 2017 : сб. материалов Междунар. форума студенческой и учащейся молодежи. – Минск : Четыре четверти, 2017. – С. 82–84.

УДК 629.33

ДИЗЕЛЬНОЕ ТОПЛИВО И ЕГО ХАРАКТЕРИСТИКИ**А.Т. БЕЙТУЛЛАЕВ, Г.Д. ДУБОВ**
(Представлено: Л.И. ЛАГУН)

Дизельное топливо после бензина является самым массовым продуктом, применяемым на автомобильном транспорте. Основные потребители дизельного топлива – железнодорожный транспорт, грузовой автотранспорт, водный транспорт, военная техника, дизельные электрогенераторы, сельскохозяйственная техника, а также легковой дизельный автотранспорт.

Автомобильное дизельное топливо получают путем прямой перегонки или каталитического крекинга керосино-соляровых фракций нефти; оно состоит в основном из керосиновых, газойлевых, а иногда и лигроиновых фракций. По групповому составу дизельное топливо преимущественно содержит нафтеновые и парафиновые углеводороды и незначительное количество ароматических углеводородов.

Дизельное топливо обладает рядом специфических характеристик, определяющих не только эффективность работы двигателя, но и влияющих на срок службы узлов топливной системы.

Основной характеристикой дизельного топлива принято считать **цетановое число**. Оно характеризует работу двигателя с точки зрения воспламенения дизельного топлива и его сгорания. От цетанового числа, в свою очередь, зависит мощность, дымность и шумность двигателя. Обычный диапазон значений цетанового числа колеблется от 40 до 50. Фактически, эта цифра означает срок задержки возгорания (отрезок времени от подачи топлива в цилиндр до его воспламенения). Более высокое цетановое число означает меньший период воспламенения, и, соответственно, лучшее горение топлива. Кроме того, при его повышении улучшаются экологические характеристики выхлопа. Однако если этот показатель превышает 60, то прирост мощности двигателя прекращается. В свою очередь, дизельное топливо с низким цетановым числом производить проще, поэтому на практике изготавливают дизельное топливо с цетановым числом не менее 40-45. ГОСТ 305-2013 гласит, что дизельное топливо должно иметь цетановое число не менее 45.

Следующий важный показатель - это плотность дизельного топлива. Крайне высокая плотность топлива означает, что в его составе присутствует больше тяжелых фракций. Для нормальной работы дизельного мотора наличие тяжелых фракций является негативным аспектом, так как испаряемость и процессы распыла в камере сгорания ДВС ухудшаются. В топливной системе и самих цилиндрах дизеля от езды на таком горючем постепенно накапливаются отложения и нагар.

Согласно действующим стандартам по ГОСТу:

- плотность летнего дизельного топлива – 860 кг/м³;
- плотность зимнего дизтоплива – не более 840 кг/м³;
- плотность арктического дизеля – не более 830 кг/м³.

Меньшая плотность дизельного топлива для зимы позволяет такому горючему сохранять текучесть и противостоять застыванию в условиях низких температур. Характеристика плотности дизельного топлива определяет не только порог его застывания и замерзания. Плотность дизельного топлива также указывает на количество энергии, которое выделяет горючее. Более высокий показатель плотности означает большее количество выделяющейся энергии в процессе сгорания в рабочей камере дизельного ДВС. Чем выше будет плотность дизельного топлива, тем большим окажется КПД двигателя. Дополнительно плотность повлияет на расход дизельного топлива на 100 км. Более плотное дизельное топливо в топливном баке заметно повышает экономичность двигателя.

Зимнее или арктическое дизельное топливо для дизельного мотора всегда имеет меньшую плотность. Для высвобождения энергии и получения необходимой отдачи от силового агрегата потребуется сжигать большее количество такой солярки сравнительно с более плотным топливом, которое используется в летний период. Этим объясняется повышенный расход менее плотного дизельного топлива зимой.

Каждую зиму владельцы автомобилей и другой техники с дизельным двигателем сталкиваются с такой проблемой, как замерзание горючего. Для того чтобы выяснить, как справляться с этой проблемой, необходимо разобраться в механизмах ее возникновения. В состав дизельного топлива входят тяжелые углеводороды. Некоторые из них являются представителями парафиновой группы. Под воздействием низких температур эти углеводороды кристаллизуются, что увеличивает вязкость горючего и, впоследствии, приводят к его частичному или полному застыванию.

Показатели, особенно актуальные в зимнее время года, – это температура помутнения и температура замерзания дизельного топлива.

Температура помутнения – температура, при которой начинается процесс кристаллизации содержащегося в топливе парафина. При этой температуре парафин неравномерно распределяется в объеме топлива, образуя своеобразные «облака». **Температура замерзания** – температура, при которой дизельное топливо теряет текучесть.

Классификация дизельного топлива в соответствии с техническим регламентом:

Согласно ГОСТу Р 52368-2005 дизельное топливо классифицируется по двум параметрам:

1. Предельное содержание серы, отраженное в показателе «ВИД» топлива, а именно:

вид I – содержание серы не более 350 (мг/кг);

вид II – содержание серы не более 50 (мг/кг);

вид III с содержанием серы менее 10 (мг/кг).

2. Температура применения (климатическая зона, в которой дизельное топливо может применяться). Для умеренной климатической зоны дизельное топливо разделяют по шести сортам: А, В, С, D, E, F, (таблица 1).

Таблица 1. – Требования к топливу для умеренного климата

Наименование показателя	Сорт					
	A	B	C	D	E	F
Предельная температура фильтруемости, °С, не выше	+5	0	-5	-10	-15	-20

Для районов с холодным климатом дизельное топливо условно разделяют на пять классов: 0, 1, 2, 3, 4.

Термин «температура фильтруемости» также впервые введен новым ГОСТ Р 52368-2005 и обозначает ту температуру, ниже которой дизельное топливо не проходит с необходимой скоростью (расходом) через стандартный эталонный фильтр (таблица 2).

Таблица 2. Требования к топливу для холодного и арктического климата

Наименование показателя	Класс				
	0	1	2	3	4
Предельная температура фильтруемости, °С, не выше	-20	-26	-32	-38	-44
Температура помутнения, °С, не выше	-10	-16	-22	-28	-34

Стандарт подразделяет топлива на летнее (Л), межсезонное (Е), зимнее (З), арктическое (А) и поначалу упраздняет старые сорта и классы.

К летнему топливу Л относят сорта А, В, С, D.

К межсезонному топливу Е – сорта Е, F.

К зимнему топливу З – классы 0, 1, 2, 3.

К арктическому топливу А – класс 4.

Как видим, дизельное топливо обладает рядом специфических характеристик, определяющих не только эффективность работы двигателя, но и влияющих на срок службы узлов топливной системы.

ЛИТЕРАТУРА

- Итинская, Н.И. Топливо, масла и технические жидкости / Н.И. Итинская, Н.А. Кузнецов. – М. : Агропромиздат, 1989. – 304 с.
- Кузнецов, А.В. Топливо и смазочные материалы / А.В.Кузнецов. – М : КолосС, 2004. – 199 с.
- Топливо, смазочные материалы и технические жидкости : учеб. пособие / В.В. Остриков [и др.]. – Тамбов : Изд-во Тамб. гос. техн. ун-та, 2008. – 304 с.
- СТБ 1658-2012. Топлива для двигателей внутреннего сгорания. Топливо дизельное. Технические условия.

УДК 629.33

СРАВНИТЕЛЬНЫЙ АНАЛИЗ КАЧЕСТВА ДИЗЕЛЬНОГО ТОПЛИВА

А.Т. БЕЙТУЛЛАЕВ, Г.Д. ДУБОВ
(Представлено: Л.И. ЛАГУН)

К дизельным топливам относятся топлива для двигателей с воспламенением топливно-воздушной смеси от сжатия (дизелей). Широкое распространение дизелей обусловлено их преимуществами перед бензиновыми двигателями: высокая экономичность (удельный расход топлива на 30...40% ниже), отсутствие системы зажигания, возможность использования более тяжелых видов топлива, меньшая пожарная опасность, более высокая приемистость.

На разных автозаправочных станциях в марте 2018 года были взяты 5 образцов дизельного топлива. Три из которых в городе Новополоцк Витебской области Республики Беларусь на автозаправках «Белоруснефть», «Татнефть» и «Лукойл» и два вида из города Себеж Псковской области Российской Федерации на автозаправках «Лукойл» и «Сургутнефтегаз». В лабораторных условиях был произведен сравнительный анализ качества этих образцов топлива.

1. Замер цетанового числа образцов дизельного топлива.

С помощью прибора ОКТАН-ИМ произведен замер цетанового числа. Согласно ГОСТу 305-2013 дизельное топливо должно иметь цетановое число не менее 45 (табл. 1).

Таблица 1. – Результаты измерения цетанового числа образцов дизельного топлива

Образец дизельного топлива	Полученное цетановое число
1. Сургутнефтегаз	48,7
2. Лукойл (РФ)	47
3. Лукойл (Республика Беларусь)	47,8
4. Татнефть	47,2
5. Белоруснефть	47,4

Из таблицы следует, что наибольший показатель цетанового числа имеет образец с автозаправки «Сургутнефтегаз», а наименьший показатель имеет топливо с автозаправки «Лукойл», взятое в РФ. Более высокое цетановое число означает меньший период воспламенения и, соответственно, лучшее горение топлива. Кроме того, при его повышении улучшаются экологические характеристики выхлопа. Так что можно сказать, что все образцы топлива соответствуют ГОСТ 305-2013

2. Замер плотности образцов дизельного топлива.

Так же в лабораторных условиях на установке Плотномер ВИП-2МП произведен замер плотности дизельного топлива. Согласно ГОСТу 305- 82 плотность зимнего дизельного топлива – не более 840 кг/м³, арктического – не более 830 кг/м³ (образцы на АЗС взяты в марте 2018 года) (табл. 2).

Таблица 2. – Результаты измерения плотности образцов дизельного топлива

Образец дизельного топлива	Полученное цетановое число
1. Сургутнефтегаз	805,5
2. Лукойл (РФ)	823
3. Лукойл (Республика Беларусь)	828,2
4. Татнефть	828,7
5. Белоруснефть	829

Из таблицы следует, что менее плотное топливо марки «Сургутнефтегаз», а более плотное топливо марки «Белоруснефть». Можно сказать, что тяжелые фракции в образцах отсутствуют. Для нормальной работы дизельного мотора наличие тяжелых фракций является негативным аспектом, так как испаряемость и процессы распыла в камере сгорания ДВС ухудшаются. В топливной системе и самих цилиндрах дизеля от езды на таком горячем постепенно накапливаются отложения и нагар. Меньшая плот-

ность дизельного топлива для зимы позволяет такому горючему сохранять текучесть и противостоять застыванию в условиях низких температур. Соответственно, в условиях сильного мороза топливо «Сургутнефтегаз» будет прокачиваться по топливной системе легче остальных. Дополнительно плотность влияет на расход дизельного топлива на 100 км. Более плотное дизельное топливо в топливном баке заметно повышает экономичность двигателя. Из этого следует, что на топливе «Беларуснефть» двигатель будет более экономичен, чем на других образцах. По результатам измерения все виды топлива соответствуют ГОСТу 305-82.

3. Измерение низкотемпературных показателей образцов дизельного топлива.

Особенно актуальные показатели для зимнего дизельного топлива – температуры помутнения и застывания. С помощью установки УТЗ-60М получены следующие показатели (табл. 3).

Таблица 3. – Температурные показатели образцов дизельного топлива

Образец дизельного топлива	Температура помутнения, °С		Температура замерзания, °С	
	полученное значение	требуемое значение	полученное значение	требуемое значение
1. Сургутнефтегаз	<-47	-22	-	-32
2. Лукойл (РФ)	-18	-16	-32	-26
3. Лукойл (РБ)	-9	-	-20	-20
4. Татнефть	-10	-	-27	-20
5. Беларуснефть	-11	-	-28	-20

Дизельное топливо с автозаправки «Сургутнефтегаз» охладили до температуры -47°С, при такой температуре помутнения не наблюдалось. Так как процесс охлаждения очень длительный, было принято решение ограничиться данной температурой, подтверждающей соответствие заявленным низкотемпературным показателям. По заявленным автозаправочными станциями характеристикам образец «Сургутнефтегаз» является топливом 2 класса, «Лукойл» РФ-1 класса, остальные образцы – Сорта F. Таким образом, все образцы соответствуют заявленным низкотемпературным показателям.

Из этого всего следует, что важными показателями для зимнего дизельного топлива являются: цетановое число, плотность и температуры помутнения и застывания. При плохих показателях этих параметров ухудшается эффективность работы двигателя, холодный запуск, срок службы узлов топливной системы и др.

ЛИТЕРАТУРА

1. Итинская, Н.И. Топливо, масла и технические жидкости / Н.И. Итинская, Н.А. Кузнецов. – М. : Агропромиздат, 1989. – 304 с.
2. Кузнецов, А.В. Топливо и смазочные материалы / А.В.Кузнецов. – М : КолосС, 2004. – 199 с.
3. Топливо, смазочные материалы и технические жидкости : учеб. пособие / В.В. Остриков [и др.]. – Тамбов : Изд-во Тамб. гос. техн. ун-та, 2008. – 304 с.
4. СТБ 1658-2012. Топлива для двигателей внутреннего сгорания. Топливо дизельное. Технические условия.

УДК 629.017

ВЛИЯНИЕ УГЛОВ УСТАНОВКИ КОЛЁС НА УПРАВЛЯЕМОСТЬ АВТОМОБИЛЯ**Ю.А. ГАСС, Д.В. ЛОПАТИН**
(Представлено: И.В. ХОМИЧ)

Рассматриваются параметры установки колёс и даются их определения. Выявляется влияние каждого параметра на управляемость автомобиля. Приводятся примеры различных сочетаний и области их применения.

Для создания наименьшего сопротивления движению, уменьшения износа шин и снижения расхода топлива колеса должны катиться в вертикальных плоскостях, параллельных продольной оси автомобиля. С этой целью колеса устанавливаются на автомобиле с развалом в вертикальной и со сходимением в горизонтальной плоскости.

Различают следующие углы установки колёс.

Угол продольного наклона (кастор) (рис. 1) - угол между осью поворота колеса и вертикалью на виде сбоку. Считается положительным, если ось наклонена назад относительно направления движения.

Развал (рис. 1) – наклон плоскости колеса к перпендикуляру, восстановленному к плоскости дороги. Если верхняя часть колеса наклонена наружу автомобиля, то угол развала положительный, а если внутрь – отрицательный.

Схождение (рис. 1) – угол между продольной осью автомобиля и плоскостью, проходящей через центр шины управляемого колеса. Схождение считается положительным, если плоскости вращения колес пересекаются перед автомобилем, и отрицательным, если они, наоборот, пересекаются сзади [1].

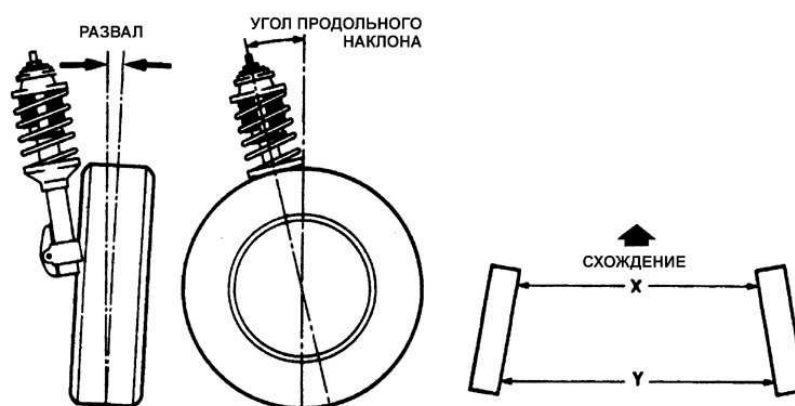


Рисунок 1. – Углы установки колёс

Цель работы – изучить влияние различных углов установки колёс и их комбинаций на управляемость автомобиля.

Регулировка углов установки колёс начинается с кастора. Кастор влияет на стабилизирующий момент и на изменение развала колес при повороте руля. Чем больше кастор, тем больше стабилизирующий эффект при вывернутых колёсах.

Из-за того, что точка контакта колеса (при положительном касторе) с поверхностью лежит несколько позади оси поворота, при отклонении колеса от прямолинейного движения возникают боковые силы, стремящиеся вернуть колесо в начальное положение.

Кроме того, если в начальный момент отклонения колес от нейтрального положения при прохождении поворота только внешнее колесо имеет необходимый отрицательный развал, то по мере увеличения отклонения колес в сторону поворота внешнее колесо приобретает ещё больший отрицательный развал, а отрицательный развал внутреннего колеса уменьшается или даже становится положительным. Таким образом, кастор благоприятно влияет на сцепление управляемых колес в поворотах.

При повороте в одну из сторон одно из колес стремится приподнять шасси, другое же само приподнимается. Таким образом, для поворота колес необходимо не только преодолеть силы трения, но и затратить некоторое количество энергии на приподнимание шасси. Благодаря этому и выше описанным боковым силам водитель может «чувствовать» автомобиль.

Нулевой кастор (иногда даже отрицательный) обеспечивает остроту рулевого управления. А при отсутствии гидроусилителя – еще и легкость поворота колес. Характерный пример – «Волга» со шкворневой подвеской. Но, к примеру, в дрифте его, напротив, делают еще «положительнее» – в сочетании с большим выворотом колес это позволяет держать хороший угол заноса [2].

После регулировки кастора выставляется развал. В большинстве случаев под «развалом» понимают статический развал управляемых колес, задаваемый при техническом обслуживании автомобиля. В некоторых автомобилях регулировке подлежит и статический развал неуправляемых колес.

Основное назначение статического развала управляемых колес – уменьшение передачи на руль их вибрации, возникающего вследствие наезда на мелкие неровности покрытия. Вместо того, чтобы передаваться через рулевую трапецию на руль, вибрация гасится за счет упругости покрышек.

У большинства автомобилей, за исключением авто, оснащенных подвеской «Макферсона», развал для передних управляемых колес, как правило, имеет небольшое положительное значение – в пределах от 0' до 45', изредка до 2°. Такое его значение позволяет снизить усилия на управляемых колесах и уменьшить передачу на рулевое управление рывков, возникающих при проезде неровностей дороги. У таких автомобилей большой отрицательный развал является признаком износа подвески либо её неправильной регулировки, и приводит к быстрому износу шин, ухудшению сцепных свойств на ровной дороге и нарушению курсовой устойчивости автомобиля [3].

Например, у ряда моделей Honda задняя подвеска состоит из верхних и нижних поперечных рычагов, находящихся под наклоном. Причем нижние длиннее верхних. Когда пружины проседают, рычаги встают в «горизонт», и нижний выталкивает низ колеса наружу [2].

На автомобилях с подвеской «макферсон» используется нулевой или небольшой отрицательный развал, что связано с отличием иных установочных параметров данной подвески, вызванным её конструктивными особенностями.

Также отрицательный развал устанавливается на гоночных автомобилях, предназначенных для езды по овалам, на внутренних колесах.

Что в дрифте, что в кольцевых гонках отрицательный развал при поворотах позволяет сохранить максимальную площадь контакта шин с покрытием. Дело в том, что боковая сила в этом случае стремится перевести развал в положительный, уменьшая пятно контакта, а отрицательные значения компенсируют это явление.

Иной раз можно наблюдать сильно положительный развал. Характерный пример этого – Tatra, которая с переходом на хребтовую раму и подвеску с качающимися полуосями получила в порожнем варианте заметный положительный развал колес задней тележки. Но при загрузке углы выводятся в ноль, обеспечивая полное пятно контакта и равномерный износ шин [2].

В гражданских автомобилях с помощью схождения компенсируют эффект криволинейного качения колеса при углах развала, отличных от 0, чтобы износ резины был минимален.

Для спортивных автомобилей допустимо «играть» сходимением в целях изменения характера управляемости автомобиля. Например, можно добавить немного отрицательного схождения для переднеприводного автомобиля, это поможет компенсировать его тенденцию к недостаточной поворачиваемости [3].

Любой автомобиль очень чувствителен к изменению угла схождения задних колес. Если у машины отрицательное схождение, тогда в повороте наружное заднее колесо будет слегка развернуто от центра поворота - и задняя ось будет стремиться обогнать переднюю. Автомобиль приобретает склонность к заносу, и наоборот, при положительном схождении нагруженное колесо направлено к центру поворота – оно работает на недостаточную поворачиваемость. Слегка сведенные к центру колеса увеличивают стабильность движения по скоростной прямой и помогают подвеске самой «подруливать» в нужную сторону, компенсируя порывы бокового ветра. Поэтому рекомендуется настроить небольшое положительное схождение.

Существует четыре наиболее распространённых варианта углов установки колес.

Вариант 1. Углы продольного наклона стоек устанавливаются в «минус». Передние колеса как бы сдвигаются назад, к брызговикам колесных ниш. Ситуация, довольно часто встречающаяся на старых машинах либо после установки проставок, поднимающих заднюю часть автомобиля. Результат: легкий руль, быстрые отклики на его малейшие отклонения. Однако машина стала излишне нервной и вертлявой, что особенно заметно при скорости 80–90 км/ч и выше. У автомобиля нестабильны отклики при входе в поворот, норовит рискнуть в сторону, требуя от водителя постоянного подруливания. Ситуация осложняется при выполнении маневра «переставка».

Вариант 2. Кастор положительный, углы схождения и развала выставляются в «ноль». Рулевое колесо стало упругим и информативным, и немного более "тяжелым". Автомобиль едет четко, понятно

и правильно. Исчезли вертлявость, невнятные взаимосвязи и траекторные рыскания. На «переставке» автомобиль легко опередил предыдущий вариант.

Вариант 3. Излишне "положительный" развал. Изменять его без коррекции схождения нежелательно, поэтому устанавливается еще и положительное схождение. Опять "полегчал" руль, стали ленивее отклики на входе в поворот, увеличилась боковая раскачка кузова. Однако при моделировании экстремальной ситуации "чувство руля" теряется. С появлением скольжений неожиданно рано осложняется попадание в заданный коридор на "переставке" и машина начинает скользить слишком рано. В быстрых поворотах доминируют сильнейшие проскальзывания передней оси.

Вариант 4. Спортивный вариант: все углы устанавливаются в "минус", за исключением кастора. Автомобиль с такими настройками повороты проходит увереннее и быстрее, как и маневр "переставка". Отсюда и лучший результат. Однако увеличивается износ шин [4].

Таким образом, существует большое количество простых и весьма эффективных способов менять характер автомобиля, не прибегая к дорогостоящим заменам узлов и деталей. Главное, не пренебрегать регулировками - зачастую они оказываются очень важными. Наиболее распространённым является второй вариант. Он наиболее логичен для повседневной езды, причем как с частичной, так и с полной нагрузкой. Необходимо лишь учитывать, что, увеличивая продольный наклон стойки, не только улучшается поведение машины, но и повышается стабилизирующее усилие на руле. Последний, наиболее «быстрый» вариант настройки больше подходит околоспортивной публике. Углы установки колес имеют важнейшее значение для управляемости автомобиля и безопасного передвижения, поэтому лучше всего использовать заводские настройки.

ЛИТЕРАТУРА

1. Вахламов, В.К. Конструкция, расчет и эксплуатационные свойства автомобиля : учеб. пособие / В.К. Вахламов. – М. : Академия, 2007. – 560 с.
2. Drom.ru – автомобильный портал [Электронный ресурс]. – Режим доступа: <https://www.drom.ru/info/misc/48237.html>. – Дата доступа: 25.04.2018.
3. DRIVE2.RU Сообщество машин и людей [Электронный ресурс]. – Режим доступа: <https://www.drive2.ru/b/1049540/>. – Дата доступа: 25.04.2018.
4. За рулем.рф [Электронный ресурс]. – Режим доступа: https://wiki.zr.ru/Регулировки_кастора. – Дата доступа: 25.04.2018.

УДК 629.018

СПОСОБЫ РЕГУЛИРОВКИ УГЛОВ УСТАНОВКИ КОЛЁС И ОСНОВНЫЕ ПРИЧИНЫ ИХ НАРУШЕНИЯ

Д.В. ЛОПАТИН, Ю.А. ГАСС
(Представлено: *И.В. ХОМИЧ*)

Производителем предусмотрены определенные значения установки передних и задних колес. Более того, практически у каждой модели эти значения хоть и похожи, но индивидуальны. Это оптимизирует ходовые качества, управляемость и прямо влияет не только на износ шин и экономичность поездки, но и на комфорт.

Периодичность проверки углов установки колес зависит от интенсивности эксплуатации и от конструктивных особенностей автомобиля. Если у одних моделей достаточно большие допуски (например, суммарное схождение колес на оси от -10 до +40 мин), то достаточно проверять развал-схождение раз в год-полтора, или через 20–30 тысяч километров. А если допустимая разбежка всего плюс-минус три минуты, то углы установки колес стоит проверять в два раза чаще, например, после каждой сезонной смены резины [1].

Цель работы – определение основных причин нарушения углов установки колёс и способов их регулировки.

