

**Министерство науки и высшего образования Российской Федерации**  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
**АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**  
**(ФГБОУ ВО «АмГУ»)**

Факультет математики и информатики  
Кафедра математического анализа и моделирования  
Направление подготовки 01.04.02– «Прикладная математика и информатика»  
Направленность (профиль) образовательной программы – «Математическое  
и программное обеспечение информационных систем»

ДОПУСТИТЬ К ЗАЩИТЕ  
И.о. зав. кафедрой  
\_\_\_\_\_ Н.Н. Максимова  
« \_\_\_\_\_ » \_\_\_\_\_ 2021 г.

**МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ**

на тему: Разработка приложения для автоматического формирования отчетов  
о работе цифровых передатчиков

Исполнитель  
студент группы 952 ом

\_\_\_\_\_

(подпись, дата)

А.С. Герман

Руководитель  
доцент, канд. физ.-мат. наук

\_\_\_\_\_

(подпись, дата)

Е.М. Веселова

Руководитель научного  
содержания программы  
магистратуры  
профессор, д-р физ.-мат. наук

\_\_\_\_\_

(подпись, дата)

А.Г. Масловская

Нормоконтроль  
доцент, канд. физ.-мат. наук

\_\_\_\_\_

(подпись, дата)

Е.М. Веселова

Рецензент  
доцент, канд. техн. наук

\_\_\_\_\_

(подпись, дата)

Л.В. Никифорова

Благовещенск 2021

**Министерство науки и высшего образования Российской Федерации**  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
**АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**  
**(ФГБОУ ВО «АмГУ»)**

Факультет математики и информатики  
Кафедра математического анализа и моделирования

УТВЕРЖДАЮ

И.о. зав. кафедрой

\_\_\_\_\_ Н.Н. Максимова

«\_\_\_\_\_» \_\_\_\_\_ 2021 г.

**З А Д А Н И Е**

К магистерской диссертации студента: Герман Анны Сергеевны.

1. Тема магистерской диссертации: Разработка приложения для автоматического формирования отчетов о работе цифровых передатчиков (утверждена приказом от 05.02.2021 №181-уч)
2. Срок сдачи студентом законченной работы: 23.06.2021 г.
3. Исходные данные к магистерской диссертации: отчет по преддипломной практике, средства разработки приложения – среда программирования VBA на базе табличного процессора MS Excel, среда программирования QtCreator4.15, кроссплатформенная библиотека фреймворка с открытым исходным кодом Qt 5.9, учебная и научная литература.
4. Содержание выпускной квалификационной работы (перечень подлежащих разработке вопросов): анализ методов и технологий обработки табличных данных; определение критерий и требований, необходимые для создания доступного и функционального программного обеспечения; разработка программного обеспечения, которое отвечает поставленным требованиям; апробация программного обеспечения на реальных данных, выгруженных из базы данных; использование результатов магистерской диссертации в профессиональной деятельности.
5. Перечень материалов приложения: листинг программы

6. Консультанты по магистерской диссертации: рецензент – Никифорова Л.В., канд. техн. наук, доцент кафедры информационных и управляющих систем, ФГБОУ ВО «Амурский государственный университет», доцент; нормоконтроль – Веселова Е.М., доцент, канд. физ.-мат. наук.

7. Дата выдачи задания: 02.03.2021 г.

Руководитель магистерской диссертации: Веселова Елена Михайловна, доцент, канд. физ.-мат. наук, доцент.

Задание принял к исполнению (02.03.2021): \_\_\_\_\_ Герман А.С.

## РЕФЕРАТ

Магистерская диссертация содержит 70 с., 21 рисунок, 1 таблицу, 1 приложение, 23 источника.

СИСТЕМА SD, ИНЦИДЕНТ, COM, OLE, VBA, WORD, ЕСТЕСТВЕННО-ЯЗЫКОВАЯ ОРИЕНТИРОВАННОСТЬ, ПРОГРАММНЫЙ МОДУЛЬ, VBA, EXCEL, QT, DLL, ФРЕЙМВОРК, ГРАФИЧЕСКИЙ ИНТЕРФЕЙС ПОЛЬЗОВАТЕЛЯ

В работе исследован процесс автоматизации оформления отчетов о работе цифровых телевизионных передатчиков. В связи с большим объемом манипуляций необходимых для формирования отчета, разработка программного обеспечения, автоматизирующая этот процесс, позволит сократить затрачиваемое на оформление отчетов время и снизить количество ошибок, возникающих по невнимательности персонала, что является весьма актуальной задачей.

Цель работы – разработка приложения для автоматического формирования отчетов о работе цифровых передатчиков.

В магистерской диссертации в среде программирования VBA табличного процессора Excel было разработано приложение для автоматического оформления отчетов о работе цифровых телевизионных передатчиков на основе данных об инцидентах, представленных в таблицах Excel. В работе проведен анализ методик и технологий, применяемых для обработки табличных данных, разработан программный модуль облегчающий работу с большим объемом табличной информации, а также внешняя подключаемая библиотека фреймворка для более удобного создания графического интерфейса пользователя.

## СОДЕРЖАНИЕ

Введение	7
1 Программное обеспечение «Амурского ОРТПЦ»	11
2 Способы программной реализации алгоритмов обработки таблиц Excel	15
2.1 Формулировка задачи	15
2.2 COM и OLE	16
2.3 VBA (Visual Basic for Applications)	17
2.4 Среда и язык программирования	17
2.5 Естественно-языковая ориентированность	18
2.6 Пользовательский интерфейс макросов VBA	19
2.7 Причины выбора Qt как базового инструмента для создания пользовательского интерфейса	21
2.8 Параллельные вычисления в среде VBA Excel	22
2.9 Основные требования к программному продукту	26
3 Разработка программного модуля для обработки таблиц и формирования отчетов	28
3.1 Структура и логика работы программного модуля	28
3.2 Функции общего назначения и константы	30
3.3 Работа с таблицами	32
3.4 Регулярные выражения	35
3.5 Сортировка и фильтрация	36
3.6 Подстановка по ключевому элементу	38
3.7 Подстановочные шаблоны	39
4 Разработка программного модуля для создания графического интерфейса пользователя	42
4.1 Структура и логика работы программного модуля	42
4.2 Подключение библиотеки фреймворка	45
4.3 Работа с окнами	45

4.4 Работа со слоями	46
4.5 Работа с элементами графического интерфейса	46
4.6 Работа со списком быстрого вывода	48
4.7 Пример применения фреймворка	48
5 Реализация программного обеспечения для формирования отчетов о работе цифровых передатчиков	51
5.1 Структура программного обеспечения	51
5.2 Графический интерфейс пользователя	52
5.3 Контроль ошибок	56
Заключение	58
Библиографический список	60
Приложение А Листинг программы «Формирование отчетов о работе цифровых передатчиков»	63

## ВВЕДЕНИЕ

На территории области в настоящий момент времени действует филиал РТРС «Амурский ОРТПЦ». Эта организация является подразделением Российской телевизионной и радиовещательной сети, и основным оператором цифрового эфирного теле- и радиовещания на территории Амурской области.

Одним из производственных подразделений «Амурского ОРТПЦ» является отдел оперативного управления, в котором оперативный дежурный отслеживает состояние передатчиков на передающих станциях, а также проводит работы по устранению и недопущению аварийных ситуаций [1].

Одной из задач оперативного дежурного является непосредственный отчет о работе передатчиков. Само же формирование отчетов отнимает достаточно большое количество времени, так как каждый отчет формируется вручную на основе табличных данных, выгруженных из специальной программы. Дежурный не всегда может успеть сформировать необходимый отчет к положенному сроку, в связи с чем возник вопрос об автоматизации процесса.

Большинство мощных пакетов обработки данных включают общие инструменты, позволяющие расширять и автоматизировать возможности программного продукта, например, интегрированные программные среды на основе сценарных языков программирования[2].

К сожалению, для работы с такими мощными инструментами требуется определенная подготовка, и для большинства рядовых пользователей предоставленные возможности остаются недоступными[2]. Возникает потребность в более простом и интуитивно понятном инструменте, который мог бы удовлетворить возникшие потребности в формировании производных данных на основе табличных[3].

Помимо разработки самого инструмента по обработке табличных данных, главная проблема заключается так же в создании графического пользовательского интерфейса в среде VBA, так как иногда при разработке макроса требует-

ся создание многочисленных, но при этом простых по своей структуре и функционалу, пользовательских форм. В среде VBA имеется встроенный редактор графического интерфейса, однако, разработка интерфейса в нем довольно трудоемка и дает весьма громоздкую кодовую реализацию [4]. Разработка внешнего фреймворка может облегчить и ускорить создание простых пользовательских форм с минимальным объемом необходимого программного кода, что, конечно же, приведет к сокращению времени разработки интерфейсов для макросов.

Цель работы – разработка приложения для автоматического формирования отчетов о работе цифровых передатчиков.

Для достижения поставленной цели необходимо выполнить следующие задачи:

1. Проанализировать методику формирования отчетов, определив необходимое количество входной информации.
2. Определить критерия и требования, необходимые для создания доступного и функционального программного обеспечения, способного решать широкий перечень задач, связанных с автоматизацией обработки таблиц.
3. Разработать программное обеспечение, которое отвечает поставленным требованиям простоты и функциональности.
4. Сформулировать основные требования к графическому интерфейсу, разработать графический интерфейс пользователя.
5. Проанализировать результаты работы программного обеспечения.

Новизна работы заключается в разработке уникального программного инструментария, включающего в себя набор функций для гибкого редактирования таблиц, быстрого создания графического интерфейса пользователя в среде разработки VBA в MS Excel и использующего принцип естественно-языковой ориентированности на русскоязычного разработчика, который позволяет облегчить понимание программного кода с минимальными начальными знаниями программирования.

Практическая значимость заключена в создании на основе разработанно-

го программного инструментария приложения по автоматическому формированию отчетов по работе цифровых передатчиков для отдела «ОУС» организации «Амурский ОРТПЦ», позволяющее снизить количество ошибок при их создании и так же уменьшить временные затраты необходимые для ручной обработки большого массива данных.

Результаты работы докладывались и обсуждались на четырех научных конференциях: III Всероссийской национальной научной конференции студентов, аспирантов и молодых ученых «Молодежь и наука: актуальные проблемы фундаментальных и прикладных исследований», 06-10 апреля 2020 г., Комсомольский-на-Амуре государственный университет; XVIII региональной научной конференции «Физика: фундаментальные и прикладные исследования, образование», Хабаровск, 10-12 ноября 2020 г.; XXIX научной конференции Амурского государственного университета «День науки», 23-25 ноября 2020 г.; региональная научно-практическая конференция «ТОГУ - Старт: фундаментальные и прикладные исследования молодых», 10-12 апреля 2021 г.; XXX научной конференции Амурского государственного университета «День науки», 15 апреля 2021 г.; на научно-методических семинарах кафедры математического анализа и моделирования ФГБОУ ВО «Амурский государственный университет» (2020-2021 гг.).

Доклад на тему «Параллельные вычисления в среде VBA Excel» отмечен диплом I степени на секции «Математическое и компьютерное моделирование» XXIX научной конференции Амурского государственного университета «День науки», 24 ноября 2020 г.

Также за период обучения в магистратуре была получена Благодарность председателя Законодательного Собрания Амурской области за активную и плодотворную работу в составе студенческой проектной группы по созданию малых космических аппаратов и полезных нагрузок, включая первый студенческий спутник Дальневосточного федерального округа и Амурского государственного университета – «АмурСат».

По теме магистерской диссертации опубликовано шесть работ [18 – 23], одна статья в журнале, входящем в РИНЦ, Вестник АмГУ (серия «Естественные и экономические науки») и 5 работ в сборниках материалов конференций.

Результаты работы поддержаны студенческим грантом Амурского государственного университета.

Магистерская диссертация состоит из введения, пяти глав, заключения, библиографического списка и одного приложения.

В первой главе кратко описано программное обеспечение, применяемое оперативными дежурными в организации «Амурский ОРТПЦ и тип входных данных для их дальнейшей обработки.

Во второй главе проведен анализ предметной области, определены способы программной реализации алгоритмов обработки таблиц Excel обоснован выбор методов и программных сред для решения поставленной задачи.

В третьей главе были сформулированы требования к необходимому функционалу для обработки табличных данных, приведен список основных команд, содержащихся в разработанном модуле и показаны примеры его применения.

В четвертой главе были сформулированы требования к графическому интерфейсу пользователя, обоснован выбор библиотеки фреймворка и показаны примеры его применения.

В пятой главе проведена реализация программного обеспечения для формирования отчетов о работе цифровых передатчиков, приведена структура программного обеспечения и логика работы программного обеспечения.

## 1 ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ «АМУРСКОГО ОРТПЦ»

На рабочем месте оперативному дежурному часто приходится сталкиваться с различными программами. Одной из них, на базе которой и формируется дальнейший отчет, является так называемая система SD.

Система SD в работе необходима для обеспечения автоматизации оперативного управления в процессах эксплуатации и технического обслуживания передающих станций РТПС. Она является основной системой и в ней оперативным дежурным чаще всего приходится сталкиваться с «инцидентами».

Инцидент – это некое аварийное событие, которое свидетельствует о нарушении нормальной работы технического средства. Регистрируется он либо на основе аварийного сообщения из подсистемы другой системы АСКиУ (в таком случае обозначается инцидент меткой FM), так и в ручном режиме – действиями оперативного дежурного.

Ежедневно в SD фиксируется до 1000 инцидентов. В зависимости от внешних условий их количество может, как уменьшаться (минимальное число за сутки достигало 300), так и увеличиваться (последнее максимальное фиксированное количество инцидентов достигало 4000). Каждый инцидент в индивидуальном порядке рассматривается оперативным дежурным. Он впоследствии определяет причину аварийной остановки и при необходимости редактирует созданный инцидент. В описание инцидента заносятся такие причины, как нарушение целостности структуры ТП сигнала мультиплекса вследствие неустановленной помехи, работа передатчика пониженной или повышенной мощностью, аварийное отключение электроэнергии по вине энергоснабжающей организации, восстановление нормальной работы передатчика после подачи электроэнергии на входе станции, режим защиты передатчика и другие.

Одной из обязанностей оперативного дежурного является обязательное составление отчета в трех экземплярах о количестве и типе инцидентов. Отчет формируется оперативным дежурным ночной рабочей смены, до восьми утра по местному амурскому времени. Иными словами, отчеты содержат инциден-

ты, произошедшие в течение суток (от 00:00 одного дня до 00:00 другого дня по московскому времени, или же от 06:00 утра одного дня до 06:00 другого по амурскому времени).

Первый отчет формируется для начальства Амурского ОРТПЦ. Так же этот отчет называют местной сводкой. В силу специфики SD инциденты для этого отчета необходимо корректировать: переводить время на местное, так как каждый инцидент фиксируется в базе по московскому времени. Инциденты сначала сортируются по причине, а после внутри по пункту установки. В каждом инциденте необходимо указывать тип технического средства. Пример отчета указан на рисунке 1.

Отключение электроэнергии по вине энергоснабжающей организации											
Источник	Пункт устан.	ТС	ТРК/МУж	Начало нарушения		Продолжительность	Причина	Окончание нарушения		Описание	
				дата	время			дата	время		
Вручную	Гуран	АМС-Опора-100292420	РТПС-1 (+6) РТПС-2 (+7)	07.12.2019	7:33:34	3:44:16	Отключение электроэнергии по вине энергоснабжающей организации с переходом на работу от ИБП	07.12.2019	11:17:50	Аварийное отключение электроэнергии. Работа от ИБП с 7:33:34 по 11:17:50.	
Вручную	Буссе	АМС-Башня-91356	РТПС-1 (+6) РТПС-2 (+7)	08.12.2019	3:19:36	1:37:24	Отключение электроэнергии по вине энергоснабжающей организации с переходом на работу от ИБП	08.12.2019	4:57:00	Аварийное отключение электроэнергии. Работа от ИБП с 03:19:36 до 04:57:00.	
Вручную	Буссе	АМС-Башня-91356	РТПС-1 (+6) РТПС-2 (+7)	08.12.2019	4:57:00		Отключение электроэнергии по вине энергоснабжающей организации			Аварийное отключение электроэнергии. Работа от ИБП с 03:19:36 до 04:57:00.	
Вручную	Гуран	АМС-Опора-100292420	РТПС-1 (+6) РТПС-2 (+7)	08.12.2019	3:19:24		Отключение электроэнергии по вине энергоснабжающей организации с переходом на работу от ИБП			Аварийное отключение электроэнергии. Работа от ИБП с 03:19:24.	

Рисунок 1 – Местная сводка

Второй отчет формируется для РКН Хабаровск. Третий отчет – для РКН Амур. В эти отчеты вносят инциденты только с нарушениями вещания свыше 10 минут. Обе эти формы отличаются форматом: отчет в Хабаровск формиру-

ется в формате .xlsx, а отчет в РКН Амурв формате docx. Время инцидентов в обоих случаях остается московским. Пример отчета для РКН Амур и РКН Хабаровск представлены на рисунке 2 и рисунке 3, соответственно.

