

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет
вычислительной техники

Кафедра
МОиПЭВМ

Направление подготовки 09.03.04 «Программная инженерия»

БАКАЛАВРСКАЯ РАБОТА

на тему

**Программный имитатор оптико-электронных
средств наблюдения**

Студент	_____	Варакин Андрей Алексеевич
	(подпись, дата)	(ФИО полностью)
Руководитель	_____	Князев В. Н.
	(подпись, дата)	(фамилия, инициалы)
Нормоконтролёр	_____	Такташкин Д.В.
	(подпись, дата)	(фамилия, инициалы)

Работа допущена к защите (протокол заседания кафедры от _____ № _____)

Заведующий кафедрой	_____	Макарычев П.П.
	(подпись)	(фамилия, инициалы)

Работа защищена с отметкой _____ (протокол заседания ГЭК от _____ № _____)

Секретарь ГЭК	_____	Попова Н.А.
	(подпись)	(фамилия, инициалы)

Пенза, 2020

Федеральное государственное бюджетное образовательное учреждение
высшего образования
ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет
Вычислительной техники

Кафедра
МО и ПЭВМ

«Утверждаю»
Заведующий кафедрой
_____ П.П. Макарычев
«30» декабря 2019 г

ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
БАКАЛАВРА

1. Студент Варакин Андрей Алексеевич гр. 16ВП1 факультета ВТ
<Фамилия, имя, отчество полностью>
2. Тема работы Программный имитатор оптико-электронных средств наблюдения

Тема утверждена приказом ПГУ № 1488/о от "30" декабря 2019

3. Руководитель работы к.т.н., доцент кафедры «МОиПЭВМ» Князев В.Н.

4. Задание на работу (назначение разработки, исходные данные и т.п.)

Назначение: Автоматизация процесса разработки и тестирования специального программного обеспечения (СПО) с минимизацией привлечения реальных образцов оптико-электронных систем наблюдения (ОЭСН), уменьшение затрат на разработку, а также возможность вести разработку и вести разработку и тестирования СПО с использование данных от ОЭСН по его протоколу информационно технического взаимодействия и обучение будущих операторов.

Среда выполнения: Astra Linux Common Edition («Орел») v2.12 и выше.

Языковые средства разработки: C++, UML.

Инструментальные средства разработки: Qt 5.14, Visio 2016, Microsoft Project 2019, Doxygen, Umbrello.

Технологии разработки: RUP, ООП.

Основные функции:

а) Имитация работы с устройством

б) Работа с настройками устройства

в) Обучение будущих операторов

г) Просмотр информации по устройству

д) Имитация устройства для разработки программного обеспечения

5. Перечень подлежащих разработке вопросов

1) Анализ предметной области и постановка задачи

2) Проектирование

3) Реализация

4) Контроль качества программного обеспечения

5) Оформление пояснительной записки

6. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1	<i>Анализ предметной области постановка задачи</i>	15%	13.04.2020– 21.04.2020	
2	<i>Проектирование</i>	25%	22.04.2020– 05.05.2020	
3	<i>Реализация</i>	40%	06.05.2020– 30.05.2020	
4	<i>Контроль качества программного обеспечения</i>	5%	31.05.2020– 02.06.2020	
5	<i>Оформление пояснительной записки</i>	15%	03.06.2020– 13.06.2020	

Дата выдачи задания "30" _____ декабря _____ 2019

Руководитель бакалаврской работы _____
(подпись, дата)

Задание к исполнению принял студент _____
(подпись, дата)

Работу к защите допустить
Декан факультета _____
(подпись, дата)

Реферат

Пояснительная записка содержит 133 листов, 32 рисунков, 6 таблиц, 20 использованных источника, 6 приложения.

ПРОГРАММНЫЙ ИМИТАТОР ОПТИКО-ЭЛЕКТРОННЫХ СРЕДСТВ
НАБЛЮДЕНИЯ, РАЗРАБОТКА ПО, ОБУЧЕНИЕ, СНИЖЕНИЕ ЗАТРАТ,
RUP, C++, DOXYGEN.

Объектом разработки является программный имитатор оптико-электронных систем наблюдения.

Цель работы – разработка программного имитатора оптико-электронных средств наблюдения, который позволяет ускорить разработку программного обеспечения для устройства и обеспечить обучение операторов.

Результатом работы является программный имитатор оптико-электронных систем наблюдения.

Технология разработки – ООП, RUP.

Разработка программного обеспечения проводилась на языке программирования C++ в среде QT5.14 в операционной системе Astra Linux Common Edition («Орел»).

					ПГУ 09.03.04 – 08ВР161.06 ПЗ			
Изм.	Лист	№ докум.	Подп.	Дата				
Разраб.		Варакин А.А			«Программный имитатор оптико-электронных средств наблюдения» Пояснительная записка	Лит.	Лист	Листов
Пров.		Князев В.Н					4	133
Реценз.						гр. 16ВП1		
Н. контр.		Такташкин Д.В						
Утв.		Макарычев П.П						

Содержание

Введение.....	7
1 Анализ предметной области и требований.....	8
1.1 Анализ предметной области.....	8
1.2 Сравнительный анализ программных продуктов-аналогов.....	9
1.3 Анализ требований для программного средства	10
1.3.1 Функциональные требования.....	11
1.3.2 Нефункциональные требования	15
2 Проектирование программного имитатора	17
2.1 Выбор технологий и инструментальных средств разработки	17
2.2 Планирование разработки и оценка бюджета	17
2.3 Проектирование интерфейса пользователя	27
3 Реализация программных средств имитатора	33
4 Тестирование и анализ качества программного кода имитатора.....	36
4.1 Тестирование программного обеспечения	36
4.1.1 Функциональное тестирование.....	36
4.2 Метрики кода.....	39
4.2.1 Комплексный показатель качества кода (Maintainability Index)	39
4.2.2 Метрика Холстеда.....	43
Заключение	45
Список использованных источников	46
Приложение А. Глоссарий	49
Приложение Б. Исходный код программы.....	52
Приложение В. Результаты тестирования	119
Приложение Г. Страницы интерфейса.....	124

Приложение Д. Изображения аналогов	130
Приложение Е. Акт о внедрении результатов.....	132

Введение

В последнее время информационные технологии внедряются во все сферы человеческой жизни. Все страны мира уделяют этому сектору огромное внимание, Россия также не является исключением.

В настоящее время вопрос производства комплекса оптико-электронных средств наблюдения (ОЭСН) для вооружённых сил российской федерации (ВСРФ) стоит очень остро. Комплексы ОЭСН активно внедряются в ВСРФ.

Комплекс ОЭСН представляет собой совокупность систем, устройств и оборудования, предназначенные для ведения обзора местности в дневном и ночном (тепловизионном) канале, обнаружения и распознавания объектов, определения координат объектов, ориентиров и местных предметов. Эти комплексы очень сложны в производстве и требуют большого финансирования. Исходя из этого, целесообразно создать программный имитатор, который в точности повторял бы характеристики и возможности комплекса ОЭСН.

Имитатор создаётся с целью минимизации затрат производства, обучения операторов комплексов ОЭСН, создания и тестирования программного обеспечения (ПО) для аппаратных комплексов.

Согласно Указу Президента Российской Федерации от 9 мая 2017 г. № 203 «О стратегии развития информационного общества в Российской Федерации на 2017 – 2030 годы» и Распоряжению Правительства Российской Федерации от 28 июля 2017 г. № 1632-р «Цифровая экономика Российской Федерации» следует, что в целях обеспечения национальных интересов необходимо сформировать новую технологическую основу для развития экономики [1, 2].

Исходя из перечисленной информации целью выпускной квалификационной работы было выбрано актуальное направление по созданию программного имитатора оптико-электронных средств наблюдения для того, чтобы существенно сократить затраты на производство программного обеспечения для комплекса ОЭСН и обучения операторов.

1 Анализ предметной области и требований

1.1 Анализ предметной области

Назначением средств имитации аппаратно-программных комплексов средств автоматизации (АП КСА) является автоматизация процесса разработки и тестирования специального программного обеспечения с минимизацией привлечения натуральных АП КСА, а именно - уменьшение затрат на работу программистов в изделии, а также возможность вести разработку и тестирования СПО с использованием данных от АП КСА до момента готовности и поставки образца АП КСА потребителю (разработчику СПО), что в условиях применения разрабатываемых перспективных образцов АП КСА может составлять существенное время. Кроме того, данное программное обеспечение может быть использовано для обеспечения проверок и стыковок, а также обучения в условиях стенда, без привлечения натуральных АП КСА.

Оптико-электронная система наблюдения (ОЭСН) предназначена для:

- ведения обзора местности в дневном и ночном (тепловизионном) канале;
- обнаружения и распознавания наземных и воздушных объектов;
- определения координат объектов, ориентиров и местных предметов.

Средства имитации АП КСА включаются в систему на уровне реализации транспорта с натурными АП КСА. Данный подход обеспечивает возможность отладки как собственно высокоуровневого СПО, так и модулей взаимодействия с АП КСА. Для протоколов взаимодействия основанных на UDP или TCP/IP взаимодействии, реализован набор самостоятельных приложений, использующих UDP или TCP/IP соединение и реализующих протокол взаимодействия с АП КСА на стороне имитируемого устройства. Для устройств, подключаемых по шине CAN, данный уровень абстракции реализован как библиотека сервиса доступа к

шине CAN, имитирующая обмен по шине CAN используя UDP. Взаимодействующим компонентом в данном случае является приложение «Эмулятор CAN сети».

Любой имитируемый объект управления (ОУ) может быть описан как набор АП КСА, с которых он получает данные. С применением средств имитации АП КСА можно создать как точное имитирование существующего ОУ (строгое перечисление АП КСА), так и любой абстрактный ОУ (по набору АП КСА) в соответствии с обрабатываемой задачей.

CAN является синхронной шиной с типом доступа Collision Resolving (CR, разрешение коллизии), который, в отличие от Collision Detect (CD, обнаружение коллизии) сетей (Ethernet), детерминировано (приоритетно) обеспечивает доступ на передачу сообщения, что особо ценно для промышленных сетей управления (fieldbus). Передача ведётся кадрами. Полезная информация в кадре состоит из идентификатора длиной 11 бит (стандартный формат) или 29 бит (расширенный формат, надмножество предыдущего) и поля данных длиной от 0 до 8 байт. Идентификатор говорит о содержимом пакета и служит для определения приоритета при попытке одновременной передачи несколькими сетевыми узлами [3].

1.2 Сравнительный анализ программных продуктов-аналогов

Оптико-электронные имитаторы средств наблюдения являются узко направленными и специальными средствами для вооружённых сил разных стран в следствии чего был проведён анализ доступных источников и было выявлено, что в открытом доступе нет никакой информации о таких программных средствах. Исходя из этого, было принято решение создать собственный программный имитатор для оптико-электронных средств наблюдения.

Ниже представлены два оптико электронных комплекса «Ирония» и «Адунок» схожих по характеристикам с устройством, для которого был разработан программный имитатор.

Многофункциональный оптико-электронный комплекс «Ирония» предназначается для ведения наблюдения и сбора информации о местности, в том числе с обнаружением различных объектов. Обеспечивается наблюдение в любое время суток с возможностью определения координат цели и передачей данных о ней. Проектом предусматривается создание двух версий комплекса: монтируемой на технике и переносной. Также в состав комплекса входят дополнительные средства обнаружения, позволяющие защитить разведчиков от внезапного нападения противника [4].

Адунок – белорусский автоматизированный дистанционно-управляемый наблюдательно-огневой комплекс, разработки вооружённых сил Беларуси. Он представляет собой дистанционно управляемый модуль, оснащённый камерами широкого и узкого поля зрения, тепловизором и лазерным дальномером. Огневая составляющая может комплектоваться различными типами вооружения. Вся система смонтирована на поворотной платформе, обеспечивающей круговое наблюдение и поражение целей [5].

Изображения этих комплексов находятся в приложении Д.

1.3 Анализ требований для программного средства

Анализ требований – один из важнейших этапов разработки программного обеспечения, включающий в себя сбор и систематизацию требований к будущей системе [6].

Можно выделить функциональные и нефункциональные требования к системе. Функциональные требования описывают функции, которые должна выполнять система, поведение системы при взаимодействии с ней пользователя. Нефункциональные требования описывают свойства и ограничения системы, которые не относятся к её поведению [7, 8].

1.3.1 Функциональные требования

В результате анализа предметной области и продуктов-аналогов были выделены следующие функциональные требования к программе. Для оператора были выделены следующие:

- прочитывать состояния;
- установить параметры;
- посмотреть информационный обмен.

Для сервера были выделены следующие требования:

- прочитывать состояния;
- установить параметры;
- посмотреть информационный обмен.

Диаграмма вариантов использования представлена на рисунке 1.