Практически любое вмешательство в подвеску автомобиля требует последующей регулировки углов установки колес. Исключение составляют работы по замене стоек стабилизатора. Кроме того, регулировку углов необходимо постоянно контролировать, т. к. в процессе воздействия нагрузок, создаваемых при движении, детали подвески претерпевают огромные, а иногда и критические изменения. Так, при попадании в глубокую яму-выбоину может согнуться рычаг, рулевая тяга, либо надорваться сайлентблок. Это приведет к изменению ориентации колеса автомобиля относительно его движения и относительно других колес. Как следствие - вероятность потери нормальной управляемости и повышенный и неравномерный износ шин. Если автомобиль управляется привычно, спицы рулевого колеса расположены ровно, а при визуальном беглом осмотре шин видно, что их износ равномерен, то не нужно ничего проверять и регулировать. [2]

Показанием к посещению стенда регулировки углов установки колес являются любые изменения в управляемости автомобиля: "рыскание" машины по дороге, невозвращение самостоятельно рулевого колеса после поворота, писк резины при входе в поворот на незначительной скорости. Любое, даже очень незначительное, изменение положения спиц рулевого колеса и разница числа оборотов рулевого колеса при максимальном повороте влево и вправо, неравномерный износ шин с внешней стороны и с внутренней - все это следствия нарушений углов установки колес автомобиля. Пренебрежение данными явлениями может легко привести не только к значительным финансовым тратам в виде покупки новых шин вместо испорченных, но и к реальным аварийным ситуациям.

Перед регулировкой углов установки колёс необходимо проверить состояние сайлентблоков, шаровых опор, рулевых тяг и наконечников. Если имеются люфты, проводить измерение углов установки колес бессмысленно. Также необходимо проверять давление в шинах. Кроме того, необходимо учитывать особенности автомобилей. Так, например, BMW и Opel, требуют дополнительной загрузки автомобиля, как если бы в нем находились водитель и передний пассажир. В BMW нужно загружать еще и заднее сиденье по центру. [3]

В принципе регулировки можно произвести и подручными средствами. Для этой процедуры достаточно иметь уровень, отвес, транспортир и линейку.

Но жизнь не стоит на месте и средства измерения углов установки колес постоянно совершенствуются и меняются. Линейку и уровень в прошлом веке сменили оптические приборы. Примером такого прибора может служить рогачевская "Система СКО-1М". Прибор в базовой комплектации позволяет измерять углы на автомобилях с колесами до 18 дюймов и выпускается по сей день в том виде, в котором его изобрели в середине прошлого века.

Затем появились лазерные стенды. На смену пришли всевозможные приспособления, выводящие информацию об углах развала-схождения колес на экран монитора. Все это и мастер, и клиент могут воочию наблюдать в режиме реального времени. А в памяти компьютера собрана база данных по углам установки колес от практически всех производителей.

Далее совершенствование приборов идет по способу доставки информации от колеса машины к монитору. Более старые приборы имели связь колеса автомобиля с компьютером через кабели. Это

доставляло массу неудобств. Потом кабели заменили беспроводными способами связи. Но все равно оставалась проблема: на каждое колесо нужно ставить отдельный прибор, который и служит источником сигнала для компьютера и сообщает, под каким углом колесо установлено. Кроме того, при замерах каждое колесо машины нужно вывешивать и прокручивать вместе с прибором, производя так называемую компенсацию биения. Все эти процедуры требуют времени и дополнительных манипуляций.

Поколение так называемых 3D-приборов для измерения углов установки колес принесло в работу сервисов значительные изменения! Теперь на колеса устанавливаются маркированные пластиковые таблицы, информацию с которых при прокатывании машины взад-вперед считывают видеокамеры компьютера. Информация сразу же сравнивается с номинальными значениями производителя в программном обеспечении.

Все это выводится на монитор с анимационными подсказками технологического процесса регулировок. Данная технология измерений позволила улучшить не только скорость, но и качество данной услуги. Помимо стандартных параметров (развал, сходжение, кастер, соосность) современный стенд позволяет замерить, например, предельный угол выворота управляемых колес, что не будет лишним после ремонта рулевого управления. По завершении регулировок клиенту выдается протокол-распечатка, где видно, с какими показаниями углов приехал автомобиль, какие требования производителя и с какими он выезжает.

Но 3D-стенды поначалу имели огромный недостаток – высокую стоимость. Поэтому и услуга с использованием данного прибора не из самых дешевых. Если в процессе регулировки необходимо разработать резьбовые соединения рулевых тяг или наконечников, сняв их с автомобиля, стоимость данных работ увеличивается. [4]

Часто после регулировок на стенде руль стоит прямо, а автомобиль имеет увод (не едет ровно). При прямолинейном движении приходится подруливать, выравнивая стремящуюся съехать с дороги машину. Рулевое колесо при движении прямо имеет несимметричное расположение спиц. Или еще одна довольно распространенная ситуация: машина ехала прямо, никуда не тянуло, руль стоял ровно, но были неисправности в подвеске. После замены сайлентблоков нижнего рычага, например, и наконечника рулевой тяги на хорошие запчасти и регулировки углов установки колес на 3D-стенде, выдаётся распечатка, которая свидетельствует о том, что углы выставлены в соответствии с параметрами завода-изготовителя. Но после вышеперечисленных манипуляций всё равно сохраняется не сильный, но ощутимый увод в сторону.

Чаще всего эта проблема не связана с правильностью регулировок углов установки колес. Причины бокового увода могут скрываться в ходовой части. Однако в 90% случаев боковой увод вызывается шинами. Это объясняется тем, что за время, которое автомобиль двигался с относительно изношенными деталями подвески (сайлентблоки и рулевой наконечник до замены), шины "притерлись" к определенному профилю протектора, который и позволял осуществлять прямолинейное движение, никуда не подруливая. Теперь установки были актуализированы, а шины остались на прежних местах, имея неравномерный профиль протектора, который и "уводит" машину. Решение – перестановка местами шин, что, впрочем, не всегда помогает. Если перестановкой не удалось ситуацию исправить, то поможет только покупка новых шин [1].

С новыми шинами тоже не все так однозначно. Шины многих производителей не имеют изначально курсовой устойчивости и грешат конусностью. Конусность вызвана небольшим смещением протектора и/или брекера по отношению к геометрической середине шины на несколько десятых миллиметра. Конусность визуально не заметна и не измеряется в условиях автомастерской.

Вот цитата из знаменитой немецкой ELSA:

Производитель допускает некоторую конусность протектора шины. Вследствие этого при качении колеса возникает боковая сила, передаваемая на подвеску, которая может приводить к самопроизвольному повороту (уводу) автомобиля. Это явление можно устранить путем целенаправленной перестановки колес.

Из-за смещения жесткость на внутреннем и внешнем плечах шины разная, что приводит к различным силам, действующим на опорную поверхность колеса. Поэтому брекер и протектор "упираются" в дорожное полотно с разным усилием. В результате получается конус. Вызванное конусностью усилие может в зависимости от скорости стать настолько большим, что автомобиль начнет уводить в какую-либо сторону. Если сила на одном колесе оси составляет, например, 50 Н, на другом колесе также 50 Н, а эти силы еще и направлены в одинаковую сторону, то они суммируются. Благодаря переворачиванию шины на диске боковой увод можно компенсировать, так как силы будут действовать в таком случае в противоположные стороны.

Так как по шине направление, в котором действует сила, вызванная конусностью, не увидеть, то определить, какая шина вызвала боковой увод, можно только пробными поездками и целенаправленной перестановкой дисков и шин. Шина состоит из многих элементов и материалов, которые в конце трудо-

емкого процесса производства отчасти вулканизируются. Из-за этого получаются различные погрешности, которые могут стать заметными при более или менее высоких боковых усилиях. Данные усилия могут проявиться также и на новых шинах. Боковой увод при сильном ускорении отчасти заложен в конструкции автомобилей с передним приводом. Различные условия трения левого и правого колес, в том числе неровности дорожного покрытия (выбоины), а в связи с этим и неравномерное сцепление с дорогой, сильно влияют на ходовые качества [4].

Таким образом, качество регулировки углов установки колёс не зависит от типа стенда. Определяющим фактором является мастерство и опыт специалиста, хотя и считается, что 3D-стенды обеспечивают большую точность. Практика же показывает, что человеческий фактор по-прежнему остается определяющим. Главное различие подобных стендов для рядового клиента – время выполнения операции. Самыми «быстрыми» являются 3D-стенды, где регулировка одной оси занимает 10–15 мин, затем идут инфракрасные – 25–30 минут, а наиболее «задумчивые» – кордовые (40–50 мин). Основной же причиной нарушения углов установки колёс, и как следствие управляемости автомобиля, является неисправное состояние ходовой части автомобиля.

ЛИТЕРАТУРА

1. Автопортал ABW.by [Электронный ресурс]. – Режим доступа: <https://www.abw.by/novosti/experience/193942/>. – Дата доступа: 17.07.2018.
2. Самохин, С. Секреты подвескостроения У-У-К. Ч. 1 / С. Самохин, А. Солнцев // Автомобиль и сервис. – 2018. – № 1. – С. 40–44.
3. Самохин, С. Секреты подвескостроения У-У-К. Ч. 2 / С. Самохин, А. Солнцев // Автомобиль и сервис. – 2018. – № 2. – С. 26–31.
4. Автопортал ABW.by [Электронный ресурс]. – Режим доступа: <https://www.abw.by/novosti/automarket/187428/>. – Дата доступа: 17.07.2018.

УДК 629.33

АНАЛИЗ ОТРАБОТАННОГО МОТОРНОГО МАСЛА

А.В. ГУЩА, М.А. ЧЕРНИЧЕНКО
(Представлено: Л.И. ЛАГУН)

Провести полноценную экспертизу отработанного моторного масла подручными средствами нельзя, но существуют простые и доступные методы проверки пригодности моторного масла к эксплуатации.

Один из методов – так называемый «капельный тест». Пожалуй, это самый простой метод, позволяющий определить, пригодно ли масло к использованию по назначению, будь то неправильное хранение ранее приобретенного масла или уже используемого на протяжении определенного периода и которым может воспользоваться любой владелец автомобиля и проверить состояние моторного масла.

Что нужно для такого теста? Собственно, ничего особенного, обычный лист офисной бумаги А4 или фильтровальной бумаги. Пробы на фильтровальной бумаге выглядят более "свежими" и, как правило, высыхают быстрее.

Нужно прогреть авто до рабочей температуры, вынуть масляной щуп и капнуть каплю масла на бумагу. После просушить все при комнатной температуре в течение суток или более. Далее необходимо внимательно и детально рассмотреть оставшийся от капли масла отпечаток, тем самым определяя цветность формы оставленного масляного рисунка, равномерность растекания масла по четырем составным частям «капельной пробы»:

1 – ядро или центр капли, соответствующий первичной зоне капли до ее растекания по бумаге; здесь оседают все тяжелые нерастворимые механические примеси;

2 – краевая зона (темное/черное кольцо), окаймляющее ядро малорастворимыми в масле органическими примесями; кольцо отсутствует как при чистом масле, так и при очень грязном масле, а ядро имеет ровный цвет;

3 – зона диффузии – широкое серое кольцо за ядром – через краевую зону масла с легкими растворенными органическими примесями;

4 – кольцо чистого масла – самое внешнее светлое кольцо, если в нем начинает проявляться потеря моюще-диспергирующих присадок. Это кольцо – не частое явление (рис. 1).

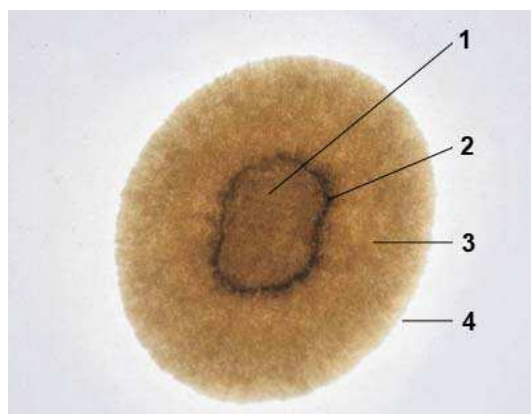


Рисунок 1. – Отпечаток капли отработанного моторного масла

Чистое масло дает большое светлое пятно, исчезающее через несколько суток. 4-й элемент «капельной пробы» через несколько часов также исчезает. Если 3-й и 4-й элементы имеют рваную (амебообразную) форму, то масло обводнено. А стойкий желтоватый, или светло-коричневый цвет зоны диффузии говорит о значительном окислении масла из-за аварийного перегрева ДВС.

Чем светлее и равномернее цвет ядра и зоны диффузии, тем работоспособнее масло. При росте механических примесей темнеет ядро, зона диффузии и теряется краевая зона. При потере присадок уменьшается зона диффузии, расширяется внешнее светлое кольцо. Появление внешнего кольца чистого масла означает момент, когда начинают исчерпываться моюще-диспергирующие свойства масла.

Отсутствие зоны диффузии или “свертывание” пятна из-за потери присадок, как правило, из-за воды в масле; густое черное мазеобразное ядро с блестками металла, коричневое или желтое кольцо свидетельствуют о браковочном состоянии масла, оно подлежит срочной смене.

Одной из таких технологий определения качества отработанного моторного масла является описание по SAE (society of automotive engineers - Американская Ассоциация автомобильных инженеров).

Для эксперимента взяты три вида отработанного моторного масла. Масло с автомобиля Mitsubishi ASX, Motors 5w-40 полусинтетика, пробег 9500 км. ELF10w-40 полусинтетика с автомобиля Renault Scenic, с пробегом 11000. Автомобиль Mazda 6, масло Mazda Original oil ultra 5w-40 полусинтетика, пробег 18000.

Рассмотрены три капли, оставленные на бумаге этими образцами моторных масел (рис. 2).



Рисунок 2. – Отпечаток от капель отработанного моторного масла

Внутренний круг показывает степень загрязнения масла: сажей пылью металлами. Можно сказать, что все масла среднего качества, явных металлов и грязи в них нет. Второй круг показывает собственно само качество масла. Чем оно темнее, тем качество масла хуже. Все образцы среднего качества, но масло с пробегом 9500 км имеет более темный круг и можно сказать, что оно хуже других образцов. Контур указывает на наличие воды в масле, более сглаженный круг говорит о том, что в масле не было воды, а более четкий круг означает, что в масле присутствовала вода. И последний 4 круг указывает на содержание топлива в масле. Можно сказать, что все масла среднего качества, и дальнейшее их использования не рекомендуется.

Это простой, но эффективный способ по оценке состояния масла позволяет вовремя производить его замену, что в свою очередь увеличивает срок эксплуатации двигателя внутреннего сгорания. Желательно проводить такие контрольные проверки состояния моторного масла перед плановой его заменой, что даст возможность заранее выявить имеющиеся неполадки в работе двигателя вашего авто. Таким образом, постоянно держать под контролем состояние самой важной части вашего автомобиля – его двигателя, не забывая о важности и необходимости своевременной смены масла в двигателе.

Опытные автомобилисты производят замену масла, ориентируясь не на пробег двигателя по спидометру, а по моточасам работы двигателя, так как эксплуатация автомобиля по городу (частое изменение скоростного режима и частое стояние в пробках) существенно влияет на состояние масла в двигателе.

ЛИТЕРАТУРА

1. Моторные масла / Р. Балтенас [и др.]. – СПб. : Альфа-Лаб, 2000. – 272 с.
2. Гнатченко, И.И. Автомобильные масла, смазки, присадки : справ. пособие / И.И. Гнатченко, В.А. Бородин, В.Р. Репников. СПб. : Полигон АСТ, 2000. – 360 с.
3. Итинская, Н.И. Топливо, масло и технические жидкости / Н.И. Итинская, Н.А. Кузнецов. – М. : Агропромиздат, 1989. – 304 с.
4. Хитрюк, В.А. Автомобильные эксплуатационные материалы : учеб. пособие / В.А. Хитрюк, А.К. Трубилов. – Минск : РИПО, 2013. – 323 с.

УДК 629.33

ОТРАБОТАННОЕ МОТОРНОЕ МАСЛО**А.В. ГУЩА, М.А. ЧЕРНИЧЕНКО**
(Представлено: Л.И. ЛАГУН)

Отработанное масло – это смазочный материал, изготовленный из нефти, но через некоторое время он загрязняется. Со временем любое масло, независимо от того, является оно минеральным или синтетическим, теряет свои свойства и становится непригодным для дальнейшей эксплуатации.

Отработанное масло – это продукт, образованный в результате деятельности электростанций, производственной деятельности, кроме того его источником являются миллионы простых потребителей, обладающие автомобилем. Современными стандартами установлены следующие виды отработанных масел, которые подлежат утилизации:

- трансмиссионные масла, не используемые как топливо в авиационном и судовом транспорте, а также в легковых и грузовых авто;
- трансмиссионные жидкости, не применяемые в качестве топлива для техники тяжелого типа, локомотивах, грузовых авто, с дизельными силовыми агрегатами;
- тормозные жидкости;
- гидравлические смазки;
- в двигателях, функционирующих на альтернативном топливе;
- смазочные материалы, используемые в газовых моторах.

Утилизации подлежат и промышленные виды отработанных масел. К таким относятся компрессорные и подшипниковые смазочные составы, электроизоляционные масла, теплоносители, смазки для кабелей и иные. Масло отработанное может нести угрозу, так как относится к опасным отходам второго и третьего класса. Оно опасно не только для окружающей среды, но и для людей. Оно считается даже более опасным, чем свежая нефть, так как в данном продукте содержатся механические и химические смеси, образованные в результате функционирования оборудования.

После использования смазочных материалов, значительная часть отработки попадает в дренажные и канализационные системы, загрязняя землю и питьевую воду. Часть масла перерабатывается, но на данный момент его количество не превышает 20% от общего объема. Так же как пачкается одежда ребенка, который играет в песочнице, так и загрязняется в результате эксплуатации отработка масла. Применение такого масла возможно повторно, если оно будет очищено от загрязнителей. Идея перерабатывать смазочные материалы возникла более 80 лет назад, но осуществлять ее начали намного позже. Вначале сырье сжигали и получали энергию, но около 40 лет назад его стали добавлять в свежее масло. Применение отработки масла зависит во многом от перерабатывающей технологии:

- продукт можно восстанавливать на месте использования. Применяются системы фильтрации, которые продлевают срок его эксплуатации. Метод популярен на крупных заводах и предприятиях;
- регенерация с целью получить новый материал для смазки. Данный процесс обычно предполагает фильтрацию сырья или воздействие с помощью тепла. Масло отработанное мало чем будет отличаться от масла, изготовленного из сырой нефти;
- переработка с целью получить топливный материал. Этот процесс включает в себя такие процедуры как фильтрация и отстаивание.

Нередко масло сжигается без предварительной подготовки. Этот вид переработки разрешен только в том случае, если оборудование, с помощью которого производится утилизация, проходит по определенным нормам и стандартам.

Организации, специализирующиеся на переработке отходов, собирают, загружают и перевозят отработанные масла в емкостях, специально разработанных для этого. Сегодня достаточно большое количество организаций и фирм предоставляют такие услуги всем желающим юридическим и физическим лицам, берут на себя ответственность за необходимую своевременную транспортировку машинного масла, эксплуатация которого больше не представляется возможной. Желающим воспользоваться этими услугами, достаточно созвониться с такой организацией и решить все вопросы относительно доставки, а также оплаты предоставляемых услуг.

В 2015 году на территории Республики Беларусь открылся завод по переработке отработанного моторного масла в поселке Крупский Минской области (рис.).



Рисунок. – Завод в п. Крупский - МДД-БЕЛ

Завод отчасти решает проблему утилизации моторных отходов. Фуры с цистернами привозят отработанное масло на завод. Далее к цистернам подсоединяют шланги, из машины отработанное масло (моторное, гидравлическое или компрессорное), поступает в приемные резервуары. В резервуарах масло проверяют, его отправляют в лаборатории, где дают заключение. Переработанную готовую продукцию переливают в прибывшие цистерны и отправляют заказчику. Ежедневно завод перерабатывает до 40 т жидкости.

Если достаточного количества отработанного масла еще не набралось, не всем автолюбителям захочется обращаться в организацию, предоставляющую услуги по утилизации. В некоторых ситуациях, нередко возникающих в хозяйстве, допустимо применение отработанных смазочных материалов. Например, отработанное масло может использоваться в качестве горючих материалов для буржеек или каких-нибудь других механизмов, расположенных в нежилых помещениях. Отработанные смазывающие материалы славятся возможностью защиты от коррозии, поэтому нередко практикуется их использование в качестве покрытия определенных автомобильных комплектующих. Смазывающие свойства дают возможность улучшить качество работы бензопил, избавиться от надоедливого скрипа дверных петель. Такое полезное вещество можно эксплуатировать в хозяйстве по-разному. Нередко плотность отработанного масла способствует защите деревянных конструкций от гниения, атаки насекомых или других негативных эффектов. Желательно хранить определенное количество отработанной жидкости, поскольку необходимость в таких веществах возникает довольно часто. Использование уже отработанного масла актуально для поддержки интенсивности горения костров в походных условиях. Нередко смазочными средствами пропитываются железнодорожные шпалы.

Миф: масло, созданное из отработанных масел, хуже обычного.

На самом деле: такое масло ничем не уступает обычным и соответствует всем заявленным на его упаковке характеристикам. По сути, это совершенно такое же моторное масло, но созданное с меньшей себестоимостью. В настоящее время существуют эффективные технологии получения полноценного базового масла из отработанного. Они широко применяются в европейских странах, где никого не смущает уведомление на канистре, говорящее о том, что база продукта получена регенерацией отработки.

ЛИТЕРАТУРА

1. Моторные масла / Р. Балтенас [и др.]. – СПб. : Альфа-Лаб, 2000. – 272 с.
2. Гнатченко, И.И. Автомобильные масла, смазки, присадки : справ. пособие / И.И. Гнатченко, В.А. Бородин, В.Р. Репников. СПб. : Полигон АСТ, 2000. – 360 с.
3. Итинская, Н.И. Топливо, масло и технические жидкости / Н.И. Итинская, Н.А. Кузнецов. – М. : Агропромиздат, 1989. – 304 с.
4. Хитрюк, В.А. Автомобильные эксплуатационные материалы : учеб. пособие / В.А. Хитрюк, А.К. Трубилов. – Минск : РИПО, 2013. – 323 с.

УДК 629.331(075)

ДИАГНОСТИКА СОВРЕМЕННЫХ БЕНЗИНОВЫХ ДВИГАТЕЛЕЙ**Д.В. ЗАКРЕВСКИЙ, Н.Я. ЗУБОВИЧ**
(Представлено: В.В. КОСТРИЦКИЙ)

Представлен наиболее оптимальный порядок проведения диагностики современных бензиновых двигателей, рассмотрено оборудование и основные неисправности, а также осциллограммы исправного и неисправного двигателя.

подавляющее большинство современных автомобилей оснащено бензиновым двигателем внутреннего сгорания. Благодаря новейшим разработкам автомобилестроителей, двигатель может прослужить своему хозяину верой и правдой на протяжении не одного десятилетия при условии правильной эксплуатации и своевременной диагностики.

Преимущества своевременной диагностики двигателя:

- Значительная экономия материальных средств. Диагностика современных бензиновых двигателей позволяет вовремя обнаружить небольшие неисправности и устранить их, дабы избежать в дальнейшем серьезной поломки, устранение которой будет стоить значительно дороже или привести к полному выходу из строя двигателя.
- Гарантия безопасности на дорогах. Неисправность двигателя (в особенности, электронных систем) может привести к непредсказуемому поведению автомобиля при движении или к невозможности тронуться после остановки.

В данной работе мы рассмотрели наиболее эффективные методы диагностирования двигателя при помощи современных средств и выбрали наиболее оптимальный порядок проведения диагностики.

Порядок проведения диагностики таков:

- проверка механики двигателя;
- проверка системы зажигания;
- диагностика с помощью системного сканера и осциллографа.

Первый этап. Диагност в первую очередь должен уметь различать неисправности механики двигателя и неисправности системы управления двигателем. Оценив состояние механики двигателя, можно принять решение о целесообразности диагностики и ремонта системы управления двигателем. Оценка состояния механики двигателя производится по графику разрежения во впускном коллекторе на режиме прокрутки двигателя стартером. Для проведения теста, датчик разрежения Dх должен быть подсоединён к впускному коллектору двигателя диагностируемого автомобиля. Пуск двигателя необходимо заблокировать путём отключения зажигания либо подачи топлива. Датчик разрежения подключается к USB Autoscore и его сигнал записывается в течение 5-6 секунд в режиме прокрутки двигателя стартером.

По полученному графику можно выявить неправильное взаимное положение коленчатого и распределительных валов, закоксованные впускные клапана, забитый катализатор, оценить равномерность открытия впускных клапанов, состояние цилиндропоршневой группы. Если двигатель исправен, то полученный график по форме близок к синусоиде без заметных поцилиндрических различий по форме и амплитуде (рис. 1).

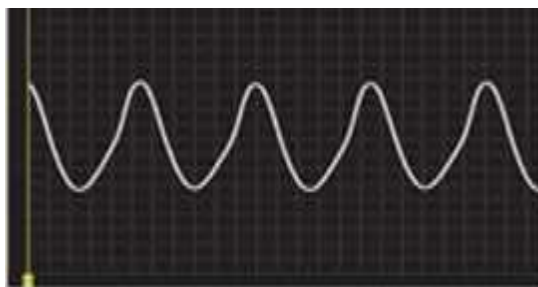


Рисунок 1. – График исправного двигателя

В случае неправильного взаимного положения коленчатого и газораспределительного валов, график разрежения во впускном коллекторе принимает "пилообразную" форму (рис. 2).