№ п/п	Предприятие	Населенный пункт	Оборудование (ТВК, МОЩНОСТЬ ПВД)	Программа	Причина перерыва	Дата начала перерыва	Время начала перерыва (мск)	Продолжительность	Дата окончания перерыва	Время окончания перерыва	Охват населения (тыс. чел.)	Дата доклада	Округ
1	Амурский ОРТПЦ	Иланский (Илановский р-н)	ПРД-TV-50D-92808 (22 ТВК, 0,05 кВт)	РTRC-1 «+6»	Отключение электроэнергии по вине энергоснабжающей организации	04.12.2019	0:03:28	0:10:30	04.12.2019	0:13:58	0,787	04.12.2019	ДФО
2	Амурский ОРТПЦ	Иланский (Илановский р-н)	ПРД-EL3 K050 UAV-175797 (28 ТВК, 0,05 кВт)	РTRC-2 «+7»	Отключение электроэнергии по вине энергоснабжающей организации	04.12.2019	0:03:28	0:10:30	04.12.2019	0:13:58	0,8	04.12.2019	ДФО
3	Амурский ОРТПЦ	Иланский (Илановский р-н)	ПРД-TV-50D-92808 (22 ТВК, 0,05 кВт)	РTRC-1 «+6»	Отключение электроэнергии по вине энергоснабжающей организации	04.12.2019	1:04:40	0:16:21	04.12.2019	1:21:01	0,787	04.12.2019	ДФО
4	Амурский ОРТПЦ	Иланский (Илановский р-н)	ПРД-EL3 K050 UAV-175797 (28 ТВК, 0,05 кВт)	РTRC-2 «+7»	Отключение электроэнергии по вине энергоснабжающей организации	04.12.2019	1:04:40	0:16:21	04.12.2019	1:21:01	0,8	04.12.2019	ДФО
5	Амурский ОРТПЦ	Магдагачи (Магдагачинский р-н)	ПРД-TV-600D-92757 (23 ТВК, 0,5 кВт)	РTRC-1 «+6»	Нарушения целостности структур ТП сигнала мультиплекса	04.12.2019	1:34:03	0:22:50	04.12.2019	1:56:53	10,07	04.12.2019	ДФО
6	Амурский ОРТПЦ	Магдагачи (Магдагачинский р-н)	ПРД-TV-600D-92757 (23 ТВК, 0,5 кВт)	РTRC-1 «+6»	Нарушения целостности структур ТП сигнала мультиплекса	04.12.2019	1:56:54	0:15:51	04.12.2019	2:12:45	10,07	04.12.2019	ДФО
7	Амурский ОРТПЦ	Магдагачи (Магдагачинский р-н)	ПРД-TV-600D-92757 (23 ТВК, 0,5 кВт)	РTRC-1 «+6»	Нарушения целостности структур ТП сигнала мультиплекса	04.12.2019	2:18:17	0:50:50	04.12.2019	3:09:07	10,07	04.12.2019	ДФО
8	Амурский ОРТПЦ	Магдагачи (Магдагачинский р-н)	ПРД-TV-600D-92757 (23 ТВК, 0,5 кВт)	РTRC-1 «+6»	Нарушения целостности структур ТП сигнала мультиплекса	04.12.2019	9:30:29	0:10:05	04.12.2019	9:40:34	10,07	04.12.2019	ДФО
9	Амурский ОРТПЦ	Магдагачи (Магдагачинский р-н)	ПРД-TV-600D-92757 (23 ТВК, 0,5 кВт)	РTRC-1 «+6»	Отключение электроэнергии по вине энергоснабжающей организации	04.12.2019	11:53:47	0:13:27	04.12.2019	12:07:14	10,07	04.12.2019	ДФО
10	Амурский ОРТПЦ	Магдагачи (Магдагачинский р-н)	ПРД-EL3 MPO UAV SD 164935 (23 ТВК, 0,5 кВт)	РTRC-2 «+7»	Отключение электроэнергии по вине энергоснабжающей организации	04.12.2019	11:53:47	0:13:27	04.12.2019	12:07:14	10,07	04.12.2019	ДФО

Рисунок 2 – Отчет для РКН Хабаровск

- Доклад
- об аварийной (нештатной) ситуации на сети связи общему пользованию, повлекшей прекращение оказания услуг связи и вещания

№ п/п	Дата, время (мск) начала перерыва в оказании услуг связи и вещания, кто, когда доложил	Место прекращения оказания услуг связи и вещания (федеральный округ, регион, населенный пункт)	Оператор связи, на сети которого произошел перерыв в оказании услуг связи и вещания	Последствия перерыва в оказании услуг связи и вещания	Население в зоне вещания или количество абонентов	Причина перерыва в оказании услуг связи и вещания	Принимаемые меры, предполагаемая дата возобновления оказания услуг связи и вещания	Дата, время (мск) окончания перерыва в оказании услуг связи и вещания	Общее время перерыва в оказании услуг связи и вещания
1а	2а	3а	4а	5а	6а	7а	8а	9а	10а
1а	03.12.2019-04:14:21 АСКиУ, РTRC	ДФО, Амурская область, Магдагачи	РTRC-1 «+6»	=	10,07	Отключение электроэнергии по вине энергоснабжающей организации	=	03.12.2019-05:20:07	1:05:46
2а	03.12.2019-04:14:21 АСКиУ, РTRC	ДФО, Амурская область, Магдагачи	РTRC-2 «+7»	=	10,08	Отключение электроэнергии по вине энергоснабжающей организации	=	03.12.2019-05:20:07	1:05:46
3а	03.12.2019-06:12:03 АСКиУ, РTRC	ДФО, Амурская область, Талдан	РTRC-1 «+6»	=	3,689	Отключение электроэнергии по вине энергоснабжающей организации	=	03.12.2019-08:12:42	2:00:39
4а	03.12.2019-06:12:03 АСКиУ, РTRC	ДФО, Амурская область, Талдан	РTRC-2 «+7»	=	3,7	Отключение электроэнергии по вине энергоснабжающей организации	=	03.12.2019-08:12:42	2:00:39
5а	03.12.2019-08:12:42 АСКиУ, РTRC	ДФО, Амурская область, Талдан	РTRC-1 «+6»	=	3,689	Неисправность передатчика	=	03.12.2019-08:23:40	0:10:58

Рисунок 3 – Отчет для РКН Амур

Каждый отчет формируется на основе выгрузки формата .xlsx из системы SD и преимущественно вручную. Сама система является государственным продуктом и не подразумевает подключение к ней постороннего программного обеспечения для выгрузки информации прямо из нее. Вместо этого система

SD позволяет выгружать из нее данные в форматах .xlsx и .xls. Пример выгрузки данных отображен на рисунке 4.

Источник	Филиал	Подраздел	Пункт	устан.	Тип	ТС	Наруш.	Приоритет	Время появл.	Причина	Время окончания	ТТ	Статус	ТРК/МУХ	Од	
Из FMI/PM	Амурский Цех	Благое Архара	Архари	Аварийно	ПРД-FUS	НР03 ULV DECT	Средний	26.11.2019	8:23:28	Работа передатчика	26.11.2019	8:23:35	TT000000; Выполнено	РTRC-2 (+7);	ПФ	
Из FMI/PM	Амурский Цех	Тында	Дипкун	Тында	Аварийно	ПРД-EL3	K050 UAV-1FECT	Низкий	26.11.2019	9:33:20	НЕИСПРАВНОСТЬ ПЕ	26.11.2019	9:33:26	TT000000; Выполнено	РTRC-2 (+7);	ПФ
Из FMI/PM	Амурский Цех	Свобод Кировский	Зей	Аварийно	ПРД-EL3	K050 UAV-1FECT	Низкий	26.11.2019	10:17:25	НЕИСПРАВНОСТЬ ПЕ	26.11.2019	10:17:31	TT000000; Выполнено	РTRC-2 (+7);	ПФ	
Из FMI/PM	Амурский Цех	Благое Норск	Селемд	Аварийно	ПРД-TV	50D-92822 (P1ECT	Низкий	26.11.2019	10:58:12	Нарушение целостнос	26.11.2019	10:58:25	TT000000; Выполнено	ВТРК-МУХ Амурский На	На	
Из FMI/PM	Амурский Цех	Благое Феверальск	Сег	Аварийно	ПРД-EL3	K050 UAV-1FECT	Низкий	26.11.2019	10:59:53	Нарушение целостнос	26.11.2019	11:00:07	TT000000; Выполнено	РTRC-2 (+7);	На	
Из FMI/PM	Амурский Цех	Благое Феверальск	Сег	Аварийно	ПРД-TV	50D-92853 (P1ECT	Низкий	26.11.2019	10:59:54	Нарушение целостнос	26.11.2019	11:00:07	TT000000; Выполнено	ВТРК-МУХ Амурский На	На	
Из FMI/PM	Амурский Цех	Благое Архара	Архари	Аварийно	ПРД-FUS	НР03 ULV DECT	Средний	26.11.2019	11:17:51	Работа передатчика	26.11.2019	11:18:01	TT000000; Выполнено	РTRC-2 (+7);	ПФ	
Из FMI/PM	Амурский Цех	Свобод Магдагачи	Маг	Аварийно	ПРД-EL3	MP02 UAV SECT	Низкий	26.11.2019	12:06:40	Нарушение целостнос	26.11.2019	12:06:46	TT000000; Выполнено	РTRC-2 (+7);	На	
Из FMI/PM	Амурский Цех	Свобод Магдагачи	Маг	Аварийно	ПРД-FUS	НР03 ULV DECT	Низкий	26.11.2019	12:10:22	Нарушение целостнос	26.11.2019	12:10:30	TT000000; Выполнено	РTRC-2 (+7);	На	
Из FMI/PM	Амурский Цех	Свобод Свободный	Ам	Аварийно	ПРД-FUS	НР03 ULV DECT	Средний	26.11.2019	13:36:17	Работа передатчика	26.11.2019	13:36:26	TT000000; Выполнено	РTRC-2 (+7);	ПФ	
Из FMI/PM	Амурский Цех	Свобод Магдагачи	Маг	Аварийно	ПРД-EL3	MP02 UAV SECT	Низкий	26.11.2019	13:41:12	Нарушение целостнос	26.11.2019	13:41:18	TT000000; Выполнено	РTRC-2 (+7);	На	
Из FMI/PM	Амурский Цех	Благое Норск	Селемд	Аварийно	ПРД-EL3	K050 UAV-1FECT	Низкий	26.11.2019	14:32:54	Нарушение целостнос	26.11.2019	14:33:03	TT000000; Выполнено	РTRC-2 (+7);	На	
Из FMI/PM	Амурский Цех	Благое Норск	Селемд	Аварийно	ПРД-TV	50D-92822 (P1ECT	Низкий	26.11.2019	14:32:54	Нарушение целостнос	26.11.2019	14:33:10	TT000000; Выполнено	ВТРК-МУХ Амурский На	На	
Из FMI/PM	Амурский Цех	Тында	Дипкун	Тында	Аварийно	ПРД-EL3	K050 UAV-1FECT	Низкий	26.11.2019	14:55:05	НЕИСПРАВНОСТЬ ПЕ	26.11.2019	14:55:13	TT000000; Выполнено	РTRC-2 (+7);	ПФ
Из FMI/PM	Амурский Цех	Благое Зеленый Бор	М	Аварийно	ПРД-EL3	MP01 UAV SECT	Средний	26.11.2019	15:59:54	Работа передатчика	26.11.2019	15:20:02	TT000000; Выполнено	РTRC-2 (+7);	ПФ	
Из FMI/PM	Амурский Цех	Благое Константиновск	Аварийно	ПРД-EL3	MP01 UAV SECT	Низкий	26.11.2019	15:51:03	Неисправность модуль	26.11.2019	15:54:05	TT000000; Выполнено	РTRC-2 (+7);	По		
Из FMI/PM	Амурский Цех	Благое Магдагачи	Маг	Аварийно	ПРД-EL3	MP02 UAV SECT	Низкий	26.11.2019	16:28:15	Нарушение целостнос	26.11.2019	16:28:22	TT000000; Выполнено	РTRC-2 (+7);	На	
Из FMI/PM	Амурский Цех	Благое Зеленый Бор	М	Аварийно	ПРД-EL3	MP01 UAV SECT	Средний	26.11.2019	17:07:00	Работа передатчика	26.11.2019	17:07:06	TT000000; Выполнено	РTRC-2 (+7);	ПФ	
Из FMI/PM	Амурский Цех	Свобод Свободный	Ам	Аварийно	ПРД-FUS	НР03 ULV DECT	Средний	26.11.2019	17:57:32	Работа передатчика	26.11.2019	17:57:42	TT000000; Выполнено	РTRC-2 (+7);	ПФ	
Из FMI/PM	Амурский Цех	Свобод Свободный	Ам	Аварийно	ПРД-FUS	НР03 ULV DECT	Средний	26.11.2019	18:12:25	Работа передатчика	26.11.2019	18:12:35	TT000000; Выполнено	РTRC-2 (+7);	ПФ	
Из FMI/PM	Амурский Цех	Свобод Кировский	Зей	Аварийно	ПРД-EL3	K050 UAV-1FECT	Низкий	26.11.2019	19:35:03	НЕИСПРАВНОСТЬ ПЕ	26.11.2019	19:35:09	TT000000; Выполнено	РTRC-2 (+7);	ПФ	
Из FMI/PM	Амурский Цех	Тында	Бам	Сховоро	Аварийно	ПРД-TV	400D-92615 (FECT	Низкий	26.11.2019	19:55:34	Нарушение целостнос	26.11.2019	19:55:49	TT000000; Выполнено	ВТРК-МУХ Амурский На	На
Из FMI/PM	Амурский Цех	Свобод Шимановск	Маг	Аварийно	ПРД-FUS	НР03 ULV DECT	Средний	26.11.2019	20:17:34	Нарушение целостнос	26.11.2019	20:17:42	TT000000; Выполнено	РTRC-2 (+7);	На	
Из FMI/PM	Амурский Цех	Свобод Магдагачи	Маг	Аварийно	ПРД-EL3	MP02 UAV SECT	Низкий	26.11.2019	21:54:35	Нарушение целостнос	26.11.2019	21:54:41	TT000000; Выполнено	РTRC-2 (+7);	На	
Из FMI/PM	Амурский Цех	Тында	Уруша	Сховоро	Аварийно	ПРД-UXATE-100-10034	Низкий	26.11.2019	23:11:04	Нарушение целостнос	26.11.2019	23:11:10	TT000000; Выполнено	РTRC-2 (+7);	На	
Из FMI/PM	Амурский Цех	Тында	Уруша	Сховоро	Аварийно	ПРД-UXATE-100-10034	Низкий	26.11.2019	23:11:11	Нарушение целостнос	26.11.2019	23:11:17	TT000000; Выполнено	РTRC-2 (+7);	На	
Из FMI/PM	Амурский Цех	Тында	Уруша	Сховоро	Аварийно	ПРД-Поляис	ТБИ-10	Низкий	26.11.2019	23:11:04	Нарушение целостнос	26.11.2019	23:11:11	TT000000; Выполнено	ВТРК-МУХ Амвский На	На

Рисунок 4 – Пример выгрузки из системы SD инцидентов

Как видно из рисунка 4 выгрузка содержит в себе помимо важных сведений, также избыточную информацию, которая в отчет не требуется. Помимо основного блока инцидентов, выгруженных из системы SD в местный отчет, вносятся данные о выездах и прибытии АПГ, с пояснением о ходе выполнения работ, а также информацию о работе сплайсеров, с пояснением ошибок в них.

Еще одним видом отчетности является недельная сводка. Формируется она в тот же временной интервал, что и суточная, в ночную смену перед началом недели. В этот отчет входят суточные отчеты за прошедшую неделю, и она имеет такой же внешний вид, что и местная сводка.

## 2 СПОСОБЫ ПРОГРАММНОЙ РЕАЛИЗАЦИИ АЛГОРИТМОВ ОБРАБОТКИ ТАБЛИЦ EXCEL

### 2.1 Формулировка задачи

Обобщим описанную ранее задачу. Входными данными является таблица Excel, далее необходимо обработать данные, сгруппировать и сформировать отчеты в формате Excel и Wordи схематически изображённая на рисунке 5.

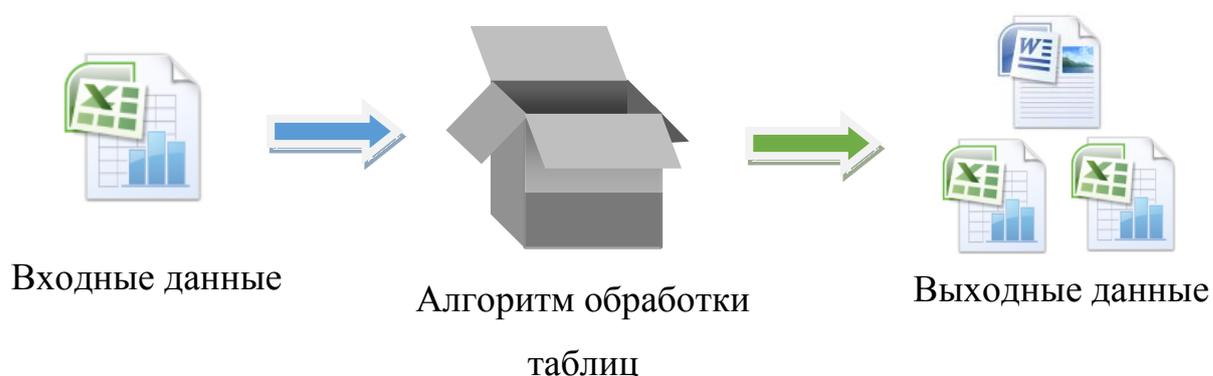


Рисунок 5 – Обобщенная схема задачи

Данную задачу можно решать несколькими разными подходами: использование макросов или использование OLEинтерфейса Excel.

Первый подход заключается в использовании интегрированной в приложения MicrosoftOffice среды программирования VBA (VisualBasicforApplications) для написания алгоритма обработки данных - макроса.

Второй подход связан с использованием технологии OLEдля получения содержания таблиц, обработки и формирования новых документов Excel, через внешнее стороннее приложение. Этот случай примечателен тем, что внешнее приложение, осуществляющее запросы к Excel, может быть реализовано практически в любой доступной средепрограммирования[5].

Далее рассмотрим подробнее каждый из описанных выше подходов, обсудим технологические моменты и возможные трудности реализации.

## 2.2 COMиOLE

COM (англ. ComponentObjectModel – модель компонентного объекта) – это технологический стандарт от компании Microsoft, предназначенный для создания программного обеспечения на основе взаимодействующих компонентов объекта, каждый из которых может использоваться во многих программах одновременно. Стандарт воплощает в себе идеи полиморфизма и инкапсуляции объектно-ориентированного программирования[5].

В COM любая часть программного обеспечения реализует свои сервисы как один или несколько объектов COM. Каждый такой объект поддерживает один или несколько интерфейсов, состоящих из методов [5].

Клиенты могут получить доступ к сервисам объекта COM только через вызовы методов интерфейсов объекта — у них нет непосредственного доступа к данным объекта [5].

На основе технологии COM были разработана технология связывания и внедрения объектов в другие документы и объекты – OLE (англ. ObjectLinkingandEmbedding)[5].

OLE позволяет передавать часть работы от одной программы редактирования к другой и возвращать результаты назад.

Все основные программные продукты, входящие в состав MicrosoftOffice, имеют собственный OLE-сервер предоставляющий достаточно полным набором методов для чтения, редактирования и создания документов, поддерживаемых продуктами MicrosoftOffice[5].

Такой подход позволяет клиент-приложению быстро получить доступ к данным, содержащимся в нужных нам файлах таблиц, а после сформировать новые документы и поместить туда необходимые нам данные. Как отмечалось ранее, само клиентское приложение может быть создано в любой программной среде.