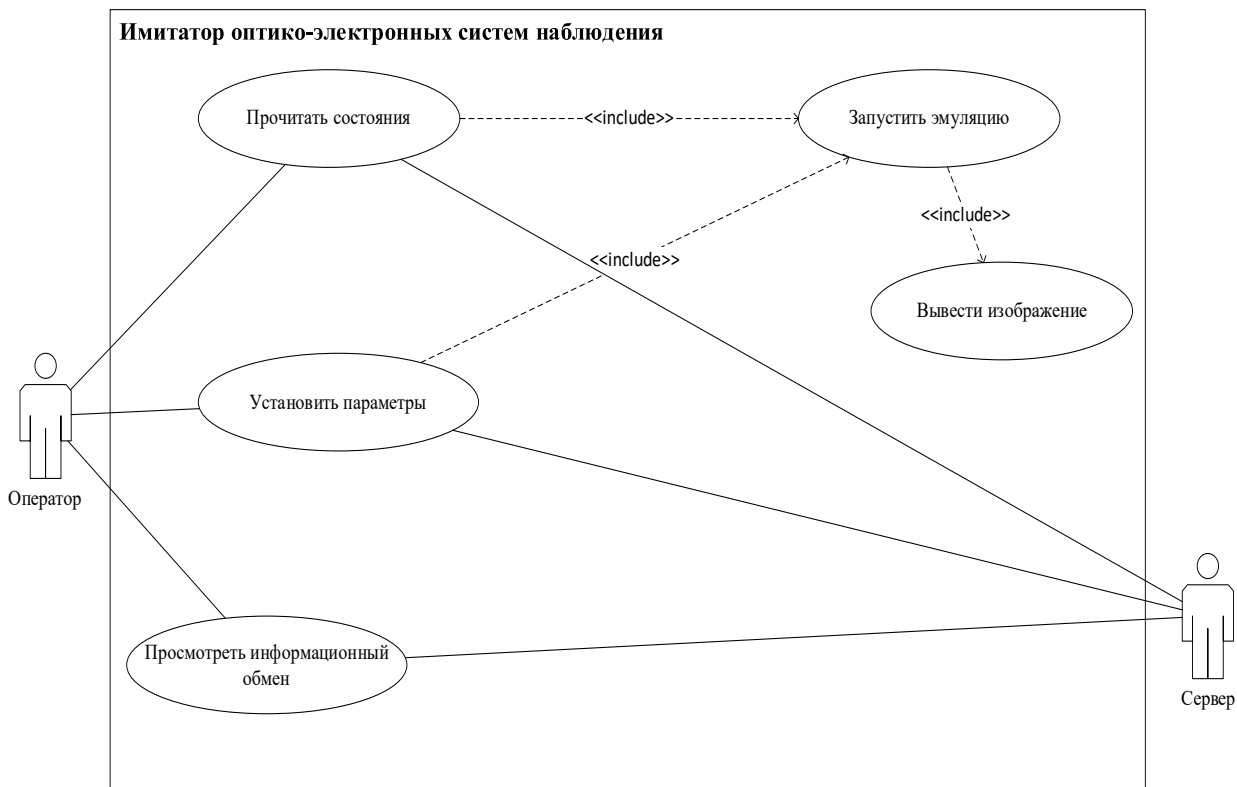


Рисунок 1 – Диаграмма вариантов использования для оператора и сервера
ОЭСН

Рассмотрим некоторые прецеденты более подробно, для этого приведем их подробную спецификацию на рисунках 2 и 3.

Прецедент: Запустить эмуляцию
ID: 1
Краткое описание: Оператор ОСЭН перед работой включает имуляцию и осуществляет последующую ее настройку
Главные актёры: <ol style="list-style-type: none"> 1. Оператор 2. Сервер
Второстепенные актёры: Нет
Предусловия: нет
Основной поток: <ol style="list-style-type: none"> 1. Оператор запускает эмуляцию 2. Сервер производит чтение сохранённого состояния 3. Оператор выставляет нужные состояния 4. Оператор выводит изображение
Постусловия: Нет
Альтернативные потоки: Нет.

Рисунок 2 – Спецификация варианта использования «Запустить эмуляцию»

Прецедент: Установить параметры
ID: 2
Краткое описание: Оператор выбирает определённые состояния для работы имитатора
Главные актёры: <ol style="list-style-type: none"> 1. Оператор 2. Сервер
Второстепенные актёры: Нет
Предусловия: нет
Основной поток: <ol style="list-style-type: none"> 1. Оператор устанавливает нужный режим работы 2. Оператор выставляет нужные состояния 3. Сервер производит чтение состояния
Постусловия: Нет
Альтернативные потоки: Нет.

Рисунок 3 – Спецификация варианта использования «Установить параметры»

Данные спецификации описаны для вариантов использования «Запустить эмуляцию» и «Установить параметры», которые предназначены для оператора ОЭСН.

Для варианта использования «Установить параметры» составим диаграмму деятельности. На рисунке 4 приведена диаграмма деятельности для прецедента «Установить параметры». Данный вариант использования применяется оператором ОЭСН.

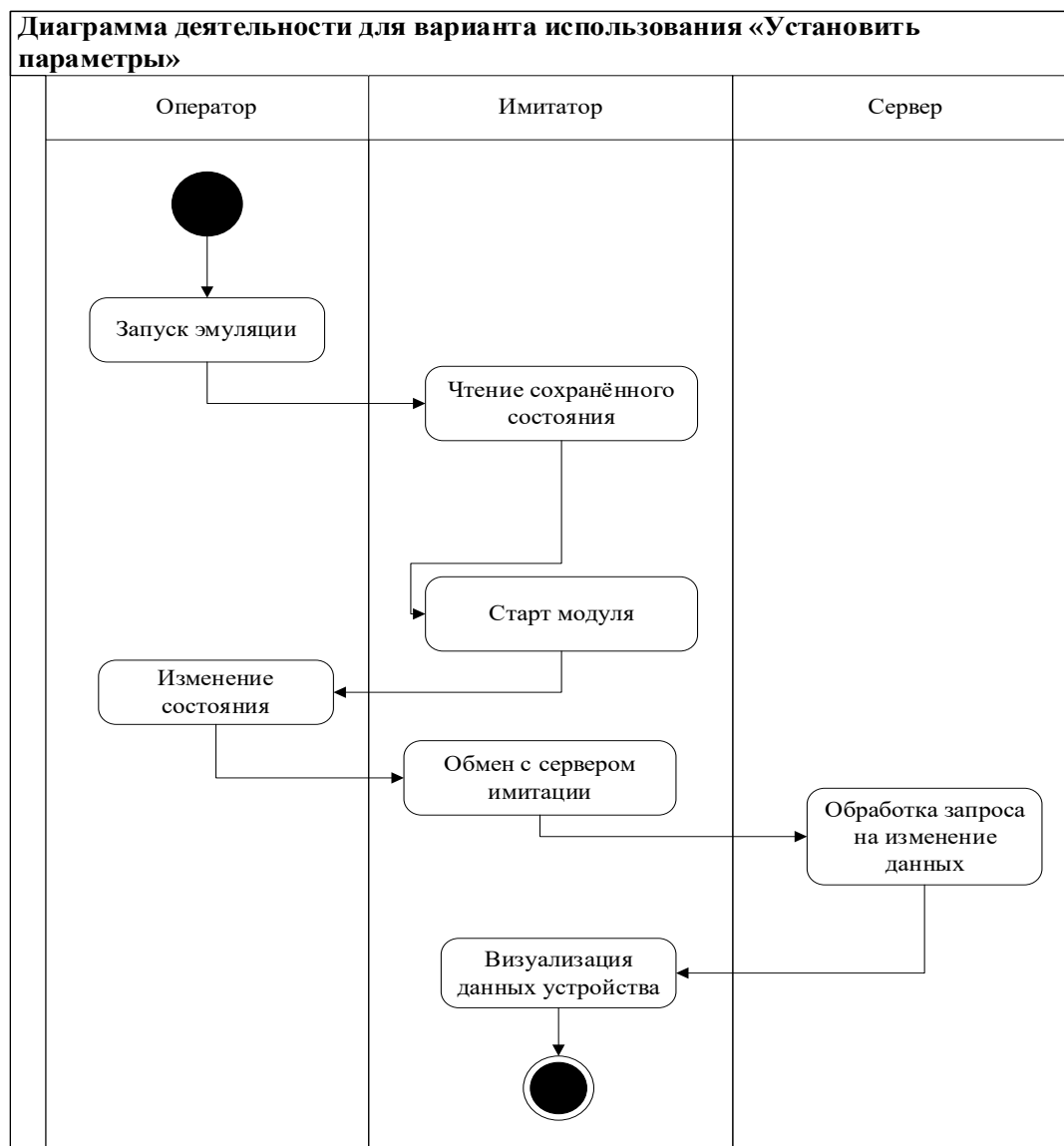


Рисунок 4 – Диаграмма деятельности для варианта использования «Установить параметры»

На рисунке 5 приведена диаграмма деятельности для прецедента «Посмотреть информационный обмен». Данный вариант использования применяется оператором ОЭСН.

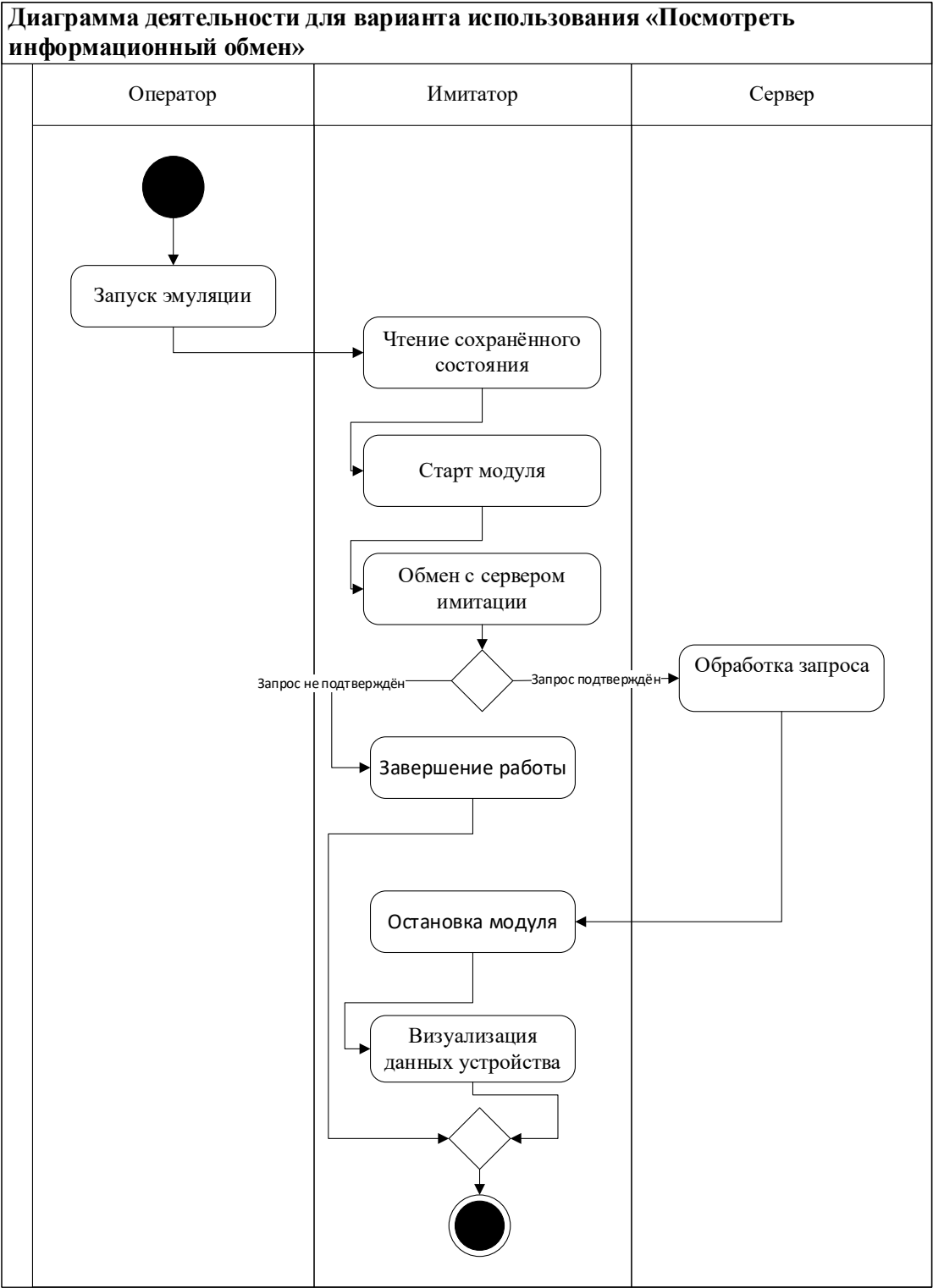


Рисунок 5 – Диаграмма деятельности для прецедента «Посмотреть информационный обмен»

На рисунке 6 представлена диаграмма последовательности для прецедента «Запустить эмуляцию».