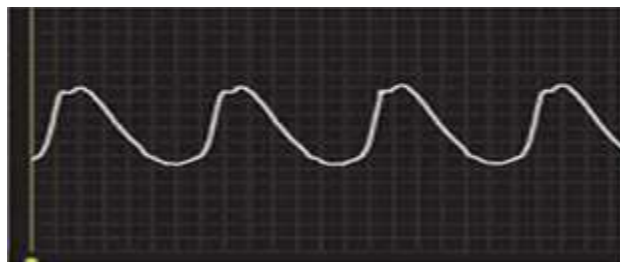


Рисунок 2. – График разрежения во впускном коллекторе при неправильном взаимном положении коленчатого и газораспределительного валов

По направлению наклона "пилы" можно выявить, распределвал смещён в сторону опоздания или опережения. Например, если разрежение, создаваемое отдельно взятым цилиндром, возрастает медленно, а падает быстро, то это указывает на опережение открытия и закрытия впускных клапанов.

"Зашумленный" график разрежения во впускном коллекторе на режиме прокрутки стартером указывает на значительный нагар на тарелках впускных клапанов (рис. 3).

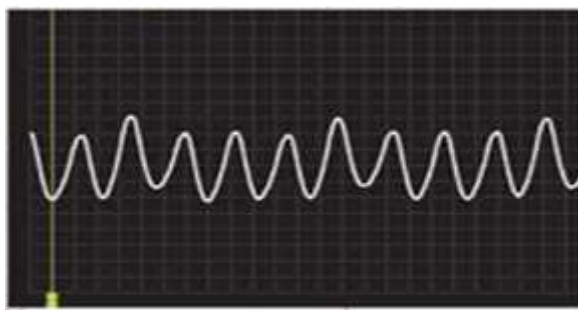


Рисунок 3. – Значительный нагар на тарелках впускных

Периодично повторяющиеся поцилиндровые различия по форме и/или амплитуде указывают на неодинаковое количество всасываемой топливовоздушной смеси для разных цилиндров, вызванное неисправностями механики двигателя.

Таким образом, диагност за очень короткое время может оценить состояние механики двигателя и принять решение, проводить диагностику системы управления двигателем или прежде более глубоко продиагностировать механику двигателя и устранить выявленные неисправности.

В случае необходимости, можно оценить состояние компрессионных колец и рабочих поверхностей цилиндров по уровню прорыва газов в картер двигателя. Уровень прорыва газов через сопряжения "кольца цилиндр" можно оценить и поцилиндрово сравнить по графику пульсаций давления в картере двигателя. График пульсаций давления в картере работающего двигателя можно получить с помощью датчика разрежения Dх. Для этого датчик необходимо подсоединить к колодцу масляного шупа, предварительно вынув шуп уровня масла, и подключить датчик к USB Autoscope. После пуска двигателя диагност увидит на мониторе компьютера график пульсаций давления в картере двигателя (рис. 4).

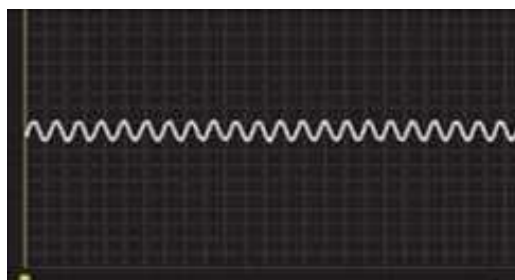


Рисунок 4. – График пульсаций давления в картере двигателя

По полученному графику можно выявить увеличенный прорыв газов в картер по циклично повторяющемуся увеличенному уровню прорыва газов в картер двигателя (рис. 5).

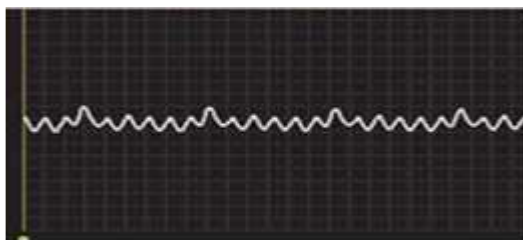


Рисунок 5. – График увеличенного уровня прорыва газов в картер двигателя

Увеличенный прорыв газов в картер может быть вызван неплотным прилеганием компрессионных колец к зеркалу цилиндра из-за их износа или поломки. Непостоянные неисправности механики двигателя, например, подвисающий клапан, неплотно закрывающийся клапан, проявляются как потраивание двигателя. Выявить их можно по неравномерным пульсациям отработавших газов. Для получения графика пульсаций отработавших газов, датчик Dх должен быть установлен в выхлопную трубу диагностируемого автомобиля и подключен к USB Autoscope. Теперь необходимо запустить двигатель. После пуска двигателя на мониторе компьютера диагност сможет просмотреть график пульсаций отработавших газов. В случае, если все цилиндры двигателя работают нормально и без перебоев, то пульсации на полученном графике имеют также равномерный характер (рис. 6).

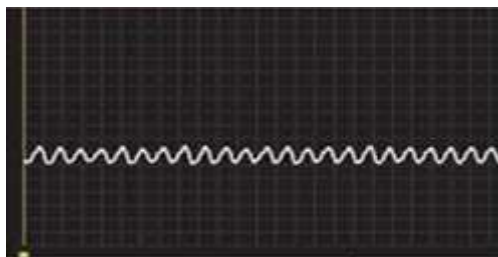


Рисунок 6. – График пульсаций отработавших газов исправного двигателя

"Потраивание", а также "троение" одного из цилиндров выявляется на графике пульсаций отработавших газов по отклонению формы и амплитуды пульсаций работающего с перебоями цилиндра (рис. 7).

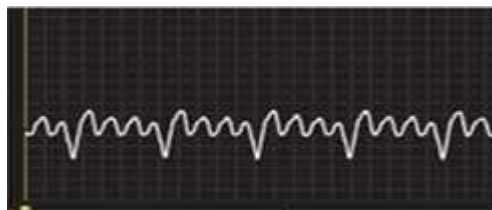


Рисунок 7. – График пульсаций отработавших газов двигателя, работающего с перебоями

Оценив состояние механики двигателя и убедившись в нормальном её состоянии, диагност может с уверенностью проводить диагностику и ремонт системы управления двигателем. Но практика показала, что около 60% двигателей, поступивших "на диагностику инжектора", имеют значительные неисправности в механической части двигателя. Такие автомобили в первую очередь требуют диагностики и ремонта/регулировок механической части двигателя. Если это упустить, как зачастую и происходит, то диагностика и ремонт системы управления двигателем оказываются неэффективными.

Второй этап. Диагностика системы управления бензинового двигателя всегда должна начинаться с проверки исправности системы зажигания. Диагностика систем зажигания проводится с помощью мотортестера.

Мотортестер, по сути, является осциллографом, специально адаптированным для диагностики автомобильной электроники, и в первую очередь высоковольтных цепей систем зажигания.

Для подключения к высоковольтным цепям систем зажигания, мотортестеры оснащаются высоковольтными датчиками различных типов и снабжены специальным режимом отображения осциллограм-

мы высокого напряжения систем зажигания – "парад цилиндров". В этом режиме в реальном времени отображаются параметры импульсов зажигания, такие, как пробивное напряжение, время и напряжение горения искры (для каждого цилиндра индивидуально).

Любая неисправность в системе зажигания, как в первичной, так и во вторичной цепи, определённым образом влияет на форму и параметры импульсов высокого напряжения во вторичной цепи системы зажигания. По отклонениям формы/параметров высоковольтных импульсов можно выявить неисправность. Например, если изолятор свечи зажигания загрязнён или "засвинцован", то высокое напряжение стекает по загрязнённому изолятору свечи на "массу" и разряд между электродами свечи не происходит, что приводит к пропускам зажигания (рис. 8).

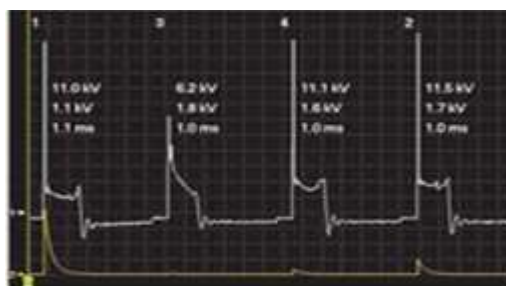


Рисунок 8. – График двигателя с загрязнённым изолятором свечи

Стеkanie высокого напряжения на "массу" может быть выявлено по заниженному пробивному напряжению и характерному изменению формы импульса высокого напряжения, отображаемому с помощью мотортестера.

В системах зажигания с силиконовыми высоковольтными проводами часто возникает обрыв высоковольтного провода из-за выгорания токопроводящего сердечника (рис. 9).



Рисунок 9. – График двигателя с оборванным высоковольтным проводом

Искровой разряд в таком случае может иметь очень короткое время горения либо вообще отсутствовать в зависимости от длины выгоревшего участка и режима работы двигателя.

Таким образом, по отображаемым мотортестером осциллограммам напряжений в высоковольтных цепях, диагност может выявить неисправности системы зажигания.

Практически в 30–40% продиагностированных систем зажигания выявляются неисправности.

Третий этап. Диагностика системы управления двигателем через диагностический разъём с помощью системного сканера.

В блоки управления двигателем современных автомобилей встроена функция самодиагностики, позволяющая выявлять неисправности датчиков, электропроводки и самого блока. Системный сканер считывает через диагностический разъём из памяти блока управления коды, выявленных системой самодиагностики неисправностей. Проведение диагностики с помощью системного сканера целесообразно в случае, если на панели приборов диагностируемого автомобиля загорелась сигнальная лампа "CHECK ENGINE". Система самодиагностики достоверно выявляет обрывы и короткие замыкания в проводке датчиков и исполнительных механизмов. Отклонения же параметров компонентов системы управления двигателем, система самодиагностики в большинстве случаев выявить не способна. Для подобных ситуаций, блоки управления также оснащаются функциями, позволяющими передавать фактические параметры (параметры, измеряемые датчиками и рассчитываемые блоком управления) через сканер для отображения на ПК. Данная функция позволяет диагносту получить дополнительную информацию для выявления истинной неисправности. Но так как дискретность передаваемых через сканер данных

очень низка, функция передачи фактических величин пригодна для проверки параметров лишь на установившихся режимах работы двигателя и медленно изменяющихся параметров.

В случае, если диагностика системы управления двигателем с помощью системного сканера выявила неисправность какого-либо датчика или исполнительного механизма, её обязательно нужно перепроверить дополнительными измерениями. С помощью осциллографа диагност должен просмотреть осциллограммы входных и выходных электрических сигналов датчиков и исполнительных механизмов, и только после этого можно делать какие-либо выводы. Дополнительная перепроверка считанных с помощью системного сканера ошибок необходима из-за того, что очень часто система самодиагностики сохраняет ложный код неисправности. Например, в случае, если в систему выпуска отработавших газов подмешивается воздух перед местом установки лямбдазонда, то диагност с помощью сканера вероятно считает такую ошибку. Дополнительный воздух в систему выпуска отработавших газов зачастую попадает через прогоревшие уплотнительные прокладки или трещины выпускного коллектора. Из-за негерметичности, состав газов в выпускной системе изменяется, и лямбдазонд регистрирует повышенное содержание кислорода в отработавших газах. Система управления двигателем на это реагирует обогащением топливовоздушной смеси, но лямбдазонд по-прежнему регистрирует высокий уровень содержания кислорода. В результате в память неисправностей записывается ошибка работы лямбдазонда. Подобная ошибка может быть записана в память неисправностей системы самодиагностики по причине загрязнения датчика массового расхода воздуха. Загрязнение расходомера воздуха происходит в основном из-за несвоевременной замены воздушного фильтра.

Из-за загрязнения, выходной сигнала датчика перестаёт соответствовать количеству протекающего воздуха, кроме того значительно повышается его инерционность. Вследствие этого блок управления уже неправильно рассчитывает необходимое количество топлива, что приводит к обеднению топливовоздушной смеси, особенно на переходных режимах. Лямбдазонд при этом фиксирует повышенный уровень содержания кислорода в отработанных газах, но система самодиагностики заносит в память неисправностей код ошибки именно лямбдазонда, а не датчика расхода воздуха. Проверить исправность датчика массового расхода воздуха можно на режиме резкой перегазовки по осциллограмме напряжения его выходного сигнала (рис. 10).

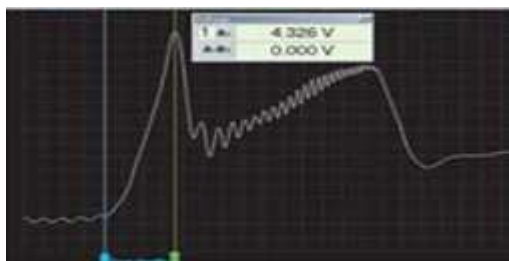


Рисунок 10. – График исправного датчика массового расхода воздуха

Осциллограмму напряжения выходного сигнала датчика необходимо записать, характерные её участки измерить. Иногда случается, что система самодиагностики не способна выявить присутствующую неисправность. Например, в памяти блока управления "судорожно дёргающегося" двигателя автомобиля Nissan Maxima никаких кодов неисправностей сохранено не было, все фактически параметры, отображаемые через сканер, также были в норме. Но при этом с помощью осциллографа была получена следующая осциллограмма напряжения выходного сигнала датчика частоты вращения двигателя (рис. 11).



Рисунок 11. – График напряжения выходного сигнала датчика частоты вращения неисправного двигателя

По осциллограмме были обнаружены чётко выраженные пропуски и нарушения формы импульсов, чего не должно быть на сигнале датчика, именуемого как "Engine Speed Sensor". Так как искажённые импульсы повторялись систематично, под подозрением оказался задающий зубчатый диск. Датчик был установлен на стыке двигателя и коробки передач. После того как датчик был снят, через его посадочное отверстие можно было осмотреть задающий зубчатый диск с шириной зуба 3 мм.

Путём медленного проворачивания коленчатого вала визуально были выявлены серьёзные повреждения зубьев диска в двух местах. Диск был повреждён при замене ремня ГРМ, когда для фиксации коленчатого вала вместо зубчатого венца маховика по ошибке был застопорён задающий зубчатый венец датчика частоты вращения двигателя.

Система самодиагностики также не способна достоверно выявить отклонения параметров датчиков. А такие "невидимые" для системы самодиагностики неисправности, как, например, старение или отравление лямбдазонда, разгерметизация его измерительных камер, загрязнение датчика массового расхода воздуха и т. д. приводят к очень серьёзным нарушениям в работе системы управления двигателем, а сохранённые коды ошибок при этом не соответствуют действительности. Выявить истинные неисправности в таком случае можно по осциллограммам напряжений сигналов.

Таким образом, по итогам исследования можно сделать следующие выводы.

1. Работа сканером более проста и удобна, однако обмен информацией между ЭБУ и сканером происходит достаточно медленно, и важные моменты сигнала можно попросту не обнаружить.

2. Мотортестер позволяет оценить сигнал датчика очень качественно и подробно, не пропустив ни малейшей детали, хотя трудоемкость его применения выше, чем у сканера.

3. Современный автомобиль невозможно представить без электронной системы управления, поэтому многие СТО стремятся предоставлять услуги для диагностирования этой системы, при этом зачастую обходятся только сканером, а как показывает практика, для оценки состояния ЭСУД такого диагностического оборудования недостаточно. Использование мотортестера и осциллографа требует специальных знаний, умений и навыков, которые лежат на границе дисциплин, связанных с устройством автомобиля и дисциплин электротехники и электроники.

ЛИТЕРАТУРА

1. Савич, Е.Л. Техническая эксплуатация автомобилей : учеб. пособие. В 3 ч. Ч. 2. Методы и средства диагностики и технического обслуживания автомобилей / Е.Л. Савич. – Минск : Новое знание ; М. : ИНФРА-М, 2015. – 364 с.
2. Режим доступа: www.InjectorService.com.ua. – Дата доступа: 04.05.2018.

УДК 629.331(075)

**АЛЬТЕРНАТИВНЫЕ МЕТОДЫ ДИАГНОСТИРОВАНИЯ СИСТЕМ COMMON RAIL
СОВРЕМЕННЫХ ДИЗЕЛЬНЫХ ДВИГАТЕЛЕЙ****Д.С. ШЕВЧЕНКО, В.В. ВОЙТЕХОВИЧ***(Представлено: В.В. КОСТРИЦКИЙ)*

Проведен анализ эффективности, трудоемкости и результативности альтернативного метода диагностирования систем common rail современных дизельных автомобилей. Выявлены преимущества и недостатки альтернативной методики диагностирования двигателя.

Введение. Система впрыска Common Rail является современной системой впрыска топлива дизельных двигателей. Работа системы Common Rail основана на подаче топлива к форсункам от общего аккумулятора высокого давления – топливной рампы. Система впрыска разработана специалистами фирмы Bosch.

Применение данной системы позволяет достигнуть снижения расхода топлива, токсичности отработавших газов, уровня шума дизеля. Главным преимуществом системы Common Rail является широкий диапазон регулирования давления топлива и момента начала впрыска, которые достигнуты за счет разделения процессов создания давления и впрыска.

Конструктивно система впрыска Common Rail составляет контур высокого давления топливной системы дизельного двигателя. В системе используется непосредственный впрыск топлива, т.е. дизельное топливо впрыскивается непосредственно в камеру сгорания. Система Common Rail включает топливный насос высокого давления, клапан дозирования топлива, регулятор давления топлива (контрольный клапан), топливную рампу и форсунки. Все элементы объединяют топливопроводы.

Топливо из топливного бака забирается топливоподкачивающим насосом (низкого давления), и через топливный фильтр поступает в топливный насос высокого давления (ТНВД). ТНВД подаёт топливо в напорную магистраль, которая играет роль аккумулятора давления. Блок управления регулирует производительность ТНВД для поддержания необходимого давления в магистрали по мере расхода топлива.

Топливная магистраль соединяется топливопроводами с форсунками. В каждую форсунку встроен управляющий клапан — электромагнитный или пьезоэлектрический. По команде от блока управления клапан открывается, впрыскивая необходимую порцию топлива в цилиндр [1].

Преимущества. Давление, при котором происходит впрыск топлива, можно поддерживать вне зависимости от скорости вращения коленчатого вала двигателя, и оно остаётся практически постоянно высоким в течение всего цикла подачи топлива, что особенно важно для стабилизации горения на холостом ходу и на малых оборотах при работе с частичной нагрузкой;

При использовании аккумуляторной системы подачи топлива момент начала и конца подачи может в широких пределах регулироваться ЭБУ. Это позволяет более точно дозировать топливо, а также осуществлять подачу топлива несколькими порциями в течение рабочего цикла — для более полного сгорания топлива;

Конструкция common rail проще, чем у системы ТНВД с форсунками, её ремонтпригодность выше.

Недостатки. Более сложные форсунки, которые требуют относительно частой замены, по сравнению с традиционной системой подачи топлива;

Система перестаёт работать при разгерметизации любого элемента высокого давления, например, при неисправности одной из форсунок, когда её клапан постоянно находится в открытом положении;

Более высокие требования к качеству топлива, чем у традиционных систем.

В легковых автомобилях используется несколько видов систем Common Rail. Упрощенно их можно разбить на два типа (электромагнитные и пьезоэлектрические) и четыре производителя (Bosch, Continental, Delphi, Denso). Самым универсальным является лидер рынка - Bosch, который производит оба типа форсунок: электромагнитные и пьезоэлектрические. В гораздо меньших масштабах оба вида форсунок производят Delphi и Denso. Continental (Siemens) ограничивается исключительно пьезоэлектрической техникой [2].

В качестве примера рассмотрим систему Common Rail фирмы Bosch.

Диагностика дизелей с системой Common Rail. Для того чтобы точно определить в чем именно заключается неисправность надо принимать во внимание и оценивать состояние трех взаимосвязанных частей:

1. Состояние механических узлов двигателя (турбина, цилиндропоршневая часть дизеля, правильность установки фаз ГРМ).
2. Электронная система управления двигателем (исправность датчиков, проводки).
3. Гидравлическая часть (система топливоподачи, ТНВД, форсунки).

Если проверять подряд все названные компоненты это займет очень много времени. Чтобы ускорить поиск возможных отклонений блок управления снабжен функцией мониторинга исправности датчиков и исполнительных механизмов. При обнаружении блоком отклонений в показании, каких-либо датчиков в оперативной памяти прописывается ошибка. В зависимости от значимости ошибки впрыскивание топлива продолжается или двигатель останавливается. С помощью сканера в меню параметры можно посмотреть реальные показания и реакцию датчиков на различные возмущения. Важной особенностью Common Rail является то что каждая форсунка управляется индивидуально т.е. на каждой форсунке прописываются свои коэффициенты топливной коррекции, по которым можно судить об состоянии каждого цилиндра и соответствующей форсунки.

В качестве примера приведем данные сканирования некоторых параметров в системе Common Rail на холостом ходе (табл. 1).

Таблица 1. – Данные сканирования параметров на холостом ходе

Частота вращения коленвала	750 об/мин
Температура охлаждающей жидкости	88
Положение педали подачи топлива	0%
Предписанное давление в аккумуляторе высокого давления	261 бар
Реальное давление	264 бар
Цикловая подача одной форсунки	13.8 ME
Сквозность управляющего сигнала электромагнитного клапана регулятора давления	16%
Коррекция равномерности работы двигателя	
Для цилиндра 1	-3.35
Для цилиндра 3	0.00
Для цилиндра 4	2.58
Для цилиндра 2	0.65

Коррекция предназначена для компенсации отклонений по цилиндрам в механике двигателя и гидравлике системы впрыска топлива, возникающих при серийном производстве. Неравномерность определяется с помощью датчика оборотов. Если какой-то цилиндр отличается более чем на 30% от заданного значения, это считается не нормальным и блок управления пытается выправить положение изменением топливоподачи. Если мы обнаружили значительную топливо-коррекцию в каком-то цилиндре, это может быть связано или с низкой компрессией в цилиндре (прогар клапана, например) или неисправностью форсунки.

При неисправностях, которые не выводят систему из строя, блок управления ограничивает топливоподачу. К таким дефектам относятся неисправности датчиков температуры, слишком низкое давление наддува, неполадки с замером расхода воздуха, или выход из строя датчика положения педали газа.

Из соображений безопасности система останавливает двигатель при следующих условиях:

- Выходит из строя форсунка или сильно падает давление в аккумуляторе
- Превышение давление топлива в рампе выше MAX (порядка 1500бар)
- Выход из строя электромагнитного клапана регулирования давления

Двигатель невозможно завести при неработающих датчиках оборотов и положения распределительного вала.

На основании вышесказанного диагностику желательно начинать с подключения сканера (рис. 1), чтобы определить возможные направления поиска и потому что это наименее трудоемкая операция.

Неисправности, требующие диагностики при помощи гидравлических тестеров:

1. Невозможность запуска двигателя, или двигатель глохнет после запуска - рекомендуется провести тест баланса мощности по цилиндрам при помощи сканера (для систем Bosch), или провести очередную проверку инжекторов отключая их по очереди (для системы Delphi)

2. Если имеются неисправности, отличающиеся от описанных выше, например, вибрация двигателя, выброс черного/белого дыма при запуске или на холостом ходу.

Проблема может заключаться в разности количества топлива, проходящего через каждый инжектор.

Процедура диагностики зависит от симптома [3].



Рисунок 1. – Сканер AUTOCOM CDP PLUS

Тест контура низкого давления. Огромное количество проблем с этими системами лежит не в области высокого давления, а в низкой цепи от бака до ТНВД. Комплект CRDI-500 предназначен для быстрой проверки состояния цепи низкого давления через подключение к трубопроводу в удобном месте (например, у топливного фильтра или на выходе из бака или перед ТНВД) с помощью стандартных быстросъемных адаптеров. С их помощью можно подключиться к 70% топливных магистралей без дополнительных адаптеров.



Рисунок 2. – Комплект CRDI-500

Вначале требуется проверить правильность подсоединений, если они правильные – ищем утечки в штуцерах. Если утечек нет – Проверяем исправность и соответствие топливного фильтра.

Далее требуется отсоединить топливную трубку от топливного фильтра и подсоединить манометр низкого давления, или вакуумметр (CRT-1050), в зависимости от системы двигателя.

Для этого требуется трубка для подключения манометра (CRT-1052), адаптер соединения с трубкой (CRT-1054), адаптер соединения (CRT-1053), заглушка топливного фильтра (CRT-1055).

Запускаем двигатель и держим на холостых оборотах примерно 5 с, затем глушим двигатель. Читаем показания манометра или вакуумметра. Анализируем эти данные.

В таблицах 2 и 3 представлены показания для манометра и вакуумметра. [4]

Таблица 2. – Показания манометра

Этап	Давление	Вывод
1	1,5–3 кг/см ²	Норма
2	4–6 кг/см ²	Загрязнение фильтра или топливопровода
3	0–1,5 кг/см ²	Утечка из насоса или топливопровода

Таблица 3 – Показания вакуумметра

Этап	Вакуум	Вывод
1	8–18 смHg	Норма
2	20–60 смHg	Загрязнение фильтра или топливопровода (насос в норме)
3	0–7 смHg	Утечка воздуха в системе или неисправность насоса

Тест контура высокого давления. Еще одна интересная особенность Common Rail: состояние электромагнитного клапана можно определить с помощью обратного слива топлива. Только на инжекторе Common Rail можно померить обратный слив топлива и посредством этой проверки увидеть состояние электромагнитного клапана форсунки Common Rail. В случае негерметичности электромагнитного клапана то количество топлива, которое должно попасть в камеру сгорания, попадет в обратный слив, и таким образом проявляется неисправность автомобиля. Проанализировать эту информацию можно, используя специальный аппарат.

Прибор состоит из набора мензурок, имеющих или не имеющих деления. От них отходят прозрачные трубки с быстросъемным соединением на конце. Подключая их к форсункам Common Rail прямо на двигателе, можно наблюдать равномерность обратного сброса (как заполняются эти мензурки) на различных режимах за определенный промежуток времени. Процедура не занимает много времени, но с диагностической точки зрения очень эффективна.