## **2.2 VBA (Visual Basic for Applications)**

Visual Basic for Applications (Visual Basic для приложений) — упрощённая реализация языка программирования Visual Basic, встроенная в линейку продуктов Microsoft Office [5].

VBA является интерпретируемым языком. VBA близок к Visual Basic, будучи языком, построенным на COM, позволяет использовать все доступные в операционной системе COM объекты и компоненты ActiveX.

На языке VBA реализуются макросы в продуктах Microsoft Office. VBA предоставляет практически неограниченный доступ к возможностям того или иного продукта [6].

Использование макросов удобно в первую очередь тем, что программный код всегда находится в непосредственном доступе, а исполнительная и программная среда являются едиными. Это обстоятельство позволяет быстро отлаживать и корректировать программный код [6].

Встроенные инструменты позволяют записывать макрос на основе последовательности действий пользователя, что позволяет быстро сформировать код будущей программы, а инструменты отладки позволяют обнаружить ошибки и пошагово проследить ход исполнения программы. К сожалению, данной возможности лишен подход с использованием стороннего клиент-приложения [6, 7].

Данные обстоятельства совместно с простотой самого языка программирования VBA делают данный подход наиболее предпочтительным и гибким для решения задач подобного рода.

## **2.3 Среда и язык программирования**

Для данной работы был выбран табличный процессор на сегодняшний день Microsoft Excel, входящий в пакет программ Microsoft Office.

Каждый продукт из пакета Microsoft Office обладает интегрированной средой программирования Visual Basic for Applications (VBA). Наличие интегрированной среды разработки дает поистине неограниченные возможности для расширения возможностей родительского приложения [8].

Основным отличие VBA от своего довольно известного прародителя VisualBasic состоит в том, что VBA является интерпретируемым языком - это значит, что программы, написанные на нем, выполняются только в среде поддерживающего его приложения[2]. Данное обстоятельство является большим достоинством данного языка: программы хранятся в виде исходного кода, и не подвергается изменениям со стороны исполняющего приложения при их запуске. Открытость и доступность исходного кода дает возможность вносить оперативные изменения и доработки в программу, а также осуществлять отладку, при возникновении ошибок, на любом устройстве, где она будет запущена[9, 2].

Функциональные возможности VBA ничем не уступают его старшему собрату. Он позволяет создавать не только простые сценарии для автоматизации функций офисного приложения, но и разрабатывать полноценные программы со сложными оконными интерфейсами; подключать внешние динамические библиотеки (DLL), написанные на любом другом языке программирования; получать доступ к жесткому диску, локальной сети и интернету; общаться с другими приложениями по средством COM-технологии и многое другое[9, 2].

#### **2.4 Естественно-языковая ориентированность**

Интересной особенностью VBA в MSOffice русской локализации является возможность использовать кириллицу в названиях функций, имен переменных и констант[2].

Пользуясь такой возможностью, было решено использовать при разработке программного модуля только русские названия и имена переменных, что должно помочь сделать продукт более понятным для конечного пользователя. Тем не менее, данное утверждение является достаточно спорным. Многие профессиональные программисты и педагогические работники в сфере информационных технологий придерживаются различных диаметрально противоположных взглядов[10].

Одни считают, что использование родного языка для описания программных операций вредит при изучении языков программирования, особенно в школьный период. Использование малоизвестного языка синтаксических кон-

струкций языка программирования при обучении программированию исключает отождествление программных команд с обыденным смыслом слов и предотвращает желание оперировать смылосодержащими формами, интуитивно заменять строгие команды на синонимичные и созвучные слова или варьировать словоформы, как это происходит в естественном языке в процессе повседневного общения[10]. Использование незнакомых слов сродни вводу специальной профессиональной терминологии, которая имеет узкое применение и исключает двусмысленность[11].

Другое мнение связано с положительным оцениванием перспектив использования родного языка для программирования. Применение родного языка позволяет добиться понятности команд и, как следствие, повысить читаемость кода[11, 12]. Этот эффект особенно сильно должен проявить себя в крупных проектах с большим количеством различных используемых классов, функций, свойств и переменных, что приводит к невозможности запомнить все и делает необходимым давать развернутые имена, которые позволят быстро и однозначно понять, что они обозначают[11]. Чтобы давать грамотно полные имена на иностранном языке, необходимо хорошее знание его, а использование транслитерации является весьма сомнительной альтернативой [13].

VBA позволяет использовать гибридный подход как компромисс между двумя выше изложенными точками зрения: использования родных англоязычных синтаксических конструкций и имен операторов с русскоязычными именами пользовательских функций и переменных.

Используемые подходы в данной работе носят экспериментальный характер и не ставят цели навязать какую-либо идеологическую концепцию.

## **2.5 Пользовательский интерфейс макросов VBA**

Тем не менее, при разработке программ в интегрированной среде программирования VBA (VisualBasicforApplication) вскоре на первый план выходят проблемы связанные с задачей создания пользовательского интерфейса для облегчения взаимодействия пользователя с программой. Для этих целей в среде VBA имеется встроенный инструмент для создания пользовательских форм,

представляющий собой полноценный *конструктор графического пользовательского интерфейса* (GUI-редактор) и позволяющий создавать диалоговые окна на основе технологии WinForms[14]. Другим важным аспектом является возможность полноценно использовать COM технологию в VBA программах, а, следовательно, и все имеющиеся на машине ActiveX компоненты [9], что позволяет расширять возможности, как самих программ, так и пользовательских интерфейсов. Однако, достоинство одновременно являются и недостатком: использование сторонних компонентов ограничивает переносимость программы на другие устройства в виду необходимости установки и регистрации этих компонентов в операционной системе, что не всегда возможно в случае ограниченного доступа пользователя на целевом устройстве.

Альтернативой может быть использование сторонних фреймворков, позволяющих быстро и эффективно создавать сложные интерфейсы. Существует множество готовых решений, в том числе кроссплатформенных [15, 16], но в силу особенностей среды VBA использование данных библиотек либо крайне затруднительно, либо невозможно в принципе [9].

При анализе предметной области можно достаточно без проблем сформулировать следующие требования к продукту, предъявляемые разработчиком:

1) *Быстрый старт*. Программный продукт должен быстро подключаться к среде программирования VBA и иметь минимум обязательных преднастроек.

2) *Простота использования*. Минимум базовых команд формирующих основной функционал продукта.

3) *Гибкость*. Возможность необязательной подстроки, изменения формы, цвета и иных "косметических" и функциональных особенностей интерфейса.

Разработчик, используя фреймворк, создает интерфейс, целевым конечным пользователем которого являются люди без профессиональных знаний в области программирования и информационных систем, что порождает свои специфические требования к разрабатываемым интерфейсам:

1) *Интуитивная понятность*. Пользователю не требуются специальные навыки для работы с интерфейсом.

2) *Интерактивная справка*. Пользователь должен иметь возможность получить исчерпывающую информацию о том, что от него требуется в данный момент, без необходимости изучения инструкции.

3) *"Наставничество"*. Пользователь должен своевременно быть уведомлен о совершении ошибки и получить информацию о том, что именно было сделано неверно.

Стоит отметить, что такие требования продиктованы спецификой использования интерфейсов в рамках макросов. В подавляющем большинстве случаев интерфейс используется для ввода входных параметров для исполнения программы макроса и в меньшей степени для отображения полученной информации. Пользователь использует макрос с четким пониманием, какой результат он хочет получить, поэтому от разработчика требуется с помощью доступных средств ясно и однозначно показать, какую именно информацию пользователю нужно передать программе посредством графического интерфейса.

Удовлетворение потребностей конечного пользователя ложится на плечи разработчика, поэтому для удовлетворения требований разработчика необходимо обеспечить его как инструментами, так и готовыми решениями, позволяющими при минимальных затратах сил и времени наиболее полно обеспечить пользователя всем необходимым.

## **2.6 Причины выбора Qt как базового инструмента для создания пользовательского интерфейса**

Как уже говорилось ранее, разработка макроса может требовать создания многочисленных, но при этом простых пользовательских форм, а встроенное решение дает довольно громоздкую кодовую реализацию интерфейса, что затягивает время разработки программного продукта. Чтобы решить эту проблему можно привлечь существующие технологии с открытым исходным кодом и довольно гибким инструментарием для создания графического интерфейса пользователя [16].

Быстрое создание графических форм пользовательского интерфейса должно сочетать в себе возможность при минимальном объеме команд созда-

вать полноценный и гибкий интерфейс. Это может быть достигнуто возможностью внутрикодowego создания форм интерфейса, что позволяет «на лету» генерировать и модифицировать элементы интерфейса. Наличие возможности автоматической масштабирования и компоновки элементов пользователя на форме, исключая необходимость их точного позиционирования на форме и значительно ускоряя процесс создания интерфейса [16].

Чтобы решить подобную задачу можно использовать кроссплатформенную библиотеку с открытым кодом Qt. Главный плюс такого подхода заключается в том, что эта библиотека содержит довольно гибкий фреймворк для создания графических интерфейсов пользователя любой сложности и при этом является свободно распространяемым программным продуктом [16].

Базовые функции Qt позволяют формировать пользовательские формы с автопозиционированием элементов интерфейса, такие как вертикальные и горизонтальные списки. Элементы, помещенные в такие списки, автоматически выравниваются в соответствии со своим размером и размерами соседних элементов, в нефиксированных направлениях элементы могут свободно растягиваться. Формы, построенные на таких автопозиционных структурах, полностью и свободно масштабируемы. В терминологии Qt невидимый элемент, который хранит в себе другие элементы интерфейса и определяет их позиционирование и выравнивание, называется слоем. Такой подход позволяет не задумываться о точном позиционировании элементов, притом, конечный результат выглядит вполне приемлемым и строгим в графическом оформлении [16].

## **2.7 Параллельные вычисления в среде VBA Excel**

Для сложных вычислительных задач повсеместно используется технология параллельных вычислений. Однако, дело в том, что среда VBA в своей реализации не поддерживает возможность создания параллельных потоков для обработки данных [2]

Известно несколько искусственных приемов реализации одновременного запуска VBA программ во множестве потоков [9]. Все их можно разделить на 2

группы: реализованные только средствами среды VBA и с использованием сторонних средств.

В рамках магистерской диссертации было реализовано 2 метода, осуществляющие параллельные вычисления. Первый способ использует только стандартными средствами VBA. Существует возможность создания внутри программы дочернего процесса Excel и удаленного запуска скрипта в созданном процессе, что позволяет из одной программы запустить несколько копий полностью автономных программ. Преимуществом данного метода является сравнительная простота и полная автономность программы. Однако, существенный недостаток в большой ресурсоемкости и, в частности, высокие требования к объему оперативной памяти: каждая параллельная программа работает как самостоятельная копия Excel.

Второй способ основан на использовании внешней динамической библиотеки. Нужные данные сначала считываются VBA программой, после чего передаются в распоряжение внешней библиотеки, где уже реализуется многопоточная обработка, и возвращаются обратно в управляющую программу. Таким образом, реализация параллельных вычислений дается на откуп внешней библиотеке, которая может быть реализована в любой среде программирования. Для данной работы была создана динамическая библиотека (DLL) в среде BorlandC++ Builder 6. Основным преимуществом является относительно большая производительность и гораздо меньшее требование к оперативной памяти, так как все потоки будут созданы внутри одного родительского процесса. К недостаткам можно отнести необходимость использования дополнительной среды программирования, что влечет, своего рода, проблемы "языкового барьера" - неэквивалентность многих типов данных VBA и других языков, согласование которых зачастую являются нетривиальной задачей.

В таблице 1 собраны данные измерений (А) однопоточного исполнения программы и описанных выше методов многопоточной обработки: (В) только средствами VBA и (С) использование внешней библиотеки. Контрольным измерением была обработка таблицы из 2000 записей, для имитации сложных вы-

числений на каждой записи делалась задержка в  $10^7$  пустых циклов (данный способ выбран в виду наиболее эквивалентного повторения в разных программных средах). Тестирование проводилось на машине с центральным процессором IntelCorei7-3630QM 2.40 ГГц. Данный процессор имеет 4 физических ядра, каждое из которых работает в 2 потока - операционной системе доступно 8 параллельных потоков.

Таблица 1 – Результаты тестирования

<i>Кол-во потоков, шт</i>	<i>1</i>			<i>2</i>		<i>3</i>	
	<i>A</i>	<i>B</i>	<i>C</i>	<i>B</i>	<i>C</i>	<i>B</i>	<i>C</i>
Создание потоков, с		0,45	0,01	1,08	0,01	1,65	0,01
Обработка, с	143,42	58,47	22,41	33,68	11,36	23,39	7,53
Всего, с	147,27	62,68	25,88	38,67	14,94	28,53	11,12

<i>Кол-во потоков, шт</i>	<i>4</i>		<i>5</i>		<i>6</i>	
	<i>B</i>	<i>C</i>	<i>B</i>	<i>C</i>	<i>B</i>	<i>C</i>
Создание потоков, с	2,16	0,01	2,88	0,01	3,57	0,01
Обработка, с	19,64	5,68	18,01	4,58	18,29	3,87
Всего, с	25,29	9,19	25,59	8,08	25,43	7,39

В таблицу не были включены операции чтения и записи в файл, они примерно равны для всех программ при любом количестве потоков и составили в среднем 1,7 с для чтения и 1,9 с для записи.

На рисунке бсхематично представлена временная и условная диаграммы выполнения алгоритма для случая 4 потоков. Для удобства сравнения результатов при различном количестве потоков принимали условную временную диаграмму, где за создание потоков принималось время от начала создания первого потока до завершения создания последнего, а за время исполнение потоков – ожидание завершения всех потоков после создания последнего.

Интересным является момент, что вынесение обработки в отдельный процесс и производство вычислений в фоновом режиме даже при одном потоке сократило время обработки более чем в два раза по сравнению с обычным выполнением VBA программы в активном окне.

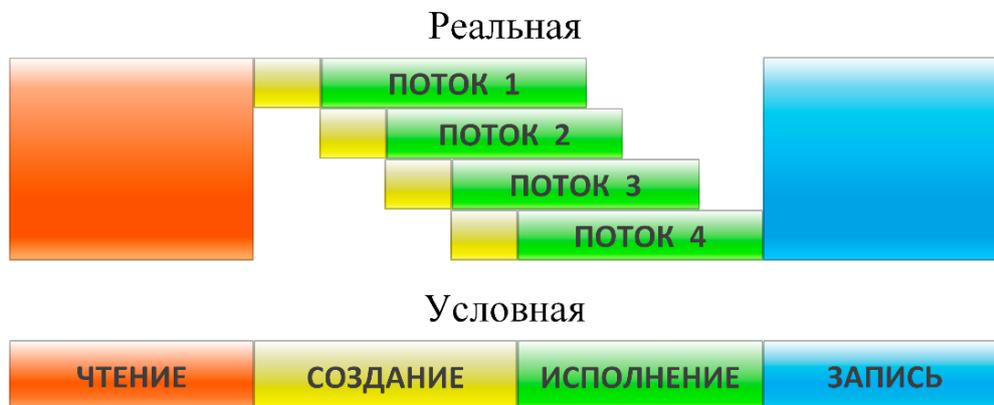
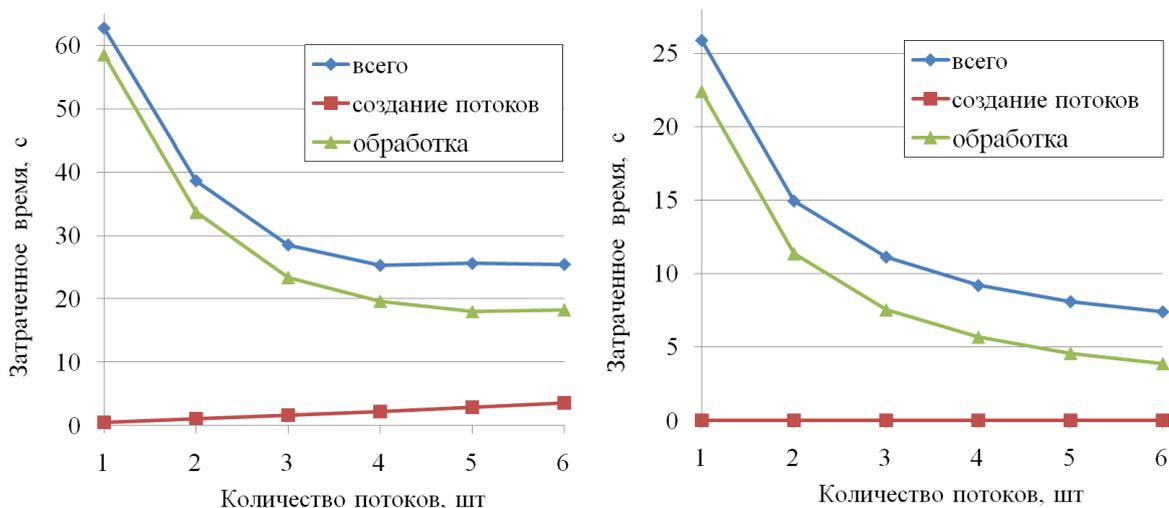


Рисунок 6 – Временная и условная диаграммы выполнения алгоритма для случая 4 потоков



слева – распараллеливание вычислений без сторонних средств;

справа – с помощью внешней динамической библиотеки;

Рисунок 7 – Графики зависимости затраченного машинного времени от количества созданных потоков

На рисунке 7 приведены графики иллюстрирующие данные из таблицы. Можно заметить, для случая (В), время создания процессов увеличивается с ро-

стом количества потоков. Это объясняется тем, что данные передаются в каждую копию программы отдельно, в то время как в случае (С) внешней библиотеки, данные передаются все разом, а создаваемые потоки имеют доступ ко всем данным одновременно.

При всех достоинствах данных методов, нужно оценивать целесообразность применения многопоточной обработки. Согласно закону Амдала, параллельные вычисления тем эффективнее, чем выше доля данных, которые могут быть обработаны независимо.

## **2.8 Основные требования к программному продукту**

Подведем краткий итог предъявляемых требований к разрабатываемому программному продукту.

Программный модуль должен отвечать следующим принципам:

1) автономность и универсальность: использование структуры программы в виде файла-библиотеки;

2) гибкость и простота: наименьший возможный набор функций доступных для пользователя, обеспечивающий решение поставленных задач;

3) доступность и понятность: использование русских полных имен для обозначения предоставляемых функций однозначно отражающих их суть и назначение;

4) прозрачная интегрируемость: возможность использовать как можно больший перечень инструментов родительского приложения - необходимо расширять возможности, а не ограничивать их.