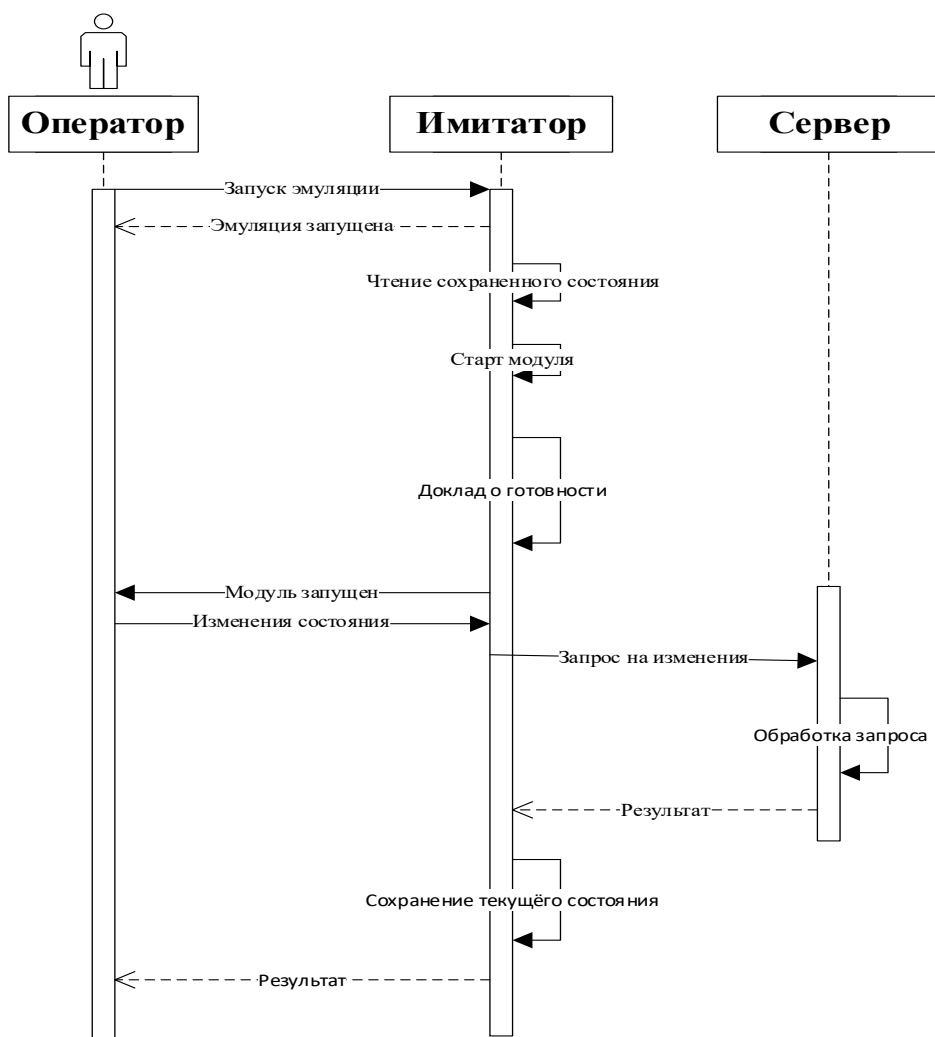


Рисунок 6 – Диаграмма последовательности для прецедента «Запустить эмуляцию»

Проанализировав функциональные требования, перейдём к нефункциональным.

1.3.2 Нефункциональные требования

Помимо функциональных требований к программному средству также можно предъявить ряд нефункциональных требований:

- операционная система (ОС): Astra Linux Common Edition («Орел») v2.12 и выше;
- язык программирования C++ стандарта C++11, компилятор g++ из состава средств разработки ОС;
- пользовательский интерфейс должен быть разработан с использованием Qt 5 из состава средств разработки ОС;

– имитатор должен быть разработан как подгружаемый модуль (plugin), функционирующий по правилам средств имитации МУ «Эмулятор CAN сети».

2 Проектирование программного имитатора

2.1 Выбор технологий и инструментальных средств разработки

Реализация приложения проводилась на основе методологии разработки Rational Unified Process (RUP). Для разработки приложения была выбрана именно эта методология, потому что она является одной из самых популярных на сегодняшний день. В рамках этой методологии используется итеративная модель разработки [9, 10]. Целью каждой итерации, размерность которых составляет от 2 до 6 недель, должно стоять получение промежуточной, но вполне функциональной версии конечного продукта [10]. Такой подход позволяет работающей над проектом команде быстро реагировать на изменения требований к продукту, с легкостью отслеживать его качество с ранних стадий разработки, а также вовремя обнаруживать и устранять любые возможные риски.

Qt – кроссплатформенный фреймворк для разработки программного обеспечения на языке программирования C++. Есть также привязки ко многим другим языкам программирования, таким как Python – PyQt, PySide, Java – Qt Jambi, PHP – PHP Qt и другие [11].

Qt позволяет запускать написанное с его помощью программное обеспечение в большинстве современных операционных систем путём простой компиляции программы для каждой системы без изменения исходного кода. Включает в себя все основные классы, которые могут потребоваться при разработке прикладного программного обеспечения, начиная от элементов графического интерфейса и заканчивая классами для работы с сетью, базами данных и XML. Является полностью объектно-ориентированным, расширяемым и поддерживающим технику компонентного программирования [11].

2.2 Планирование разработки и оценка бюджета

Управление проектами – область деятельности, в ходе которой определяются и достигаются четкие цели проекта при балансировании между

объемом работ, ресурсами, временем, качеством и рисками. Главным фактором успеха проектного управления является наличие четкого заранее определенного плана и минимизация отклонений от этого плана [12].

Для определения трудоемкости и стоимости проекта необходимо провести его грамотное планирование. Для этого был определен список задач, решение которых необходимо для создания приложения, и перечень ресурсов, с помощью которых эти задачи будут решаться.

В таблице 1 представлены задачи, представляющие из себя этапы создания проекта.

Таблица 1 – Перечень задач проекта

№	Название задачи	Длительность	Предшественники
1	Начало проекта	–	–
2	Анализ требований	–	–
3	Поиск литературы (научно технических пособий)	7 дней	1
4	Принятие окончательного решения	1 день	3
5	Анализ требований завершен	–	4
6	Выбор средств реализации проекта	–	–
7	Анализ возможных программных средств	5 дней	1
8	Выбор программного средства для реализации	2 дня	7
9	Выбор средств реализации завершен	–	8
10	Реализация проекта	–	–
11	Проектирование пользовательского интерфейса	4 дня	5;9
12	Создание математических моделей приборов	7 дней	11
13	Реализация математических моделей	12 дней	12

Продолжение таблицы 1

№	Название задачи	Длительность	Предшественники
14	Программирование сопутствующих и обслуживающих классов	19 дней	11
15	Реализация проекта завершена	–	13;14
16	Отладка	–	–
17	Тестирование работы отдельных приборов	5 дней	15
18	Тестирование работы сопутствующих классов	5 дней	15
19	Исправление ошибок	7 дней	17;18
20	Отладка завершена	–	19
21	Внедрение программного средства в общий проект	–	–
22	Внедрение программного средства	3 дня	20
23	Поиск и исправление ошибок при внедрении	4 дня	22
24	Внедрение завершено	–	23
25	Конец проекта	–	24

Из этой таблицы можно сделать вывод, что для реализации программного средства нужно пройти через решение 25 задач.

В таблице 2 представлен список затрачиваемых ресурсов, распределение которых приведено в таблице 3.

Таблица 2 – Список ресурсов проекта

Название ресурса	Тип	Стандартная ставка	Ставка сверхурочных
Программист1	Трудовой	500 Р\час	1000 Р\час
Программист2	Трудовой	500 Р\час	1000 Р\час
Программист3	Трудовой	500 Р\час	1000 Р\час
Интернет	Затраты	5000 Р\мес	

Продолжение таблицы 2

Название ресурса	Тип	Стандартная ставка	Ставка сверхурочных
Компьютер	Материальны	2000 Р	

Таблица 3 – Назначения ресурсов

№	Название задачи	Ресурс	Единицы
1	Начало проекта		
2	Анализ требований		
3	Поиск литературы (научно технических пособий)	Программист1 Компьютер Интернет	100% 1 шт
4	Принятие окончательного решения	Программист1 Программист2 Программист3	100% 100% 100%
5	Анализ требований завершен		
6	Выбор средств реализации проекта		
7	Анализ возможных программных средств	Программист2 Компьютер Интернет	100% 1 шт
8	Выбор программного средства для реализации	Программист2 Компьютер Интернет	100% 1 шт
9	Выбор средств реализации завершен		
10	Реализация проекта		
11	Проектирование пользовательского интерфейса	Программист1 Компьютер Интернет	50% 1 шт

Продолжение таблицы 3

№	Название задачи	Ресурс	Единицы
13	Реализация математических моделей	Программист1 Программист2 Программист3 Компьютер Интернет	25% 75% 75% 3 шт
14	Программирование сопутствующих и обслуживающих классов	Программист1 Компьютер Интернет	50% 1шт
15	Реализация проекта завершена		
16	Отладка		
17	Тестирование работы отдельных приборов	Программист2 Компьютер	100% 1 шт
18	Тестирование работы сопутствующих классов	Программист3 Компьютер	100% 1 шт
19	Исправление ошибок	19	Исправление ошибок
20	Отладка завершена	20	Отладка завершена
21	Внедрение программного средства в общий проект	21	Внедрение программного средства в общий проект
22	Внедрение программного средства	Программист1 Компьютер Интернет	100% 1 шт
23	Поиск и исправление ошибок при внедрении	Программист1 Компьютер Интернет	100% 1 шт
24	Внедрение завершено		
25	Конец проекта		

Из представленных таблиц можно увидеть, что решением задач будут заниматься 3 программиста, которым для этого необходимы компьютеры, доступ к интернету.

Теперь после выбора программных средств планирования можно провести все необходимые вычисления с помощью системы управления проектами Microsoft Project 2019.

На рисунке 7 изображено введенное в Project содержание таблицы 1 с вычисленными датами начала и завершения каждой задачи. За начало проекта была принята дата 19.02.2020

	Режим задачи	Название задачи	Длительность	Начало	Окончание	Предшественн	Названия ресурсов	Добавить новый столбец
1		Начало проекта	0 дней	Ср 19.02.20	Ср 19.02.20			
2		Анализ требований	8 дней	Ср 19.02.20	Пт 28.02.20			
3		Поиск литературы	7 дней	Ср 19.02.20	Чт 27.02.20	1	Программист1;Комп	
4		Принятие окончательного решения	1 день	Пт 28.02.20	Пт 28.02.20	3	Программист1;Проект	
5		Анализ требований	0 дней	Пт 28.02.20	Пт 28.02.20	4		
6		Выбор средств реализации	7 дней	Ср 19.02.20	Чт 27.02.20			
7		Анализ возможных вариантов	5 дней	Ср 19.02.20	Вт 25.02.20	1	Программист2;Комп	
8		Выбор программного обеспечения	2 дней	Ср 26.02.20	Чт 27.02.20	7	Программист2;Комп	
9		Выбор средств реализации	0 дней	Чт 27.02.20	Чт 27.02.20	8		
10		Реализация проекта	23 дней	Пн 02.03.20	Ср 01.04.20			
11		Проектирование	4 дней	Пн 02.03.20	Чт 05.03.20	5;9	Программист1[50%]	
12		Создание математической модели	7 дней	Пт 06.03.20	Пн 16.03.20	11	Программист1[50%]	
13		Реализация математической модели	12 дней	Вт 17.03.20	Ср 01.04.20	12	Программист1[25%]	
14		Программирование	19 дней	Пт 06.03.20	Ср 01.04.20	11	Программист1[50%]	
15		Реализация проекта	0 дней	Ср 01.04.20	Ср 01.04.20	13;14		
16		Отладка	12 дней	Чт 02.04.20	Пт 17.04.20			
17		Тестирование работоспособности	5 дней	Чт 02.04.20	Ср 08.04.20	15	Программист2;Комп	
18		Тестирование работоспособности	5 дней	Чт 02.04.20	Ср 08.04.20	15	Программист3;Комп	
19		Исправление ошибок	7 дней	Чт 09.04.20	Пт 17.04.20	17;18	Программист1;Проект	
20		Отладка завершена	0 дней	Пт 17.04.20	Пт 17.04.20	19		
21		Внедрение программного обеспечения	7 дней	Пн 20.04.20	Вт 28.04.20			
22		Внедрение программного обеспечения	3 дней	Пн 20.04.20	Ср 22.04.20	20	Программист1;Информационные ресурсы	
23		Поиск и исправление ошибок	4 дней	Чт 23.04.20	Вт 28.04.20	22	Программист1;Комп	
24		Внедрение программного обеспечения	0 дней	Вт 28.04.20	Вт 28.04.20	23		

Готово Новые задачи : Автоматическое планирование

Рисунок 7 – Задачи проекта, введенные в среду Microsoft Project 2019

Календарный график работ, состоящий из значков и стрелок, называется диаграммой Ганта. Значки представляют собой задачи, длина которых пропорциональна длительности работ. Стрелки показывают связи между значками – работами. Диаграмма Ганта для программного имитатора опτικο-электронных средств наблюдения представлена на рисунке 8.