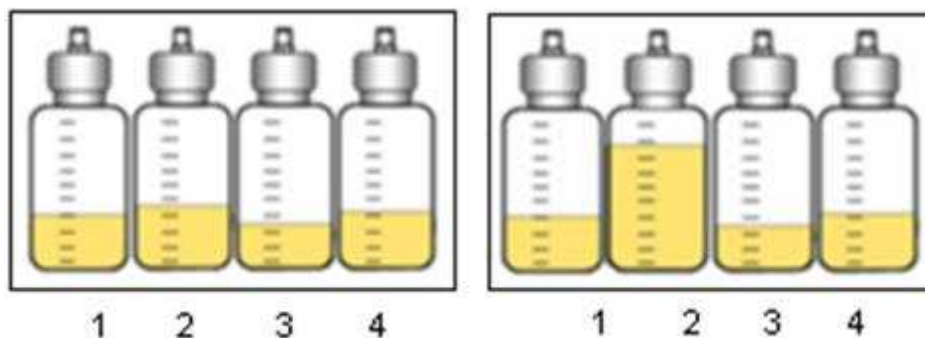


Рисунок 3. – Колбы (CRT-1030)

Инжектор, у которого показания в три раза отклоняются от нормы подлежит замене.

Проверка ТНВД. Для проверки используем Тестер Common Rail.

1. Очищаем от дизельного топлива клапан-регулятор (CRT-1020), адаптер клапана-регулятора (CRT-1023) и заглушки (CRT-1021 или CRT-1022).
2. Удаляем все 4 трубки форсунок с рампы и отсоединяем коннектор датчика давления в рампе.
3. Подсоединяем клапан-регулятор (CRT-1020), заглушки (CRT-1021 или CRT-1022), пыльники (CRT-1035), адаптер-коннектор (CRT-1041, CRT-1042, CRT-1043) и индикатор высокого давления (CRT-1040) к рампе, как показано на рисунке.
4. Отсоединяем коннектор IMV от ТНВД.
5. Прокручиваем двигатель в течение 5-6 секунд. Для получения наиболее точных результатов проведите тест дважды и выбираем наибольший параметр как измерительный эталон.

Если давление топлива, показанное на индикаторе соответствует спецификации, ТНВД в норме.

Спецификация по параметрам давления ТНВД:

BOSCH System: 1000~1500 ATM

DELPHI System: 1050~1600 ATM

Тест регулятора давления топлива. Для его проведения повторяем схему предыдущего опыта, только к обратке регулятора соединяем мерную мензурку. Вращаем стартером двигатель в течении 5 с и замечаем количество топлива вытекающего с обратки, оно не должно превышать 10 мм [5].

Таким образом, по итогам исследования можно сделать вывод:

Выполнение диагностики топливной системы при помощи альтернативных методов значительно проще, однако достаточно точно позволяют выявить неисправность, как и традиционные методы.

Альтернативный метод диагностики имеет преимущество в том, что требует значительно меньших затрат, так как при использовании данного метода не требуется дорогостоящее оборудование, используемое на станциях диагностики.

ЛИТЕРАТУРА

1. Системы современного автомобиля [Электронный ресурс] – Режим доступа: <http://systemsauto.ru/>. – Дата доступа 25.09.2018.
2. Википедия, современная энциклопедия [Электронный ресурс] – Режим доступа: <https://ru.wikipedia.org/>. – Дата доступа 25.09.2018.
3. Автомастер – ремонт, диагностика автомобилей и двигателей [Электронный ресурс]. – Режим доступа: <https://www.avtomaster21.ru/>. – Дата доступа 25.09.2018.
4. Диагностика дизельных двигателей [Электронный ресурс]. – Режим доступа: <http://www.common-rail.ru>, свободный. – Дата доступа 25.09.2018.
5. Kia CRT-1500 Common Rail Tester – Руководство Пользователя [Электронный ресурс]. – Режим доступа: <http://www.common-rail.ru>. – Дата доступа 25.09.2018.

УДК 621.7.012.3

**АНАЛИЗ ТЕХНОЛОГИЙ ВРЕЗАНИЯ ПРИ ФРЕЗЕРОВАНИИ КАРМАНОВ
НА СТАНКАХ С ЧПУ****Е.О. ЖИХОРЕВ***(Представлено: д-р техн. наук, проф. Н.Н. ПОПОК,
канд. техн. наук. Р.С. ХМЕЛЬНИЦКИЙ)*

Проанализированы технологии врезания, доступные в SolidCAM при фрезеровании карманов. Представлены их преимущества и недостатки. На примере детали «Корпус», производимой на ОАО «Измеритель», представлено сравнение затрачиваемого времени и количества кадров в управляющей программе для всех видов технологий.

Введение. В машиностроении корпусные детали имеют широкое распространение, так как они являются основной несущей частью машины, механизма или прибора. В соответствии с классификатором ЕСКД деталь корпусная – деталь коробчатой формы, имеющая одно или несколько базовых отверстий, с помощью которых другие детали могут монтироваться внутри корпуса в определенной кинематической взаимосвязи, а также корпуса приводных и распределительных систем, характеризующихся наличием рабочих отверстий, называемых базовыми [1].

Ранее [2,3] проводилось исследование, в котором было выявлено, что одним из наиболее часто встречающихся конструктивных элементов (КЭ) в корпусах является «карман». Из 26 проанализированных деталей, изготавливаемых в условиях ОАО «Измеритель», данный КЭ встречается во всех 100%. Ввиду этого рассмотрение технологий обработки кармана является актуальным.

Основная часть. При фрезеровании корпуса на станке с ЧПУ обработка кармана занимает значительную часть времени фрезерной операции, а также немалую часть управляющей программы (УП). Обработывается данный конструктивный элемент концевыми фрезами [4].

В процессе программирования обработки кармана технолог-программист использует два типа циклов: технология врезания и цикл снятия основного припуска кармана. Задавая различные комбинации этих двух циклов, он имеет возможность манипулировать временем обработки кармана.

Можно отметить, что при этом не учитывается отвод режущего инструмента (РИ) (в нашем случае концевой фрезы). Причина этого то, что данная часть программы занимает малую часть как по количеству кадров в УП, так и по времени. Кроме того, почти в 100% случаев отвод производится по нормали к контуру обработки на расстояние 1...2 мм.

При разработке управляющих программ обработки корпусных деталей в SolidCAM используются три варианта реализации данного цикла, а именно врезание вдоль одной оси, по двум осям (маятниковой подачей) и по трем осям (врезание по винтовой траектории) [5].

Сравнение всех трех вариантов реализации цикла врезания произведем на примере обработки кармана в детали «Корпус» (рис.) из алюминиевого сплава АМг4 концевой фрезой Ø16 мм. Глубина врезания – 10 мм. Расстояние от верхней поверхности заготовки, при достижении которой инструмент начинает перемещаться на рабочей подаче, зададим равное 2 мм. Результаты замеров представлены в таблице 1.

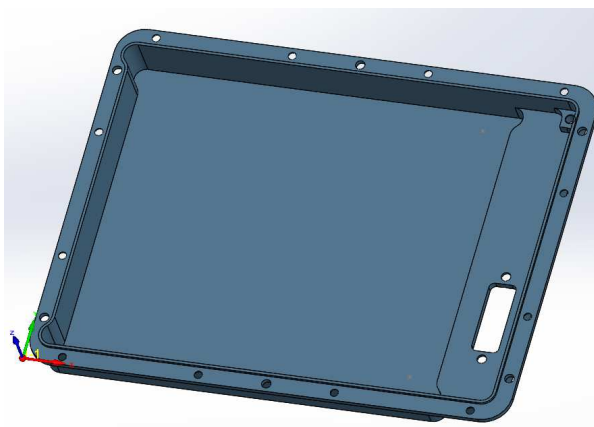
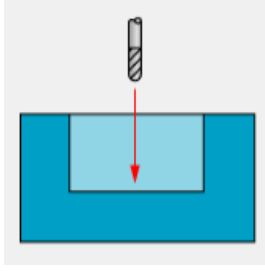
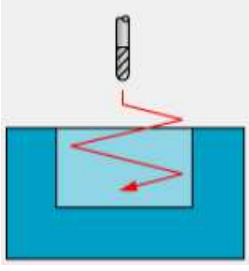
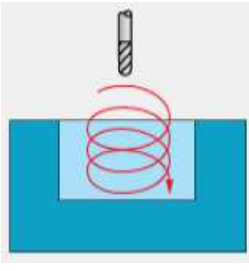


Рисунок. – Деталь «Корпус»

Таблица 1. – Время врезания РИ при различных технологиях

	По одной оси	По двум осям	По трем осям
Траектория движения			
Величина хода РИ	Глубина общего хода – 12 мм	Величина смещения по оси X – ± 9 мм Угол врезания - 15°	Диаметр спирали – 18 мм Угол схода спирали – 15°
Режимы резания	Подача, $S_m=15$ мм/мин; Обороты, $n=1300$ об/мин.	Подача, $S_m =30$ мм/мин; Обороты, $n=1300$ об/мин.	Подача, $S_m = 30$ мм/мин; Обороты, $n = 1300$ об/мин.
Время врезания	$T_{\text{врезания}} = 0,74$ мин	$T_{\text{врезания}} = 1,45$ мин	$T_{\text{врезания}} = 1,78$ мин
Количество кадров	3	7	36

Проанализировав данные, приведенные в таблице 1, можно выделить следующие преимущества и недостатки вышеописанных вариантов врезания инструмента в заготовку (табл. 2).

Таблица 2. – Преимущества и недостатки циклов врезания

Технологии врезания	По одной оси (вертикально)	По двум осям (линейное)	По трем осям (спиральное, винтовое)
Преимущества	– малое количество кадров в УП; – возможность программирования в любой точке кармана	– нагрузка на РИ распределяется по двум осям; – более высокие режимы резания по сравнению с вертикальным врезанием	– нагрузка распределена по трем осям; – траектория более компактна, чем линейное врезание; – более высокие режимы резания
Недостатки	– большая осевая нагрузка на РИ; – наименьшие режимы резания	– невозможно запрограммировать в любой точке кармана из-за траектории перемещения РИ; – занимает больше всего места в УП	– невозможно использовать на старых станках из-за особенностей конструкции самого станка, либо системы ЧПУ; – значительно большее количество кадров в УП по сравнению с линейным

Как видно из таблицы 2, третий вариант имеет наибольшее количество плюсов, но и недостатки являются существенными. При обработке на современных станках типа SPINNER VC860, DBZ проблем с этим не возникает – данные станки имеют возможность движения по трем координатам. В случае же обработки на старых станках типа ИР320ПМФ4, ИР500ПМФ4, ИС500ПМФ4, МАНО МС5 использование данного варианта врезания невозможно из-за отсутствия возможности перемещения по трем координатным осям. Поэтому вместо винтового врезания приходится использовать врезание по двум координатам, что конечно приводит к увеличению времени обработки, и как следствие к снижению производительности.

Выводы:

- врезание по трем осям занимает больше всего времени (1,78 мин против 1,45 по двум и 0,74 по одной оси);
- врезание по трем осям занимает больше места в УП, чем по двум;
- врезание по трем координатам, в отличие от линейного способа, могут осуществлять преимущественно современные станки;
- несмотря на наименьшие затраты по времени, первый способ врезания используется редко по нескольким причинам: во-первых, увеличивается износ инструмента и, следовательно, затраты на него; во-вторых, этот способ требует самых малых режимов резания из всех трех, что ведет к увеличению времени обработки. Кроме этого, врезание по двум координатам рекомендуют сами производители режущего инструмента, указывая в каталогах угол входа РИ в материал заготовки. Учтя эти рекомендации, тех-

нолог-программист сможет уменьшить износ инструмента в ходе обработки, а следовательно, повысить производительность,

Врезание по двум осям при обработке кармана является оптимальным с точки зрения затрат времени, времени работы инструмента, размера самого цикла в УП и универсальности при программировании на различные модели станков.

ЛИТЕРАТУРА

1. Классификатор ЕСКД. Классы 71, 72, 73, 74, 75, 76. Приложение. – М. : Изд-во стандартов, 1986. – 36 с.
2. Жихорев, Е.О. Программирование технологий врезания и обработки конструктивных элементов в виде «карманов» деталей на фрезерных станках с ЧПУ при помощи САМ-систем / Е.О. Жихорев, И.Н.Домкин.
3. Zhykharau, Y. Analysis of constrictive elements of housing details processed on CNC milling machine / Y. Zhykharau, N. Popok // European and National Dimension in Research. Machine-building : X Junior Researcher's Conference, Novopolotsk, 10-11 May 2018. – P. 158–159.
4. Sandvik Coromant [Электронный ресурс] // Старт / Техническая информация / Фрезерование / Обзор технологических решений / Методы обработки отверстий и выборки / Вскрытие карманов. – Режим доступа: https://www.sandvik.coromant.com/ru-ru/knowledge/milling/application_overview/holes_and_cavities/creating_openings/pages/default.aspx. – Дата доступа: 26.09.2018.
5. Руководство SolidCAM 2006 R10.1. – 106 с.

УДК 621.7.012.3

**ОПРЕДЕЛЕНИЕ ОПТИМАЛЬНОГО ЦИКЛА СНЯТИЯ ОСНОВНОГО ПРИПУСКА
ПРИ ФРЕЗЕРОВАНИИ КАРМАНОВ НА СТАНКАХ С ЧПУ****Е.О. ЖИХОРЕВ***(Представлено: д-р техн. наук, проф. Н.Н. ПОПОК,
канд. техн. наук. Р.С. ХМЕЛЬНИЦКИЙ)*

В настоящее время каждая САМ-система предлагает множество готовых циклов обработки, доступных для использования при разработке управляющей программы (УП), при фрезеровании, точении, сверлении, резьбонарезании и т.д. Кроме того, с помощью циклов можно программировать и другие части УП: подвод, отвод, врезание, переход между обрабатываемыми контурами и другие [1].

Одним из наиболее часто используемых циклов при программировании фрезерования корпусов является цикл обработки кармана [2]. Программирование обработки кармана имеет ряд сложностей:

1) большинство карманов имеет сложную геометрическую форму, из-за чего значительно усложняется расчет траектории перемещения инструмента;

2) в некоторых случаях карман имеет внутри себя площадки (бобышки), высота которых выше минимального уровня кармана. Их наличие требуется учитывать в процессе расчета траектории.

Циклы, доступные при программировании, значительно упрощают процесс написания управляющей программы. Достаточно указать контур, глубину, шаг врезания инструмента и технологию, после чего запустить расчет траектории.

Именно технология снятия основного припуска составляет большую часть времени фрезерования кармана. В зависимости от ее выбора и задаваемых параметров технолог-программист осуществляет управление временем обработки.

Рассмотрим преимущества и недостатки циклов фрезерования кармана, доступных в SolidCAM (табл. 1) [3].

Таблица 1 – Преимущества и недостатки циклов снятия основного припуска

Цикл обработки	Растровый	Контурный	Растровый+контурный
Преимущества	- самое меньшее количество кадров в УП; - самая простая с точки зрения расчета траектории	- самое меньшее количество следов от инструмента	
Недостатки	- невозможность задания попутного фрезерования		- невозможность задания попутного фрезерования; - невозможность убрать лишний черновой проход

Необходимо отметить, что обработка врезанием (плунжерное фрезерование) не рассматривается. причина этого, так как при написании УП на производствах Беларуси данная технология не используется.

Для определения оптимального цикла с точки зрения затрат времени смоделируем обработку кармана в детали «Корпус» (рис. 1). Материал детали – алюминиевый сплав АМг4.

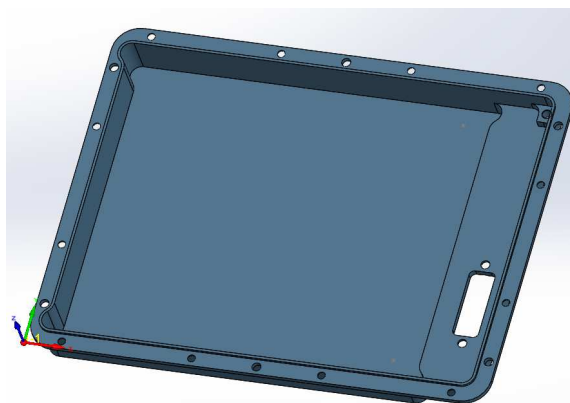


Рисунок 1. – Деталь «Корпус»

Условия обработки:

- материал детали – алюминиевый сплав АМг4.
- используемый инструмент: концевая фреза $\varnothing 16$ мм;
- режимы резания: черновая подача – 140 мм/мин, чистовая – 160 мм/мин, по оси Z – 30 мм/мин, скорость вращения шпинделя – 1300 об/мин;
- глубина кармана – 10 мм., размеры – 121x163 мм., величина чернового припуска на боковую сторону – 0,5 мм;
- тип врезания – по одной оси (вертикально).

Результаты моделирования представлены в Таблице 2. Траектория движения инструмента представлена на рисунке 2.

Таблица 2 – Результаты вычислений

Тип технологии	Растровая	Контурная	Растровая+контурная
Время обработки, $T_{\text{очн}}$, мин	15,43	15,97	18,99
Количество кадров УП	33	50	41

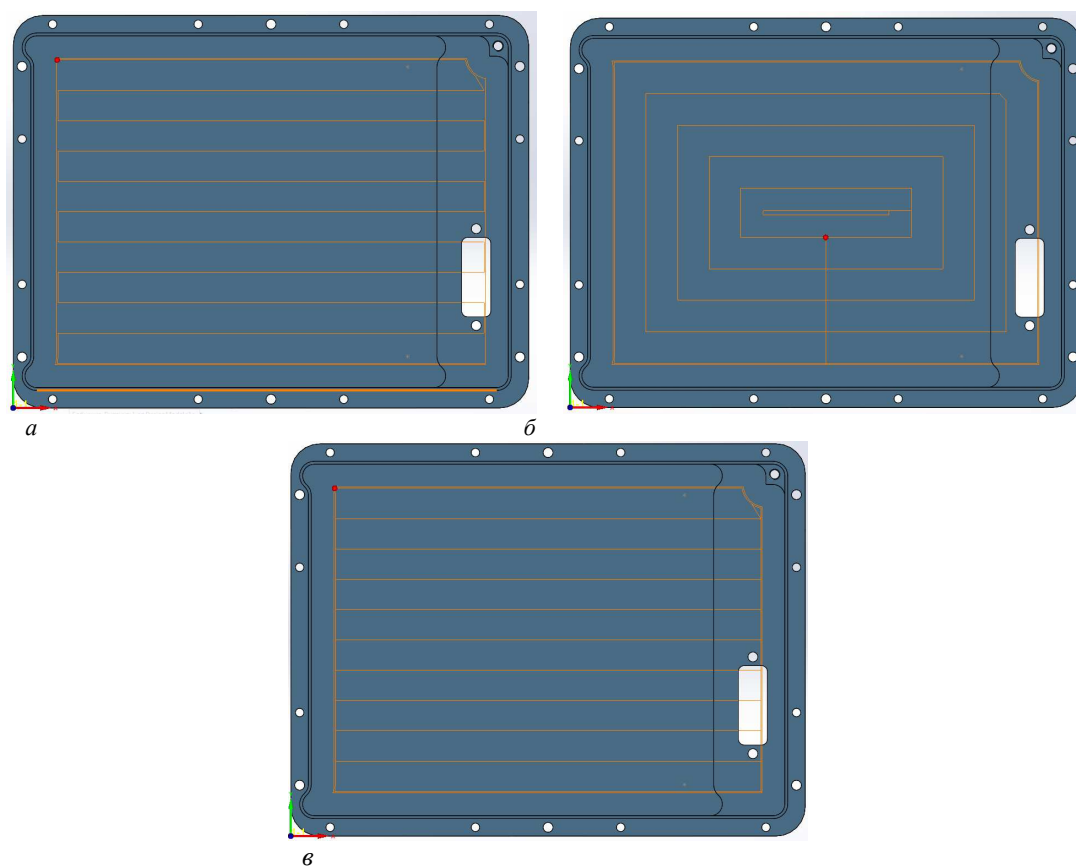


Рисунок 2. – Траектория движения РИ [3]
a – растровая; *б* – контурная; *в* – растровая+контурная

Выводы:

- цикл, сочетающий растровую и контурную обработки, практически идентичен по траектории обычному растровому, однако он имеет лишний проход инструмента по контуру, избавиться от которого нельзя. Из-за этого время обработки больше, чем у первого способа;
- если сравнивать растровый и контурный циклы, то более выгодна первая технология снятия припуска. Однако на практике применяется преимущественно контурный способ, несмотря на большую нагрузку системы ЧПУ и немного большее время обработки, ввиду того, что он оставляет меньшее количество следов на поверхности, а также при его использовании врезание осуществляется по центру кармана, что позволяет безопасно использовать цикл врезания по двум координатам, являющийся наиболее оптимальным.

ЛИТЕРАТУРА

1. Sandvik Coromant [Электронный ресурс] // Старт / Техническая информация / Фрезерование / Обзор технологических решений / Методы обработки отверстий и выборки / Вскрытие карманов. – Режим доступа: https://www.sandvik.coromant.com/ru-ru/knowledge/milling/application_overview/holes_and_cavities/creating_openings/pages/default.aspx. – Дата доступа: 26.09.2018.
2. Жихорев, Е.О. Программирование технологий врезания и обработки конструктивных элементов в виде «карманов» деталей на фрезерных станках с ЧПУ при помощи САМ-систем / Е.О. Жихорев, И.Н. Домкин.
3. Руководство SolidCAM 2006 R10.1. – 106 с.

УДК 629.331(075)

**СИСТЕМА НЕПОСРЕДСТВЕННОГО ВПРЫСКА ТОПЛИВА
И ЕЕ НЕДОСТАТКИ В БЕНЗИНОВЫХ ДВИГАТЕЛЯХ****О.Д. ПЛЮЩЕНКО, П.В. ЮШКЕВИЧ**
(Представлено: В.В. КОСТРИЦКИЙ)

Рассмотрены устройство и недостатки системы с непосредственным впрыском бензинового двигателя.

Введение. Система непосредственного впрыска топлива является самой современной системой впрыска топлива бензиновых двигателей. Работа системы основана на впрыске топлива непосредственно в камеру сгорания двигателя.

Впервые система непосредственного впрыска была применена на двигателе GDI (GasolineDirectInjection – непосредственный впрыск бензина), устанавливаемом на автомобили компании Mitsubishi. В настоящее время система непосредственного впрыска используется в двигателях многих автопроизводителей. Передовики Audi (двигатели TFSI) и Volkswagen (двигатели FSI, TSI), которые практически полностью перешли на бензиновые двигатели с непосредственным впрыском.

Двигатели с непосредственным впрыском имеют в своем активе BMW (двигатели N54, N63), Infiniti (двигатели M56), Ford (двигатели EcoBoost), GeneralMotors (двигатели Ecotec), Hyundai (двигатели Theta), Mazda (двигатели Skyactiv), Mercedes-Benz (двигатели CGI).

Применение системы непосредственного впрыска позволяет достичь до 15% экономии топлива, а также сокращения выброса вредных веществ с отработавшими газами.

Устройство системы непосредственного впрыска топлива. Конструкция системы непосредственного впрыска топлива рассмотрена на примере системы, устанавливаемой на двигатели FSI (FuelStratifiedInjection – послойный впрыск топлива). Система непосредственного впрыска составляет контур высокого давления топливной системы двигателя и включает топливный насос высокого давления, регулятор давления топлива, топливную рампу, предохранительный клапан, датчик высокого давления и форсунки впрыска.

Данная система включает в себя:

1. топливный бак;
2. топливный насос;
3. топливный фильтр;
4. перепускной клапан;
5. регулятор давления топлива;
6. топливный насос высокого давления;
7. трубопровод высокого давления;
8. распределительный трубопровод;
9. датчик высокого давления;
10. предохранительный клапан;
11. форсунки впрыска;
12. адсорбер;
13. электромагнитный запорный клапан продувки адсорбера.

Принцип действия системы непосредственного впрыска. Система непосредственного впрыска в результате работы обеспечивает несколько видов смесеобразования:

- послойное;
- стехиометрическое гомогенное;
- гомогенное.

Многообразие в смесеобразовании определяет высокую эффективность использования топлива (экономия, качество образования смеси, ее полное сгорание, увеличение мощности, уменьшение вредных выбросов) на всех режимах работы двигателя.

Послойное смесеобразование используется при работе двигателя на малых и средних оборотах и нагрузках. Стехиометрическое (другое наименование – легковоспламеняемое) гомогенное (другое наименование – однородное) смесеобразование применяется при высоких оборотах двигателя и больших нагрузках. На бедной гомогенной смеси двигатель работает в промежуточных режимах.

При послойном смесеобразовании дроссельная заслонка почти полностью открыта, впускные заслонки закрыты. Воздух поступает в камеры сгорания с большой скоростью, с образованием воздушного вихря. Впрыск топлива производится в зону свечи зажигания в конце такта сжатия. За непродолжи-

тельное время до воспламенения в районе свечи зажигания образуется топливно-воздушная смесь с коэффициентом избытка воздуха от 1,5 до 3. При воспламенении смеси вокруг нее остается достаточно много чистого воздуха, выступающего в роли теплоизолятора. Гомогенное стехиометрическое смесеобразование происходит при открытых впускных заслонках, дроссельная заслонка при этом открывается в соответствии с положением педали газа. Впрыск топлива производится на такте впуска, что способствует образованию однородной смеси. Коэффициент избытка воздуха составляет 1. Смесь воспламеняется и эффективно сгорает во всем объеме камеры сгорания.