5) программный модуль должен предоставлять быстрый доступ к следующим инструментам:

- редактирование таблиц с поддержкой табличных формул,
- сортировка и фильтрация таблиц,
- регулярные выражения,
- экспресс-блоки,
- графические объекты,
- подстановка табличных данных в шаблоны Word-файлов с сохранени-

ем форматирования,

- сохранение результата в широком перечне форматов (в частности .doc, .docx, .pdf).

## 3 РАЗРАБОТКА ПРОГРАММНОГО МОДУЛЯ ДЛЯ ОБРАБОТКИ ТАБЛИЦ И ФОРМИРОВАНИЯ ОТЧЕТОВ

### 3.1 Структура и логика работы программного модуля

Структура программы продиктована спецификой программной среды. В VBA определение каждого класса должно быть описано в отдельном модуле (*ClassModule*). Для удовлетворения требования быстрой переносимости программного модуля в другие файлы с табличными данными, целесообразно делать все в одном единственном класс-модуле, в таком случае весь код может быть экспортирован в едином файле [2].

Таким образом, разработанный программный модуль представляет собой один большой класс, включающий в себя все необходимые инструменты. Чтобы получить доступ ко всем его функциям и свойствам, необходимо сначала создать копию экземпляра данного класса. В состав класса входят конфигурационные свойства, позволяющие задавать настройки, которые будут запоминаться внутри созданного экземпляра.

Возьмем за правило, что работа программы не должна изменять исходный файл с данными, для обработки данных будем использовать отдельную копию приложения Excel, которую назовем виртуальным приложением, а открытую в нем таблицу - виртуальной таблицей. На самом деле никакой разницы между открытой таблицей с исходными данными и виртуальной нет, это лишь условность, принятая для четкого разграничения таблицы изначальной и таблицы с промежуточными вычислениями, перед формированием отчета.

Использование промежуточной виртуальной таблицы позволяет аккумулировать данные из множества разных таблиц в одну, после чего обрабатывать все данные сразу и формировать на их основе отчет. Для исключения путаницы подстановка непосредственно из исходной таблицы в шаблон не допускается.

Таким образом, процесс создания отчета сводится к трем шагам:

- 1) копирование из исходной таблицы необходимых данных в виртуаль-

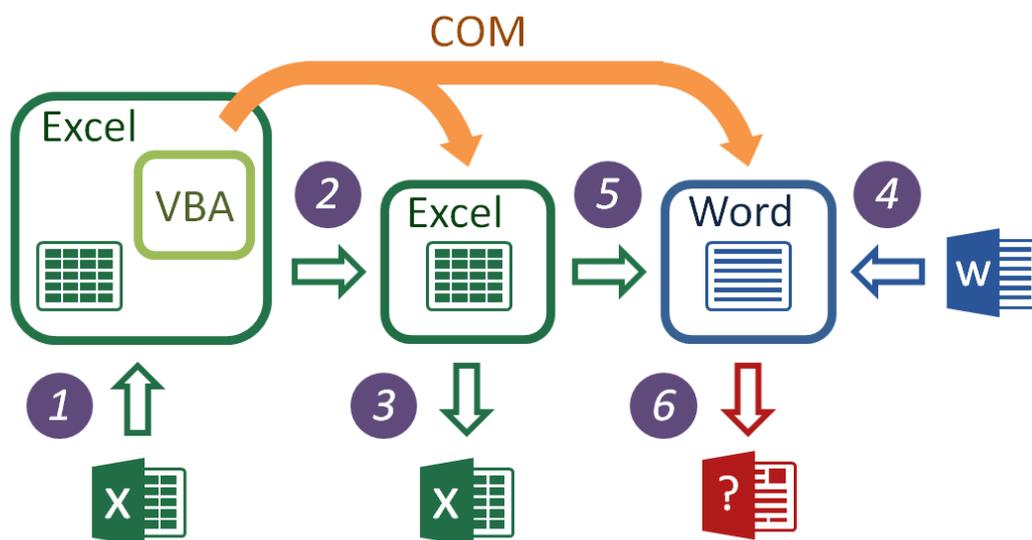
ную таблицу;

2) редактирование, сортировка, фильтрация и прочие операции обработки данных в виртуальной таблице;

3) подстановка данных из виртуальной таблицы в шаблон отчета с последующим сохранением отчета.

Разработанный программный модуль предоставляет лишь набор инструментов, пользователь сам решает, что с ними делать. Формирование отчета не является самоцелью, предлагаемый набор может быть использован для автоматизации быстрой обработки таблиц, склейки множества таблиц в одну и для решения множества других задач.

На рисунке 8 представлена диаграмма с возможными этапами сценария, созданного с помощью разработанного программного модуля. Управление процессом осуществляется VBA программой, посредством технологии COM программа может открывать и взаимодействовать с копиями других приложений MSOffice.



1 – загрузка таблицы с исходными данными; 2 – копирование данных в виртуальную таблицу с последующей обработкой; 3 – сохранение виртуальной таблицы; 4 – загрузка шаблона; 5 – подстановка данных в отчет; 6 – сохранение отчета в необходимом расширении

Рисунок 8 – Диаграмма возможных этапов сценария

### 3.2 Функции общего назначения и константы

Часто используемые стандартные типы функций в разработанном программном модуле получили функции обертки с русскоязычными именами. Отметим, что при разработке программного модуля была принята условность записывать константы и перечисляемые типы курсивом, а также функции-обертки, возвращающие постоянные значения.

Рассмотрим самые востребованные из них.

**Function**Сообщение( \_

**Optional** Текст*AsString*, \_

**Optional** Заголовок*AsString* = "", \_

**Optional** ВидИконки*AsИКОНКА*, \_

**Optional** ВидКнопок*AsКНОПКИ*, \_

**Optional** НажатаяКнопка*AsОТВЕТ\_СООБЩЕНИЯ* = -1) \_

*AsInteger*

Выводит на экран окно с сообщением, является оберткой стандартной функции **MsgBox**.

Принимает первым аргументом текст сообщения, а вторым заголовок окна. Аргумент *ВидИконки* задает вид сообщения и отображаемой иконки, принимает фиксированное значение из списка: *НЕТ\_ИКОНКИ*, *ОШИБКА*, *ВОПРОС*, *ПРЕДУПРЕЖДЕНИЕ*, *ИНФОРМАЦИОННАЯ*. Аргумент *ВидКнопок* задает количество и название кнопок, также является перечисляемым типом данных, принимая следующие значения из списка: *ОК*, *ОК\_ОТМЕНА*, *ПРЕРВАТЬ\_ПОВТОР\_ПРОПУСТИТЬ*, *ДА\_НЕТ\_ОТМЕНА*, *ДА\_НЕТ*, *ПОВТОР\_ОТМЕНА*.

Последний аргумент *НажатаяКнопка* сделан для удобства сравнения кода нажатой кнопки с заданным значением аргумента, если оно совпадает, функция вернет значение 1, соответствующее логическому *True*; если не совпадает, будет возвращено значение 0, соответствующее логическому *False*. Это позволяет более удобно использовать функцию в логических конструкциях. В

случае, если аргумент не задан, будет возвращен код нажатой кнопки. Аргумент принимает фиксированное именованное значение из списка: ОТВЕТ\_ОК, ОТВЕТ\_ОТМЕНА, ОТВЕТ\_ПРЕРВАТЬ, ОТВЕТ\_ПОВТОР, ОТВЕТ\_ПРОПУСТИТЬ, ОТВЕТ\_ДА, ОТВЕТ\_НЕТ.

**Function**Длина(Массив*AsVariant*, \_

**Optional**Размерность*AsInteger* = 1) *AsInteger*

Функция возвращает длину массива, принятым в первом аргументе, в размерности заданным вторым аргументом. Если тип данных первого аргумента не является массивом, тогда он преобразуется в тип строки и возвращается длина этой строки. Таким образом, функция является общей для определения длин массивов и строк.

**Function**КопироватьФайл(ПутьОткуда*AsString*, \_

ПутьКуда*AsString*) *AsBoolean*

Оболочка функции **FileCopy**. Копирует файл по пути в первом аргументе, в место указанное во втором аргументе. При удачном копировании возвращает *True*, при неудачном - *False*. Автоматически проверяет существование копируемого файла, при отсутствии файла выводится текстовое сообщение и возвращается значение *False*.

**Function**ПутьКЭтойКниге() *AsString*

Возвращает путь к файлу, из которого запущена программа.

**Function**ФайлСуществует(ПутьКФайлу*AsString*) *AsBoolean*

Возвращает *True*, если файл по указанному пути существует, в противном случае возвращает значение *False*. Также может применяться для проверки существования папок.

**Function**Дата(**Optional**Формат*AsString* = "dd.mm.yy") *AsString*

**Function**Время(**Optional**Формат*AsString* = "h:m") *AsString*

Данные функции являются оболочкой одной и той же стандартной функции **Now**, которая возвращает текущую дату и время. Однако функции сделаны таким образом, чтобы при использовании без аргумента сразу возвращать текущую дату в общепринятом формате *день.месяц.год*и время в формате *ча-*

сы:минуты. Функции также принимают пользовательский шаблон форматирования как свой единственный аргумент.

**Function**УбратьПробелы(Сторока**AsString**) **AsString**

Оболочка функции **Trim**. Убирает крайние пробелы в строке аргумента и возвращает результат.

В модуле также были реализованы функции-обертки для русификации именных значений булевого типа данных:

**Function**ДА() **AsBoolean**

Возвращает значение *True*.

**Function**НЕТ() **AsBoolean**

Возвращает значение *False*.

**Function**ПУСТАЯ\_СТРОКА() **AsString**

Возвращает пустую строку.

### 3.3 Работа с таблицами

Для перемещения и копирования табличных данных предусмотрен широкий перечень команд.

**Sub**КопироватьАктивныйЛист()

Копирует всю информацию с исходной таблицы в виртуальную.

**Sub**КопироватьСтолбец(Столбец **AsVariant**, \_

**Optional**НовыйСтолбец**AsVariant**)

Копирует указанный столбец из исходной таблицы в другой указанный столбец в виртуальной таблице. Столбец может быть указан как порядковым номером, так и буквенным эквивалентом. Если второй аргумент не указан, то для получения адреса назначения столбца будет вызвана функция **СледующийСтолбец**(см. ниже).

**Sub**КопироватьОбласть(Область **AsString**, \_

НоваяОбласть**AsString**)

Копирует указанную область из исходной таблицы в указанную область в виртуальной таблице. Области должны быть заданы числобуквенными кодами с разделителем в виде двоеточия. Области должны быть равны по количеству

строк и столбцов.

**PropertyLet**ФиксСтроки(Количество*AsInteger*)

Свойство задает количество строк от начала таблицы, которые будут игнорироваться при копировании функциями **КопироватьАктивныйЛист** и **КопироватьСтолбец**. Используется для исключения из таблицы заголовочных строк.

**Sub**ОткрытьТаблицу(ПутьКФайлу*AsString*)

**Sub**ВыбратьКнигу(НазваниеКниги*AsString*)

**Sub**ВыбратьЛист(НазваниеЛиста*AsString*)

Процедуры позволяющие: открыть новый файл с таблицей; выбрать книгу и назначить как активную, если открыто больше одного файла с таблицами; выбрать лист в активной книге и назначить как активный лист, из которого будут проводиться операции копирования в виртуальную таблицу.

**Sub**ЗакрытьТаблицу()

Закрывает приложение с открытой таблицей, возвращая в активное положение приложение, из которого исполняется данная команда. Если ранее не была открыта таблица функцией **ОткрытьТаблицу**, то результатом будет закрытие основного приложения, где вызвана команда.

**Function**ТекущийСтолбец() *AsInteger*

Возвращает значение счетчика текущего столбца.

**Function**СледующийСтолбец() *AsInteger*

Возвращает значение счетчика текущего столбца увеличенного на единицу.

**PropertyLet**УстановитьТекущийСтолбец(Столбец*AsInteger*)

Позволяет присвоить счетчику текущего столбца произвольное значение.

Последние три команды используются для удобного последовательного заполнения столбцов виртуальной таблицы.

**Sub**СоздатьВиртуальнуюТаблицу()

Создает скрытую копию приложения Excel с пустой таблицей, которую называем "*виртуальной*".

**Sub**ЗакрыватьВиртуальнуюТаблицу()

Закрывает копию приложения Excel с "виртуальной" таблицей, запущенную функцией **СоздатьВиртуальнуюТаблицу**.

**Sub**ПоказатьВиртуальнуюТаблицу()

Делает копию приложения Excel с "виртуальной" таблицей, запущенную функцией **СоздатьВиртуальнуюТаблицу** видимой для пользователя, таким образом, позволяя пользователю получить доступ к "виртуальной" таблице.

**Sub**ДобавитьСтолбцы(НомерСтолбца*AsVariant*, \_

**Optional**Количество*AsInteger* = 1, \_

**Optional**Формат*AsString* = "General")

**Sub** ДобавитьСтроки(НомерСтроки*AsInteger*, \_

**Optional**Количество*AsInteger* = 1, \_

**Optional**Формат*AsString* = "General")

Вставляют указанное количество строк или столбцов по указанному адресу. При отсутствии аргумента, указывающего количество вставляемых элементов, будет вставлен один. Последний аргумент позволяет задать формат отображения в ячейках, вставленных строк или столбцов.

**Sub**УдалитьСтолбец(НомерСтолбца*AsVariant*)

**Sub**УдалитьСтроку(НомерСтроки*AsInteger*)

Удаляют указанный столбец или строку в виртуальной таблице.

**Function**АктивнаяКнига() *AsExcel.Workbook*

**Function**АктивныйЛист() *AsExcel.Worksheet*

Возвращают ссылку на объект соответствующий активной книге или активному листу.

**Function**ВсяТаблица() *AsRange*

Возвращает ссылку на рабочую область виртуальной таблицы.

**Function**Столбец(КодСтолбца*AsVariant*) *AsRange*

Возвращает ссылку на область ячеек указанного столбца виртуальной таблицы.

**Function** Ячейки(КодЯчеек *AsString*) *AsRange*

Возвращает ссылку на произвольную область ячеек виртуальной таблицы.

**Property Let** ЗначениеВЯчейке(Ячейка *AsString*, Значение *AsVariant*)

Присваивает ячейки виртуальной таблицы указанное значение.

**Property Get** ЗначениеВЯчейке(Ячейка *AsString*)

Возвращает значение в указанной ячейке виртуальной таблицы.

**Property Let** ФормулаВЯчейке(Столбец *As Variant*, \_

Строка *As Integer*, \_

Формула *As String*)

**Property Let** ФормулаВОбласти(Область *As Variant*, Формула *As String*)

**Property Let** ФормулаВСтолбце(Столбец *As Variant*, Формула *As String*)

Данная группа методов позволяет задавать формульные выражения в ячейке, столбце и произвольной области ячеек виртуальной таблицы.

**Function** КоличествоСтрок(**Optional** Столбец *AsVariant* = 1) *AsInteger*

Возвращает количество подряд идущих от начала таблицы не пустых и не состоящих из одних пробелов строк в указанном столбце. Если столбец не указан, строки считаются в первом столбце таблицы.

### 3.4 Регулярные выражения

Инструменты для работы с регулярными выражениями представлены следующим перечнем команд.

**Property Let** РегВырИскатьВсеСовпадения(Флаг *AsBoolean*)

Свойство, задающее опцию поиска всех совпадений, удовлетворяющих шаблону регулярного выражения. В случае присвоения значения *False*, будет возвращено только первое совпадение.

**Property Let** РегВырНеУчитыватьРегистр(Флаг *AsBoolean*)

Свойство, задающее опцию учета регистра, при поиске совпадений, удовлетворяющих шаблону регулярного выражения.

**Sub** РегВырСЗаменойОбласти( Шаблон *AsString*, \_

СтрокаЗамены *AsString*, \_

ОбластьПоиска*AsString*)

**Sub**РегВырСЗаменойСтолбца( Шаблон*AsString*, \_  
СтрокаЗамены*AsString*, \_  
Столбец*AsVariant*)

Заменяют строкой замены в каждой ячейке указанной области или столбца найденные совпадения, соответствующие шаблону регулярного выражения, в виртуальной таблице.

**Sub**РегВырНовыйСтолбец( Шаблон *AsString*, \_  
Столбец*AsVariant*, \_  
НовыйСтолбец*AsVariant*, \_

**Optional** НомерСовпадения*AsInteger* = 0)

Заменяет строкой замены в каждой ячейке указанного столбца исходной таблицы найденные совпадения, соответствующие шаблону регулярного выражения, и вставляет их в указанный столбец виртуальной таблицы.

Заметим, что данные команды используют подключаемый модуль, который необходим активировать перед их использованием, для этого нужно пройти Tools→References и в появившемся окне отметить MicrosoftVBScriptRegularExpressions актуальной версии (рисунок 9).

### 3.5 Сортировка и фильтрация

**Sub**СортироватьТаблицу( \_  
**Optional**ДиапазонСортировки*AsRange*, \_  
**Optional**КлючевойДиапазон1*AsRange*, \_  
**Optional**ПорядокСортировки1*AsСОПТИРОВКА* = ПО\_АЛФАВИТУ, \_  
**Optional**КлючевойДиапазон2*AsRange*, \_  
**Optional**ПорядокСортировки2*AsСОПТИРОВКА* = ПО\_АЛФАВИТУ, \_  
**Optional**КлючевойДиапазон3*AsRange*, \_  
**Optional**ПорядокСортировки3*AsСОПТИРОВКА* = ПО\_АЛФАВИТУ)

Сортирует виртуальную таблицу в указанном диапазоне ячеек по ключевым поддиапазнам, которых может быть до трех штук. Сортировка происходит в иерархической последовательности в каждом ключевом диапазоне, т.е.

второй ключевой диапазон влияет на сортировку совпавших значений в первом ключевом диапазоне. Для каждого ключевого диапазона может быть определен порядок сортировки принимающий следующие фиксированные именные значения: ПО\_ВОЗРАСТАНИЮ, ПО\_УБЫВАНИЮ, ПО\_АЛФАВИТУ, ПО\_АЛФАВИТУ\_ОБРАТНО.