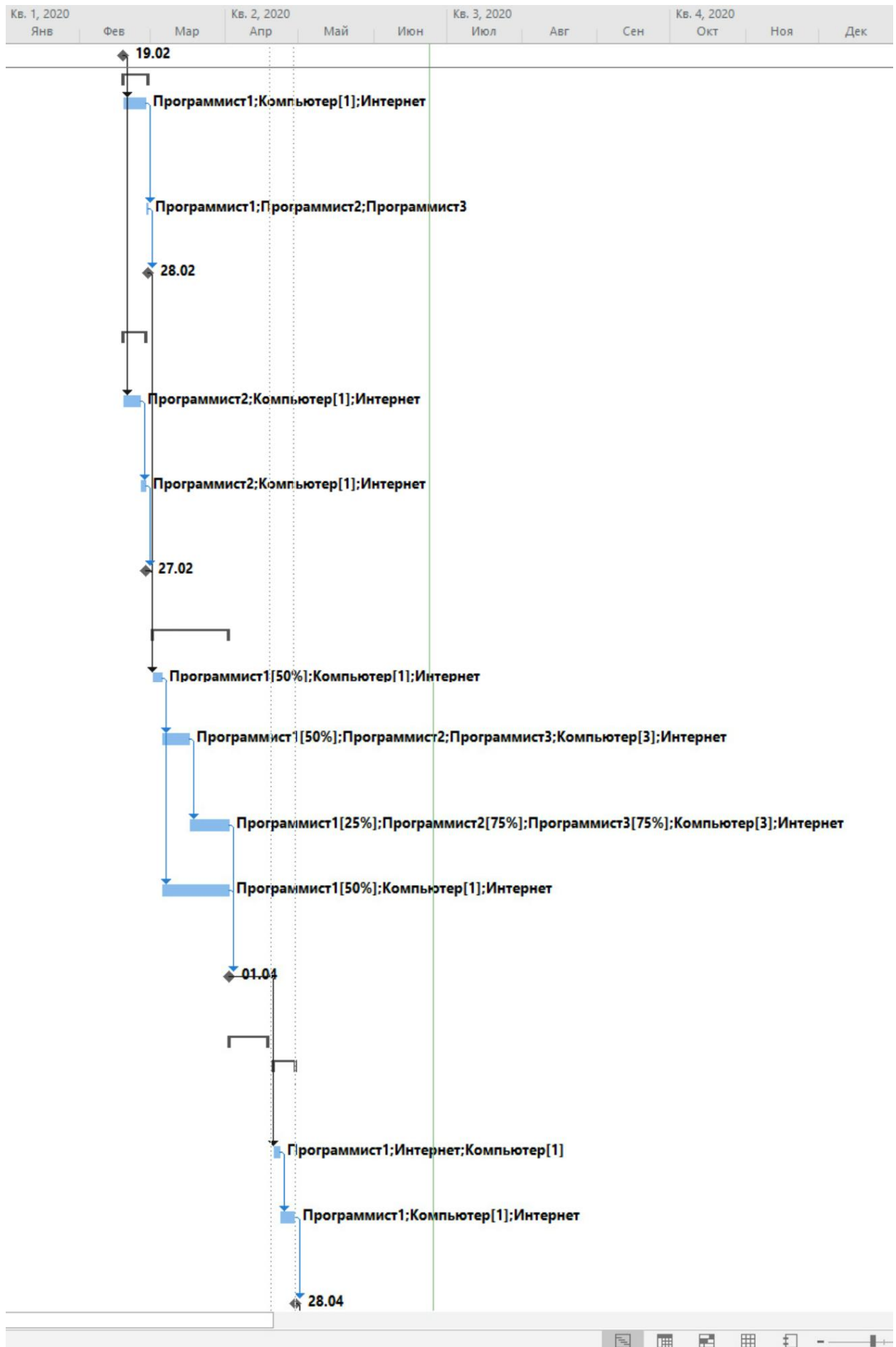


Рисунок 8 – Диаграмма Ганта

На рисунке 9 изображен обзор затрат проекта, на котором можно увидеть подсчитанную Microsoft Project 2019 длительность реализации проекта – 50 дней (что совпадает с вычисленной ранее вручную), и предполагаемые затраты – 424 000 руб.

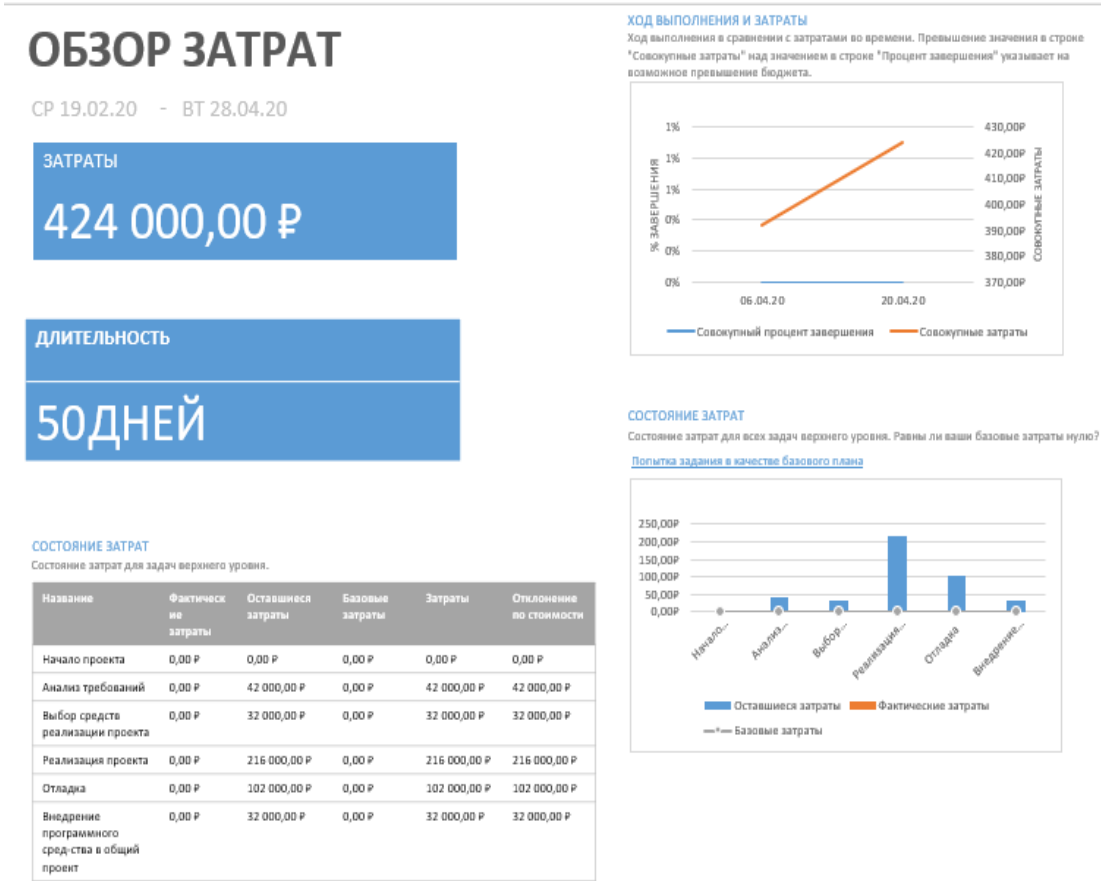


Рисунок 9 – Обзор затрат проекта

Также приведем графики ресурсов. На рисунке 10 представлен график для материального ресурса «Компьютер».

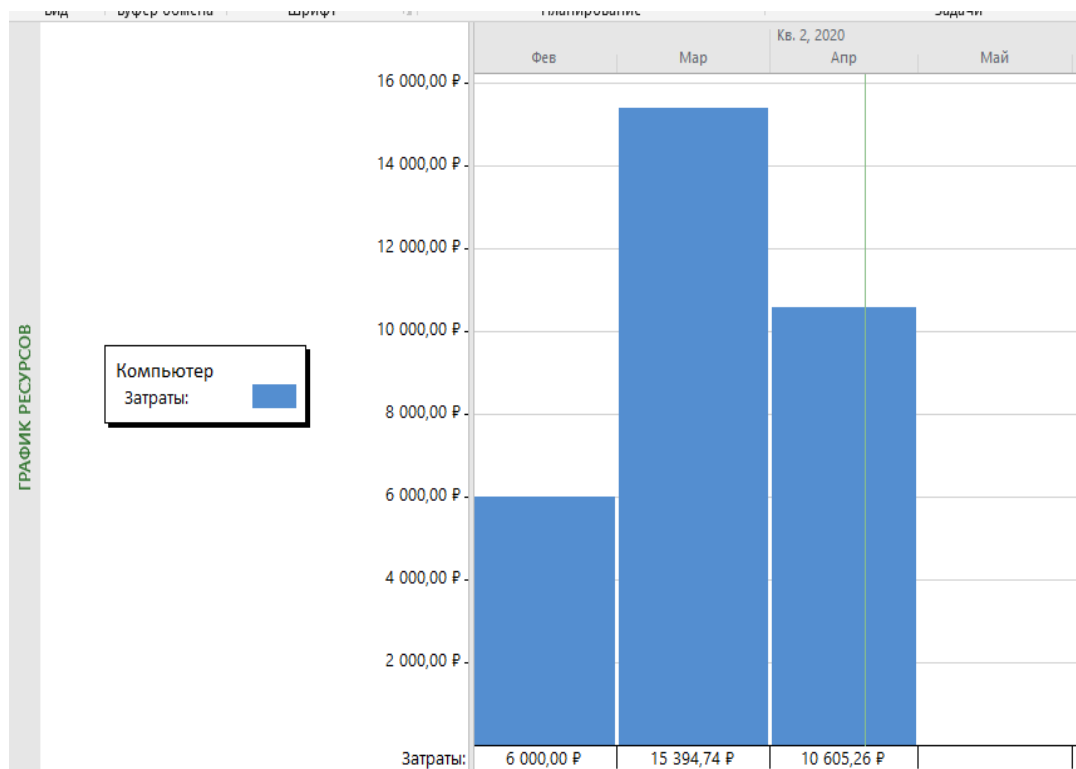


Рисунок 10 – График ресурсов для ресурса «Компьютер»

На рисунках 11 – 13 представлены графики для трудовых ресурсов «Программист1», «Программист2» и «Программист3».

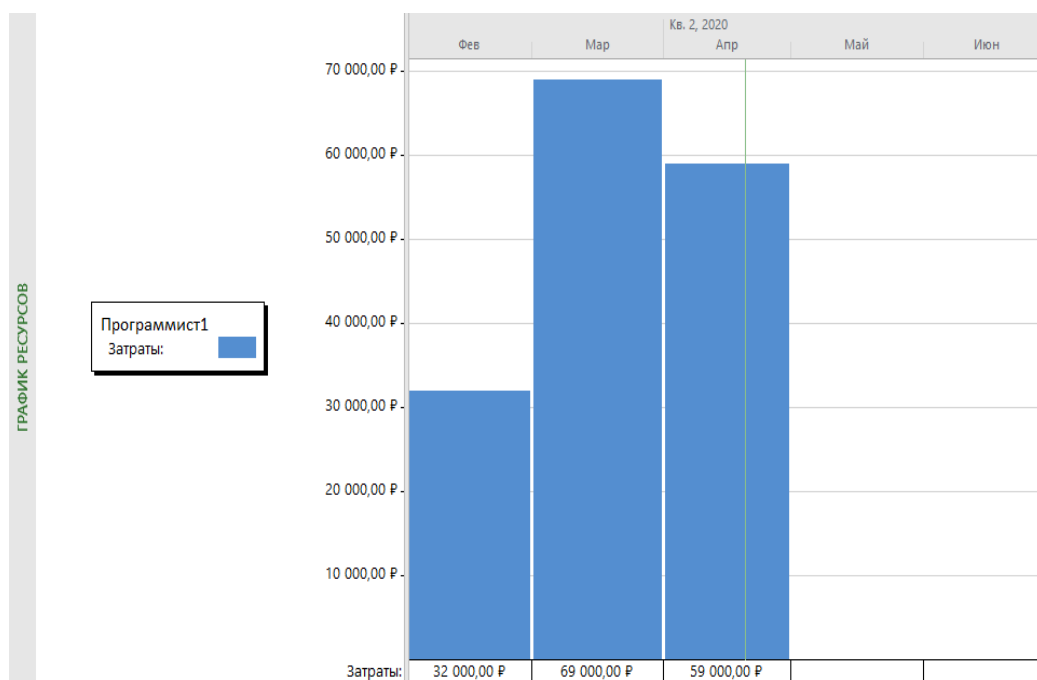


Рисунок 11 – График ресурсов для ресурса «Программист1»