Бедная гомогенная смесь образуется при максимально открытой дроссельной заслонке и закрытыми впускными заслонками. При этом создается интенсивное движение воздуха в цилиндрах. Впрыск топлива производится на такте впуска. Коэффициент избытка воздуха поддерживается системой управления двигателем на уровне 1,5. При необходимости в состав смеси добавляются отработавшие газы из выпускной системы, содержание которых может достигать до 25%.

Недостатки системы питания с непосредственным впрыском. Собственно о минусах подобной системы питания было известно с самого начала. Сложность и высокая стоимость сюрпризом не были – опыт внедрения непосредственного впрыска накопился изрядный. Надежность сложных систем честно постарались увеличить. Правда, цену особенно опустить не пытались.

Как известно, для подачи топлива непосредственно в цилиндры нужен насос высокого давления. Вообще-то и в системах «обычного» распределенного впрыска в системе питания давление немаленькое, но у прямого впрыска оно примерно в 10 раз больше.

На дизельных моторах непосредственный впрыск и ТНВД появился существенно раньше, и ресурс узлов был не таким уж низким. У бензиновых все получилось иначе: насосы оказались весьма недолговечными. Потому что дизтопливо имеет более высокие смазочные свойства, чем бензин, и без специальных смазывающих присадок ресурс всех узлов трения очень мал.

Современные мембранные ТНВД не так зависят от смазки, как поршневые, но, тем не менее, нуждаются в ней. Да и в целом насос высокого давления – штука довольно хрупкая, любые загрязнения выведут его из строя. Улучшить ситуацию смогли введением стандарта на смазывающие присадки в топливо. Конечно, 15% масла, как в двухтактные моторы, добавлять не стали, но топливо Евро-4 и выше обязательно содержит небольшое количество специальных смазок. Не в последнюю очередь – именно для ТНВД на бензиновых машинах.



Рисунок 1. – ТНВД на бензиновых машинах

С форсунками ситуация аналогичная, они дороже и менее надежны, чем на системах распределенного впрыска. Требования к их работе тоже намного выше. Небольшое изменение факела распыла, даже без изменения общего расхода подачи, ведет к серьезным нарушениям работы мотора. В результате для сохранения работоспособности резко растут требования по чистоте топлива и рабочей температуре.

Пьезофорсунки еще и имеют ограниченное количество циклов срабатывания, чувствительны к перегреву, а также обладают склонностью при выходе из строя «лить» бензин, что может вызвать гидроудар при запуске. Особенно это характерно для очень распространенных «высокоточных» пьезофорсунок Bosch, которые имеют ограниченный ресурс, а компания на протяжении последних десяти лет не может создать действительно хорошо работающий вариант.

Склонность к закоксовке впускных клапанов и худшие условия их работы проявились на моторах Мицубиси довольно быстро. Обычно форсунки подают бензин на впускной клапан и охлаждают его. И заодно смывают с него отложения. У «непосредственного» мотора такой возможности нет, клапан греется сильнее, больше нагревает воздух, а масло из системы вентиляции картера и из сальника клапана

постепенно образует «шубу», которая затрудняет газообмен и приводит к зависанию клапанов и его перегреву. Особенно тяжело приходится моторам с повышенным расходом масла, а в самой критической группе риска – моторы, которые часто работают с малой нагрузкой, то есть в пробках.

Плохие пусковые качества из-за неудовлетворительного испарения топлива при пуске тоже проявились давно. Оказалось, что оптимизация формы факела впрыска на холодном и горячем моторе должна производиться более тщательно. Любое попадание топлива на стенки цилиндра приводит к резкому увеличению количества несгоревшего топлива и попаданию его в масло. А при запуске при отрицательных температурах большое значение приобретает качество распыла бензина: оно должно оказаться намного выше, чем при обычной работе, и давление топлива на пуске должно быть очень высоким. Поначалу этого не учли.

Попадание топлива в масло из-за неисправностей топливного насоса высокого давления – в общем-то чисто конструктивный недостаток насосов Bosch, но в силу их широкого распространения и общности конструкций насосов свойственен почти всем моторам с непосредственным впрыском. Бензин в масле не так уж и страшен, но в больших количествах ведет к снижению вязкости масла до критической, что приводит к повреждениям моторов.

Почти у всех проблем есть пути решения. Например, двойной впрыск, когда топливо подается и в цилиндры, и во впускной трубопровод – это справляется сразу со сложностью с закоксовкой клапанов, экологичностью и плохим запуском в холода. Такая схема применялась на некоторых двигателях Volkswagen EA888, но продавались они исключительно в США и были заточены под жесткие экологические нормы Калифорнии. Но в конце 2014-го комбинированный впрыск появился и у нас – на моторе 6AR-FE (2 литра, 150 л. с.) Toyota Camry последнего поколения. Пока сложно судить о надежности, ибо пробеги машин пока небольшие в основной массе, однако предпосылки хорошие.



Рисунок 2. – Под капотом Toyota Camry XLE 2015 г.в.

С поршневыми кольцами и топливными насосами приходится разбираться чисто конструктивными методами, экспериментируя с формой – часто «дизайн» поршневой группы производители дорабатывают уже после того, как машина вышла на рынок и поразила всех угаром масла. Так, скажем, делала Toyota в 2005 году, доводя до ума моторы серии ZZ (еще без непосредственно впрыска), а позже – Volkswagen с уже упомянутыми выше EA888. Насосы высокого давления тоже стараются сделать надежнее – эта задача технически выполнима.

Заключение. Система очень сложная и дорогая – накладным для производителей выходит не только себестоимость конечной продукции, но и исследования с экспериментами. А маркетологи не дают возможности по 10 лет заниматься испытаниями, требуют все более новых моторов с еще более привлекательными характеристиками.

ЛИТЕРАТУРА

1. Коваленко, О.Л. Электронные системы автомобилей : учеб. пособие / О.Л. Коваленко ; Сев. (Арктич.) федер. ун-т им. М.В. Ломоносова. – Архангельск : ИПЦ САФУ, 2013. – 80 с.
2. Режим доступа: <http://chiptuner.ru/content/sensor/>. – Дата доступа: 01.05.2016.
3. Режим доступа: http://injectorservice.com.ua/docs/temperature_sensor_diagnostics.pdf. – Дата доступа: 01.05.2016.

УДК 621.924.1

**СРАВНИТЕЛЬНОЕ ИССЛЕДОВАНИЕ ИЗНОСОСТОЙКОСТИ
ШЛИФОВАННЫХ И НЕШЛИФОВАННЫХ ОБРАЗЦОВ
С КОМБИНИРОВАННЫМ ПОКРЫТИЕМ 95ХС И TiN****Д.А. БАШЛАЧЁВ, Е.В. ПУЙМАН***(Представлено: д-р техн. наук, проф. Н.Н. ПОПОК,**Изучена структура и состав комбинированного газотермического и ионноплазменного покрытия 95ХС и TiN, исследована износостойкость покрытия на нешлифованных и шлифованных образцах.*

Проблема повышения износостойкости деталей машин, работающих в условиях фрикционного взаимодействия, а также режущих инструментов, функционирующего при повышенных температурах и давлениях, становится все более значимой в связи с постоянным увеличением уровня действующих нагрузок, скоростей и рабочих температур, а также применением агрессивных сред. Одним из наиболее перспективных способов существенного повышения физико-механических, триботехнических характеристик и антикоррозионных свойств, а также выносливости сталей и сплавов является модифицирование поверхностных слоев этих материалов концентрированными потоками ионов металлов и газов [1, 2], интенсивно развиваемое в Полоцком государственном университете [3]. Важной особенностью этого метода является сохранение прочностных свойств подложки за счет пониженной температуры ионно-плазменной обработки, что весьма существенно для повышения ресурса работоспособности материалов с упрочненными поверхностными слоями. Весьма перспективно использование для покрытий высоколегированных инструментальных сталей мартенситного типа. Дополнительные возможности для повышения прочности и износостойкости газотермических покрытий с метастабильными фазами открывает использование технологии ионно-плазменной обработки их поверхностных слоев.

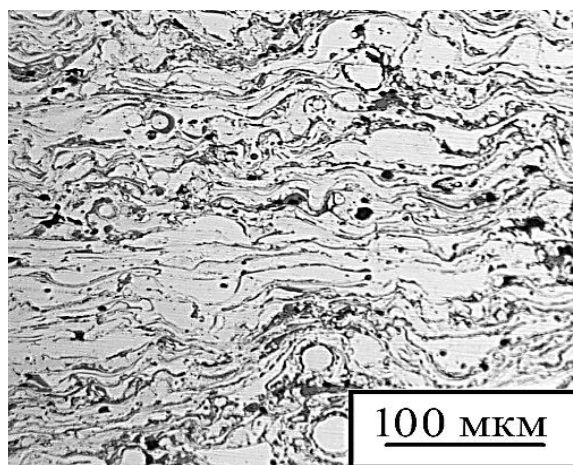
На рисунке 1 представлена характерная микроструктура напыленного газотермического покрытия из инструментальной стали 95ХС.

Газотермическое покрытие из инструментальной стали 95ХС в дальнейшем подвергалось ионно-плазменной обработке нитридом титана.

Рентгеновская дифрактограмма образцов покрытия из стали 95Х18 с осажденным слоем TiN представлена на рисунке 2.

С целью оценки износостойкости образцов с комбинированным покрытием были проведены экспериментальные исследования шлифованных и нешлифованных образцов на машине трения МТ-393, при следующих режимах: усилие прижима – 50Н, скорость – 200мин⁻¹, время – 200сек.

На рисунке 3 показана машина трения МТ-393 и внешний вид образцов при проведении исследований износостойкости.



**Рисунок 1. – Микроструктура газотермического покрытия
из проволоочной стали 95ХС**

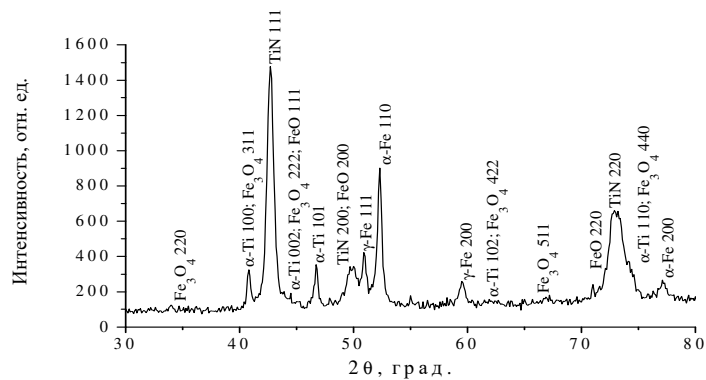


Рисунок 2. – Фрагмент рентгеновской дифрактограммы (CoKa) покрытия из стали 95XC и TiN

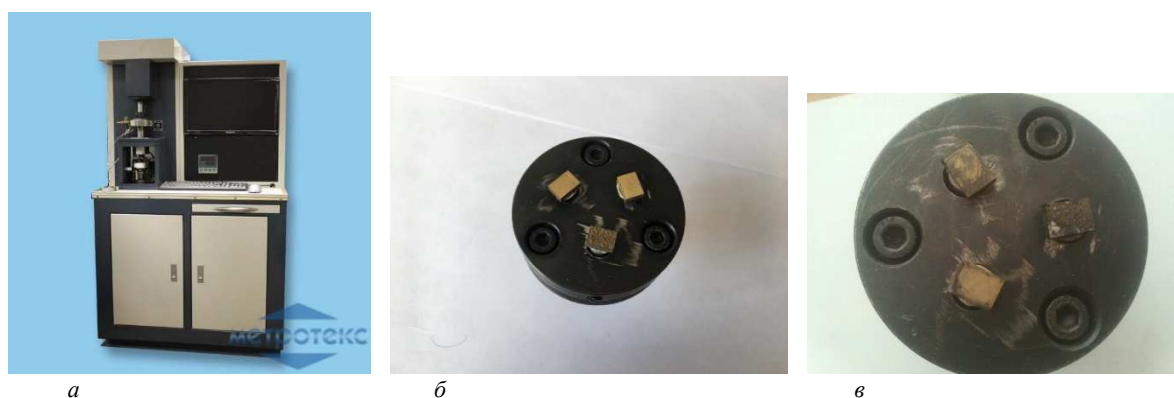


Рисунок 3. – Внешний вид машины трения (а) и образцов до (б) и после (в) испытания

Результаты исследования износа образцов представлены в таблице и на рисунке 4.

Таблица. – Износ шлифованных и нешлифованных образцов

Номер образца	Вес до, г/мг	Вес после, г/мг	Весовой износ, г/мг	Материал
1	4,510	4,505	0,005	95XC+TiN шлифованный
2	4,470	4,440	0,030	95XC+TiN нешлифованный

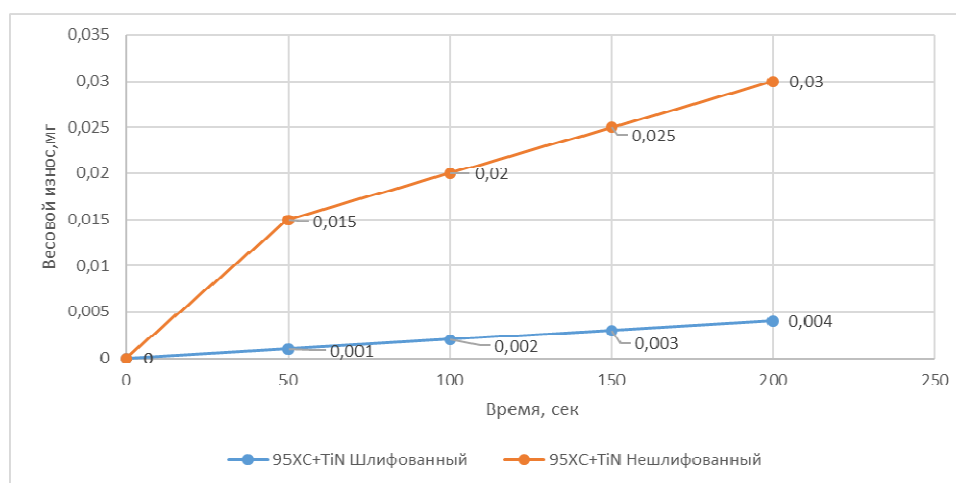


Рисунок 4. – Зависимости износостойкости образцов от времени испытания: 1 – шлифованный; 2 – нешлифованный

Как видно из графика, износостойкость образцов, полученных нанесением покрытия из нитрида титана на шлифованную поверхность газотермического покрытия 95ХС выше, чем нешлифованных.

Выводы:

1. Газотермическое покрытие из инструментальной стали 95ХС с осаждённым покрытием из TiN имеет мелкозернистую структуру и фазовый состав, способствующие повышению микротвёрдости поверхности образцов.

2. Нанесение покрытий из нитрида титана на шлифованную поверхность 95ХС обеспечивает повышение износостойкости образцов по сравнению с нешлифованными образцами до 3-4 раз.

ЛИТЕРАТУРА

1. Хирвонен, Дж.К. Ионная имплантация / Дж.К. Хирвонен. – М. : Металлургия, 1985. – 285 с.
2. Риссел, Х. Ионная имплантация / Х. Риссел, И. Руге ; пер. с нем. В.В. Климова, В.Н. Пальянова ; под ред. М.И. Гусевой. – М. : Наука, 1983. – 360 с.
3. Попок, Н.Н. Ионные источники: виды, конструкции, применение = Ion sources: variety, construction, application : пособие для студентов и магистрантов машиностр. Специальности / Н.Н.Попок, С.В. Дербуш, А.Н. Попок. – Новополоцк : Полоц. гос. ун-т, 2017. –72 с.

УДК 621.785

МЕТОДЫ КОНТРОЛЯ НАПРЯЖЕНИЙ СВАРНЫХ СОЕДИНЕНИЙ**С.В. СКУРЬЯТ***(Представлено: М.В. СЕМЕНЧЕНКО)*

Современные способы контроля качества сварных соединений позволяют обнаружить внутренние дефекты и определить их размер. Своевременно обнаружить внутренние напряжения можно рентгенографическим методом и методом магнитной памяти металла. В статье рассмотрены особенности обоих методов, их преимущества, недостатки и используемое оборудование, а также предложен метод контроля напряжений сварных соединений по величине биения их поверхности.

На современном этапе развития технологий существует множество различных методов контроля, но для точного контроля наличия сварочных напряжений широко используются следующие методы:

- рентгенографический метод;
- метод магнитной памяти металла.

Рентгенографический метод позволяет проецировать на специальную рентгеновскую пленку или бумагу с помощью рентгеновских лучей, внутреннюю структуру сварного шва. Проекция внутренней структуры позволяет не только определить локализацию дефектов с информацией об их размере, но и оценить строение сварного шва [1].

Принцип рентгенографического метода основан на поглощении лучей плотными средами. Плотность структуры оказывает непосредственное влияние на количество пройденных лучей: чем плотнее структура, тем меньше лучей сможет через нее пройти. Имеющиеся внутри сварного шва дефекты, нарушают плотность структуры, тем самым обеспечивая возможность прохождения большего количества лучей. Пройдя через сварной шов, рентгеновское излучение поступает на регистрирующее устройство, которое обрабатывает поступившие данные и выдает информацию о наличии размере и расположении дефектов [1].

Устройство рентгенографического метода состоит из излучающегося элемента, который имеет вид вакуумного сосуда внутри которого находится три электрода [2]:

- анод;
- накал катода;
- катод.

Основными конструктивными элементами устройства является анод и катод. Рентгеновский луч возникает, когда частице придется сильное ускорение. Данное ускорение может произойти при высокоэнергетическом переходе в облачке атома.

Данный метод имеет следующие преимущества:

- точный и надежный метод, используемый в том числе при контроле сварных соединений трубопроводов;
- позволяет увидеть скрытые дефекты;
- предоставляет возможность определения абсолютных и относительных размеров имеющихся дефектов;
- высокая производительность контроля.

Несмотря на преимущества в использование этого метода есть и недостатки:

- зависимость правильности установки параметров;
- используемое в процессе контроля высокую стоимость;
- необходимость использования специальных расходных материалов для отображения результатов контроля;
- негативное влияние на здоровье людей.

Метод магнитной памяти относится к неразрушающему контролю. Он построен на принципе распределения собственного магнитного поля на поверхности изделия. Позволяет определить зону концентрации напряжения, неоднородность структуры и дефекты сварного шва путем отображения остаточной намагниченности, сформировавшейся естественным образом в процессе сварки [3, 4].

Устройства и программное обеспечение метода магнитной памяти. К устройствам метода относятся [5]:

- Измеритель концентрации напряжений является системой измерения, регистрации и обработки данных диагностики напряжённо-деформированного состояния оборудования и конструкций с использованием метода магнитной памяти металла.
- Специализированное высокочувствительное двенадцати канальное сканирующее устройство со встроенными трехкомпонентными феррозондовыми преобразователями и аналого-цифровым преобразователем.

– Сканирующее устройство предназначено для контроля трубопроводов, сосудов, протяженных сварных соединений. Изготовлено в виде "тележки" на четырех колёсах, имеющее восемь феррозондовых преобразователей в двухкомпонентном их расположении для измерения одновременно нормальной и тангенциальной составляющих магнитного поля и счётчик длины.

К программному обеспечению метода магнитной памяти относится [5]:

– "ММП-Система". Специализированная программа обработки данных о напряжённо-деформированном состоянии оборудования и конструкций по методу магнитной памяти металла. Программа позволяет произвести обработку результатов диагностического контроля и выполнить их анализ по методу магнитной памяти с оформлением отчета.

– "МПП-Ресурс". Специализированная программа оценки остаточного ресурса металла оборудования по усталостному повреждению в зонах концентрации напряжений по данным диагностики с использованием метода магнитной памяти металла.

Преимущества использования данного метода:

– не требуется использование специальных намагничивающих устройств, так как основу метода составляет явление намагничивания узлов оборудования и конструкций в процессе их работы;

– возможность определения мест концентрации напряжений, появившихся под воздействием рабочих нагрузок, непосредственно в процессе контроля. Местоположение концентраторов может быть неизвестно;

– предварительная подготовка контролируемой поверхности не требуется. Определить наличие внутренних напряжений можно даже без зачистки металла;

– используемое оборудование отличается компактными размерами, наличием регистрирующих устройств и автономного питания. Как следствие, контрольные мероприятия могут выполняться в различных местах, в т.ч. труднодоступных;

– конструктивное исполнение специальных сканирующих устройств позволяют выполнять контроль трубопроводов, сосудов, оборудования со скоростью 100 м/ч и более.

Данный метод не имеет явных недостатков.

Рентгенографический метод и метод магнитной памяти металла предполагают использование специализированного оборудования, которое не всегда есть в наличии на многих предприятиях. В качестве альтернативного варианта может быть предложен способ контроля напряжений сварных соединений за счет проверки их геометрических параметров. Коробление изделия будет свидетельствовать о наличии внутренних напряжений, которые можно снизить путем термического воздействия.

Нами разработано приспособление, позволяющее проконтролировать наличие остаточных напряжений по величине биения поверхности сварного изделия. Предложен метод для снижения напряжения сварного соединения путем пропускания электрического тока в режиме термоциклирования. Разработанное приспособление позволило оценить влияние термического воздействия на геометрические параметры изделия.

Таким образом, после рассмотрения принципов и методов работы, преимущества и недостатков рентгенографического метода и метода магнитной памяти металла можно сделать следующий вывод: перечисленные методы эффективны для контроля сварочных напряжений. Их использование позволяет:

– контролировать с высокой производительностью сварные швы различной толщины;

– вести наиболее точный контроль дефектов с определением места их локализацией.

В качестве альтернативного варианта может выступать метод контроля остаточных напряжений по величине биения поверхности сварного изделия.

ЛИТЕРАТУРА

1. Рентгенографический контроль сварных соединений [Электронный ресурс] // Сварка под рентген. – Режим доступа: <https://www.samsvar.ru/stati/svarka-pod-rentgen.html>. – Дата доступа: 10.09.2018.
2. Рентгенографический контроль сварных соединений [Электронный ресурс] // Путеводитель по сварке и пайке. – Режим доступа: <http://svarkaipayka.ru/tehnologia/drugoe/rentgenograficheskiy-kontrol-svarnyih-soedineniy.html>. – Дата доступа: 10.09.2018.
3. Метод магнитной памяти металла [Электронный ресурс] // Эталон. – Режим доступа: <http://etalon-rk.ru/metod-magnitnoj-pamyati-metalla/>. – Дата доступа: 10.09.2018.
4. Метод магнитной памяти металла – новое направление в технической диагностике [Электронный ресурс] // Энергодиагностика. – Режим доступа: <http://www.energodagnostika.ru/about-mmm-method.html>. – Дата доступа: 10.09.2018.
5. Метод магнитной памяти металла. Обзор методики и техники работ [Электронный ресурс] // Диагностика подводных трубопроводов. – Режим доступа: <http://www.morinzheologia.ru/download/dpt.mpm.info.pdf>. – Дата доступа: 10.09.2018.

УДК 656.1

**СОВЕРШЕНСТВОВАНИЕ ОРГАНИЗАЦИИ ДОРОЖНОГО ДВИЖЕНИЯ
В ГОРОДЕ НОВОПОЛОЦКЕ****Е.С. ЦАРЬКОВА***(Представлено: И.В. ХОМИЧ)*

Выполнен анализ статистических данных для г. Новополоцка, выявлены основные причины возникновения дорожно-транспортных происшествий. Изучены наиболее известные организации, занимающиеся разработкой программного обеспечения в области дорожной инфраструктуры и наиболее распространенное программное обеспечение в Республике Беларусь, а также европейские разработки в создании интеллектуальные транспортные системы. Предложены решения по использованию интеллектуальных транспортных систем в г. Новополоцке.

В связи с ростом парка автомобилей в современном мире необходимо более углубленно заниматься развитием транспортной инфраструктуры, на некоторых участках дороги помимо грамотной организации дорожного движения необходимо внедрять ИТС (интеллектуальные транспортные системы). На гистограммах можно отследить рост числа автомобилей в Витебской области (рис.1), количество автомобилей за 3 года увеличилось практически в 2 раза, следовательно, вероятность возникновения ДТП (дорожно-транспортное происшествие) возрастает.



Рисунок 1. – Количество транспортных средств в личной собственности в Витебской области

Использование ИТС позволит решить две задачи – обеспечение безопасности дорожного движения и обеспечение сохранности автодорог.

Интеллектуальная транспортная система – система, интегрирующая современные информационные, коммуникационные и телематические технологии, технологии управления и предназначенная для автоматизированного поиска и принятия к реализации максимально эффективных сценариев управления транспортной системой региона (города, дороги), конкретным транспортным средством или группой транспортных средств, с целью обеспечения заданной мобильности населения, максимизации показателей использования дорожной сети, повышения безопасности и эффективности транспортного процесса, комфорта для водителей и пользователей транспорта [1].

Ниже приведен перечень наиболее известных организаций, занимающихся разработкой программного обеспечения в области дорожной инфраструктуры:

ERTICO – занимается разработкой различных программ, направленных на развитие европейских инновационных технологий в области развития дорожной инфраструктуры, применением интеллектуальных транспортных систем в целях управления дорожным движением, повышением мобильности населения и грузов, улучшением качества жизни людей, повышением безопасности на дорогах и снижением вредного воздействия автотранспорта на окружающую среду.

ERTRAC (The European Road Transport Research Advisory Council) – программа координации взаимодействия Европейских исследовательских институтов в дорожном и транспортном комплексе в целях структурирования и оптимизации научно-исследовательских работ в интересах стран Евросоюза.