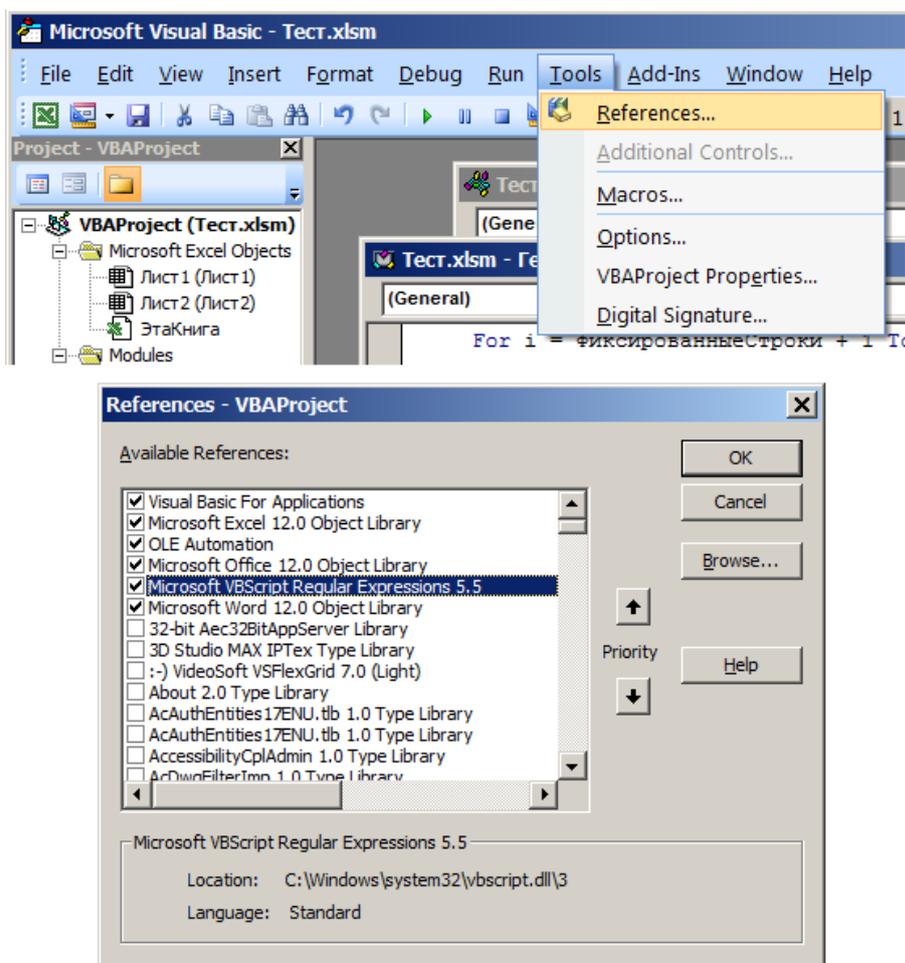


Рисунок 9 – Окно подключения модулей в программной среде VBA

**Sub**ФильтроватьТаблицу( \_

**Optional**НомерКлючевогоСтолбцаAsInteger = 1, \_

**Optional**Критерий1 AsVariant, \_

**Optional**УсловиеAsКРИТЕРИЙ\_УСЛОВИЕ= 0, \_

**Optional**Критерий2AsVariant = "")

Применяет фильтр для ключевого столбца виртуальной таблицы по одно-

му критерию или совокупности двух критериев. При использовании двух критериев фильтрации, необходимо указать условие, принимающее именные значения: И, ИЛИ – соответствующие логическим операциям конъюнкции и дизъюнкции между двумя критериями.

Результатом выполнения фильтрации будет производная таблица, созданная в отдельном листе виртуальной книги, новый лист будет назначен как активный виртуальный лист, с которым будут проводиться дальнейшие операции.

#### **Sub** УдалитьФильтр()

Удаляет созданный командой **ФильтроватьТаблицу** лист с результатом фильтрации. После удаления активным виртуальным листом будет назначен предыдущий, содержащий исконную виртуальную таблицу

### **3.6 Подстановка по ключевому элементу**

Иногда случается так, что нужная информация для отчета находится в нескольких таблицах и для формирования конечной таблицы необходимо их совместить, опираясь на некоторый повторяющийся в обеих таблицах ключевой элемент. Чтобы облегчить решение такой задачи был добавлен следующий функционал:

**Sub** ПодставитьИзТаблицы(НачальныйЛист *AsString*, \_  
ПодстановочныйЛист *AsString*, \_  
СписокСтолбцовПодстановки *AsVariant*, \_  
СписокСтолбцовСсылок *AsVariant*, \_  
КлючСтолбецОригинал1 *AsString*, \_  
КлючСтолбецСсылка1 *AsString*, \_  
**Optional**КлючСтолбецОригинал2 *AsString* = "", \_  
**Optional**КлючСтолбецСсылка2 *AsString* = "", \_  
**Optional**КлючСтолбецОригинал3 *AsString* = "", \_  
**Optional** КлючСтолбецСсылка3 *AsString* = "")

Данная процедура позволяет подставить в таблицу на листе с названием *НачальныйЛист* из таблицы на листе *ПодстановочныйЛист* значения столбцов *СписокСтолбцовСсылок* в *СписокСтолбцовПодстановки*.

За раз возможно подставлять сразу несколько столбцов, списки столбцов оформляются в виде число-буквенного кода через точку с запятой. Количество подставляемых столбцов должно совпадать с количеством столбцов назначения подстановки.

Оставшиеся аргументы задают попарно ключевые столбцы, т.е. столбцы, значения которых должны совпасть, чтобы подстановка была возможна. Таким образом, возможно использовать до 3 ключевых столбцов.

Если в таблице доноре ключевые столбцы имеют не уникальные значения элементов (повторяются), то будет подставлено первое совпадение. В случае, если соответствий не было найдено, то поле будет оставлено пустым, а в окно отладки выведено сообщения о пропуске строки с указанием ее номера и ключевых столбцов.

### **3.7 Подстановочные шаблоны**

В данном контексте шаблоном будем называть файл в формате .docx, который является готовой основой для подстановки в него необходимых данных из таблицы.

Каждая подстановка характеризуется символьным кодом, оставляемым в теле шаблона, при подстановке данные коды будут найдены и заменены указанными значениями. Коды подстановок были разделены на две категории: табличные и константные.

Первые должны быть расположены в ячейках таблицы и не повторяться в документе шаблона, в них будут подставляться наборы табличных данных путем расширения таблицы.

Вторые являются постоянными, т.е. заменяемыми одним конкретным значением. Они могут повторяться и быть расположены в любых местах документа, включая колонтитулы.

Рассмотрим команды для работы с файлами-шаблонами и подстановками.

**Function**ОткрытьФайлШаблона(ПутьКФайлуAsString) AsBoolean

Создает скрытую копию приложения Word и открывает файл-шаблон по указанному пути. В случае успеха возвращает *True*, если же шаблон не был

успешно открыт или не найден, возвращает значение *False*.

**Function**ПоказатьШаблон() *AsString*

Делает видимым для пользователя копию приложения Word, запущенную командой **ОткрытьФайлШаблона**.

**Sub**ЗакрытьФайлШаблона()

Закрывает копию приложения Word, запущенную командой **ОткрытьФайлШаблона**.

**Sub**СохранитьШаблонКак(Путь *AsString*, \_

**Optional**РасширениеФайла*AsRACШИПЕНИЕ* = DOCX, \_

**Optional**ОптимизироватьДляПечати*AsBoolean* = False)

Сохраняет открытый документ шаблона по указанному пути. Сохраняемое расширение файла задается фиксированными значениями из списка: DOC, DOCX, RTF, PDF. При отсутствии аргумента, указывающего расширение, отчет будет сохранен в формате .docx. Последний аргумент используется при выборе формата .pdf, определяет степень сжатия изображений в сохраняемом документе.

**Sub**ПодставитьВШаблон( \_

КодПодстановки*AsString*, \_

ПодставляемоеЗначение*AsString*, \_

**Optional**ТипСодержимого*AsТИП\_СОДЕРЖАНИЯ* = ТЕКСТОВЫЙ, \_

**Optional**Ширина*AsInteger*, \_

**Optional**Высота*AsInteger*)

Подставляет в шаблон на место кода подстановки указанное константное значение. Аргумент типа содержания может принимать следующие значения: ТЕКСТОВЫЙ, ИЗОБРАЖЕНИЕ, ПОЛЕ. Тут полем называются экспресс-блоки с генерируемым содержанием. Последние два аргумента применяются при выборе типа содержания ИЗОБРАЖЕНИЕ, и задают размеры подставляемого изображения, а путь нахождения изображения должен быть определен в аргументе подставляемого значения.

**PropertyLet**ПодставлятьВКолонтитулы (Значение *AsBoolean*)

Определяет, будет ли проводиться поиск для подстановки в колонтитулах.

**Sub**ДобавитьСоответствие(  
Код **AsString**, \_  
Столбец **AsVariant**, \_  
**Optional**ТипСодержимого**AsТИП\_СОДЕРЖАНИЯ** = ТЕКСТОВЫЙ, \_  
**Optional**Ширина**AsVariant**, \_  
**Optional**Высота**AsVariant**)

Задаёт соответствие кода подстановки и столбцом, из которого будут браться данные. Каждый вызов команды сохранят значения в накапливаемый массив. При подстановке изображения, значение ширины и высоты могут быть указаны как постоянные или как ссылка на столбец таблицы в виде кодового значения, либо в виде номера столбца со знаком минус.

**Sub**ОчиститьСоответствия()

Очищает массив соответствий, заданный ранее вызовами команды **ДобавитьСоответствие**.

**Sub**ПодставитьВШаблонТаблицу(  
**Optional**ПодставляемыйДиапазон**AsRange** = Nothing)

Подставляет в шаблон табличные данные из указанного диапазона согласно заданным командой **ДобавитьСоответствие** отношениям. В случае, если подставляемый диапазон не указан, будет использоваться все задействована область виртуальной таблицы

Для работы данных команд необходимо дополнительно активировать модуль **MicrosoftWordObjectLibrary** актуальной версии (рисунок9).

## 4 РАЗРАБОТКА ПРОГРАММНОГО МОДУЛЯ ДЛЯ СОЗДАНИЯ ГРАФИЧЕСКОГО ИНТЕРФЕЙСА ПОЛЬЗОВАТЕЛЯ

### 4.1 Структура и логика работы программного модуля

Как уже упоминалось ранее, не все современные решения возможно использовать в среде VBA. Наиболее универсальным вариантом с точек зрения функциональности, мобильности и автономности является использование динамически подключаемых библиотек (DLL) [6]. А как показано в пункте 2.8, использование динамических библиотек может сильно выиграть в производительности, нежели в случае попытки разработки фреймворка только средствами VBA.

Реализацию поставленной задачи было решено осуществлять на основе архитектурной модели MVC (англ. *Model-View-Controller*, «Модель-Представление-Контроллер»). Принцип MVC предполагает разделение управляющей логики и пользовательского интерфейса на три независимых модуля: *модель*, *представление (вид)*, *контроллер*. В рамках имеющейся задачи модуль *модели*, который отвечает за доступ к данным и реализует управляющую логику, расположен непосредственно в VBA программе, а библиотека фреймворка содержит в себе модуль *контроллера*, обрабатывающий команды пользователя, и модуль *представления (вида)*, отвечающий за графическую часть интерфейса пользователя. Схематичное устройство и взаимодействие описанных блоков изображено на рисунке 10.

За основу были взяты списочные интерфейсы, когда каждый последующий элемент управления расположен под предыдущим. С этой моделью пользовательский интерфейс хорошо знаком каждый, кто имеет сенсорный мобильный телефон, так как именно с приходом сенсорного управления данный тип интерфейсов получил наибольшую популярность, тем не менее, это обстоятельство не исключает возможность его использовать для ПК приложений. Основной важной особенностью такого интерфейса является его последо-

вательность, пользователь идет сверху вниз, исключая возможность пропустить что-то или запутаться в большой концентрации активных элементов управления. Именно такой интерфейс, по нашему мнению, хорошо подходит для малых и средних приложений VBA, когда необходимо ввести несколько входных параметров для выполнения программы.



Рисунок 10 – Схематичное представление внутренней структуры и взаимодействия VBA программы и фреймворка.

Второй важной особенностью списочных интерфейсов является их относительно быстрое создание: разработчику нет необходимости располагать каждый элемент по координатной сетке, достаточно лишь указать какие элементы необходимо отобразить на форме.

Была реализована внешняя подключаемая библиотека DLL фреймворка на основе кроссплатформенной библиотеки с открытым кодом Qt, которая содержит довольно гибкий фреймворк для создания графических интерфейсов пользователя любой сложности. Библиотека была собрана в среде разработки QtCreator 4.14 с версией Qt 5.15.

В целях удобства использования данной библиотеки в среде VBA был до-

полнительно разработан модуль интеграции для более удобного взаимодействия с библиотекой фреймворка при создании графических интерфейсов пользователя.

На рисунке 11 приведено схематичное представление взаимодействия библиотек между собой.



Рисунок 11 – Диаграмма взаимодействия модулей библиотек

Программная среда разработки VBA интегрирована в MS Excel и является его частью. Пользователь-разработчик создает графический интерфейс общаясь с библиотекой фреймворка посредством модуля интеграции, который является всего лишь программным модулем на языке VBA и представляет функционал фреймворка в удобной для программирования на VBA форме.

Модуль интеграции упрощает общение с динамической библиотекой (DLL), которая является оболочкой для более объемного и сложного фреймворка Qt, реализуя в упрощенной форме только необходимый функционал для создания графического интерфейса пользователя. Сам интерфейс пользователя уже создается и работает непосредственно на ядре Qt.

Общение VBA программы с фреймворком ограничивается спецификой подключаемых динамических библиотек, позволяющей только обмениваться данными по типу запрос-ответ. Однако, дополнительно, посредством функций WinAPI был реализован обратный вызов функции (callback функция) внутри VBA программы [17], что позволило организовать систему прерываний из библиотеки фреймворка в VBA. Таким образом, было достигнуто полноценное двустороннее общение VBA программы и библиотеки фреймворка.

Далее рассмотрим перечень основных доступных функций для работы с фреймворком.

#### **4.2 Подключение библиотеки фреймворка**

Прежде чем начинать работать непосредственно с графическими интерфейсами необходимо подключить библиотеку фрейворка.

##### **Sub**ПодключитьФреймворк()

Данная команда запускает фоновый процесс ядра Qt из динамической библиотеки.

##### **Sub**ЗакрытьФреймворк()

Закрывает открытый ранее процесс фреймворка.

После того как фреймворк становится больше не нужен, его необходимо закрыть, иначе забытый запущенный фоновый процесс может помешать нормальной работе MSExcel.

#### **4.3 Работа с окнами**

##### **Function**СоздатьОкно (

ЗаголовокОкнаAsString, \_

##### **Optional**ТаблицаСтилейAsString = \_

ТАБЛИЦА\_СТИЛЕЙ\_ПО\_УМОЛЧАНИЮ) AsLong

Создает пустое окно, принимает первым аргументом текст заголовка окна, второй аргумент является опциональным, в случае, если он не указывается, то принимает значение константы ТАБЛИЦА\_СТИЛЕЙ\_ПО\_УМОЛЧАНИЮ, которая содержит название таблицы стилей стандартного оформления. Функция возвращает порядковый номер созданного окна, он может использоваться

для обращения к конкретному окну в многооконном режиме.

**Sub**ЗакретьОкно()

Закрывает активное окно.

**Sub**ПоказатьОкно()

**Sub**СкрытьОкно()

Команды, позволяющие отобразить или скрыть активное окно, соответственно.

**Sub**ВыбратьОкно(НомерОкнаAsLong)

Делает окно под указанным номером активным, используется для выбора нужного окна в многооконном режиме.

#### **4.4 Работа со слоями**

Для автоматического позиционирования элементов на форме используются слои с компоновкой в виде вертикального и горизонтального списков.

**Sub**ДобавитьСлой\_ВертикальныйСписок()

**Sub**ДобавитьСлой\_ГоризонтальныйСписок()

Вставляет в текущую позицию слой с вертикальной или горизонтальной компоновкой элементов, соответственно.

При добавлении нового слоя на форму курсор вставки перемещается на новый слой, чтобы вернуть его на предыдущий слой используется данная команда:

**Sub**ВернутьсяНаРодительскийСлой()

#### **4.5 Работа с элементами графического интерфейса**

Следующие функции позволяют добавлять на форму новые элементы графического интерфейса пользователя.

**Function**ДобавитьКнопку(ТекстAsString, \_  
НомерПрерыванияAsLong, \_  
**Optional**ИмяОбъектаAsString) AsLong

Функция добавляет кнопку в текущую позицию на форме. Первый аргумент задает текст, отображаемый на кнопке. Второй аргумент определяет номер прерывания, данный номер будет возвращен в функцию обратного вызова при

совершении события нажатия на кнопку, что позволяет установить, какая именно кнопка была нажата. Третий опциональный аргумент позволяет задать идентификатор объекта в виде строки для получения быстрого доступа к данному элементу. Функция возвращает уникальный идентификационный номер созданного элемента.

**Function**ДобавитьНадпись(ТекстAsString, \_

**Optional**ИмяОбъектаAsString) AsLong

Добавляет статичную надпись. Первый аргумент задает текст надписи, отметим, что данный элемент Qt поддерживает форматирование HTML тегами, что позволяет форматировать выводимый текст, а также вставлять картинки. Функция возвращает уникальный идентификационный номер созданного элемента.

**Function**ДобавитьПоле(

**Optional**ТекстУдержанияAsString, \_

**Optional**ТекстСправкиAsString, \_

**Optional** ИмяОбъекта As String, \_

**Optional** ДобавитьВЗапись As Boolean = True) As Long

Добавляет текстовое поля с возможностью редактирования его содержимого. Все аргументы являются опциональными, если аргумент не указан, он принимается равным пустой строке. Аргумент *ТекстУдержания* определяет текст, который будет отображаться, когда поле остается пустым. *ТекстСправки* позволяет задать содержимое, всплывающее подсказки при наведении на элемент. *ИмяОбъекта* определяет идентификатор объекта в виде строки для получения быстрого доступа к данному элементу. Последний аргумент *ДобавитьВЗапись* определяет, будет ли включено данное поле в список быстрого вывода. Функция возвращает уникальный идентификационный номер созданного элемента.

**Function**ЗначениеЭлемента(НомерAsLong) AsString

Возвращает содержимое элемента с указанным идентификационным номером как текстовую строку.