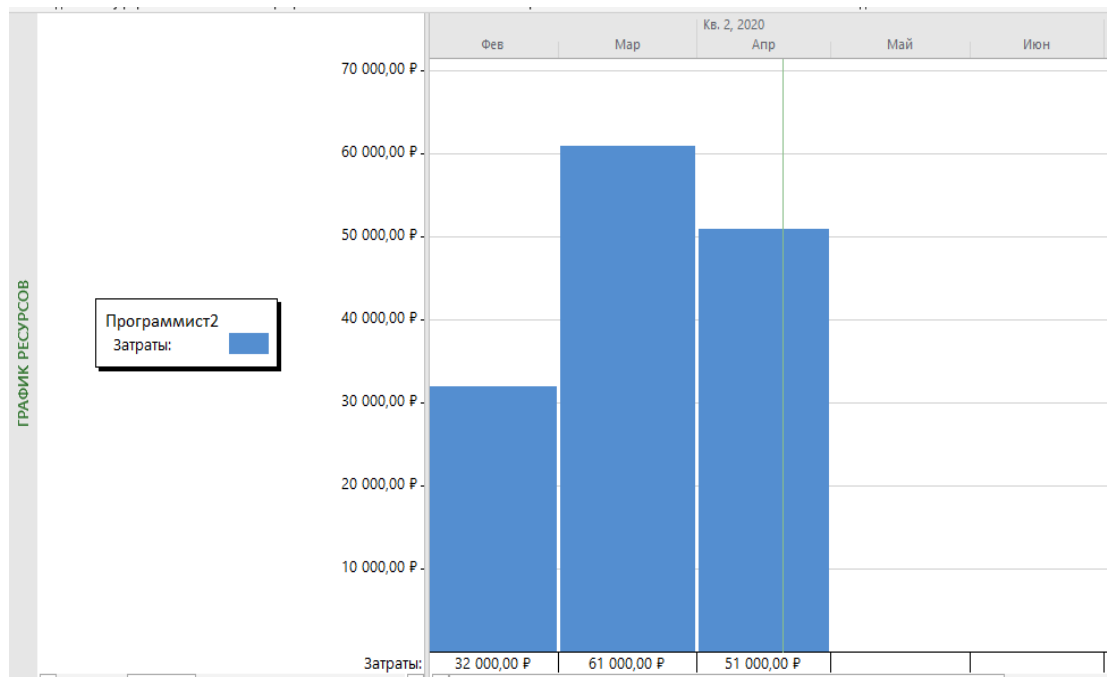


Рисунок 12 – График ресурсов для ресурса «Программист2»

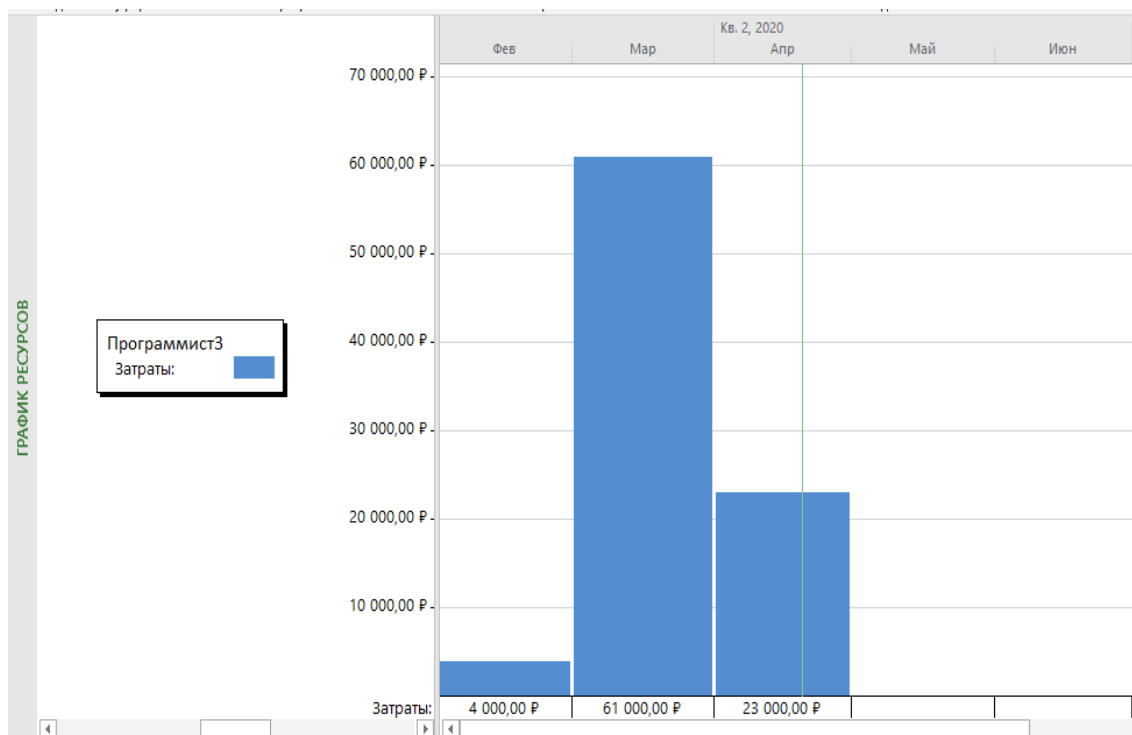


Рисунок 13 – График ресурсов для ресурса «Программист3»

Из этих рисунков видно, что нагрузка не превышает нормальную.

2.3 Проектирование интерфейса пользователя

На основе выявленных функциональных требований был спроектирован пользовательский интерфейс программы. К основным страницам программы относятся следующие:

- эмуляция CAN сети;
- справка;
- параметры;
- информационный обмен;
- отображение;
- состояние.

Рассмотрим элементы, расположенные на странице, эмуляция CAN сети (рисунок 14).

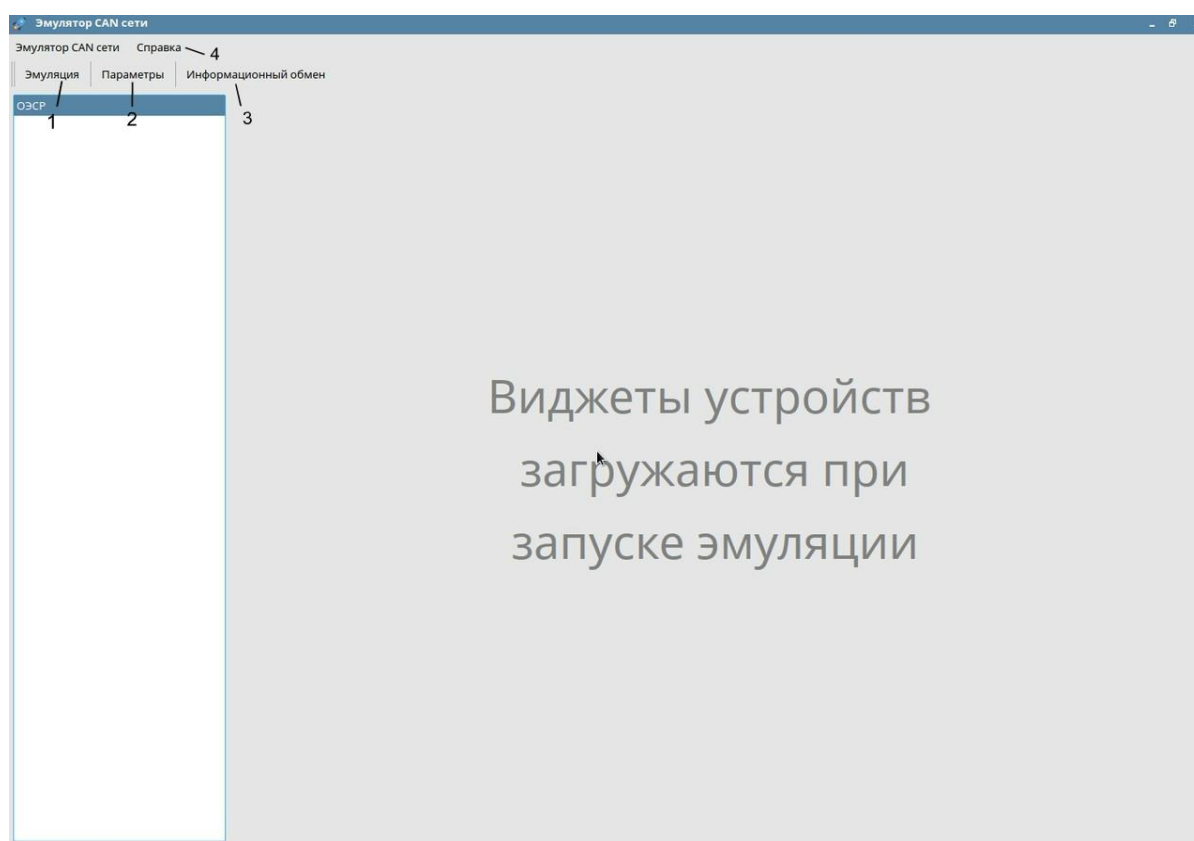


Рисунок 14 – Страница эмуляции CAN сети

Страница эмуляции CAN сети состоит из следующих компонентов:

- 1 – кнопка эмуляции;

- 2 – кнопка параметры;
- 3 – кнопка информационный обмена;
- 4 – кнопка справки.

Данная страница содержит 4 кнопки, предназначенные для перехода к функциональным возможностям программы. Нажав кнопку, отмеченную на рисунке 14 цифрой 1, оператор включает ОЭСН и попадает на страницу отображения выбранных состояний (рисунок 15)

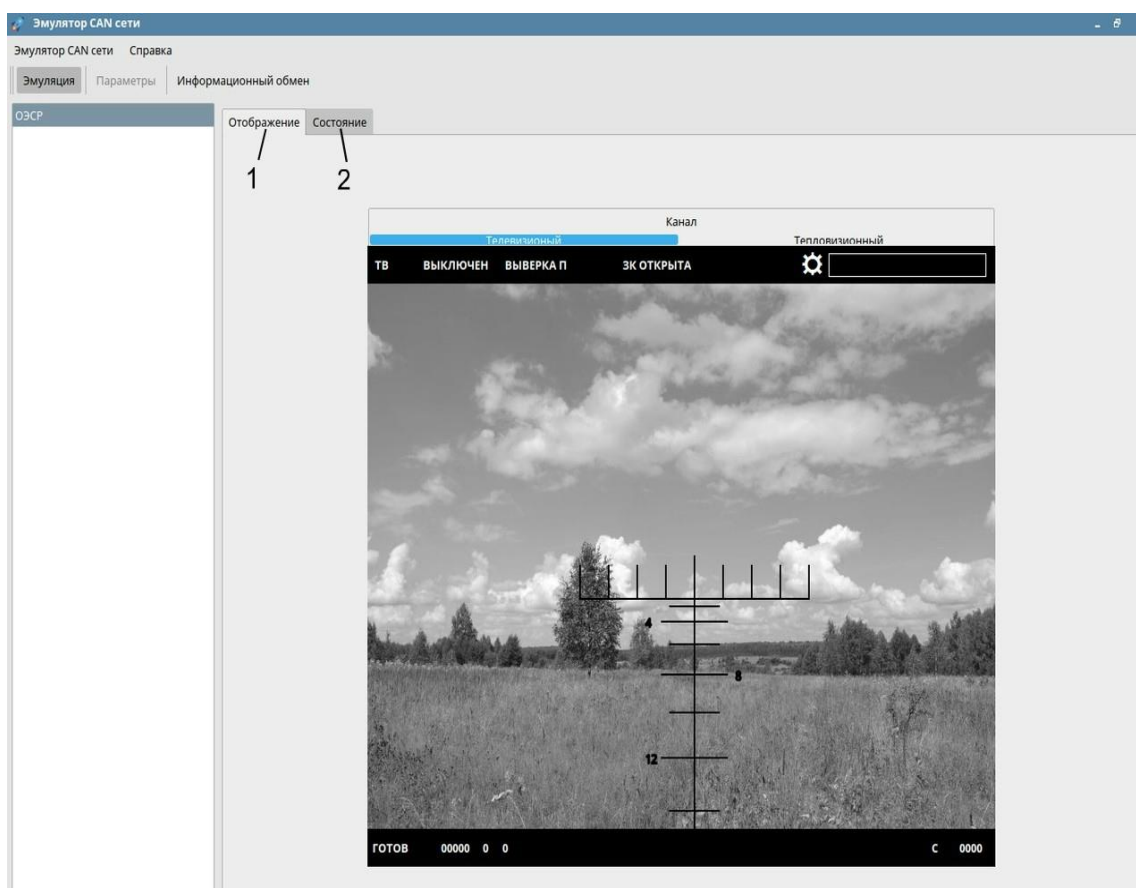


Рисунок 15 – Отображение выбранных состояний

Страница отображение выбранных состояний состоит из следующих компонентов:

- 1 – кнопка отображение;
- 2 – кнопка состояние.