FOT-Net Data, Field Operational Test Networking and Data Sharing Support – это трехлетний проект поддержки действий для эффективного обмена и повторного использования доступных наборов данных FOT (Field Operational Tests), разработки и продвижения основы для обмена данными, построение подробного каталога доступных данных и инструментов и создание международной сетевой платформы для деятельности FOT [3].

Системы:

E-Call (Emergency Call) – это система, отправляющая автоматическое сообщение в центр экстренных вызовов в случае аварии автомобиля.

ADASIS (Advanced Driver Assistant Systems Interface Specification) Использование точных картографических данных в средствах навигации для получения водителем прогноза ситуации на дороге.

GST (Global System for Telematics) – создание технологической платформы для развития сотрудничества, необходимого для развития массового рынка открытых телематических услуг, в первую очередь обеспечивающих сбор, передачу обработку информации для пользователей – участников дорожного движения, скорой помощи и служб спасения.

eSafety Forum– европейская программа по массовому внедрению систем активной и пассивной безопасности, включающая в себя работы по проекту eCall (“экстренный вызов”), созданию электронных карт для использования экстренными службами, изучению эффективности различных каналов передачи информации от автомобиля в диспетчерский центр оператора, сотрудничество с участниками американского, японского и других рынков телематических услуг, с целью выработки приоритетных задач и международных стандартов по оказанию экстренной помощи пострадавшим в аварии на дорогах, гармонизация технических решений по передаче информации от автомобиля к автомобилю или от автомобиля к дорожной инфраструктуре, организация информирования участников дорожного движения в режиме реального времени о ситуации на дорогах через специальный радиоканал.

Программное обеспечение:

FeedMAP – обеспечение постоянного обновления электронных карт. Эта структура способствует повышению эффективности транспорта и повышению безопасности дорожного движения за счет создания постоянного обновления карт по сниженным ценам для навигации и приложений Advanced Driver Assistance Systems (ADAS).

AIDE (Adaptive Integrated Driver-Vehicle Interface) Использование специального электронного оборудования и программного обеспечения (ПО), позволяющего концентрировать внимание водителя в момент обгона. Также отключаются функции приборов в салоне автомобиля, отвлекающих внимание водителя во время сложного маневра.

Программы, получившие широкое распространение при обеспечении безопасности дорожного движения:

EuroFOT (European Large-Scale Field Operational Tests on In-Vehicle Systems) – проект сбора натуралистических данных для оценки воздействия использования интеллектуальных транспортных систем под названием «интеллектуальные системы транспортных средств» или «активных систем безопасности» для оценки их влияния на безопасность транспорта и топливную экономичность [4].

HeavyRoute– программа поддержки быстрых и безопасных грузовых перевозок. Работая со всеми основными заинтересованными сторонами, проект Heavyroute будет работать над предоставлением инструментов, систем и процессов сбора и интерпретации данных, которые будут эффективно связывать дорожную инфраструктуру Европы с помощью электронных картографических систем с операторами и водителями грузовиков. [5].

IP PReVENT– программа внедрения специальных электронных устройств (ADAS – Advanced Driver Assistance Systems), позволяющих водителю получать предупреждающую информацию о возможных опасностях по ходу движения и избегать аварийных ситуаций.

MAPS&ADAS (IP PReVENT) - это подпроект по профилактической безопасности интегрированного проекта (IP) (PReVENT), финансируемый Европейской комиссией

SAFESPOT– программа поддержки появления большего количества «умных» машин на «умных» дорогах.

SpeedAlert Forum– информирование водителей о соблюдении установленного скоростного режима.

CVIS (Cooperative vehicle-infrastructure systems)– программа взаимодействия автомобилей и дорожной инфраструктуры.

EuroRoadS– программа по созданию базы данных о европейской дорожной инфраструктуре.

AGILE (Application of Galileo in the LocationBased Service Environment) SISTER (Promoting the integration of satellite and terrestrial communication with GALILEO for road transport)

AGILE – программа обеспечения коммерческого использования спутниковой системы Galileo.

ENITE (European Network on ITS Training & Education)– программа подготовки специалистов по интеллектуальным транспортным системам.

FRAME Forum – программа построения архитектуры для Европейской интеллектуальной транспортной системы.

RCI (Road Charging Interoperability) – программа развития платных дорог.

Road Traffic Information Group – программа развития информационного сопровождения участников дорожного движения.

TMC Forum (Traffic Message Channel)– программа информирования участников дорожного движения о реальной дорожной обстановке по специальному выделенному радиоканалу.

Network of National ITS Associations– программа по развитию международной сети Ассоциаций Интеллектуальных транспортных систем.

CONNECT, SIMBA – национальные и международные программы по развитию рынка интеллектуальных транспортных систем. Включают в себя программы в Странах Центральной и Восточной Европы, Бразилии, Индии, Китае, ЮАР, а с 2008 г. – в России. Национальным координатором проекта SIMBA в России является Профессиональная Ассоциация противодействия угонам транспортных средств [4].

Анализ аварийности на территории г. Новополюцка (табл. 1) показывает, что за последние 5 лет количество ДТП и погибших остается приблизительно на одном уровне

Таблица 1. – Основные показатели ДТП

Показатель \ Год	2012	2013	2014	2015	2016
Количество ДТП подлежащих учёту	30	24	27	20	29
Количество погибших/погибших несовершеннолетних	4/0	4/0	2/0	2/0	3/1
Количество раненых/раненых несовершеннолетних	29/3	20/3	28/0	21/3	29/5
Количество ДТП совершенные лицами не имеющими права управления	2	–	1	–	–
Количество ДТП совершенными лицами в состоянии алкогольного опьянения	–	–	3	5	–

В 2013, 2015 годах наблюдалось небольшое снижение ДТП, но в 2016 году оно вновь увеличилось (рис. 2).

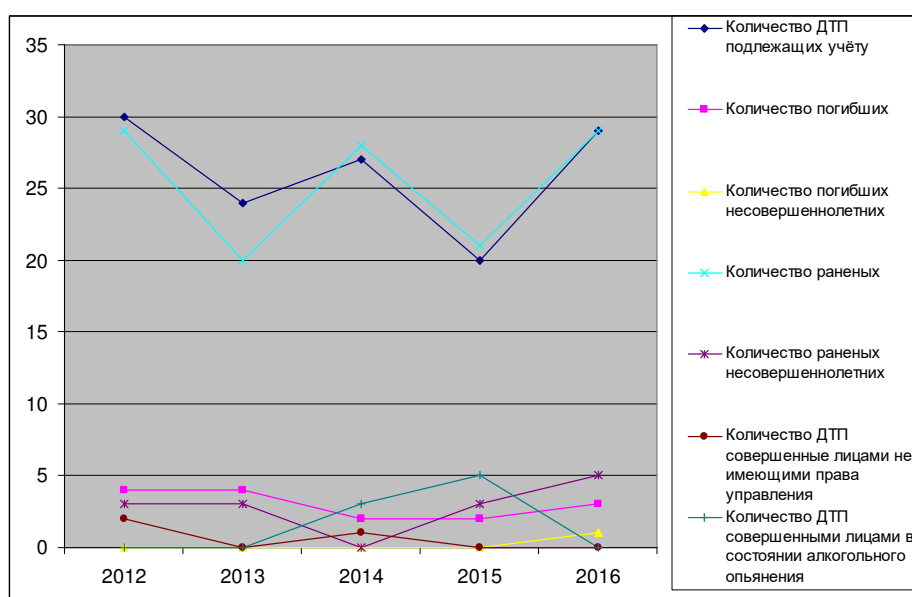


Рисунок 2. – Графики основных показателей ДТП

Анализ ДТП за 2012–2016 гг. (табл. 2) показывает, что ДТП с участием пешеходов составляет 80% от общего количества. Исследования показали, что аварии можно разделить на ДТП с участием пешеходов, со столкновением транспортных средств, с наездом на препятствия и с опрокидыванием (табл. 2–5). Собранные данные показывают, что 88 % ДТП, произошедших в 2016 г. произошло по вине водителей.

Таблица 2. – Основные показатели ДТП с участием пешеходов

Показатель\ Год	2012	2013	2014	2015	2016
Количество ДТП	21 (70%)	17 (70,8%)	21 (77,7%)	8 (40%)	23 (79,2%)
Количество погибших	2	2	1	–	3
Количество раненых	21	15	20	8	22
На пешеходных переходах:					
–Количество	15	9	10	4	16
–По вине водителя	14	9	9	4	14
–По вине пешехода	1	–	1	–	2
Вне пешеходных переходов:					
–Количество	6	8	11	4	7
–По вине водителя	2	3	5	1	3
–По вине пешехода	4	5	6	3	4

Таблица 3. – Основные показатели ДТП со столкновением ТС

Показатель\ Год	2012	2013	2014	2015	2016
Количество ДТП	7 (23,3%)	2 (8,3%)	1 (3,7%)	5 (25%)	2 (6,9%)
Количество погибших	1	–	–	–	–
Количество раненых	8	2	1	5	2

Таблица 4. – Основные показатели ДТП с наездом на препятствие

Показатель\ Год	2012	2013	2014	2015	2016
Количество ДТП	1 (3,3%)	2 (8,3%)	3 (11,1%)	4 (20%)	1 (3,5%)
Количество погибших	2	–	–	2	–
Количество раненых	–	2	3	5	1

Таблица 5. – Основные показатели ДТП с опрокидыванием

Показатель\ Год	2012	2013	2014	2015	2016
Количество ДТП	1 (3,3%)	–	2 (7,4%)	–	–
Количество погибших	–	–	1	–	–
Количество раненых	1	–	3	–	–

Поэтому программы развития информационного сопровождения участников дорожного движения, программа информирования участников дорожного движения о реальной дорожной обстановке по специальному выделенному радиоканалу, позволяющих водителю получать предупреждающую информацию о возможных опасностях по ходу движения и избегать аварийных ситуаций являются перспективными с точки зрения использования их в г. Новополоцке.

Основными местами концентрации ДТП являются улицы Молодёжная, Комсомольская и Блохина (табл. 6).

Таблица 6. – Улицы с наибольшей концентрацией ДТП

Улица/Показатель	Количество ДТП	Количество погибших	Количество раненых
Молодёжная	59	5	57
Комсомольская	8	0	8
Блохина	15	2	14

Большое количество ДТП на улице Молодёжной обусловлено её протяженностью, интенсивностью движения, в том числе и общественного транспорта. Основными причинами ДТП являются управление транспортными средствами в состоянии опьянения, нарушение проезда пешеходных переходов, а также несоблюдение скоростного режима.

Для предотвращения несоблюдения скоростного режима предложить установку радаров в местах где водители чаще всего превышают скорость. Примером данной программы в Европе может являться программа SpeedAlert Forum.

Для помощи водителям при проезде перекрестков или обгоне, возможна установка дополнительного оборудования, а также дополнительное информационное сопровождение участников дорожного движения. Примеры данных программ в Европе: Road Traffic Information Group, IP PReVENT.

Анализ статистических данных для г. Новополоцка по количеству ДТП и причинам их возникновения показал, что 88 % произошло по вине водителей. Изучены наиболее известные организации, занимающиеся разработкой программного обеспечения в области дорожной инфраструктуры и наиболее распространенное программное обеспечение в Республике Беларусь, а также европейские разработки в создании интеллектуальных транспортных системы. Предложены мероприятия для снижения количества ДТП в г. Новополоцке, такие как установка дополнительного оборудования и использование программного обеспечения Road Traffic Information Group, IP PReVENT.

ЛИТЕРАТУРА

1. Цели, задачи и сфера приложения интеллектуальных транспортных систем [Электронный ресурс]. – 2017. – Режим доступа: https://studwood.ru/916209/ekonomika/tseli_zadachi_sfera_prilozheniya_intellektualnyh_transportnyh_sistem. – Дата доступа: 09.05.2017.
2. Жанказиев, С.В. Интеллектуальные транспортные системы / С.В. Жанказиев. – М. : МАДИ, 2016. – 120 с.
3. EuroFOT [Электронный ресурс]. – 2018. – Режим доступа: <https://en.wikipedia.org/wiki/EuroFOT>. – Дата доступа: 05.04.2018.
4. FOT-Net Data [Электронный ресурс]. – 2011. – Режим доступа: <https://en.wikipedia.org/wiki/EuroFOT>. – Дата доступа: 01.01.2011.
5. Heavyroute [Электронный ресурс]. – 2018. – Режим доступа: <http://heavyroute.fehrl.org/>. – Дата доступа: 05.04.2018.

ХИМИЧЕСКИЕ ТЕХНОЛОГИИ

УДК 657.6:66

ИННОВАЦИОННЫЕ ИДЕИ В ПРОЦЕССЕ АЛКИЛИРОВАНИЯ**Е.Я. КОРБУТ***(Представлено: канд. хим. наук, доц. С.В. ПОКРОВСКАЯ)*

Рассмотрены теоретические основы процесса алкилирования фенола, а также перспективные направления совершенствования технологий производства алкилфенола на примере предприятия СООО «ЛЛК-Нафтан». Способ решения задачи заключается в замене алкилирующего агента и катализатора на более эффективный, обладающий улучшенными физико-химическими и эксплуатационными характеристиками, вводе нового оборудования в эксплуатацию, а именно выпарного аппарата, для улучшения качества разделения продуктовой смеси.

Применение смазочных масел, легированных современными присадками, обуславливает длительную и безотказную работу энергетических установок и оборудования. Качество масла может быть улучшено при введении одних присадок и совершенно не изменится при введении других того же функционального действия, поэтому присадки относятся к наукоемкой продукции, характеризующейся тем, что затраты на разработку сопоставимы с затратами на организацию их промышленного производства [1].

Процесс производства алкилфенола и додецилфенола состоит из стадий приготовления реакционной смеси, алкилирования фенола тетрамерами пропилена на катионообменном катализаторе, фракционирования додецилфенола. Однако использование сульфокатионита как катализатора вводит значительные ограничения в технологию ведения процесса алкилирования, а именно: максимальная температура в реакторе 120°С вследствие отрыва кислотных активных сульфогрупп от дивинилстирольной матрицы, меньшая селективность реакции синтеза монододецилфенола, следовательно, снижается качество целевого продукта.

Анализ патентной литературы идентифицировал возможность усовершенствования процесса получения додецилфенола путём замены низкотемпературной катионной смолы на более термически устойчивые цеолиты типа морденит. Вследствие пористой структуры цеолита с определённым размером каналов, возрастает селективность целевой реакции синтеза монододецилфенола. Монододецилфенолы при дальнейшей реакции с пентасернистым фосфором и последовательной нейтрализацией цинковыми белилами образует присадку вниинп-353, от стерической ориентации которой зависит возможность дальнейшего синтеза присадок с более сложной структурой.

Важно отметить влияние фактора чёткости разделения продуктовой смеси из реактора, так как именно по содержанию примесей оценивают пригодность использования продукта для синтеза присадок. Патентный поиск литературы определил возможность установки выпарного аппарата для разгрузки колонн блока алкилирования по фенолу и предотвращения термодеструкции, уменьшения кислотных отложений.

Рассмотрена возможность замены сырья: в качестве алкилирующего агента целесообразно использование альфа-олефинов, которые имеют тенденцию к присоединению в п-положение в кольце молекулы фенола, а также исключает образование дидодецилфенолов.

Таким образом, задачей модернизации цеха №1 предприятия сооо «ЛЛК-Нафтан» являлось увеличение селективности процесса алкилирования, рациональное использование сырья, улучшение разделения продуктовой смеси.

Целевым продуктом реакции алкилирования является монододецилфенол. В процессе важную роль играет вид алкилирующего агента. Необходимо, чтобы селективность реакции образования монододецилфенола увеличилась. При алкилировании тетрамерами пропилена, выход целевого продукта достигает 42% на продукт [2].

В таблице 1 приведём сравнительные характеристики альфа-олефинов и тетрамеров пропилена.

Таблица. – Характеристики альфа-олефинов и тетрамеров пропилена

Свойство	Альфа-олефины (C ₁₂)	Тетрамеры пропилена
Внешний вид	Прозрачная жидкость без механических примесей	Прозрачная жидкость с желтоватым оттенком
Плотность при 20 °С г/см ³	0,76	0,785
Фракционный состав:		
– Начало кипения, °С	–	170

Окончание табл.

Свойство	Альфа-олефины (C ₁₂)	Тетрамеры пропилена
– Конец кипения, °С	–	215
Содержание олефинов, % масс.:		
– C ₁₂	98,0	94,0
– Выше C ₁₂	1,0	5,0
– Ниже C ₁₂	1,0	1,0
Содержание влаги, % масс.	0,0100	0.05
Бромное число г Br ₂ /100г	83–95	110–120

Можно сделать вывод о целесообразности использования альфа-олефинов в качестве алкилирующего агента, так как исходя из стерического фактора синтеза целевого продукта додецилфенола, вероятность получения п-додецилфенола даёт возможность для синтеза в дальнейшем присадки ВНИИНП-353 как сырья для дальнейшего получения присадок ВНИИНП-354 и ВНИИНП-714. Плотность альфа-олефинов фракции C₁₂ ниже, чем тетрамеров пропилена, следовательно, уменьшатся затраты на смешение исходных реагентов для алкилирования, уменьшатся энергетические затраты на процесс. С экологической точки зрения, более выгодно алкилирование альфа-олефинами, за счёт длинных углеводородных хвостов додецилфенол на 60% быстрее разлагается, чем додецилфенол, полученный при алкилировании тетрамерами пропилена [3].

Более того, согласно данным [4], при алкилировании ароматических углеводородов α -олефинами с 10–14 атомами углерода ароматические углеводороды с длинными боковыми цепями имеют значительно более высокий выход 2-фенильного изомера, который является предпочтительным, потому что углеводородные хвосты в результате дальнейшего сульфирования вещества, придают моющие свойства и свойство солюбилизации.

Также было замечено, что алкилирование фенола α -олефинами с длинной углеродной цепью имеет ряд проблем. Чем больше длина углеводородного хвоста α -олефина, тем меньшая объёмная скорость необходима, так как растворимость реагентов уменьшится соответственно. Так как α -олефины хуже смешиваются с жидким алкилфенолом, есть необходимость в установке предварительного смесителя сырья перед мешалкой. В дополнение, использование длинноцепочечных α -олефинов как реагентов может привести к формированию коксовых отложений на поверхности катализатора. Чем выше молекулярная масса α -олефина, тем сложнее контролировать образование вторичных продуктов, а именно дидодецилфенолов.

Кроме того, есть и положительные стороны использования α -олефинов в качестве алкилирующего агента. Большая конверсия сырья, селективность реакции образования монододецилфенола и срок службы катализатора также увеличивается.

Согласно источнику [5], мольное отношение α -олефинов к фенолу составляет от 10:1 до 3:1, α -олефинов к катализатору от 20:1 до 2:1, выход монозамещённых ароматических углеводородов 80%, среди которых 70% алкилированы в пара-положение. Температура в реакторе может варьироваться от 20 °С до 260 °С, при этом предпочтительны температуры в диапазоне от 100 °С до 170 °С. В качестве катализатора рассматривается морденит с размерами пор 6,7–7,5 Å.

Таким образом, с помощью внедрения новых технологий появляется возможность увеличения эффективности процесса алкилирования, а также увеличить рентабельность продукции за счёт селективности протекания целевой реакции алкилирования и увеличения выхода целевого продукта.

Обеспечение высоких темпов производства высококачественных смазочных масел за рубежом во многом связано с более масштабным применением высокоэффективных присадок. Актуальность данной темы обусловлена необходимостью повышения технико-экономических показателей работы установок по производству соответствующих компонентов присадок. Рассмотрен механизм процесса алкилирования, сделаны соответствующие выводы о структуре алкилирующего агента, на основании патентной литературы в роли алкилирующего агента предложен додецен-1.

Предложена и обоснована схема модернизации реакторного блока установки. Проанализирован рынок катализаторов для процесса алкилирования, в результате дано обоснование использованию новых катализаторов.

ЛИТЕРАТУРА

1. Рудник, Л.Р. Присадки к смазочным материалам. Свойства и применение : пер. с англ. / Л.Р. Рудник ; под ред. А.М.Данилова. – СПб. : Профессия, 2013. – 928 с., ил.
2. ТР -2010 -01-2. Промышленный технологический регламент по процессу алкилирования цеха алкилфенольных присадок ООО «ЛЛК-Нафтан».
3. Манг, Т. Смазки. Производство, применение, свойства : справ. : пер. с англ. / Т. Манг, У. Дрезель ; под ред. В.М. Школьников. – СПб. : Профессия, 2010. – 944 с.
4. Process and system for alkylation of aromatic compounds : pat. US 6315964. Unated States Patent. Date of Patent nov.13.2001/ John F. Knifton; Prakasa Rao Anantanenl, both of Austin; Melvin Stockton, Georgetown, all of TX (US).
5. Alkylation of aromatics with alpha-olefins : pat. US 4731497. Unated States Patent. . Date of Patent nov.15.1998/ Roger A. Grey, West Chester, Pa. Atlantic Richfield Company, Los Angeles, Calif.

УДК 665.654.2

**ПЕРСПЕКТИВЫ ВНЕДРЕНИЯ ПРОЦЕССА КАТАЛИТИЧЕСКОГО КРЕКИНГА
В ОАО «НАФТАН» С ЦЕЛЬЮ СОВЕРШЕНСТВОВАНИЯ
НЕФТЕХИМИЧЕСКОГО ВЕКТОРА РАЗВИТИЯ ПРЕДПРИЯТИЯ****С.С. СТЕПАНКОВ***(Представлено: канд. техн. наук, доц. А.А. ЕРМАК)*

Рассмотрены основные тенденции развития современных нефтеперерабатывающих предприятий и перспективы использования процессов каталитического крекинга для переработки нефтяных вакуумных газойлей. Проанализированы основные достоинства и недостатки существующих технологий процесса каталитического крекинга вакуумных газойлей в зависимости от конфигурации применяемого реакторного блока и катализатора. Рассмотрены особенности технологии Deep Catalytic Cracking (DCC). На основании анализа поточной схемы ОАО «Нафтан» выявлены ресурсы сырья для блоков гидродготовки и комплекса каталитический крекинг.

Введение. В настоящее время возможности нефтепереработки многих стран для удовлетворения растущих потребностей в моторных топливах за счет увеличения объемов добычи нефти практически исчерпаны. Для решения этой актуальнейшей проблемы интересны следующие тенденции развития нефтепереработки: углубление и химизация переработки нефти, оптимизация качества моторных топлив с целью расширения ресурсов и снижения их расхода, применение альтернативных топлив. Первое из перечисленных выше направлений является генеральной линией современной мировой нефтепереработки и связано с разработкой и внедрением и совершенных интенсивных экологически безвредных термокаталитических и гидрогенизационных процессов глубокой переработки нефтяных фракций [1].

Глубина переработки нефти в ОАО «Нафтан», согласно источникам [2,3] в период с 2012 по 2015 г.г. находилась в пределах от 70,3 до 75,6%. На «Нафтани» модернизация идет непрерывно. В 2018-2019 планируется пустить установку по замедленному коксованию нефтяных остатков. Тот нефтепродукт, который сегодня не подлежит переработке, будет поступать на эту установку и перерабатываться с получением дополнительных товарных нефтепродуктов, что должно увеличить глубину переработки до желанных 90-92% [4]. Однако в долгосрочной перспективе для повышения технико-экономических показателей работы завода требуется внедрение новых перспективных процессов.

Для глубокой переработки нефти решающее значение имеет каталитический крекинг(далее КК), позволяющий из малоценного тяжелого сырья получать высококачественный компонент автобензина с ОЧ 85-93 (ИМ). При этом образуется значительное количество жирного газа, богатого пропанпропиленовой (сырье для производства акрилонитрила, пропиленгликоля, глицерина) и бутан-бутиленовой фракциями (сырье для производства высокооктановых эфиров, алкилатов). Если говорить о нефтехимической промышленности, то она испытывает растущую потребность в пропилене для производства различных изделий и жирный газ КК является его основным источником и, как следствие, сырьём для производства товаров широкого потребления, в особенности пластмасс [5]. Установки КК являются также поставщиком сырья для химической промышленности: из газойлей крекинга получают сажевое сырье и нафталин, при использовании в качестве сырья гидроочищенного вакуумного газойля, тяжелый газойль может служить сырьем для производства высококачественного «игольчатого» кокса.

За длительный период своего развития каталитический крекинг значительно совершенствовался в отношении способа контакта сырья с катализатором и в отношении применяемых катализаторов [6].

Исследовательская часть. Целью настоящей работы является рассмотрение существующих технологий каталитического крекинга и анализ возможности внедрения данного процесса в поточную схему ОАО «Нафтан», с целью обеспечения загрузки нефтехимической базы завода и увеличения его прибыли.

Применение остаточного сырья в процессе каталитического крекинга имеет огромное промышленное значение. Данный процесс позволяет повысить глубину переработки нефти, увеличив при этом выход газов, в состав которых входит значительное количество этилена и пропилена, что очень выгодно отразится на экономике предприятия.

Существует три основных направления использования каталитического крекинга:

- топливное по переработке ГОВГ с добавлением мазута – FCC(UOP), MSCC;
- топливное по переработке остаточного сырья – RCC и RFCC типа R2R;
- топливно-нефтехимическое – DCC, PetroFCC, HSFCC, FCC(MAXOFIN).

Каталитический крекинг в псевдооживленном слое FCC — наиболее важный и широко применяемый в современной нефтеперерабатывающей промышленности процесс. Особенности процесса являются сравнительно низкие капитальные затраты, надёжная работа в течение длительного времени и экс-

платационная гибкость, позволяющая производителю переходить на другую структуру выхода продуктов путём простого изменения эксплуатационных параметров. Товарный бензин имеет высокое начальное октановое число и хорошие общие октановые характеристики [7].