**Function** ЗначениеЭлементаПоИмени(ИмяОбъекта As String) As String

Возвращает содержимое элемента по заданному имени объекта как текстовую строку.

#### 4.6 Работа со списком быстрого вывода

Список быстрого вывода – это реализованный функционал для быстрого доступа к содержимому многих элементов. В процессе добавления новых элементов на форму, содержащих опциональный аргумент *ДобавитьВЗапись*, и при установлении значения ИСТИНА данного аргумента, элемент будет добавлен в специальный список записей. Далее специальными функциями можно получить значения всех элементов добавленных в данный список без необходимости обращаться к каждому элементу отдельно, что позволяет значительно экономить время при создании простых форм для заполнения.

Список быстрого вывода создается для каждого окна отдельно.

**Function**СоставитьЗаписи() AsLong

Переносит содержимое элементов в списке быстрого вывода во временный массив и возвращает количество позиций в этом массиве.

**Function**СледующаяЗапись() AsString

Возвращает текущее значение во временном массиве и перемещает курсор на следующий элемент массива.

#### 4.7 Пример применения фреймворка

Рассмотрим применение разработанной библиотеки фреймворка на примере создания простого диалогового окна.

На рисунке 12 приведен код, который создает окно с одной надписью, одним полем и двумя кнопками.

Сначала создается пустое окно командой *СоздатьОкно*. Созданное окно не является абсолютно пустой формой, там уже содержится начальный шаблон будущего интерфейса, а добавляемые элементы будут помещены в начальный родительский слой с вертикальной компоновкой.

Добавим в созданное окно надпись командой *ДобавитьНадпись* и текстовое поле командой *ДобавитьПоле*. Как уже отмечалось ранее, текстовые поля

некоторых элементов поддерживают HTML форматирования, пример использования тегов для форматирования текста можно увидеть на строках 7-9.

```
1 Public Sub СоздатьДиалоговоеОкно()  
2     СоздатьОкно "Выбор файла"  
3  
4     ДобавитьНадпись "Исходная таблица"  
5  
6     ДобавитьПоле ТекстУдержания:="Путь до таблицы", _  
7         ТекстСправки:="Укажите путь до таблицы в формате" _  
8             & "<font color=green>.xlsx</font>" _  
9             & "или <font color=green>.xls</font>", _  
10        ИмяОбъекта:="ПолеПуть"  
11  
12    ДобавитьСлой_ГоризонтальныйСписок  
13  
14    ДобавитьКнопку "Отмена", 0  
15    ДобавитьКнопку "Далее", 1  
16  
17    ПоказатьОкно  
18 End Sub
```

Рисунок 12 – Программный код создания окна графического интерфейса

Далее добавим две кнопки, чтобы расположить их горизонтально рядом друг с другом, создадим прежде новый слой с горизонтальной компоновкой командой *ДобавитьСлой\_ГоризонтальныйСписок*, затем добавим две кнопки командой *ДобавитьКнопку*. Указанные при создании кнопок цифры обозначают номер события, который будет возвращен в функцию обратного вызова при нажатии на ту или иную кнопку.

На рисунке 13 представлен код функции обратного вызова (callback) для обработки событий нажатия кнопок.

```
1 Public Sub ОбработкаСобытий(ByVal Номер As Long)  
2     Select Case Номер  
3         Case 0  
4             ЗакретьОкно  
5         Case 1  
6             Сообщение ЗначениеЭлементаПоИмени("ПолеПуть")  
7     End Select  
8 End Sub
```

Рисунок 13 – Программный код функции обработки событий нажатия кнопок

При нажатии на кнопку происходит вызов фреймворком callback функции

из программы VBA с передачей в качестве аргумента номера кнопки, которая породила данное событие. Внутри функции нажатие обрабатывается в соответствии с номером кнопки. В данном примере при нажатии на кнопку «Отмена» будет закрыто окно интерфейса, а при нажатии на кнопку «Далее» выведено сообщение с содержанием текстового поля.

На рисунке 14 изображено окно графического интерфейса, созданное в результате выполнения выше приведенного программного кода.

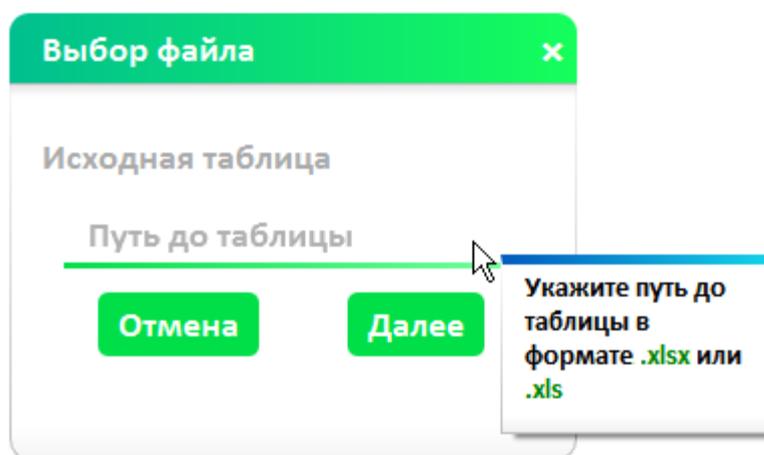


Рисунок 14 – Пример созданного окна графического интерфейса пользователя

## 5 РЕАЛИЗАЦИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ ФОРМИРОВАНИЯ ОТЧЕТОВ О РАБОТЕ ЦИФРОВЫХ ПЕРЕДАТЧИКОВ

### 5.1 Структура программного обеспечения

На основе разработанных модулей, описанных в главах 3 и 4, было реализовано программное обеспечение для автоматического формирования отчетов из таблицы с информацией о произошедших инцидентах. Программное обеспечение представляет собой комплекс из программ макросов в среде VBA и подключаемой библиотеки фреймворка в формате динамической библиотеки (DLL). Схематичное представление взаимодействия частей программного обеспечения представлено на рисунке 15.



Рисунок 15 – Схематичное представление взаимодействия частей программного обеспечения

Основной программой является VBA макрос, который включает в себя

класс-модули: модуль интерфейса (содержит функционал для работы с подключаемой внешней библиотекой фреймворка (см. главу 4)) и модуль создания отчетов (содержит функционал, облегчающий работу с таблицами и формирование отчетов (см. главу 3)).

При запуске VBAпрограммы создается окно пользовательского интерфейса с помощью внешней библиотеки фреймворка. После ведения пользователем необходимых данных в поля графического интерфейса, они передаются VBAпрограмме, которая посредством функционала модуля для создания отчетов загружает и формирует табличные данные в необходимой форме.

Сформированные данные подставляются в заранее подготовленные шаблоны отчетов. В виду того, что исполняющая программа запущена внутри Excel, она может напрямую управлять его функционалом и формирование отчета в виде таблиц не представляет трудностей. Для формирования отчета в формате MSWord, используется COMтехнология, которая позволяет вызывать и управлять другими независимыми приложениями, таким образом, исполняющая программа формирует отчет средствами самого MSWord.

Отметим, так как основная программа является макроскриптом, она может храниться как отдельно, так и в составе родительской таблицы. Для удобства использования и запуска макроса, он был сохранен в интегрированном в Excelфайл виде (расширение .xlsm– Excel книга с поддержкой макросов). В самой родительской Excel книге находятся дополнительные настройки для корректной работы программы, что также удобно для их редактирования при необходимости.

## **5.2 Графический интерфейс пользователя**

Модуль для формирования отчетов является полностью автономным макросом и может быть запущен из среды VBA, в качестве аргументов ему необходимо указать только пути начальной таблицы с выгрузкой инцидентов из системы SDи папку, куда ему следует сохранять сформированные отчеты. Однако, в целях удобства использования программы, а также, для обеспечения интуитивной понятности и исключения ошибок при использовании макроса был

разработан вспомогательный макрос реализующий вывод графического интерфейса пользователя. На рисунке 16 представлен внешний вид главного окна графического интерфейса.

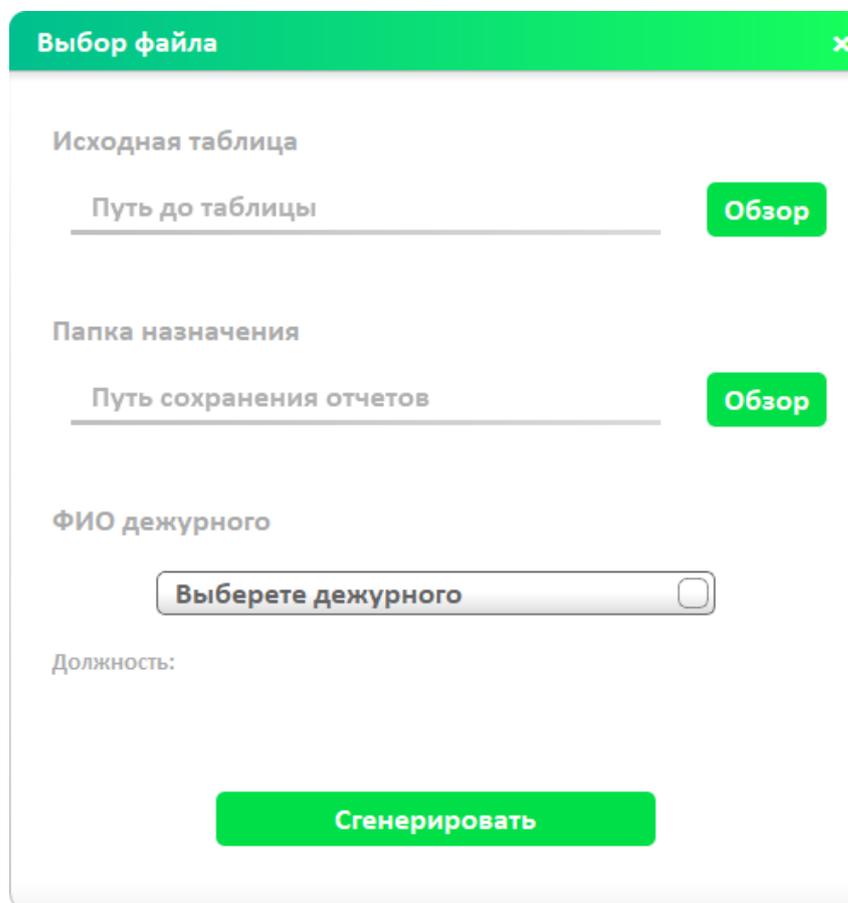


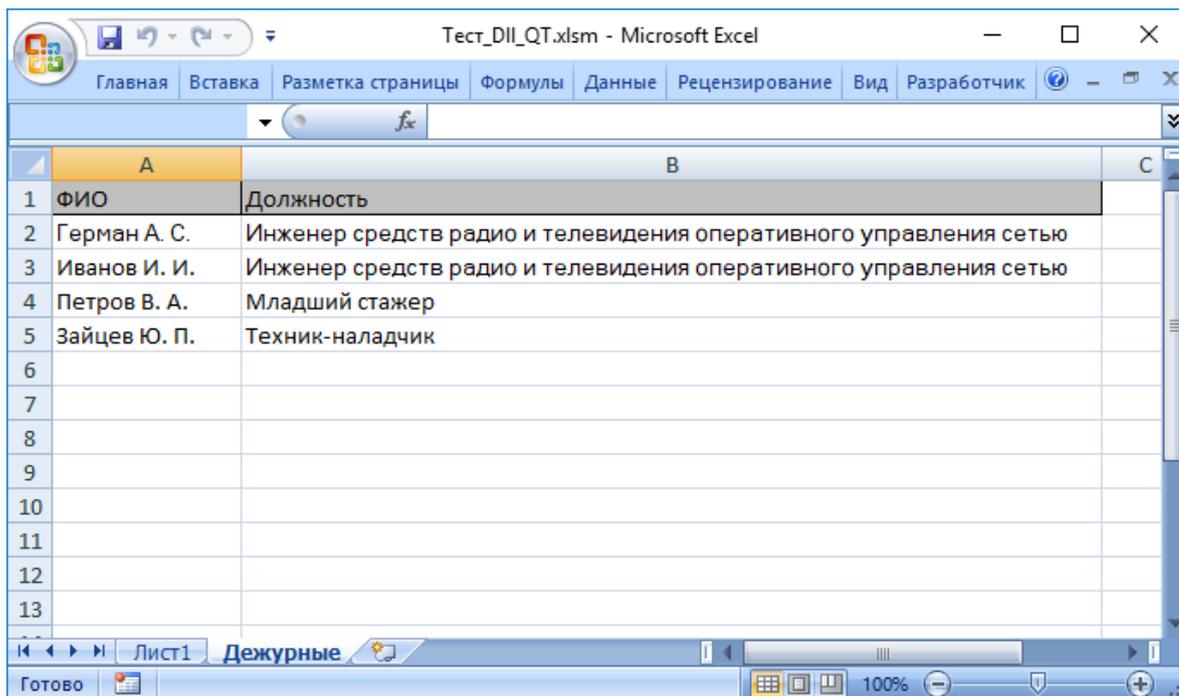
Рисунок 16 – Главное окно графического интерфейса пользователя

Интерфейс крайне прост и понятен, требуется указать путь к файлу с таблицей содержащий сведения об инцидентах в формате *.xls* либо *.xlsx*, путь к папке сохранения отчетов, а также выбрать из выпадающего списка дежурного, чьим именем будут подписаны отчеты.

Для удобства выбора файлов и папки предусмотрены кнопки «Обзор», нажатие на которые вызовет стандартный диалог проводника для выбора файла или папки, соответственно.

Содержимое выпадающего списка определяется таблицей «Дежурные» в составе родительской книги (рисунок 19), где содержится исполняемый макрос, что позволяет просто вносить изменения и добавлять новые записи в список со-

трудников.



The screenshot shows a Microsoft Excel window titled 'Тест\_DII\_QT.xlsm'. The ribbon includes 'Главная', 'Вставка', 'Разметка страницы', 'Формулы', 'Данные', 'Рецензирование', 'Вид', and 'Разработчик'. The active sheet is 'Лист1' with the title 'Дежурные'. The table contains the following data:

	A	B	C
1	ФИО	Должность	
2	Герман А. С.	Инженер средств радио и телевидения оперативного управления сетью	
3	Иванов И. И.	Инженер средств радио и телевидения оперативного управления сетью	
4	Петров В. А.	Младший стажер	
5	Зайцев Ю. П.	Техник-наладчик	
6			
7			
8			
9			
10			
11			
12			
13			

Рисунок 17 – Таблица «Дежурные»

После заполнения формы нажимаем на кнопку «Сгенерировать», далее будет отображено окно с информацией о ходе процесса формирования отчетов с процентным указанием завершенности и описанием текущего этапа (рисунок 18).

При успешном завершении создания отчетов они будут сохранены в указанную ранее папку, а пользователь будет извещен о завершении процесса: ранее неактивная кнопка «Готово» станет доступна, при нажатии на нее окно программы будет закрыто.

На рисунке 19 представлена блок-схема логики работы графического интерфейса пользователя. Отметим, что программа, отвечающая за интерфейс, служит только для общения с пользователем и не включает в себя работу с таблицами, для этого она запускает макрос, отвечающего за обработку табличных данных, и передает ему необходимые параметры, полученные от пользователя. На блок-схеме макрос формирования отчетов изображен блоком вызова вспомогательного алгоритма «Создание отчетов».