Данная страница содержит 2 кнопки, предназначенные для перехода к функциональным возможностям программы. Нажав кнопку, отмеченную на рисунке 15 цифрой 2, оператор ОЭСН переходит на страницу состояний

(рисунок 16). Более подробное описание интерфейса данной страницы в приложение Г.

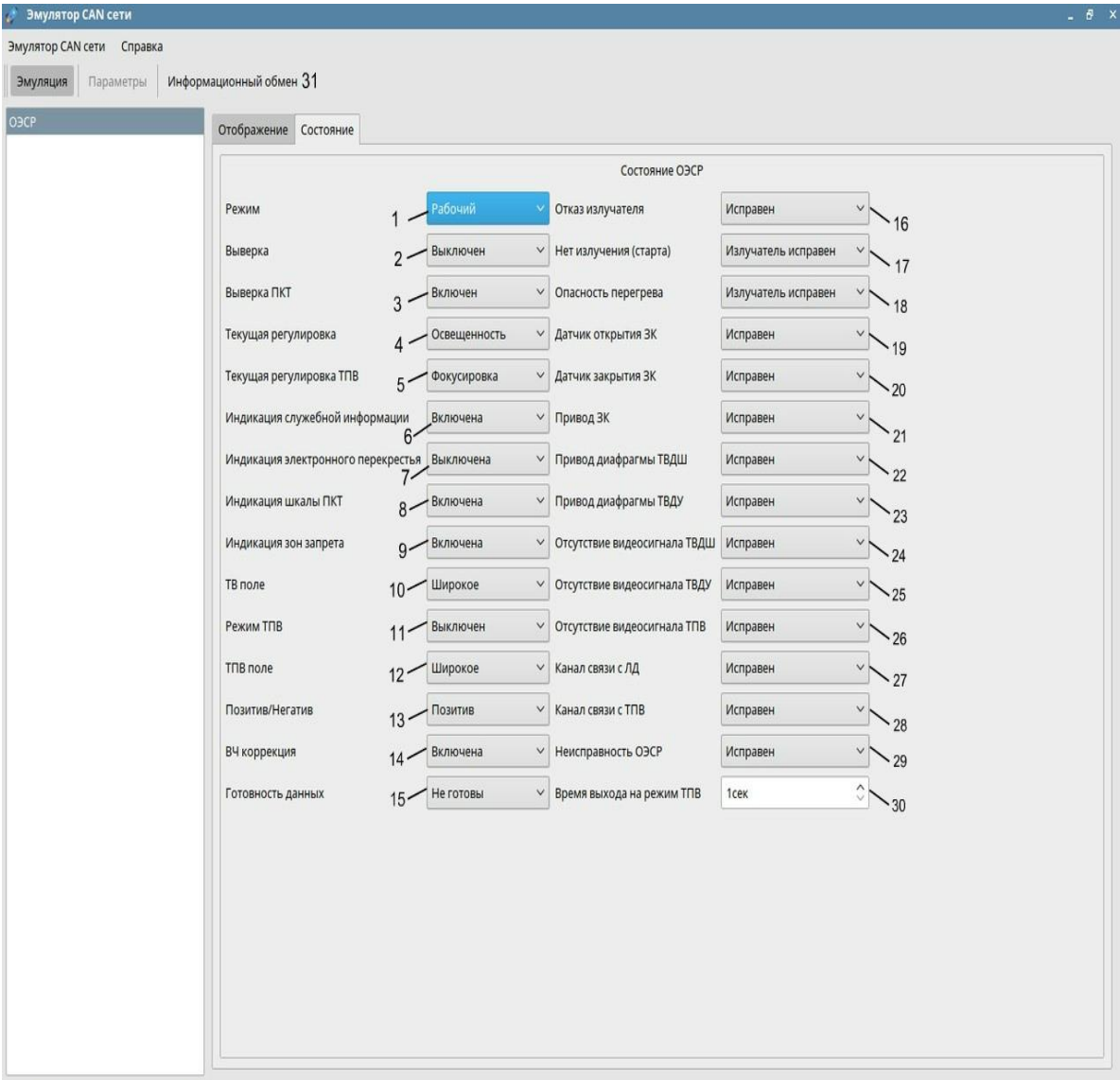


Рисунок 16 – Страница состояний

Данная страница содержит 30 кнопок, предназначенных для настройки ОЭСН. Нажав кнопку, на рисунке 16 под цифрой 31, оператор ОЭСН переходит на страницу информационный обмен (рисунок 17).

	время	snd	rcv	dsr	pri	size	тело сообщения
1	14:21:05.996						Остановка эмуляции
2	14:17:23.600	63	63	16383	7	0	
3	14:17:23.430						Старт эмуляции
4	14:14:11.745						Остановка эмуляции
5	14:13:27.168	63	63	16383	7	0	
6	14:13:27.000						Старт эмуляции
7	14:10:09.715						Остановка эмуляции
8	14:10:07.733	63	63	16383	7	0	
9	14:10:07.347						Старт эмуляции
10	14:09:42.491						Остановка эмуляции
11	10:59:36.788	63	63	16383	7	0	
12	10:59:36.617						Старт эмуляции
13	10:56:39.507						Остановка эмуляции
14	10:53:52.046	63	63	16383	7	0	
15	10:53:51.885						Старт эмуляции
16	10:53:24.285						Остановка эмуляции
17	10:34:08.758	63	63	16383	7	0	
18	10:34:08.585						Старт эмуляции
19	10:33:56.970						Остановка эмуляции
20	10:33:52.533	63	63	16383	7	0	
21	10:33:52.362						Старт эмуляции
22	10:33:49.978						Остановка эмуляции
23	10:31:59.571	63	63	16383	7	0	
24	10:31:59.395						Старт эмуляции

Рисунок 17 – Информационный обмен

Данная страница содержит 2 кнопки, кнопка на рисунке 6 под цифрой 1, предназначенных для остановки на паузу журнала информационного обмена, а кнопка на рисунке 17 под цифрой 2, предназначена для применения определённых фильтров к записям в информационном журнале. Поле изображённое, на рисунке 17 под цифрой 3, обозначает количество записей в информационном журнале.

Так же оператор ОЭНС может настроить CAN сеть, перейдя по вкладке параметры. Вкладка параметры изображена на рисунках 18 – 20.

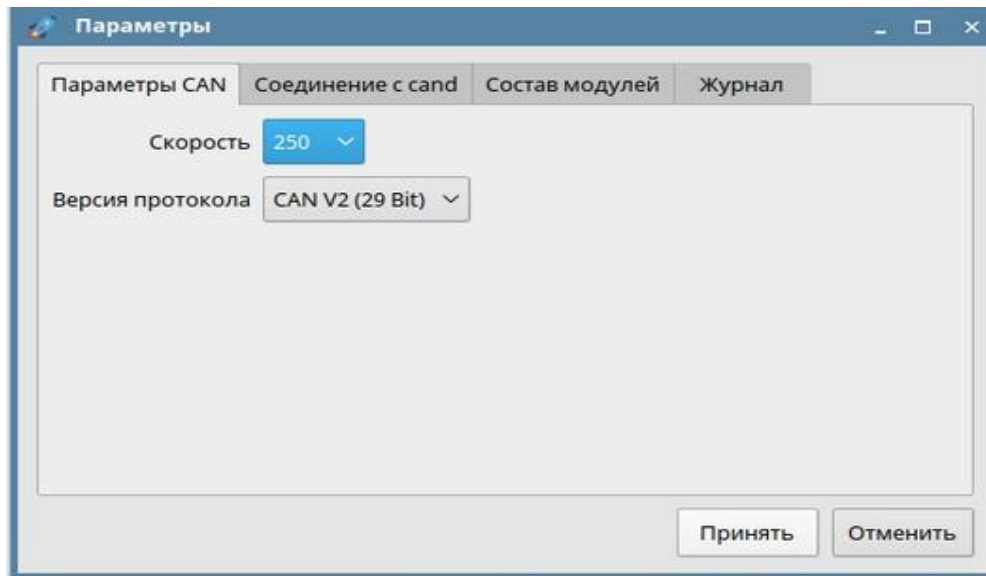


Рисунок 18 – Страница параметров (часть 1)

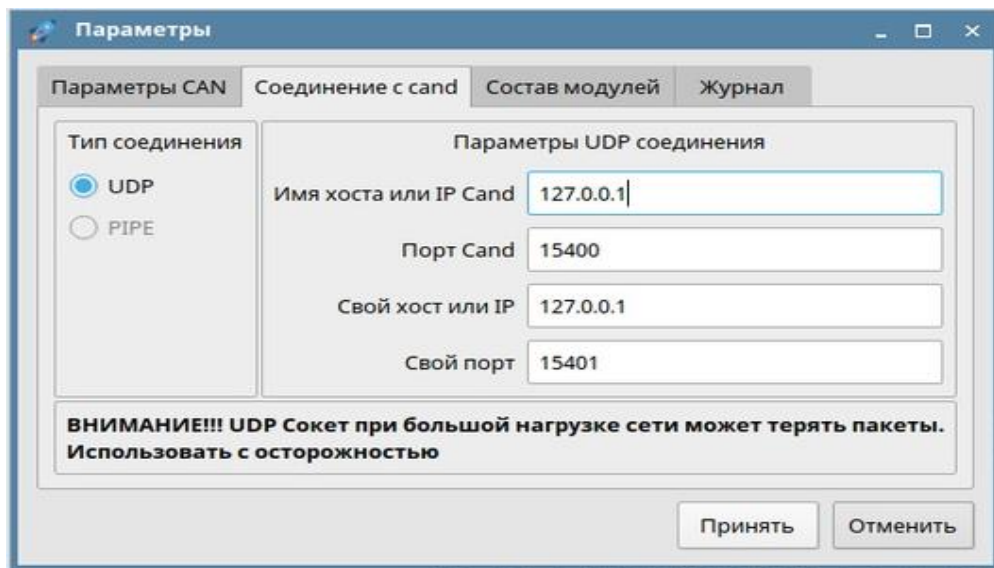


Рисунок 19 – Страница параметров (часть 2)

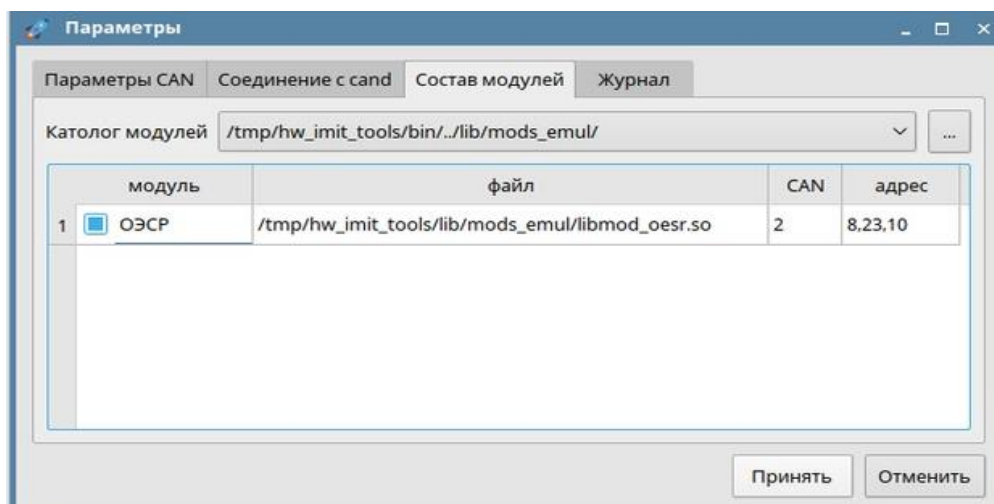


Рисунок 20 – Страница параметров (часть 3)

На рисунке 21 изображён журнал настройки CAN сети.

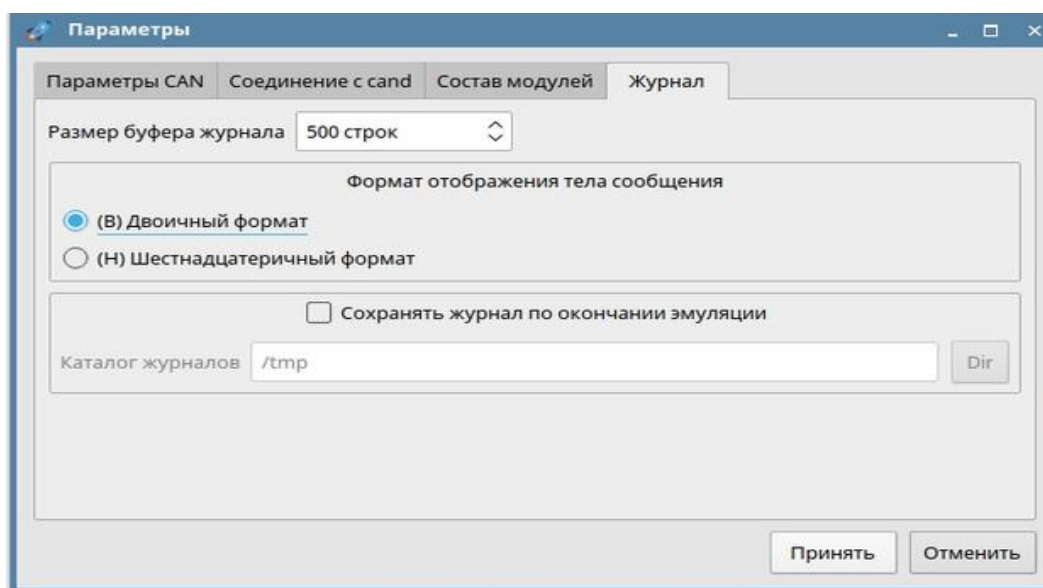


Рисунок 21 – Страница параметров (часть 4)

На рисунках 18–21, представлены параметры, которыми пользуется оператор ОЭСН для настройки CAN сети.