Фирма БАРКО разработала новую концепцию системы контактирования сырья с катализатором, где используются преимущества ультракороткого времени контакта, названную впоследствии миллисекундный каталитический крекинг **MSSC**, монтаж реакторов которой обходится дешевле, чем системы с лифт-реактором, что в итоге делает стоимость реакторно-регенераторного блока на 20-30 % дешевле [8].

Преимуществами малого времени контакта являются снижение выхода сухого газа, повышение выхода бензина (дистиллятов), повышение ОЧ бензина, уменьшение реакций дегидрирования.

Фирма ЮОП предлагает заказчикам процесс переработки остаточного сырья - **RCC**. Первая из установок **RCC** рассчитана на мазут коксуемостью не более 10 %, содержанием ванадия и никеля не более 35 мг/кг или на смесь вакуумного газойля с добавлением гудрона с коксуемостью до 12 %, содержащего до 200 мг/кг металлов и до 1000 мг/кг азота. Процесс предусматривает [9]:

- тщательное управление временем контакта сырья и катализатора в реакторе;
- использование стойких к металлам катализаторов и их пассивацию;
- высокую кратность циркуляции катализаторов;
- хорошее смешение сырья с катализатором в зоне ввода их в реактор;
- низкое парциальное давление паров сырья (за счёт подачи разбавителей – пара и др).

Вторая из упомянутых установок по переработке остаточного сырья – это запатентованная фирмой «Тотал» установка **RFCC** типа **R-2-R** (реактор-2 регенератора). Процесс рассчитан на крекинг мазута с коксуемостью до 7% [9]. Для обеспечения длительной и бесперебойной эксплуатации в проект были заложены высокая работоспособность и механическая стойкость. Для переработки тяжёлого остаточного сырья, содержащего значительные концентрации металлов и дающего сравнительно высокий выход кокса, в проект заложили усовершенствованную систему подачи сырья, уникальный принцип регенерации и систему транспортировки катализатора, обеспечивающую его устойчивую циркуляцию. [7].

Если говорить о нефтехимической промышленности, то в последнее десятилетие резко возросла потребность в пропилене, что привело к его дефициту. Одновременно возросла потребность самих НПЗ в олефинах C_3 , C_4 , C_5 в связи с расширением их использования для производства алкилатов, эфиров, полимеризатов, высококачественных бензинов нового поколения [10]. Примерно половину пропилена, потребляемого химической промышленностью, получают на НПЗ, а остальная часть поступает от парового крекинга (ПК) — пиролиза. Так как основной продукт установок ПК — этилен, то для производства пропиленов и бутиленов больше подходит каталитический процесс.

Основной альтернативой пиролизу является технология **DCC** компании Stone & Webster. **DCC** - процесс глубокого КК тяжёлого сырья (например, газойли и парафинистые остатки) до легких олефинов (от C_3 до C_5) [7]. Именно технология крекинга **DCC** обеспечит преимущественно не этиленовый, а пропиленовый вектор развития нефтехимическому комплексу и комплексу по производству оксигенатов, что позволит увеличить прибыль предприятия и обеспечить загрузкой нефтехимическую базу завода.

Относительно недавно фирмой UOP был разработан процесс **PetroFCC**, который осуществляется на катализаторе со специальными присадками, таких как «SuperZ» фирмы Intercat, введение которой обеспечивает суммарный выход углеводородов C_3 - C_4 – 33%, а пропилен до 13% [11].

Главной особенностью этого процесса является использование рецикла науглероженного катализатора с целью увеличения выхода лёгких олефинов (особенно пропилена) и ароматики. Эта технология основана на том, что современные катализаторы, несмотря на закоксованность, ещё сохраняют существенную активность, причём в ряде случаев характеристики таких науглероженных катализаторов более предпочтительны, чем свежих, содержащих точки с экстремальной активностью. Как правило, кокс откладывается на более активных центрах, а катализатор становится более мягким и селективным [6].

Наряду с **PetroFCC** выделяется процесс фирмы JCCP и Saudi Aramco – **HSFCC**. **HSFCC** – каталитический крекинг с увеличенным выходом пропилена, проводимый в реакторе с нисходящим потоком сырья и катализатора и высоком отношении объёма катализатора к объёму сырья. По данным разработчиков выход пропилена увеличивается с традиционных 3 - 6% до 19 – 20%, выход высококачественного бензина с октановым числом 100 – с 40 до 48,5%. Общая степень конверсии составляет 87 – 90% [6].

Если же рассматривать сам процесс **FCC**, то выход пропилена и бутиленов можно сделать больше, чем в режиме максимального выхода бензинов, если поднять температуру в райзере выше 540 °С и применять каталитические добавки, содержащие **ZSM-5**. Установка **FCC** может быть специально спроектирована для максимального производства этилена путём интеграции в неё технологии **MAXOFIN**. Если выход пропилена в обычных условиях **FCC** составляет, как правило, 6 %масс. или менее, то включение процесса **MAXOFIN** увеличивает этот показатель до 20 и более %масс., даже если речь идёт об обычном

сырье крекинга. Выход пропилена по сравнению с обычными установками FCC повышается путём применения каталитической добавки MAXOFIN-3 и патентованного оборудования, в том числе второго райзера, предназначенного для крекирования избыточного количества бензиновых фракций и углеводородов C₄ в дополнительное количество легких олефинов.

Сравнительная характеристика направлений представлена в таблице и приводится в усреднённом виде без учёта вариаций катализаторов и зависимости от исходного сырья [7, 9-15].

Таблица. – Материальный баланс установок каталитического крекинга различных технологий

Выход продуктов, % масс.	FCC (UOP)	MSCC	RCC	R2R	DCC	PetroFCC	HSFCC	FCC (MAXOFIN)
Сухой газ C ₁ -C ₂	3,9	4,2	2,8	6,7	12,9	9	2	11,85
в т.ч. Этилен	0,9	0,84	-	-	5,7	6	4	4,3
Углеводороды C ₃ -C ₄	27,5	17,1	14,9	14	42,3	43	42,24	47,36
в т.ч. Пропиленов	8,2	5,1	5,5	-	20,4	22	20	18,37
Бутиленов	13,1	6	9,4		15,7	14	16	12,92
в т.ч. Изобутиленов	2,6		-		7,64	-	-	3,1
Бензины	47,9	53,04	43,7	44,2	20,2	28	33	18,81
Легкий газойль	8,7	13,61	18	22,1	7,9	9,5	8	8,44
Тяжёлый газойль	5,9	7,63	11,9	6,3	7,3	5	4,76	5,19
Кокс	6,1	4,32	8,7	6,7	9,4	5,5	6	8,34

Технологии топливно-нефтехимического направления отличаются более жёсткими условиями, большей загрузкой катализатора и расходом пара, что ведёт к увеличению затрат на эксплуатацию установки, однако увеличенный выход лёгких олефинов, в частности пропилена, компенсирует это и выводит установку в экономический плюс в связи со стабильно растущим спросом на пропилен. Так же очень важной составляющей любой технологии каталитического крекинга является катализатор, от которого зависит эффективность процесса. В зависимости от состава катализатора будет отличаться и сама кинетика процесса. Решение о выборе из целой гаммы предложений различных фирм наиболее целесообразной марки катализатора для конкретной установки каталитического крекинга и нефтеперерабатывающего завода в целом является ответственным мероприятием, требующим высокой квалификации и глубоких знаний по данному процессу от специалистов предприятия [13].

Как видно из вышесказанного, некоторые технологии между собой не так уж сильно и отличаются, что может означать, что при необходимости можно использовать одно и то же оборудование для переключения между топливным и топливно-нефтехимическим направлениями в зависимости от возможностей завода и его потребностей, при условии грамотной проектировки (эффективные эжекторы, контактные устройства, циклоны и т.п.). Для этого необходимо лишь соорудить дополнительные бункеры для катализатора, т.к. для FCC и DCC они будут отличаться и наладить соответствующий режим ведения технологического процесса. Также на практике различными заводами применялся опыт модернизации установки FCC до R2R, для осуществления которой достаточно дооборудовать регенератор второй ступени. Чего не скажешь о технологиях HSFCC и PetroFCC, чьи реакторно-регенераторные блоки существенно отличаются от остальных технологий.

Однако при выборе технологии, подходящей под условия ОАО «Нафтан» стоит опираться на существующее положение дел завода: его загрузку, распределения сырья по установкам и т.д. Опираясь на проектные мощности АВТ-2, АТ-8, АВТ-6, в сумме выходит около 5 млн.т/год мазута, который разгоняется на вакуумных блоках с получением гудрона и вакуумных дистиллятов(газойлей), часть которых распределяется по установкам Юникрекинг, Мягкий Гидрокрекинг, Фенольная очистка, а остальное направляется в резервуары цеха №8. Именно с цеха №8 можно брать весь газойль и тогда, получающегося после гидроподготовки, которая исходя из качества получающейся смеси газойлей просто необходима(катализатор очень чувствителен к металлам), количества хватит для загрузки установки КК.

Исходя из того, что свободного мазута на заводе нет, сразу исключаются следующие технологии – MSCC, RCC, R-2-R. Поэтому, на данный момент, на «Нафтани» нет перспектив для внедрения технологии каталитического крекинга остатков. Также, учитывая, что отрасль нефтехимии более перспективна, то стоит рассматривать технологии топливно-нефтехимического направления.

Исходя из данных таблицы, наибольший выход пропилена достигается в технологии PetroFCC, затем DCC, HSFCC и на последнем месте FCC-MAXOFIN. По выходу этилена технологии идут в следующем порядке: HSFCC-PetroFCC-DCC-FCC. Очевидно, что по выходу целевых продуктов PetroFCC и HSFCC самые эффективные технологии. Однако у этих двух технологий очень большое соотношение

катализатор/сырьё по сравнению с DCC, которая по выходу целевых продуктов ненамного отстаёт от них. Из-за того, что в технологиях глубокой переработки требуется более активный катализатор, то его стоимость будет большой и экономически невыгодно использовать такое его количество. Из вышесказанного можно сделать вывод, что в условиях ОАО «Нафтан» DCC наиболее подходящая технология, которая обеспечит не только пропиленовый вектор развития завода, достаточную загрузку нефтехимической базы завода, но и также высококачественное сырьё для получения игольчатого кокса, который имеет немалый спрос и стоимость. Стоит также повторить, что DCC при незначительных изменениях можно перевести на режим FCC. Отсюда следует, что это самый оптимальный вариант для ОАО «Нафтан».

После выбора технологии КК встаёт вопрос о подготовке сырья, а именно вакуумного газойля, к переработке на самой установке. Из-за того, что катализатор DCC более активный, то он становится и более чувствительным к отравлению. Наибольшую опасность необратимого отравления для катализатора КК представляют металлы – никель и ванадий. Значит основным критерием гидроподготовки будет выбор технологии с наибольшей деметаллизирующей способностью. Так же не стоит забывать и о других нежелательных компонентах сырья, оказывающих негативное влияние на катализатор. Поэтому на установке ГОВГ должен быть предусмотрен блок деметаллизации и гидроочистки, который будет отличаться от остальных установок ГО лишь более жесткими условиями ведения процесса и катализатором.

При прогнозируемой мощности завода по переработке нефти до 12 млн тонн в год и повышении глубины переработки нефти с 90 до 95%, проектная мощность блока гидроподготовки сырья комплекса каталитического крекинга составит около 1,5 млн тонн в год, а самой установки каталитического крекинга около 1,3 млн тонн в год.

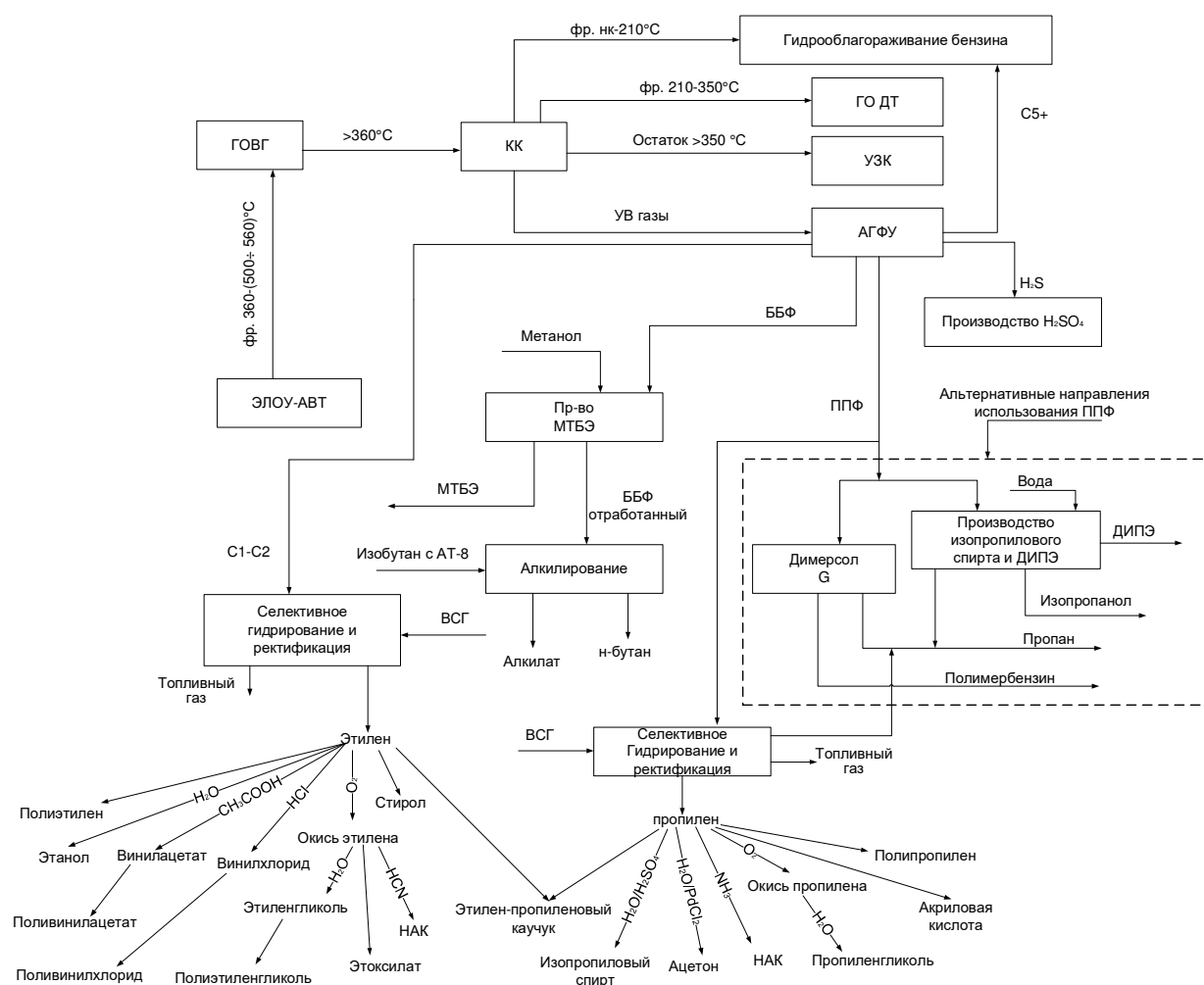


Рисунок. – Блок-схема комплекса каталитического крекинга

Остаток фр >350 °C в условиях ОАО «Нафтан» может использоваться в качестве компонента котельного топлива или как сырьё установки УЗК с целью дальнейшего получения игольчатого кокса.

В результате проведенных исследований выявлено, что внедрение КК DCC в ОАО «Нафтан» открывает перспективы для развития производства высокооктановых компонентов к бензину и расширение ассортимента продуктов из пропилена, востребованных на рынке. Данная установка каталитического крекинга уступает пиролизу по объему производства этилена, но выигрывает по производству таких низших олефинов, как пропилен и бутилены. Значимым достоинством каталитического крекинга является производство высокооктанового бензина, а также стоит отметить, что данный процесс увеличивает глубину переработки нефти.

ЛИТЕРАТУРА

1. Комплексная оценка нефтеперерабатывающих заводов и заводов по переработке тяжелых нефтей и природных битумов / З.И. Сафин [и др.] // Вестн. Казан. технол. ун-та. – 2011. – Т. 14, № 9. – С. 188–191.
2. Нефтепереработка-2015 / ЗАО Инвестиционная компания ЮНИТЕР, апрель 2015. – 19 с.
3. Развитие / Официальный сайт ОАО «Нафтан» [Электронный ресурс]. – Режим доступа: <http://www.naftan.by/ru/develop.aspx>. – Дата доступа: 07.09.2016.
4. Нефть глубокой переработки. «Белнефтехим» [Электронный ресурс]. – Режим доступа: <http://www.belneftekhim.by/print/press/about/ef845afd52323e53.html>. – Дата доступа: 03.01.2018.
5. Бурумбаева, Г.Р. Моделирование работы промышленной установки каталитического крекинга : дис. ... магистра хим. наук : 18.04.01 / Г.Р. Бурумбаева. – Томск, 2016. – 68 с.
6. Современное состояние и тенденции развития каталитического крекинга нефтяного сырья [Электронный ресурс]. – Режим доступа: <http://cyberleninka.ru/article/n/sovremennoe-sostoyanie-i-tendentsii-razvitiya-kataliticheskogo-krekinga-neftyanogo-syrya.pdf>. – Дата доступа: 03.01.2018.
7. Основные процессы нефтепереработки : справ. : пер. с англ. / Р.А. Мейерс (ред.) [и др.] ; под ред. О.Ф. Глаголевой, О.П. Лыкова. – СПб. : Профессия, 2011. – 944 с., ил.
8. Каталитический крекинг остаточного нефтяного сырья / Р.Г. Галиев [и др.] // Технология нефти и газа. – 2009. – № 1. – С. 3–11.
9. Мановян, А.К. Технология переработки природных энергоносителей / А.К. Мановян. – М. : Химия, КолосС, 2004. – 456 с., ил. – (Учебники и учеб. пособия для студентов высш. учеб. заведений).
10. Henz, H. Re-invent FCC / H. Henz // Hydrocarbon Processing. – 2004. – № 9. – С. 41–48.
11. New FCC catalyst // Hydrocarbon Processing. – 2006. – № 3. – С. 29–35.
12. Zaiting, L. DCC – A New Propylene Production Process from Vacuum Gas Oil / L. Zaiting, J. Fakang and M. Enze // NPRA Annual Meeting, AM-90-40. – 1990. – Mar. 25–27.
13. Абхметов, С.А. Технология глубокой переработки нефти и газа : учеб. пособие для вузов / С.А. Абхметов. – Уфа : Гилем, 2002. – 672 с.
14. Fundamentals of Petroleum Redining / Mohamed A. Fahim, Taher A. Alsahhaf, and Amal Elkilani // Department of Chemical Engineering / Great Britain. – 2010. – P. 513.
15. HS-FCC for Propylene: Concept to Commercial Operation [Электронный ресурс]. – Режим доступа: <https://www.axens.net/document/1199/hs-fcc-for-propylene-concept-to-commercial-operation/english.html>. – Дата доступа: 07.01.2018.

УДК 657.6:66

**МОДЕРНИЗАЦИЯ ОТДЕЛЕНИЯ ПОЛИМЕРИЗАЦИИ
ЦЕХА № 102 ЗАВОДА «ПОЛИМИР» ОАО «НАФТАН»
С ЦЕЛЬЮ ПОЛУЧЕНИЯ СОПОЛИМЕРА ЭТИЛЕНА С ВИНИЛАЦЕТАТОМ****Е.Н. ЧУГАН***(Представлено: канд. пед. наук, доц. И.В. БУРАЯ)*

Рассматривается возможность модернизации четвертого технологического потока с целью производства сополимера этилена и винилацетата. Приводятся изменения в технологической схеме потока, а также анализ свойств сополимера в зависимости от содержания в нем винилацетата.

Сополимер этилена и винилацетата (СЭВА, EVA, сэвилен)- это высокомолекулярное соединение, которое относится к группе полиолефинов, получаемое в результате сополимеризации этилена и винилацетата. Производство СЭВА с низким содержанием винилацетата (от 2 до 14%) возможно на установках по производству полиэтилена высокого давления [1]. Но в сравнении с полиэтиленом СЭВА имеет ряд преимуществ. Даже при малом содержании винилацетата заметно улучшается глянец и снижается мутность пленок из СЭВА (рис. 1). Также результаты испытания ударной прочности падающим грузом пленок одинаковой толщины свидетельствуют о резком повышении ударной прочности с ростом содержания винилацетата в сополимере (рис. 2) [1].

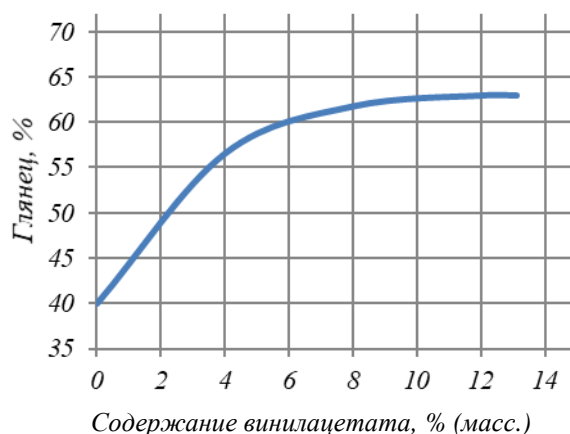


Рисунок 1. – Зависимость глянца пленок из СЭВА от содержания винилацетата

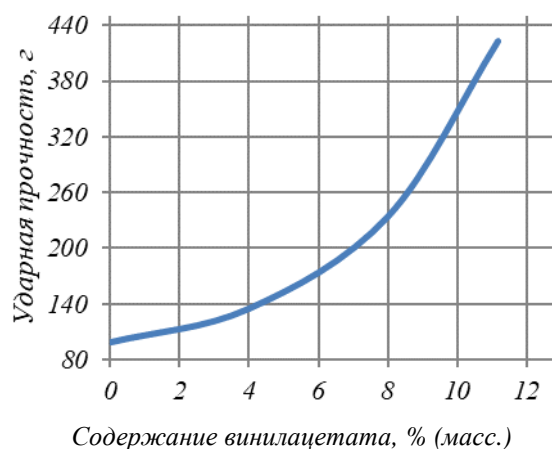


Рисунок 2. – Зависимость ударной прочности пленок из СЭВА от содержания винилацетата

Имея низкую степень кристалличности, пленки из СЭВА обладают более высокой проницаемостью для водяного пара, диоксида углерода и кислорода, чем пленки из полиэтилена. Пленки из сополимеров с различным содержанием винилацетата, даже при небольшом его содержании имеют в интервале температур от -40 до 25°C лучшую ударную прочность, чем полученные в тех же условиях пленки из полиэтилена высокого давления.

По мере увеличения содержания винилацетата в сополимере резко увеличивается стойкость к растрескиванию (рисунок 3) и снижаются температура размягчения (теплостойкость) по Вика (рисунок 4) [1].

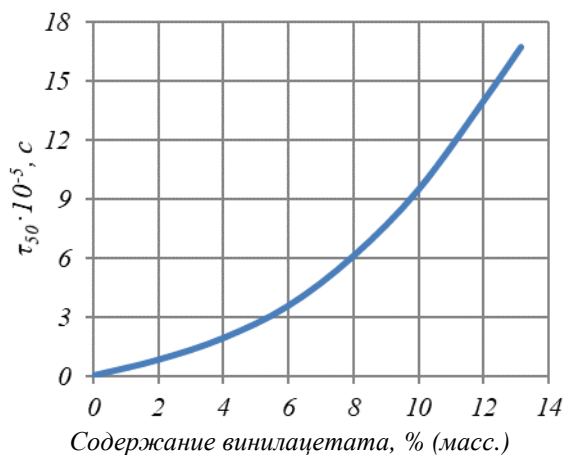


Рисунок 3. – Зависимость стойкости к растрескиванию от содержания винилацетата в СЭВА

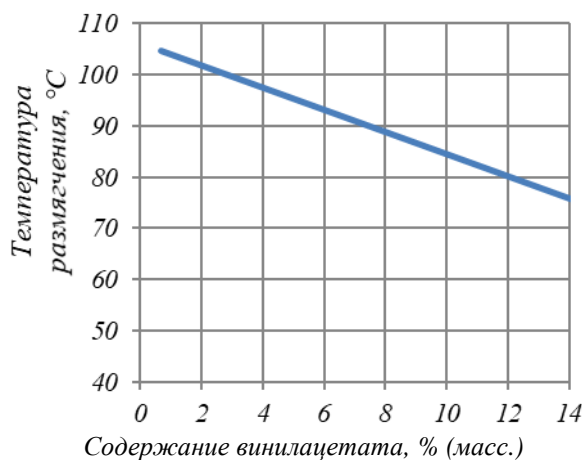


Рисунок 4. – Зависимость температуры размягчения от содержания винилацетата в СЭВА

Проведенный анализ технологической схемы цеха № 102 получения полиэтилена при высоком давлении, применяемого оборудования, а также результаты предварительного расчета основных узлов, подтвердили возможность получения СЭВА по имеющейся схеме.

Поскольку требуется ввод винилацетата в реактор, а также отделение непрореагировавшего сомономера от этилена, требуется некоторая модернизация потока. Оснащение потока заключается в установке винилацетатного насоса высокого давления Р-19 для подачи винилацетата в реактор, аналогичного используемым инициаторным насосам по характеристикам, но с большей производительностью. Насос модели «UHDE HPD 4055» фирмы Uhde High Pressure Technologies будет удовлетворять требованиям по производительности по винилацетату, которая должна составлять, согласно расчетам, 217,3 кг/ч [2].