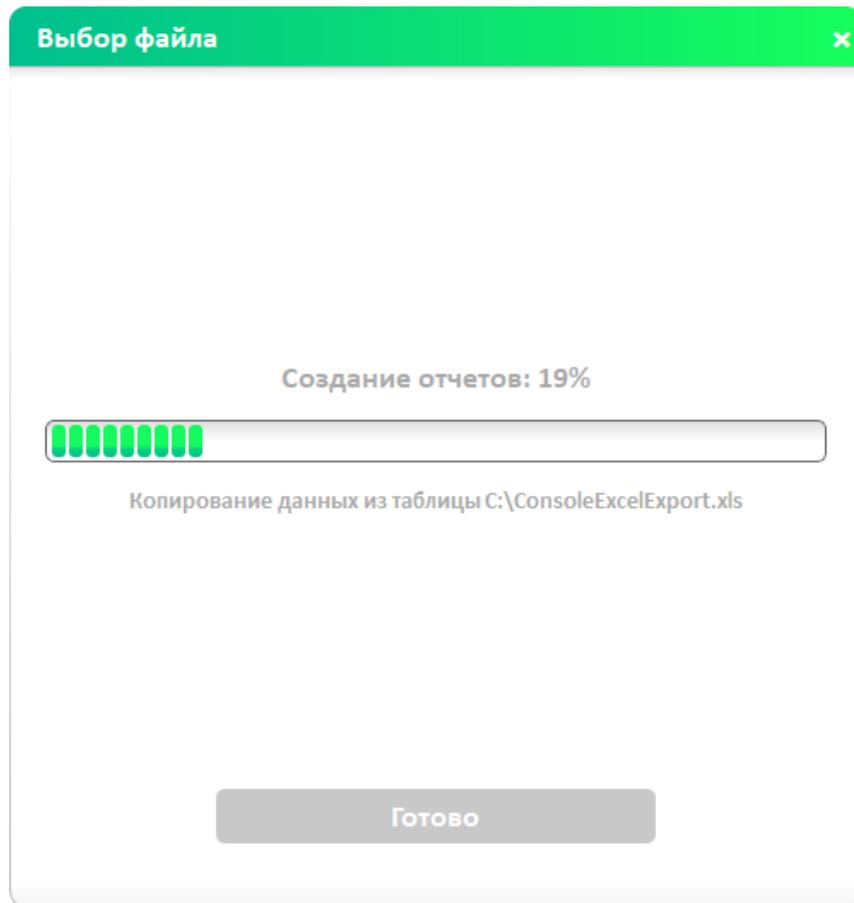


Рисунок 18 – Окно ожидания завершения формирования отчетов

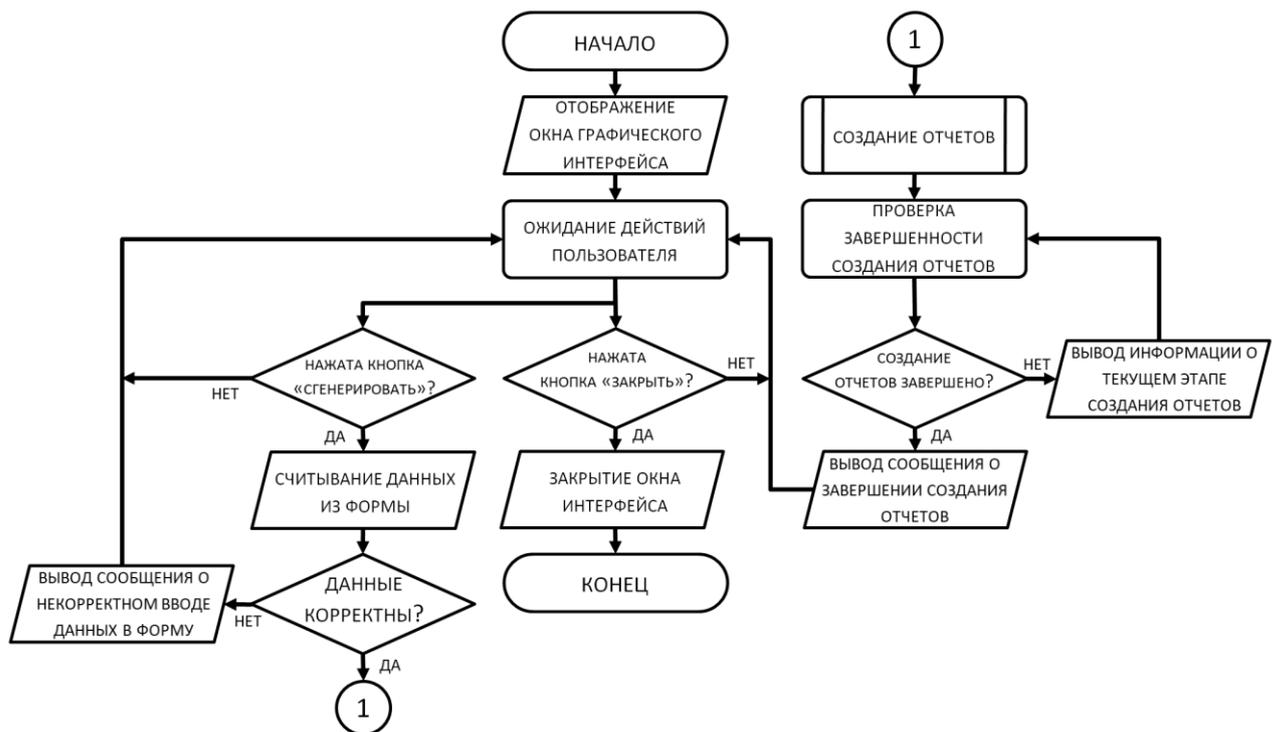


Рисунок 19 – Блок-схема работы интерфейса

### 5.3 Контроль ошибок

Разработанное программное обеспечение предусматривает обработку возникающих ошибок. При возникновении ошибки в процессе формирования отчета пользователю будет выведено сообщение о возникшей проблеме (рисунок 20), после чего пользователь может самостоятельно закрыть окно программы. Если же ошибка возникает в процессе вывода графического интерфейса, то пользователь получит диалоговое окно с текстовым уведомлением об аварийном прекращении работы программы.

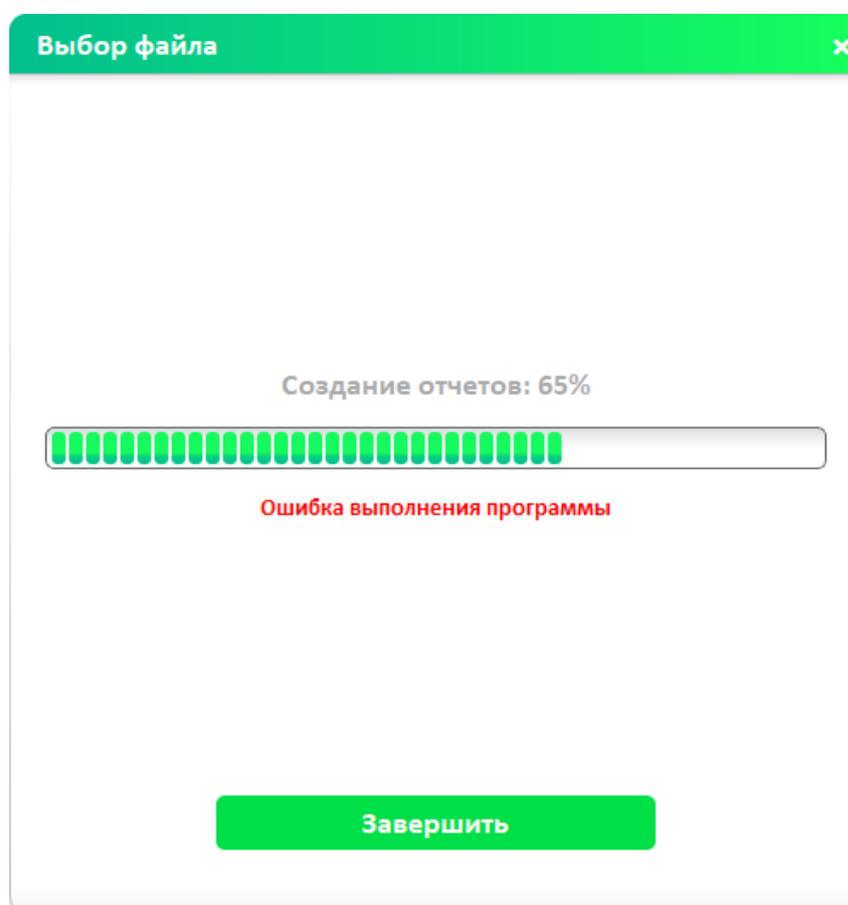


Рисунок 20 – Демонстрация возникновения ошибки в процессе формирования отчетов

Программная обработка ошибок является важным моментом при работе с подключаемыми библиотеками, которые работают в фоновом режиме, так как это позволяет правильно завершить работу программы при возникновении

внештатной ситуации, в противном случае это может привести к зависанию и последующему аварийному завершению родительского приложения Excel.

Для получения дополнительной информации о произошедшем нужно открыть окно среды VBA нажав соответствующую кнопку на вкладке «Разработчик», либо сочетанием клавиш *Alt+F11*, затем открыв окно отладочных сообщений, выбрав вкладку *View→ImmediateWindow*, либо нажав сочетание клавиш *Ctrl+G*. В открывшемся окне *ImmediateWindow* содержатся записи об основных этапах работы программы, что позволяет определить место возникновения ошибки, в конце будет выведено сообщение с информацией о возникшей ошибке. На рисунке 21 приведен пример отладочных записей при возникновении тестовой ошибки деления на ноль.

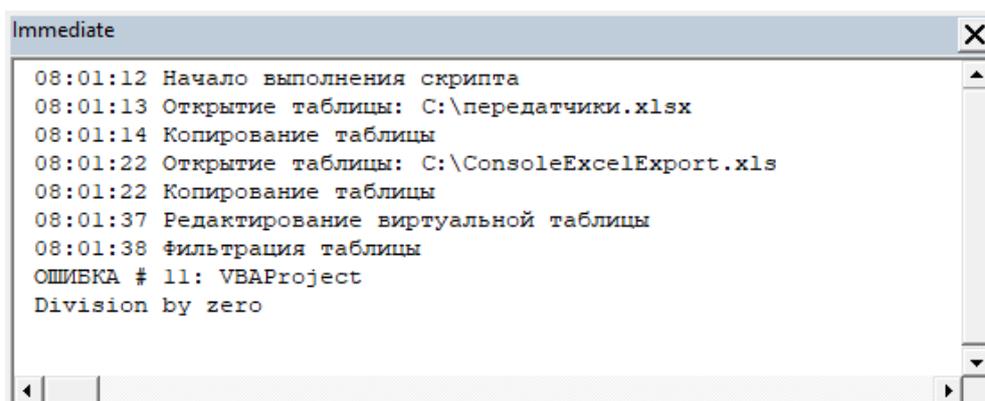


Рисунок 21 – Пример содержания окна *Immediate* при возникновении ошибки

Отметим так же, как было упомянуто в главе 3, модуль формирования отчетов имеет специальный отладочный режим, который активируется методом **ОтладочныйРежим**. В обычном режиме все операции над таблицами проводятся скрыто в фоновом режиме, при активации данной функции временные таблицы для хранения промежуточных данных, а также процесс подстановки в шаблоны становится видимым, что позволяет наглядно определить возникшую проблему и устранить ее.

## ЗАКЛЮЧЕНИЕ

Целесообразностью создания ПО для формирования отчетов на основе выгрузок из системы SD является острая необходимость в экономии времени. Инциденты формируются в зависимости от внешних обстоятельств и не поддаются статистическому прогнозированию. В день может быть более полутора тысяч инцидентов.

Результаты выполнения магистерской диссертации:

1) Выяснено, что итоговое программное обеспечение может быть реализовано двумя подходами: с использованием интегрированной в приложения MicrosoftOffice среды программирования VBA (VisualBasicforApplications) для написания алгоритма обработки данных – макроса и с использованием технологии OLE для получения содержания таблиц, обработки и формирования новых документов Excel, через внешнее стороннее приложение.

2) Выяснено, что возможность реализовать параллельные вычисления только средствами VBA, что приемлемо при необходимости ускорить обработку данных при редком использовании, но требовательно к ресурсам. Использовании внешней динамической библиотеки для выполнения параллельных вычислений более предпочтительно, т.к. имеет большие возможности и позволяет достичь хороших показателей производительности.

3) Реализовано программное обеспечение в среде программирования VBA на базе текстового процессора MSExcel и Word, которое соответствует поставленным требованиям и, по мнению автора, является наиболее оптимальным решением, объединяющим в себе простоту, интуитивную понятность и функциональную гибкость.

4) Реализована на языке C++ подключаемая динамическая библиотека в среде разработки QtCreator на основе фреймворка Qt. Дополнительно был разработан программный модуль в среде VBA для более удобного взаимодействия с подключаемой библиотекой фреймворка.

5) Представлены примеры использования разработанного программного модуля для решения потенциальных задач, которые показали хорошие функциональные возможности при минимальном количестве программного кода, что позволяет быстро создавать графический интерфейс пользователя для решения широко круга прикладных задач.

6) Сформулированы основные требования к разработке пользовательских интерфейсов с учетом специфики использования интерфейсов в программах VBA. Предложен прототип подключаемой библиотеки фреймворка для создания пользовательских интерфейсов, который должен позволить экономить время при разработке VBA приложений.

## БИБЛИОГРАФИЧЕСКИЙ СПИСОК

- 1 Официальный сайт РТРС [Электронный ресурс]. Раздел эфирного телевидения. Режим доступа: <https://amur.rtrs.ru/tv/efirnoe-televeshchanie/>
- 2 Лебедев, В. М. Программирование на VBA в MSEXCEL. / В. М. Лебедев. - М.:Юрайт, 2019. - 272 с
- 3 Большаков, С. А. Двухфазная технология формирования выходных документов в информационных системах на основе MSWord / С. А. Большаков, С. Б. Спиридонов // Машиностроение и компьютерные технологии. - 2015.- № 8. - С. 252-268.
- 4 Сайт компании Microsoft [Электронный ресурс]. Начало работы с VBA в Office. Режим доступа: <https://docs.microsoft.com/ru-ru/office/vba/library-reference/concepts/getting-started-with-vba-in-office> – 15.05.21
- 5 Сайт компании Microsoft [Электронный ресурс]. Раздел разработки Win32 и COM в библиотеке MSDN. Режим доступа: <https://msdn.microsoft.com/en-us/library/aa139672.aspx>
- 6 Сайт компании Microsoft [Электронный ресурс]. Начало работы с VBA в Office. Режим доступа: <https://docs.microsoft.com/ru-ru/office/vba/library-reference/concepts/getting-started-with-vba-in-office>
- 7 Зинюк, О. В. Методика расчета ошибки и однородности выборки средствами MSeXcel и VBA / О. В. Зинюк // Статистика и экономика. - 2011.- № 4. - С. 77-81.
- 8 Гордина, С. В. VisualBasicforApplication как система объектноориентированного программирования / С. В. Гордина // ИТС. - 2016.- № 4. - С. 60-65.
- 9 Макапов, А. А. Основные возможности VBA и MSeXcel: преимущества и недостатки / А. А. Макапов // Информация и образование: границы коммуникаций. - 2019.- № 11 (19). - С. 124-125.

10 Карташов, Д. В. Методика обучения алгоритмам и структурам данных / Д. В. Карташов, А. Н. Стась // Вестник ТГПУ. - 2015.- № 8 (161). - С. 131-134.

11 Лутцев, Е. Г. Программирование на языках, приближенных к естественному: обзор литературы / Е. Г. Лутцев // Информатика. - 2015.- № 2 (46). - С. 39-45.

12 Семенов, К. С. Сравнительный анализ лексем языков программирования и частей речи русского языка / К. С. Семенов // Материалы Всероссийской научно-практической конференции "Научная дискуссия современной молодежи: актуальные вопросы, достижения и инновации". - М.: Издательство ООО "Конверт", 2020. - С. 145-148.

13 Городняя, Л. В. О языке начального обучения параллельному программированию / Л. В. Городняя, А. С. Боярских // Вестник НГУ. Серия: Информационные технологии. - 2012.- № 10 (2). - С. 85-99.

14 Абрамова И. А. Расширения стандартного функционала и интерфейса приложений MSOffice на основе разработки пользовательских надстроек / И. А. Абрамова, В. В. Сыркин, А. П. Степанов // Наука и военная безопасность – 2020 – №2 (21). – С. 192-199.

15 Искра Н. А. Изучение и оценка походов к разработке графического интерфейса пользователя / Н. А. Искра, В. Н. Макоед, Е. Ю. Куница // Объектные системы. – 2015. – №10 – С. 63-68.

16 Грузин Н. А. Обзор и сравнение библиотек пользовательских интерфейсов: GTK, QT, WXWIDGETS / Н. А. Грузин, А. А. Голубничий // E-Scio. – 2020. – №2 (41). – С. 1-8.

17 Accessing and calling DLLs from VBA: Calling back to VBA functions from the DLL [Электронный ресурс]. Режим доступа: <https://renenyffenegger.ch/notes/development/languages/VBA/Calling-DLLs/callback> - 15.05.21

18 Герман, А.С. Алгоритм кусочной обработки данных в среде VBAExcel / А.С. Герман // Молодежь XXI века: шаг в будущее: материалы XXII региональной научно-практической конференции (20 мая 2020 г., Благовещенск)

щенск): в 4 томах. – Благовещенск: Изд-во Амурского гос. ун-та, 2020. – С. 112-113

19 Герман, А.С. Параллельные вычисления в среде VBA Excel / А.С. Герман, Е.М. Веселова // в сборнике материалов III Всероссийской национальной научной конференции студентов, аспирантов и молодых ученых «Молодежь и наука: актуальные проблемы фундаментальных и прикладных исследований», 06-10 апреля 2020 г. – Комсомольский-на-Амуре государственный университет, 2020. – С. 245-247

20 Герман, А.С. Разработка фреймворка для создания пользовательского интерфейса макросов VBA / А.С. Герман, Е.М. Веселова // Вестник АмГУ. Серия «Естественные и экономические науки». – Благовещенск: АмГУ, 2020. – Вып. 91 – С. 37-41

21 Герман, А.С. Применение Excel и Matlab/Simulink в научно-образовательной деятельности / А.С. Герман, Д.О. Струков, Д.В. Фомин, Е.М. Веселова // Физика: фундаментальные и прикладные исследования, образование: материалы XVIII региональной научной конференции, Хабаровск, 10-12 ноября 2020 г. – Хабаровск: Изд-во Тихоокеан. гос. ун-та, 2020. – С. 182-185

22 Герман, А.С. VBA Excel: реализация параллельных вычислений / А.С. Герман // в сборнике материалов XXIX научной конференции Амурского государственного университета «День науки», 23-25 ноября 2020 г. – Благовещенск: Изд-во Амурского гос. ун-та, 2020. – С. 45-46

23 Герман, А.С. Использование Qt фреймворка для создания интерфейса пользователя на VBA / А.С. Герман, Е.М. Веселова, Д.О. Струков // материалы региональной научно-практической конференция «ТОГУ - Старт: фундаментальные и прикладные исследования молодых». – Хабаровск, 2021. – С. 231-235

## ПРИЛОЖЕНИЕ А

### Листинг программы «Формирование отчетов по работе цифровых передатчиков»

```
Const ПутьКШаблонуВорд = "Шаблон РКН Амур.docx"  
Const ПутьКШаблонуЭксель = "Шаблон Сводка РКН.xlsx"  
Const ПутьКОтчетуВорд = "Сводка РКН Амур "  
Const ПутьКОтчетуЭксель = "Сводка РКН "  
Const ПапкаШаблонов = "C:\Шаблоны"  
  
DimПутьКИсходнойТаблицеAsString  
Dim ПутьКПапкеОтчета As String  
Dim ФИО, Должность As Variant  
Dim ФреймворкЗапущен As Boolean  
Dim ФВ As Фреймворк  
  
Sub Кнопка1_Щелчок()  
    Set ФВ = New Фреймворк  
    On Error GoTo ЕслиОшибка  
    ЗагрузитьСведенияОДежурных  
    With ФВ  
        .Инициализация  
        .ОпределитьФункциюОбратногоВызова (VBA.Int(AddressOf ОбработкаСобытий))  
        .ЗапуститьФреймворк  
    End With  
    Exit Sub  
ЕслиОшибка:  
    MsgBox "Ошибка запуска программы"  
    Debug.Print "ОШИБКА #" & Str(Err.Number) & ": " & Err.Source & Chr(13) & Err.Description  
    If Not ФВ Is Nothing And ФреймворкЗапущен Then  
        ФВ.АварийноеЗакрытие  
    End If  
End Sub  
  
Public Sub СоздатьИнтерфейс()  
    On Error GoTo ЕслиОшибка  
    With ФВ  
        .СоздатьОкно "Выбор файла"  
        .ЗадатьРазмерОкна 499, 530  
        .ДобавитьНадпись "Исходная таблица"  
        .ДобавитьСлой_ГоризонтальныйСписок  
        .ДобавитьПоле ТекстУдержания:="Путь до таблицы", _  
            ТекстСправки:="Укажите путь до таблицы в формате" _  
            & "<font color=green>.xlsx</font>" _  
            & " или<font color=green>.xls</font>", _  
        ИмяОбъекта:="ПолеПуть"  
        .ДобавитьКнопку "Обзор", 6
```