3 Реализация программных средств имитатора

Кодирование выполнялось на языке программирования C++ в среде Qt5.14. В результате созданы классы:

- Oesr_TV – Структура телевизионного канала ОЭСН;
- Oesr_TPV – Структура тепловизионного канала ОЭСН;
- Oesr_LD – Структура канала дальномера ОЭСН;
- Oesr_State – класс состояний;
- ModelLog::mod_oesr – наследник классов Oesr_TV, Oesr_TPV, Oesr_LD – класс модуля для эмуляции ОЭСН;

Диаграмма классов представлена на рисунке 22.

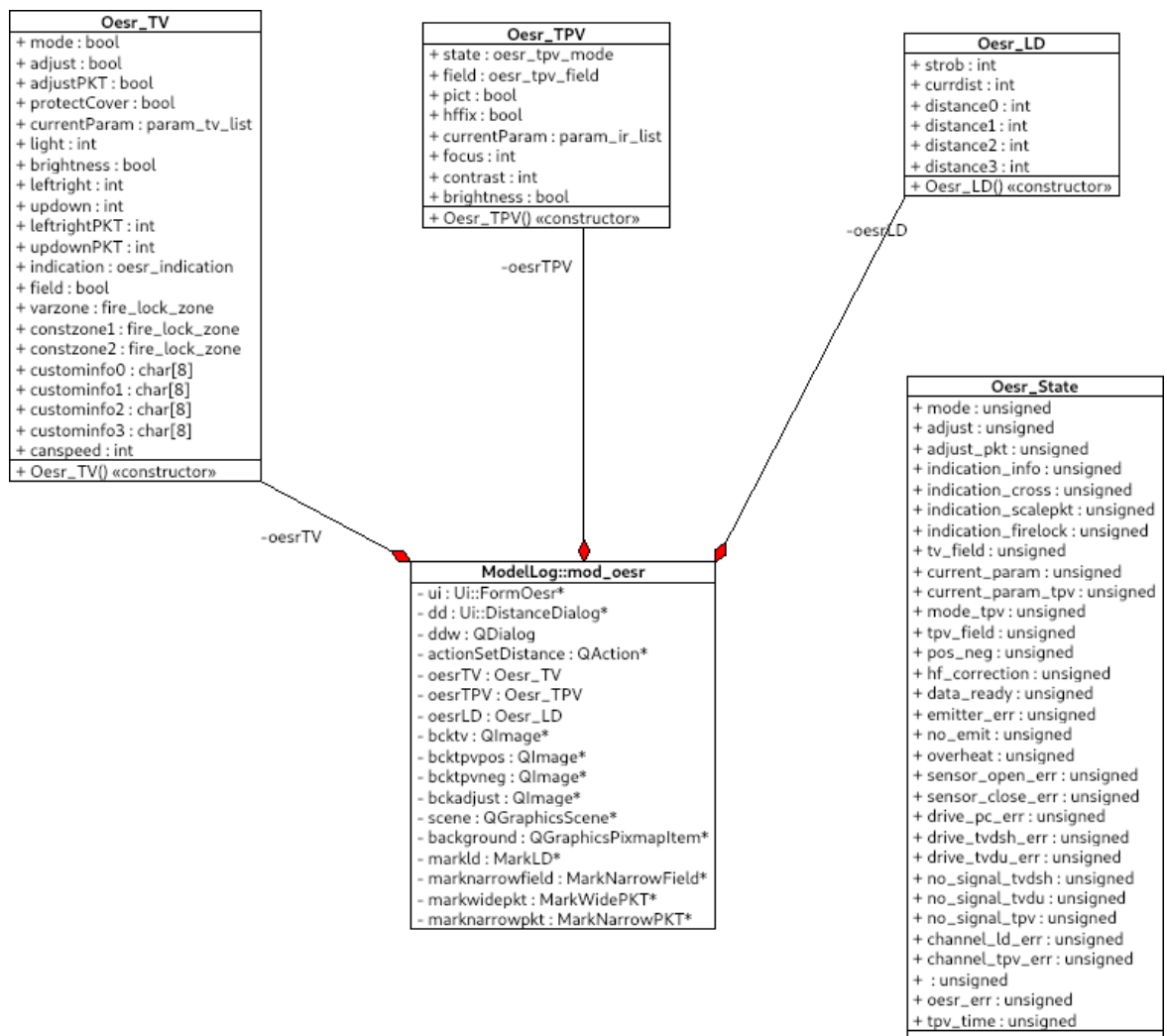


Рисунок 22 – Диаграмма классов модуля ОЭСН

Рассмотрим более подробно два класса Oesr_TV и ModelLog::mod_oesr.

Класс Oesr_TV имеет атрибуты:

- mode – режим работы 0 – дежурный, 1 – рабочий ;
- adjust – выверка;
- adjustPKT – выверка ПКТ;
- protectCover – нсостояние защитной крашки 0 – закрыта, 1 –

открыта;

- currentParam – регулируемый параметр ТВ;
- leftright – положение перекрестия по горизонтали 0 -128;
- updown – положение перекрестия по вертикали 0 -128;
- indication – режим индикации;
- field – поле зрения 0 – широкое, 1 – узкое;
- varzone – пееменная зона запрета стрельбы;
- constzone1 – постоянная зона запрета стрельбы;
- customnfo0 – пользовательская информация;
- conspeed – скорость обмена по шине CAN;

Класс ModelLog::mod_oesr имеет атрибуты:

- ui – главная форма;
- actionSetDistance – ввод показаний дальномера;
- oesrTV – телевизионный канал;
- oserTVP – тепловизионный канал;
- oserLD – дальномер;
- bcktv – графика (фон для тв);
- bcktrvpos – графика (фон для тпв позитив);
- bcktrvneg – графика (фон для тпв негатив);
- bckadjust – графика (фон для режима выверки);
- scene – холмт графического поля измерений;
- background – графика (фоновое изображение);
- markld – прицельная марка дальномера;

- marknarrowfield – метка узкого поля;
- markwidepkt – шкала ПКТ штрoкого поля зрения;
- marknarrowpkt – шкала ПЛЕ узкого поля зрения.

4 Тестирование и анализ качества программного кода имитатора

4.1 Тестирование программного обеспечения

Контроль качества программного обеспечения – неотъемлемый этап в разработке приложений. Одним из важнейших способов контроля качества является тестирование программного обеспечения.

Тестирование – способ обеспечения качества программного продукта, которое представляет собой ряд мероприятий, связанных с различного рода испытаниями тестируемого объекта для установления соответствия или несоответствия техническим требованиям и направленных на выявление дефектов [13].

Под дефектами понимаются ошибки в программном обеспечении, которые ведут к нарушению работы приложения. Необнаруженные на стадии разработки дефекты способны принести заказчику материальные убытки, а также исправление таких ошибок при эксплуатации программного обеспечения может быть затруднено и экономически невыгодно. Поэтому тестирование является неотъемлемой частью разработки приложений.

Для выпускной квалификационной работы было выбрано тестирование методом черного ящика, которое подразумевает по собой осуществление функционального тестирования, и модульное тестирование.

4.1.1 Функциональное тестирование

Функциональное тестирование – это тестирование программного обеспечения с целью проверки реализации функциональных требований. Другими словами, проверяется способность программного обеспечения в определенных условиях решать необходимые пользователям задачи [14]. Тестирование будем проводить в среде Эмулятор CAN сети под операционной системой Astra Linux Common Edition («Орел»). План тестирования

разработанного программного обеспечения имитатора ОЭСН представлен в таблице 4.

Таблица 4 – План функционального тестирования в среде Эмулятор CAN сети

№	Название теста	Описание теста	Ожидаемый результат	Результат выполнения
1	Состояние «включен»	Имитация входного дискретного сигнала	Появилось окно с телевизионным и тепловизионным каналами, и окно настройки состояния, имитатор включен	Соответствует ожидаемым результатам (рисунок В.1)
2	Состояние «выключен»	Имитация входного дискретного сигнала	Пустое окно, имитатор выключен	Соответствует ожидаемым результатам (рисунок В.2)
3	Состояние «Рабочий»	Имитация входного сообщения «Рабочий»	Появилось изображение на телевизионном канале, имитатор включен	Соответствует ожидаемым результатам (рисунок В.3)
4	Состояние «Дежурный»	Имитация входного сообщения «Дежурный»	Нет изображения на телевизионном канале, имитатор выключен	Соответствует ожидаемым результатам (рисунок В.4)
5	Состояние «Выверка»	Имитация входного сообщения «Выверка»	Выверка включена, появилось перекрестие в «Телевизионном канале»	Соответствует ожидаемым результатам (рисунок В.5)

Продолжение таблицы 4

№	Название теста	Описание теста	Ожидаемый результат	Результат выполнения
6	Состояние «Выверка ПКТ»	Имитация входного сообщения «Выверка ПКТ»	Выверка ПКТ включена, появилось специальное перекрестие в «Телевизионном канале»	Соответствует ожидаемым результатам (рисунок В.6)
7	Состояние «Индексация служебной информации»	Имитация входного сообщения «Регулировка параметра+»	Индексация служебной информации включена	Соответствует ожидаемым результатам (рисунок В.7)
8	Состояние «Режим ТПВ»	«Включить ТПВ»	ТПВ включен, появилось изображение на тепловизионном канале	Соответствует ожидаемым результатам (рисунок В.8)

Результаты тестирования представлены в приложении В. Функциональное тестирование разработанного программного обеспечения имитатора ОЭСН свидетельствует о корректности его работы.

4.1.2 Модульное тестирование

Модульное или unit-тестирование позволяет провести более глубокое исследование и анализ работоспособности отдельных модулей программного обеспечения. Существуют каркасы автономного тестирования, избавляющие пользователя от ручного воспроизведения тестов [15].

Написание модульных тестов же в отличие от функционального тестирования более похоже на тестирование методом белого ящика, когда детали реализации модулей нам известны.

Для выполнения автономного тестирования был создан и добавлен в решение тестовый проект. Результаты автономного тестирования представлены на рисунке 23.

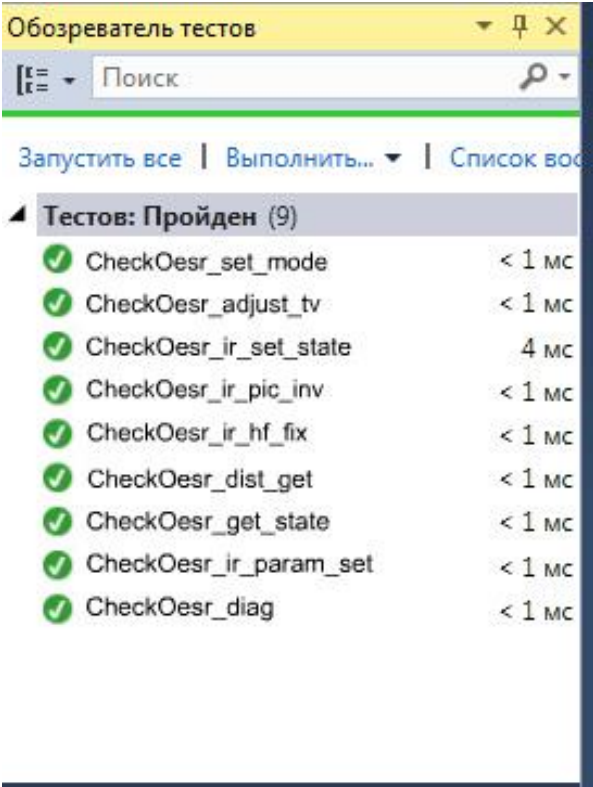


Рисунок 23 – Результаты модульных тестов

На данном рисунке в верхней части – перечень тестовых методов, статус выполнения и время их выполнения. 9 из 9 тестов выполнены успешно.

4.2 Метрики кода

Метрика программного обеспечения – это мера, представляющая собой численное значение некоторого свойства программного обеспечения или его спецификации [16].