Для хранения поступающего в цех винилацетата требуется установить емкость свежего винилацетата ТК-20, а также емкость возвратного винилацетата ТК-19 для сбора непрореагировавшего сомономера и повторного вовлечения его в процесс.

Для отделения непрореагировавшего сомомера от этилена следует установить сепараторы для возвратных газов высокого и низкого давлений. Сепаратор высокого давления V-11 является аппаратом периодического действия и эксплуатируется при переходе с производства СЭВА на полиэтилен, либо при снижении давления возвратного газа высокого давления ниже 10 МПа по причине расслоения сомомерной смеси [1]. Разделение в сепараторе происходит при дросселировании сомомерной смеси и снижении ее давления до значения давления свежего этилена, которое составляет 0,95-1,05 МПа [3]. Сепаратор низкого давления V-12 работает непрерывно. Разделение сомомерной смеси происходит предварительным охлаждением возвратного газа в теплообменнике E-19 до значений 40-50°C и последующей конденсацией винилацетата. В качестве сепараторов предполагается установка аппаратов ГС1-2,5-600, способные обеспечить требуемую степень сепарации сомомерной смеси [4].

Принципиальная технологическая схема представлена на рисунке 5.

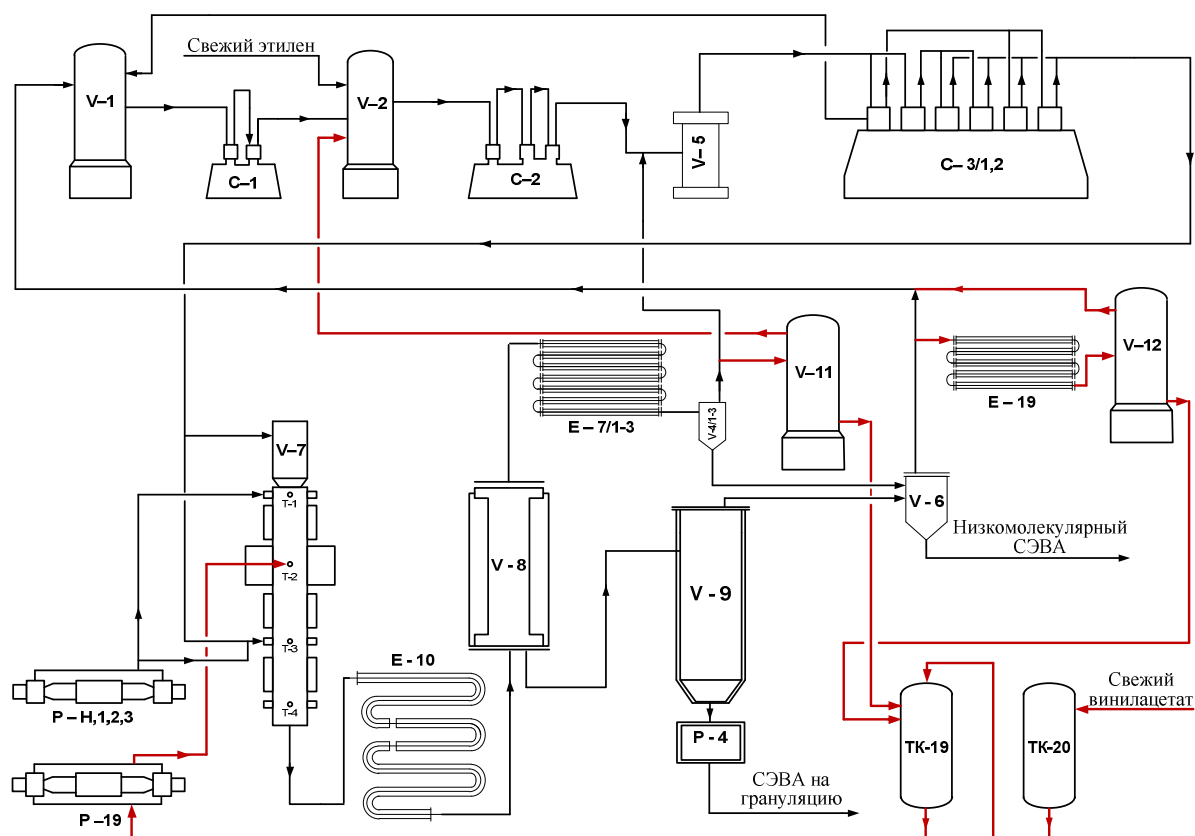


Рисунок 5. – Принципиальная технологическая схема получения СЭВА

Согласно проведенному исследованию можно сделать вывод о целесообразности модернизации четвертого технологического с целью получения сополимера этилена и винилацетата. Полученный продукт при минимальном вложении в модернизацию позволит не только покрыть его потребность в стране, но и быть успешным при поставке на экспорт, поскольку единственное его производство в СНГ находится в России, которая является монополистом в данной области.

ЛИТЕРАТУРА

1. Веселовская, Е.В. Соплимеры этилена / Е.В. Веселовская, Н.Н. Северова, Ф.И. Дунтов. – Л. : Химия, 1983. – 224 с.
2. Альфа-Интех [Электронный ресурс] // Насосы водяные высокого давления UNDE. – Режим доступа: <http://alphajet.ru/content/vodyanoi-nasos-vysokogo-davleniya-uhde>. – Дата доступа: 22.04.2018.
3. Промышленный технологический регламент производства полиэтилена низкой плотности (цех компрессии и полимеризации 1-й очереди) №102-0-11-13.
4. Петрухин, В.В. Справочник по газопромысловому оборудованию / В.В. Петрухин, С.В. Петрухин. – М. : Инфра-Инженерия, 2010. – 928 с.

УДК 657.6:66

АНАЛИЗ ВОЗМОЖНОСТИ ПРОИЗВОДСТВА СОПОЛИМЕРА ЭТИЛЕНА И ВИНИЛАЦЕТАТА В ЦЕХЕ № 102 ЗАВОДА «ПОЛИМИР» ОАО «НАФТАН»

Е.Н. ЧУГАН

(Представлено: канд. пед. наук, доц. И.В. БУРАЯ)

Рассматривается возможность производства сополимера этилена и винилацетата на мощностях цеха №102 завода «Полимир» ОАО «Нафтан». Приводится анализ литературных данных, в частности, характеристика получаемого сополимера и выбор выпускаемого марочного ассортимента. Также анализируется изменение тепловой нагрузки на автоклавный реактор при получении сополимера этилена и винилацетата.

Для этилена характерны реакции сополимеризации с другими мономерами. Это позволяет получать на основе этилена целый ряд сополимеров, обладающих различными свойствами, а также модифицировать полиэтилен, придавая ему свойства, которые невозможно получить при гомополимеризации. В качестве сополимеров в реакции сополимеризации с этиленом могут применяться: пропилен, винилхлорид, винилацетат, этилакрилат. Соплимер этилена и винилацетата (СЭВА, EVA, сэвилен) - это высокомолекулярное соединение, относящееся к группе полиолефинов, получается в результате сополимеризации этилена и винилацетата. Представляет собой гранулы от прозрачных до полупрозрачных со слабым запахом уксусной кислоты. Содержание винилацетата в сополимере может варьироваться от 2 до 45% [4].

Установки по производству СЭВА подразделяются по содержанию винилацетата в сополимере на два типа. Установки типа 1 используют для получения сополимеров, содержащих от 2 до 14% (масс.) винилацетата. На установках типа 2 получают сополимеры с содержанием винилацетата от 20 до 45% (масс.). Анализ имеющихся литературных источников по данной технологии показал, что СЭВА с низким содержанием винилацетата можно получать на установках по производству полиэтилена высокого давления без значительного переоборудования, поэтому для анализа был принят вариант производства сополимера с содержанием винилацетата в сополимере до 14% масс [1]. Для производства СЭВА с разбросом содержания винилацетата в заданных рамках возможно получение соответствующих марок сополимера, приведенных в таблице.

Таблица. – Марки СЭВА (сополимера этилена и винилацетата), производимые по ТУ 6-05-1636-97 [3]

Наименование показателя	Норма для марки		
	11104-030	11205-040	11306-075
Плотность, г/см ³	0,922 – 0,928	0,925 – 0,931	0,930 – 0,936
Показатель текучести расплава при t=190 °С, г/10	1,0 – 5,0	2,0 – 6,0	5,0 – 10,0
Массовая доля винилацетата, % масс, в пределах	5 – 7	7 – 10	10 – 14

Как видно из представленных в таблице данных показатель текучести расплава (ПТР) для данных марок колеблется от 1,0 до 10,0 г/10 мин., что свидетельствует о технической возможности получения сополимера этилена и винилацетата на четвертом технологическом потоке, где реализована возможность переработки термопласта со значениями ПТР от 0,24 до 30,0 г/10 мин [2].

Принципиальным вопросом технологии полимеризации и сополимеризации этилена при высоком давлении является обеспечение эффективного отвода тепла реакции. Таким образом, возможный экзотермический эффект реакции сополимеризации является основным критерием определения возможности получения сополимера этилена и винилацетата на имеющихся мощностях завода.

Теплота экзотермической реакции сополимеризации может быть рассчитана по правилу аддитивности [1]. Для расчета принимаем за основу СЭВА с содержанием винилацетата 10% масс, как среднее значение.

$$Q_{\text{сopol.}} = Q_{\text{пол.}}^{\text{Эт.}} \cdot 0,9 + Q_{\text{пол.}}^{\text{Ва.}} \cdot 0,1$$

где $Q_{\text{пол.}}^{\text{Эт.}}$ – тепловой эффект полимеризации 100% этилена, $\frac{\text{кДж}}{\text{кг}}$;
 $Q_{\text{пол.}}^{\text{Ва.}}$ – тепловой эффект полимеризации 100% винилацетата $\frac{\text{кДж}}{\text{кг}}$.

$$Q_{\text{сopol.}} = 3560 \cdot 0,9 + 1026 \cdot 0,1 = 3306,6 \frac{\text{кДж}}{\text{кг}}$$

Согласно произведенным расчетам, тепловая нагрузка на реактор при получении СЭВА в сравнении с получением полиэтилена снижается на 7%. Следовательно, ожидается снижение теплообразования на 1 кг полученного СЭВА в сравнении с полиэтиленом.

На заводе «Полимир» ОАО «Нафтан» реализуются одновременно два вида полимеризации: полимеризация одним потоком в трубчатом реакторе (цех №105), и полимеризация четырьмя технологическими потоками в автоклавных реакторах с мешалками (цех №102). В мировой химической промышленности наибольшее распространение получили технологии с трубчатыми реакторами по причине большей производительности относительно автоклавных реакторов. В цехе №102 завода «Полимир» ОАО «Нафтан» реализован процесс полимеризации этилена при высоком давлении в реакторах автоклавного типа с перемешивающим устройством в присутствии инициаторов (органических перекисных соединений) при точно установленных параметрах давления – в диапазоне 110-155 МПа и температур – от 160 до 280 °С.

Производство СЭВА целесообразно организовать именно на производственной площадке цеха № 102, так как его отличает невысокая производительность (от 20 до 55 тонн в сутки по продукту), большая маневренность и гибкость работы. Автоклавный реактор намного быстрее выходит на режим, чем трубчатый, а это существенно снижает издержки производства и выход некондиционной продукции. Поток цеха №102 может быстрее и с меньшими материальными затратами переходить от выпуска ПЭВД к выпуску СЭВА и наоборот. Получаемый в автоклавном реакторе сополимер имеет более равномерное строение, поскольку мешалка поддерживает постоянство концентраций по всему объему аппарата.

Одной из проблем эксплуатации автоклавного реактора в сравнении с трубчатым является низкий теплоотвод через стенку и, следовательно, ограничение максимальной температуры в аппарате. Снижение теплового эффекта позволит увеличить температуру в реакторе, что способствует увеличению конверсии сомономеров. Достигнутая конверсия полимеризации этилена в настоящее время составляет 15%.

Таким образом, в производственных условиях завода «Полимир» ОАО «Нафтан» целесообразным является выпуск базовых марок СЭВА, имеющих наивысший приоритет (спрос) и существенные конкурентные преимущества по сравнению с ПЭВД.

Согласно проведенному исследованию можно сделать вывод, что выпуск сополимера этилена и винилацетата с содержанием в нем винилацетата до 14% (масс.) на четвертом технологическом потоке вполне возможен.

На основании проведенного расчета и анализа полученных результатов можно сделать заключение о возможности использования имеющегося реактора автоклавного типа в цехе № 102 для получения сополимера этилена и винилацетата.

ЛИТЕРАТУРА

1. Веселовская, Е.В. Сополимеры этилена / Е.В. Веселовская, Н.Н. Северова, Ф.И. Дунтов. – Л. : Химия, 1983. – 224 с.
2. Промышленный технологический регламент производства полиэтилена низкой плотности (цех компрессии и полимеризации 1-й очереди) №102-0-11-13.
3. УралСинтез [Электронный ресурс] // Сэвилен. ТУ 6-05-1636-97: – Режим доступа: <http://uralsintez.ru/sevilen/>. – Дата доступа: 20.04.2018.
4. Руспласт [Электронный ресурс] // Этиленвинилацетат (ЭВА, EVA). – Режим доступа: <http://rusplast.by/catalog/eva/>. – Дата доступа: 23.04.2018.

СОДЕРЖАНИЕ

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

<i>Лабузова А.П.</i> Целесообразность разработки игрового приложения «Flamy – необычная головоломка» для мобильных устройств под управлением операционной системы Android	3
<i>Лабузова А.П.</i> Проектирование игрового приложения «Flamy – необычная головоломка» для мобильных устройств под управлением операционной системы Android	6
<i>Войтов А.В.</i> Краткий обзор библиотеки Material Design Lite	10
<i>Войтов А.В.</i> Обработка изображений с помощью библиотеки OpenIMAJ	13
<i>Гаврилов В.В.</i> Создание электронного каталога изображений Козьянковского клада арабских куфических дирхамов IX–X веков из музейного собрания Национального Полоцкого историко-культурного музея-заповедника	16
<i>Гаврилов В.В.</i> Использование HTML Application для создания оболочки электронного каталога изображений Козьянковского клада арабских куфических дирхамов IX–X веков из музейного собрания Национального Полоцкого историко-культурного музея-заповедника	19
<i>Карпович А.Д.</i> Методы оптимизации производительности мобильных игр в Unity3d	21
<i>Карпович А.Д.</i> Сравнение архитектурных паттернов проектирования игр	25
<i>Курилович А.Ю.</i> Мобильное приложение для распознавания реквизитов банковских карт под управлением iOS	29
<i>Курилович А.Ю.</i> Предварительная обработка изображений банковских карт для распознавания их реквизитов с использованием мобильных устройств под управлением iOS	32
<i>Перельгин И.М.</i> Нейронные сети и экспертные системы	36
<i>Перельгин И.М., Ананенко А.А.</i> Разработка логической структуры и тестового задания для приложения виртуальной реальности	38
<i>Попов А.И.</i> Проектирование графического интерфейса для автоматизированной информационной системы туристического центра учреждения образования «Полоцкий государственный университет»	40
<i>Попов А.И.</i> Проектирование базы данных для автоматизированной информационной системы туристического центра учреждения образования «Полоцкий государственный университет»	43
<i>Суцеский Д.В.</i> Архитектура клиент-серверных игровых приложений	46
<i>Суцеский Д.В.</i> Архитектурный шаблон проектирования Entity-Component-System	48
<i>Лапковский К.И.</i> Проблема использования чистых функций в языке программирования JavaScript	51
<i>Лапковский К.И.</i> Переопределение контекста вызова по умолчанию в языке программирования JavaScript	53
<i>Лапковский Н.И.</i> Эффективные методы SEO-продвижения сайтов, построенных на базе CMS Wordpress	55
<i>Лапковский Н.И.</i> Генерация случайных (псевдослучайных) чисел в языке программирования PHP	58
<i>Карабанов Р.Ю.</i> Пять уровней управления информационной безопасности в организации	60
<i>Карабанов Р.Ю.</i> Как выполняется шифрование в мессенджере Telegram	62
<i>Роголев В.С.</i> Выбор средств и методов решения задач. Проектирование базы данных обучающего приложения для изучения английского языка	64
<i>Роголев В.С.</i> Экономическая составляющая разработки обучающего приложения для изучения английского языка. Оценка влияния информационной среды на экономику	66
<i>Талайко П.Д.</i> Защита данных пользователя на стороне сервера посредством фреймворка Spring Security и технологии JSON Web Token	68
<i>Талайко П.Д.</i> Защита информации пользователя на клиентской части веб-приложения посредством JWT Токена и встроенных механизмов Angular 5	71
<i>Татарин Д.С.</i> Протокол Hypertext Transfer Protocol Secure. Безопасность WEB-сервисов	74
<i>Татарин Д.С.</i> Проектирование базы данных для музыкального стримингового веб-сервиса «Blzmusic»	77

Тращенко М.А. Особенности создания графических интерфейсов с помощью технологий Windows Forms	81
Тращенко М.А. Принципы моделирования базы данных на примере автоматизированной информационной системы учета товаров	84
Гончаров В.А. Проектирование интерфейса автоматизированной информационной системы «Комиссионный магазин» на Платформе 1с:Предприятие 8.3	87
Гончаров В.А. Проектирование базы данных автоматизированной информационной системы «Комиссионный магазин» на Платформе 1с:Предприятие 8.3	89
Захарова И.Ю. Блочный алгоритм обнаружения и классификации объектов с использованием сверточной нейронной сети для видеопоследовательностей большого разрешения	91
Захарова И.Ю., Воробьев Д. Анализ архитектур сверточных нейронных сетей для обнаружения и классификации объектов на видеопоследовательностях большого разрешения	94
Медведева О.Н. Расчет температурных полей двухслойного полого цилиндра методом разделения переменных	98
Корсак И.А. Визуализация и графика в Unity3d	103
Корсак И.А. Технологии виртуальной реальности в обучении персонала. Учебный класс для работы в виртуальной реальности	107
Ананенко А.А. Применение технологий виртуальной реальности в обучении специалистов на примере приложения для газовых служб	110
Ананенко А.А. Создание приложения виртуальной реальности «газорегуляторный пункт VR: разработка 3D-моделей и скриптов логики приложения»	113
Субботин А.В. Проектирование графического интерфейса приложения гибридная криптосистема	117
Субботин А.В. Разработка гибридной криптосистемы для защиты пользовательских данных	120
Андреев И.С. Проектирование графического интерфейса системы для скрытия информации на основе алгоритма последовательной стеганографии	122
Андреев И.С. Разработка системы для скрытия информации на основе алгоритма последовательной стеганографии	125
Соловьёв А.А. Тенденции развития свёрточных нейронных сетей	128
Соловьёв А.А. Библиотека машинного обучения TensorFlow	131
Потапов М.А. Проектирование интернет-магазина Telegram-каналов	134
Потапов М.А. Разработка бота-помощника для интернет-магазина в Telegram	137
Зязюля П.В. Анализ функциональных возможностей электронной торговой площадки	139
Зязюля П.В. Разработка электронной торговой площадки	141
Карнилович А.В. Защита данных приложений, разработанных на Unity3D	143
Карнилович А.В. Проектирование графического интерфейса для системы безопасного хранения персональной информации	145
Станкевич К.В. Проектирование графического интерфейса системы для скрытия информации на основе дискретных преобразований с помощью алгоритма Коха и Жао	148
Станкевич К.В. Разработка системы для скрытия информации на основе дискретных преобразований с помощью алгоритма Коха и Жао	151
Иваненко И.Е. Разработка алгоритма шифрования на основе алгоритмов кубика Рубика	154
Иваненко И.Е. Сравнительный анализ алгоритма шифрования на основе алгоритма кубика Рубика и AES	157
Смоляк А.И. Разработка системы для скрытия информации с использованием гиперболических функций	159
Смоляк А.И. Проектирование графического интерфейса системы для скрытия информации с использованием гиперболических функций	163
Маханько П.В. Проектирование графического интерфейса для веб-приложения, разработанного для обмена информацией и файлами в ограниченном кругу лиц	166
Маханько П.В. Проектирование API и базы данных для веб-приложения, разработанного для обмена информацией и файлами в ограниченном кругу лиц	169
Радченко В.С. Рассмотрение фреймворков PHP в общем виде и изучение Yii в частности	171
Радченко В.С. Создание веб-приложения с использованием PHP-фреймворка Yii	173
Макарычева В.А., Макарычев М.Ю. Управление транзакциями над файловыми системами в Java-приложениях	177

<i>Макарычева В.А., Макарычев М.Ю.</i> Система безопасного выполнения динамически генерируемого Java-кода	179
<i>Ларионов И.В.</i> Проектирование мобильного игрового приложения «Cube» под управлением операционной системы Android	181
<i>Ларионов И.В.</i> Разработка мобильного игрового приложения «Cube» под управлением операционной системы Android	183
<i>Пацанков Д.Н.</i> Разработка пользовательского интерфейса для автоматизированного рабочего места по учету автотранспорта	185
<i>Пацанков Д.Н.</i> Проектирование автоматизированного рабочего места по учету работы автотранспорта	187
<i>Денисова Е.С.</i> Возможные варианты хранения информации, анализ и выбор оптимального хранилища для дальнейшей реализации	190
<i>Денисова Е.С.</i> Подходы к разработке интерфейса для веб-приложения «Разработка и реализация математической двухключевой криптосистемы для облачного хранилища»	192
<i>Петюкевич В.В.</i> Создание схемы защищённой передачи сообщений между пользователями	194
<i>Петюкевич В.В.</i> Проектирование графического интерфейса клиентской части веб-приложения чат ...	197
<i>Лапковский К.И.</i> Проблема использования чистых функций в языке программирования JavaScript	200
<i>Лапковский К.И.</i> Переопределение контекста вызова по умолчанию в языке программирования JavaScript	202
<i>Самсонов К.С.</i> Проектирование графического интерфейса для веб-приложения, разработанного для отслеживания статуса выполняемых итераций и слежение времени их выполнения	204
<i>Самсонов К.С.</i> Проектирование базы данных для веб-приложения, предназначенного для отслеживания стадий выполнения разрабатываемых проектов	206
<i>Сметана Е.В.</i> Оптический генератор для передачи видеосигналов по оптическому каналу на поднесущих частотах	207
<i>Сметана Е.В.</i> Частотный модулятор для передачи видеосигналов по оптическому каналу на поднесущих частотах	209
<i>Змитрович С.Ю.</i> STM32CUBEMX в настройке периферии микроконтроллеров компании ST	211
<i>Змитрович С.Ю.</i> Программное обеспечение анализа аналоговых сигналов физических величин	214
<i>Становой А.О.</i> Фильтрация спекл-шума в системах дистанционного зондирования Земли	217
<i>Становой А.О.</i> Анализ базовых алгоритмов фильтрации спекл-шума	220
<i>Царькова Е.С.</i> Перчатка с датчиками изгиба	222
<i>Ловкис В.В.</i> Применение микропроцессоров для систем релейной защиты и автоматики	226

МАШИНОСТРОЕНИЕ

<i>Гавриловец Д.А.</i> Особенности технологического обеспечения переработки офисной макулатуры	229
<i>Гавриловец Д.А.</i> Совершенствование эффективности блока механического разрыва бумажной массы персональной станции переработки офисной макулатуры	232
<i>Бейтуллаев А.Т., Дубов Г.Д.</i> Дизельное топливо и его характеристики	234
<i>Бейтуллаев А.Т., Дубов Г.Д.</i> Сравнительный анализ качества дизельного топлива	236
<i>Гасс Ю.А., Лопатин Д.В.</i> Влияние углов установки колёс на управляемость автомобиля	238
<i>Гасс Ю.А., Лопатин Д.В.</i> Способы регулировки углов установки колёс и основные причины их нарушения	241
<i>Гуца А.В., Черниченко М.А.</i> Анализ отработанного моторного масла	244
<i>Гуца А.В., Черниченко М.А.</i> Отработанное моторное масло	246
<i>Закревский Д.В., Зубович Н.Я.</i> Диагностика современных бензиновых двигателей	248
<i>Шевченко Д.С., Войтехович В.В.</i> Альтернативные методы диагностирования систем Common Rail современных дизельных двигателей	254
<i>Жихорев Е.О.</i> Анализ технологий врезания при фрезеровании карманов на станках с ЧПУ	259
<i>Жихорев Е.О.</i> Определение оптимального цикла снятия основного припуска при фрезеровании карманов на станках с ЧПУ	262

Плющенко О.Д., Юшкевич П.В. Система непосредственного впрыска топлива и ее недостатки в бензиновых двигателях	265
Башлачёв Д.А., Пуйман Е.В. Сравнительное исследование износостойкости шлифованных и нешлифованных образцов с комбинированным покрытием 95ХС и TiN	268
Скурьят С.В. Методы контроля напряжений сварных соединений	271
Царькова Е.С. Совершенствование организации дорожного движения в городе Новополоцке	273

ХИМИЧЕСКИЕ ТЕХНОЛОГИИ

Корбут Е.Я. Инновационные идеи в процессе алкилирования	278
Степанков С.С. Перспективы внедрения процесса каталитического крекинга в ОАО «Нафтан» с целью совершенствования нефтехимического вектора развития предприятия	281
Чуган Е.Н. Модернизация отделения полимеризации цеха № 102 завода «Полимир» ОАО «Нафтан» с целью получения сополимера этилена с винилацетатом	286
Чуган Е.Н. Анализ возможности производства сополимера этилена и винилацетата в цехе № 102 завода «Полимир» ОАО «Нафтан»	289