.ВернутьсяНаРодительскийСлой  
.ДобавитьРазделитель 30

## Продолжение ПРИЛОЖЕНИЯ А

Листинг программы «Формирование отчетов о работе цифровых  
передатчиков»

```
.ДобавитьНадпись "Папка назначения"  
.ДобавитьСлой_ГоризонтальныйСписок  
  .ДобавитьПоле ТекстУдержания:="Путь сохранения отчетов", _  
    ТекстСправки:="Укажите путь до папки, куда будут сохранены отчеты", _  
    ИмяОбъекта:="ПолеПутьПапка"  
  .ДобавитьКнопку "Обзор", 7  
.ВернутьсяНаРодительскийСлой  
.ДобавитьРазделитель 30  
.ДобавитьНадпись "ФИО дежурного"  
.ДобавитьСлой_ГоризонтальныйСписок  
  .ДобавитьРазделитель 50  
  .ДобавитьВыпадающийСписок ФИО, 2, "Комбо"  
  .ДобавитьРазделитель 50  
.ВернутьсяНаРодительскийСлой  
.ДобавитьНадпись "<font size=2>Должность: </font>", "Должность", .ДА '  
.ДобавитьРазграничитель  
.ДобавитьСлой_ГоризонтальныйСписок  
  .ДобавитьРазделитель 100  
  .ДобавитьКнопку "Сгенерировать", 3  
  .ДобавитьРазделитель 100  
.ВернутьсяНаРодительскийСлой  
.СледующаяСтраница  
.ДобавитьРазграничитель  
.ДобавитьНадпись "Создание отчетов: 0%", "Ожидание", _  
  ВыравниваниеСтроки:=ПО_ЦЕНТРУ  
.ДобавитьПолосуПрогресса "Прогресс"  
.ДобавитьНадпись "", "Этап", ВыравниваниеСтроки:=ПО_ЦЕНТРУ, _  
  ПереносСлов:=ДА  
.ДобавитьРазграничитель  
.ДобавитьСлой_ГоризонтальныйСписок  
  .ДобавитьРазделитель 100  
  .ДобавитьКнопку "Готово", 0, "Готово"  
  .АктивироватьЭлемент("Готово") = .НЕТ  
  .ДобавитьРазделитель 100  
.ВернутьсяНаРодительскийСлой  
.ПредыдущаяСтраница  
.ПоказатьОкно  
EndWith  
  
ExitSub  
ЕслиОшибка:  
MsgBox "Ошибка создания интерфейса"  
Debug.Print "ОШИБКА #" & Str(Err.Number) & ": " & Err.Source & Chr(13) & Err.Description
```

If Not ФВ Is Nothing And ФреймворкЗапущен Then  
ФВ.АварийноеЗакрытие

## Продолжение ПРИЛОЖЕНИЯ А

Листинг программы «Формирование отчетов о работе цифровых  
передатчиков»

End If  
End Sub

```
Public Sub ОбработкаСобытий(ByVal НомерОкна As Long, _  
                             ByVal НомерЭлемента As Long)  
    Dim Закрытие As Boolean  
    On Error GoTo ЕслиОшибка  
    With ФВ  
        SelectCase НомерЭлемента  
Case -3  
            .Сообщение "Аварийное закрытие Фреймворка!"  
            Закрытие = .ДА  
Case -2  
            .Сообщение "Поехали!"  
            ФреймворкЗапущен = .ДА  
            СоздатьИнтерфейс  
Case -1  
            .Сообщение "Закрытие Фреймворка!"  
            ФреймворкЗапущен = .НЕТ  
            Закрытие = .ДА  
Case 0  
            .ВыбратьОкно НомерОкна  
            .ЗакрытьОкно  
Case 2  
            .ЗначениеЭлементаПоИмени("Должность") = "<font size=2>Должность: " & _  
                Должность(.ЗначениеЭлементаПоИмени("Комбо")) & "</font>"  
Case 3  
            .СледующаяСтраница  
            ПутьКИсходнойТаблице = .ЗначениеЭлементаПоИмени("ПолеПуть")  
            ПутьКПапкеОтчета = .ЗначениеЭлементаПоИмени("ПолеПутьПапка")  
            НомерКомбо = .ЗначениеЧислоЭлементаПоИмени("Комбо")  
            СоздатьОтчет ПутьКИсходнойТаблице, ПутьКПапкеОтчета, _  
                ФИО(НомерКомбо), Должность(НомерКомбо)  
Case 6  
            .ЗначениеЭлементаПоИмени("ПолеПуть") = _  
                .ПоказатьДиалог(ВЫБРАТЬ_ФАЙЛ, Array("Excel files", "*.xls*;*.xla*", 1))  
Case 7  
            .ЗначениеЭлементаПоИмени("ПолеПутьПапка") = _  
                .ПоказатьДиалог(ВЫБРАТЬ_ПАПКУ)  
        End Select  
    End With  
    If Закрытие Then  
        Set ФВ = Nothing  
    End If  
End Sub
```

```
End If
Exit Sub
```

## Продолжение ПРИЛОЖЕНИЯ А

### Листинг программы «Формирование отчетов о работе цифровых передатчиков»

```
ЕслиОшибка:
Debug.Print "ОШИБКА #" & Str(Err.Number) & ": " & Err.Source & _
Chr(13) & Err.Description

If Not ФВ Is Nothing And ФреймворкЗапущен Then
With ФВ
.СледующаяСтраница
.ЗначениеЭлементаПоИмени("Этап") = "<fontsize=2 color=red>" & _
" <b>Ошибка выполнения программы</b></font>"
.ЗначениеЭлементаПоИмени("Готово") = "Завершить"
.АктивироватьЭлемент("Готово") = .ДА
End With
End If
End Sub

Sub ЗадатьПрогресс(Значение As Integer, Optional Этап As String = "")
Dim Процент As Integer
With ФВ
Процент = .ЗначениеЧислоЭлементаПоИмени("Прогресс") + Значение
.ЗначениеЧислоЭлементаПоИмени("Прогресс") = Процент
.ЗначениеЭлементаПоИмени("Ожидание") = "Создание отчетов: " & Процент & "%"
If Этап <> "" Then
.ЗначениеЭлементаПоИмени("Этап") = "<font size=2>" & Этап & "</font>"
End If
If Процент >= 100 Then
.АктивироватьЭлемент("Готово") = .ДА
.ЗначениеЭлементаПоИмени("Ожидание") = "Отчет готов!"
.ЗначениеЭлементаПоИмени("Этап") = ""
End If
End With
DoEvents
End Sub

Sub СоздатьОтчет(ByVal ПутьКТаблице As String, ByVal ПутьКПапкеОтчетов As String,
ByVal Дежурный As String, ByVal Должность As String)
Dim ГО As ГенераторОтчета
Set ГО = New ГенераторОтчета
With ГО
'.ОтладочныйРежим = .ДА
.СообщениеОтладка "Начало выполнения скрипта" '###
ПутьКТаблице = .УбратьПробелы(ПутьКТаблице)
ПутьКПапкеОтчетов = .ЗамениТЬ(.УбратьПробелы(ПутьКПапкеОтчетов), _
.Символ(0), "")
```

If Not .СимволыСправа(ПутьКПапкеОтчетов, 1) = "\" Then  
ПутьКПапкеОтчетов = ПутьКПапкеОтчетов & "\"

## Продолжение ПРИЛОЖЕНИЯ А

### Листинг программы «Формирование отчетов о работе цифровых передатчиков»

```
End If
ЗадатьПрогресс 0, "Начало" '<---
.СоздатьВиртуальнуюТаблицу
.ПоказатьВиртуальнуюТаблицу
'---Загрузка дополнительной таблицы подстановки---
ЗадатьПрогресс 5, "Загрузка таблицы " & ПапкаШаблонов & "передатчики.xlsx" '<---
.ДобавитьВиртЛист "передатчики"
.ВыбратьВиртЛист "передатчики"
.ОткрытьТаблицу ПапкаШаблонов & "передатчики.xlsx"
ЗадатьПрогресс 5, "Копирование данных из таблицы " & ПапкаШаблонов & _
"передатчики.xlsx" '<---

.ФиксСтроки = 1
.КопироватьАктивныйЛист
.ЗакрытьТаблицу
.ВыбратьВиртЛист "Лист1"
.ОтобразитьАктивВиртЛист
'---Загрузка таблицы с данными---
ЗадатьПрогресс 5, "Загрузка таблицы " & ПутьКТаблице '<---
.ОткрытьТаблицу ПутьКТаблице
ЗадатьПрогресс 2, "Копирование данных из таблицы " & ПутьКТаблице '<---
.СообщениеОтладка "Копирование таблицы" ###
.ФиксСтроки = 1
.КопироватьСтолбец "A"
.КопироватьСтолбец "B"
ЗадатьПрогресс 2 '<---
.КопироватьСтолбец "D"
.КопироватьСтолбец "F"
ЗадатьПрогресс 2 '<---
.КопироватьСтолбец "I"
.КопироватьСтолбец "J"
ЗадатьПрогресс 2 '<---
.КопироватьСтолбец "K"
.КопироватьСтолбец "N"
.КопироватьСтолбец "O"
.ЗакрытьТаблицу
'---Редактирование таблиц---
ЗадатьПрогресс 2, "Разделение строк по передатчикам" '<---
.СообщениеОтладка "Редактирование виртуальной таблицы" ###
РазбитьНаНесколькоГО, "Н", "ВГТРК-МУХ Амурский"
.ФорматЯчеек("K:K; J:J") = "DD.MM.YYYY hh:mm:ss"
.ФормулаВСтолбце("J") = "=ЕСЛИ(ЕСЛИОШИБКА(ПОИСК("от ИБП " & _
"(";I1);0)>0;ДАТА(ГОД(E1);МЕСЯЦ(E1);ДЕНЬ(E1)+ " & _
"ЕСЛИ(ВРЕМЗНАЧ(ЛЕВСИМВ(ПРАВСИМВ(I1;10);8)) <" & _
```

"ВРЕМЯ(ЧАС(E1);МИНУТЫ(E1);СЕКУНДЫ(E1)); 1;0)+" & \_  
 "ВРЕМЗНАЧ(ЛЕВСИМВ(ПРАВСИМВ(11;10);8));" & \_  
 Продолжение ПРИЛОЖЕНИЯ А

Листинг программы «Формирование отчетов о работе цифровых  
 передатчиков»

```

"ДАТАЗНАЧ(E1)+ ВРЕМЗНАЧ(E1))"
.ФормулаВСтолбце("К") = "=ЕСЛИ(ЕСЛИОШИБКА(ДАТАЗНАЧ(G1)+" & _
"ВРЕМЗНАЧ(G1);G1) <J1; ЕСЛИОШИБКА(ДАТАЗНАЧ(G1)+" & _
"ВРЕМЗНАЧ(G1);G1)-ЕСЛИОШИБКА(ДАТАЗНАЧ(E1)+" & _
"ВРЕМЗНАЧ(E1);E1); ЕСЛИОШИБКА(ДАТАЗНАЧ(G1)+" & _
"ВРЕМЗНАЧ(G1);G1)-J1)" "="G1-J1"
ЗадатьПрогресс 5, "Фильтрация таблицы по времени " & ">10 мин" '<---
.ФильтроватьТаблицу .БукваВНомерСтолбца("К"), ">" & .МинутыТехВремя(10)
Dim СписокСовпадений1, СписокСовпадений2 As Variant
СписокСовпадений1 = .Массив("Неисправность СЖО передатчика", "Отключение элект-
роэнергии по вине энергоснабжающей организации", "Неисправность передатчика")
СписокСовпадений2 = .Массив("", "", "Восстановление нормальной работы передатчика
после подачи электроэнергии на входе станции.")
ЗадатьПрогресс 5, "Сортировка таблицы" '<---
.ИсключитьСтроки "F", СписокСовпадений1, "I", СписокСовпадений2
.СортироватьТаблицу .ВсяТаблица, .Столбец("С"), ПО_АЛФАВИТУ, .Столбец("Е"),
ПО_ВОЗРАСТАНИЮ
ЗадатьПрогресс 5, "Подстановка в таблицу данных передатчиков" '<---
.ФорматЯчеек("G:G; Q:Q") = "DD.MM.YYYY hh:mm:ss"
.ФормулаВСтолбце("Q") = "=ЕСЛИОШИБКА(ДАТАЗНАЧ(G1)+ВРЕМЗНАЧ(G1);G1)"
.ФормулаВСтолбце("L") = "=ЕСЛИ(ПОИСК("";C1 & ""("") < ДЛСТР(C1);" & _
"ЛЕВСИМВ(C1;(ПОИСК("";C1))-1); C1)"
.ПодставитьИзТаблицы .НазваниеВиртАктивногоЛиста, "передатчики", _
.Массив("М", "N", "P"), .Массив("D", "B", "E"), "L", "A", "H", "C"
For i = 1 To .КоличествоСтрок
.ЗначениеВЯчейке("O" & i) = i
Next i
'---Подстановка в шаблон---
ЗадатьПрогресс 5, "Добавление соответствий данных в шаблонах" '<---
.ДобавитьСоответствие "#Номер", "O"
.ДобавитьСоответствие "#Предприятие", "B"
.ДобавитьСоответствие "#Населенный_пункт", "P"
.ДобавитьСоответствие "#Оборудование", "N"
.ДобавитьСоответствие "#Программа", "H"
.ДобавитьСоответствие "#Причина_перерыва", "F"
.ДобавитьСоответствие "#Дата_начала", "J"
.ДобавитьСоответствие "#Время_начала", "J"
.ДобавитьСоответствие "#Продолжительность", "K", Формат:="hh:mm:ss"
.ДобавитьСоответствие "#Дата_окончания", "Q"
.ДобавитьСоответствие "#Время_окончания", "Q"
.ДобавитьСоответствие "#Охват_населения", "M"
.ДобавитьСоответствие "#Город", "L"
'---Создание отчета Эксель---

```

ЗадатьПрогресс 5, "Открытие шаблона отчета Эксель " & ПапкаШаблонов & \_  
ПутьКШаблонуЭксель '<---

## Продолжение ПРИЛОЖЕНИЯ А

Листинг программы «Формирование отчетов о работе цифровых  
передатчиков»

```
.ОткрытьФайлШаблонаЭксель ПапкаШаблонов & ПутьКШаблонуЭксель
.ПоказатьШаблонЭксель
ЗадатьПрогресс 5, "Подстановка данных в шаблон отчета Эксель" '<---
.ПодставитьВШаблонЭксельТаблицу
ЗадатьПрогресс 10, "Сохранение отчета Эксель " & ПутьКПапкеОтчетов & _
ПутьКОтчетуЭксель & .Дата & ".xlsx" '<---
.СохранитьШаблонЭксельКак ПутьКПапкеОтчетов & ПутьКОтчетуЭксель & _
.Дата & ".xlsx", XLSX

.ЗакрытьФайлШаблонаЭксель
'---Создание отчета Ворд---
ЗадатьПрогресс 5, "Открытие шаблона отчета Ворд " & ПапкаШаблонов & _
ПутьКШаблонуВорд '<---

.ОткрытьФайлШаблона ПапкаШаблонов & ПутьКШаблонуВорд
.ПоказатьШаблон
ЗадатьПрогресс 10, "Подстановка данных в шаблон отчета Ворд" '<---
.ПодставитьВШаблонТаблицу
ЗадатьПрогресс 10 '<---
.ПодставитьВШаблон "$Дежурный_ФИО", Дежурный
.ПодставитьВШаблон "$Должность", Должность
ЗадатьПрогресс 5, "Сохранение отчета Ворд " & ПутьКПапкеОтчетов & _
ПутьКОтчетуВорд & .Дата & ".docx"
.СохранитьШаблонКак ПутьКПапкеОтчетов & ПутьКОтчетуВорд & _
.Дата & ".docx", DOCX

.ЗакрытьФайлШаблона
.ЗакрытьВорд
ЗадатьПрогресс 5 '<---
.ЗакрытьВиртуальнуюТаблицу
.СообщениеОтладка "Окончание выполнения скрипта" '###
End With
Set ГО = Nothing
End Sub

Sub ЗагрузитьСведенияОДежурных()
Dim i As Integer
Dim n As Integer
Dim ГО As ГенераторОтчета
SetГО = NewГенераторОтчета
With ГО
.ВыбратьАктивноеПриложение
.ВыбратьЛист "Дежурные"
n = .КоличествоСтрок(Лист:=.АктивныйЛист) - 1
ReDim ФИО(n)
ReDim Должность(n)
```

ФИО(0) = "Выберете дежурного"

Должность(0) = "..."

## Продолжение ПРИЛОЖЕНИЯ А

### Листинг программы «Формирование отчетов о работе цифровых передатчиков»

```
For i = 1 To n
    ФИО(i) = .ЗначениеВЯчейке(.НомерВБуквуСтолбца(1) & i + 1, _
        Лист:=.АктивныйЛист)
    Должность(i) = .ЗначениеВЯчейке(.НомерВБуквуСтолбца(2) & i + 1, _
        Лист:=.АктивныйЛист)
Next i
End With
Set ГО = Nothing
End Sub

SubРазбитьНаНесколько(ГОAsГенераторОтчета, СтолбецAsString, _
    Исключить As Variant)
    Dim i, j, k As Integer
    Dim m As Variant
    Dim МассивСтрок As Variant
    Dim Строка, Строка2 As String
    Dim Совпадение As Boolean
    With ГО
        If Not .ЭтоМассив(Исключить) Then
            Исключить = .Массив(Исключить)
        End If
    End With
    For i = .КоличествоСтрокTo 1 Step -1
        МассивСтрок = .РазбитьСтроку(.ЗамениТЬ(.ЗначениеВЯчейке(Столбец & i), _
            ",", ";"), ";")
        If .Длина(МассивСтрок) > 1 Then
            k = 0
            For j = 0 To .Длина(МассивСтрок) - 1
                Строка = МассивСтрок(j)
                Совпадение = .НЕТ
                For Each m In Исключить
                    If .УбратьПробелы(Строка) = m Then Совпадение = .ДА
                Next
                If Not Совпадение And .Длина(.УбратьПробелы(Строка)) > 0 Then
                    If k > 0 Then
                        .ВставитьПустуюСтроку i
                        .ЗначениеВСтроке(i) = .ЗначениеВСтроке(i + 1)
                    End If
                    .ЗначениеВЯчейке(Столбец & i) = .УбратьПробелы(Строка)
                End If
            Next j
            k = k + 1
        End If
    Next j
End For
Next i
```

End With  
End Sub