4.2.1 Комплексный показатель качества кода (Maintainability Index)

Комплексный показатель качества кода (Maintainability Index) – наиболее часто используемая метрика программного обеспечения. Рассчитывается метрика по формуле:

$$MI = \max\left(0, (171 - 5,2 * \ln(HV) - 0,23 * CC - 16,2 * \ln(LoC)) * \frac{100}{171}\right),$$

где HV – Halstead Volume, вычислительная сложность, чем больше операторов, тем больше значение этой метрики;

CC – Cyclomatic Complexity, структурная сложность кода, т.е. количество различных ветвей в коде, чем больше показатель, тем больше тестов должно быть проведено для полного покрытия кода;

LoC – количество строк кода.

Комплексный показатель качества кода может принимать значения от 0 до 100 и показывает относительную сложность поддержки кода. Чем больше значение этой метрики, тем легче поддерживается код [16, 17].

Вычислительная сложность программы HV рассчитывается по следующей формуле:

$$HV = N \log_2 \eta,$$

где N – длина программы;

η – словарь программы.

Словарь программы рассчитывается по формуле:

$$\eta = \eta_1 + \eta_2,$$

где η_1 – число простых (отдельных) операторов и операций в программе;

η_2 – число простых (отдельных) операндов (констант и переменных) в программе.

Длина программы рассчитывается по формуле:

$$N = N_1 + N_2,$$

где N_1 – общее число простых (отдельных) операторов и операций в программе;

N_2 – общее число простых (отдельных) операндов в программе.

Структурная сложность кода программы (CC) вычисляется по формуле:

$$CC = e - v + 2 \times p,$$

где e – число дуг ориентированного графа программы;

v – число вершин ориентированного графа программы;

p – число компонент связности ориентированного графа программы.

Программа может быть сложной и объемной и построить управляющий граф для нее может быть затруднительно. В этом случае структурную сложность программы можно определить по формуле:

$$CC = КОЦ + КОУ + 1,$$

где КОЦ – количество операторов цикла;

КОУ – количество операторов условия – IF.

Значение CC больше 10 свидетельствует о структурной сложности программы, то затрудняет ее функциональное тестирование и требует упрощения программы или выбора другого метода тестирования.

Посчитаем комплексный показатель качества кода для фрагмента кода, отвечающего за настройку режима (листинг 1).

Листинг 1 – Настройка режима

```
int mod_oesr::SetStandby(const quint8 *data)
{
    bool setmode = data[0];
    if (oesrTV.mode != setmode)
    {
        oesrTV.mode = setmode;
        if(setmode == 1)                // переходим в рабочий режим
        {
            oesrTV.protectCover = 1;    //открываем защитную крышку
            oesrTV.adjust = 0;
            oesrTV.adjustPKT = 0;
            background->setVisible(true);
        }
        else if(setmode == 0)           // переходим в дежурный режим
        {
            oesrTV.protectCover = 0;
            oesrTV.adjust = 0;
            oesrTV.adjustPKT = 0;
            background->setVisible(false);
        }
        else
            return 4;
    }
    this->setForm();
    return 0;
}
```

Данные для расчета словаря и длины программы представлены в таблице 5.

Таблица 5 – Данные для расчета словаря и длины программы

Операторы		Операнды	
Оператор	Количество	Операнд	Количество
if	3	mod_oesr	1
int	1	oesrTV.protectCover	2
const	1	oesrTV.adjust	2
else	2	oesrTV.adjustPKT	2
return	2	background	2
:	2	setVisible	2
true	1	setmode	5
false	1	setForm	1
this	1	SetStandby	1
->	3	quint	1
()	7	data	2
;	13	oesrTV.mode	2
=	9		
==	2		
{}	4		
*	1		
!	1		
[]	1		
bool	1		
Всего: 19	Всего: 56	Всего: 12	Всего: 23

Из таблицы 5 можно увидеть, что:

Словарь программы:

$$\eta = \eta_1 + \eta_2 = 31.$$

Длина программы:

$$N = N_1 + N_2 = 79.$$

Вычислительная сложность:

$$HV = N \log_2 \eta = 79 * \log_2 31 = 76 * 4,95 = 376.$$

Количество строк кода:

$$LoC = 17.$$

Структурная сложность кода:

$$CC = КОЦ + КОУ + 1 = 2 + 1 = 3.$$

Комплексный показатель качества кода:

$$\begin{aligned} MI &= \max \left(0, (171 - 5,2 * \ln(HV) - 0,23 * CC - 16,2 * \ln(LoC)) * \frac{100}{171} \right) = \\ &= \max \left(0, (171 - 5,2 * \ln(376) - 0,23 * 3 - 16,2 * \ln(17)) * \frac{100}{171} \right) = \\ &= \max(0, 55) = 55. \end{aligned}$$

Комплексный показатель равен 55, что свидетельствует об относительной простоте поддержки программного кода, возможности его изменения и достаточно высоком качестве разработанного программного обеспечения.

4.2.2 Метрика Холстеда

Эта метрика служит для оценки объема рассматриваемого листинга. Для ее вычислений нужно определить число уникальных операторов в программе – $n1$, – (таблица 5), число уникальных операндов – $n2$, – (таблицу 5), а также их общее количество – $N1$ и $N2$ соответственно.

Из таблицы 5 можно увидеть, что:

$$n1 = 19.$$

Таблица 5 в свою очередь дает понять, что:

$$n2 = 12.$$

Для определения значений параметров $N1$ и $N2$ подсчитаем общее количество операторов и операндов в программе:

$$N1 = 56,$$

$$N2 = 23.$$

Затем вычислим словарь программы (n) и ее длину (N):

$$n = n1 + n2 = 31,$$

$$N = N1 + N2 = 79.$$

Теперь для вычисления ее объема осталось только подставить значения в приведенную ниже формулу и получить результат:

$$V = N * \log_2 n = 391,366.$$

Данное значение метрики Холстеда показывает, что эта функция является достаточно компактной.

Заключение

В рамках выполнения выпускной квалификационной работы был разработан программный имитатор оптико-электронных средств наблюдения.

Во время проведения анализа предметной области и требований были проанализированы существующие аналоги, выявлены их преимущества и недостатки, разработаны функциональные и нефункциональные требования к создаваемому программному средству.

На этапе проектирования было осуществлено планирование разработки программы и оценка бюджета, которое показало, что длительность реализации проекта составляет 50 дней и на его реализацию следует затратить 424 000 руб. Также на этом этапе были созданы концепции окон пользовательского интерфейса программного средства.

Для реализации программного имитатора использовался язык программирования C++, код на котором разрабатывался в среде программирования Qt 5.14.

Тестирование клиентской части программного средства проводилось методами функционального тестирования. Также были просчитаны метрики кода для оценки его качества.

Далее программное средство будет реализовано и внедрено в производство. В приложении Е приведен акт о его внедрении.

Как видно из всего вышесказанного, поставленная задача по реализации программного имитатора была выполнена.

Список использованных источников

- 1 Указ Президента Российской Федерации от 9 мая 2017 г. №203 «О стратегии развития информационного общества в Российской Федерации на 2017 – 2030 годы» [Электронный ресурс]. URL: http://zmedu.hostedu.ru/files/ykaz_7668.pdf (дата обращения: 21.05.2020).
- 2 Распоряжение Правительства Российской Федерации от 28 июля 2017 г. №1632 – р «Цифровая экономика Российской Федерации» [Электронный ресурс]. URL: <http://static.government.ru/media/files/9gFM4FHj4PsB79I5v7yLVuPgu4bvR7M0.pdf> (дата обращения: 21.05.2020).
- 3 CAN – шина, CAN – интерфейс [Электронный ресурс] <http://embeddedsystem.ru/index.php?page=can-bus-can-interface> (дата обращения: 22.05.2020).
- 4 Оптико-электронный комплекс наблюдения «Ирония» [Электронный ресурс] <https://topwar.ru/108862-optiko-elektronnyy-kompleks-nablyudeniya-ironiya.html> (дата обращения: 22.05.2020).
- 5 Адунок [Электронный ресурс] <https://ru.wikipedia.org/wiki/%D0%90%D0%B4%D1%83%D0%BD%D0%BE%D0%BA> (дата обращения: 22.05.2020).
- 6 Все для аналитиков [Электронный ресурс]. – URL: http://foranalysts.blogspot.com/2011/08/blog-post_17.html (дата обращения: 22.05.2020).
- 7 Арлоу, Д. UML 2 и Унифицированный процесс / Д. Арлоу, А. Нейштадт. – 2-е издание, перевод с англ. Шатохиной Н. – Санкт-Петербург: «Символ-Плюс», 2007. – 624 с.
- 8 Анализ требований разработки ПО [Электронный ресурс]. – URL: <http://unetway.com/tutorial/analiz-trebovanij-razrabotki-po/> (дата обращения: 29.05.2020).

9 Статья на Wikipedia о методологии RUP [Электронный ресурс]. URL:https://ru.wikipedia.org/wiki/Rational_Unified_Process (дата обращения: 23.05.2020).

10 Статья о методологии RUP // Информационный ресурс [Электронный ресурс]. URL:<http://www.informicus.ru/default.aspx?SECTION=6&id=73&subdivisionid=7> (дата обращения: 23.05.2020).

11 Статья про Qt [Электронный ресурс]. URL: <https://ru.wikipedia.org/wiki/Qt> (дата обращения: 23.05.2020).

12 Управление проектами [Электронный ресурс]. URL: https://ru.wikipedia.org/wiki/Управление_проектами (дата обращения: 23.05.2020).

13 Функциональное тестирование программного обеспечения [Электронный ресурс]. URL: <http://aplana.ru/services/testing/functionalnoe-testirovanie> (дата обращения: 26.05.2020).

14 Функциональное тестирование [Электронный ресурс]. URL: https://ru.wikipedia.org/wiki/Функциональное_тестирование (дата обращения: 27.05.2020).

15 Гончаренко, С. Управление проектами. Управление качеством. – 2011. – N 8. – С. 44-46.

16 Метрика программного обеспечения // Википедия [Электронный ресурс]. URL: https://ru.wikipedia.org/wiki/Метрика_программного_обеспечения (дата обращения: 27.05.2020).

17 Средство оценки качества кода // Сообщество IT-специалистов [Электронный ресурс]. URL: <https://habr.com/post/111524/> (дата обращения: 27.05.2020).

18 Umbrello [Электронный ресурс]. <https://ru.wikipedia.org/wiki/Umbrello> (дата обращения: 10.06.2020).

- 19 Doxygen [Электронный ресурс].
<https://ru.wikipedia.org/wiki/Doxygen> (дата обращения: 10.06.2020).
- 20 Плагин [Электронный ресурс].
<https://ru.wikipedia.org/wiki/%D0%9F%D0%BB%D0%B0%D0%B3%D0%B8%D0%BD> (дата обращения: 11.06.2020).

Приложение А
ГЛОССАРИЙ
(справочное)

RUP (Rational Unified Process) – методология разработки программного обеспечения, в основе которой лежит использование итеративной модель разработки [8].

UML (Unified Modeling Language) – язык графического описания, с помощью которого осуществляется объектное моделирование в области разработки программного обеспечения, проектирования систем и отображения организационных структур.

API (Application Programming Interface) – набор готовых классов, функций, процедур, констант и структур, предоставляемых приложением или операционной системой для использования во внешних программных продуктах.

Umbrello – среда UML-моделирования. Это приложение является свободным программным обеспечением, предназначенным для построения UML диаграмм на платформе Unix. Является CASE инструментом. Umbrello входит в пакет kdesdk оконного менеджера KDE, но также хорошо работает и с другими оконными менеджерами. Пользовательский интерфейс программы прост и функционален.

Umbrello поддерживает все стандартные типы UML-диаграмм. Также поддерживается импорт из C++, IDL, Pascal/Delphi, Ada, Python, Java, Perl (с помощью внешнего инструмента, доступного на uml.sourceforge.net) и экспорт диаграмм в различные языки программирования. Формат файла, используемый при хранении диаграмм, основан на XML.

Umbrello позволяет хранить данные модели в форматах DocBook и XHTML. Это было сделано с целью поддержки модели совместной разработки, когда не все разработчики имеют доступ к Umbrello. Эта возможность также незаменима при необходимости размещения контента модели на web-сайте [18].

Doxygen — это кроссплатформенная система документирования исходных текстов, которая поддерживает C++, Си, Objective C, Python, Java, IDL, PHP, C#, Фортран, VHDL и, частично, D.

Doxygen генерирует документацию на основе набора исходных текстов и также может быть настроен для извлечения структуры программы из недокументированных исходных кодов. Возможно составление графов зависимостей программных объектов, диаграмм классов и исходных кодов с гиперссылками [19].

Плагин — независимо компилируемый программный модуль, динамически подключаемый к основной программе и предназначенный для расширения и/или использования её возможностей. Плагины обычно выполняются в виде библиотек общего пользования [20].

Приложение Б
ИСХОДНЫЙ КОД ПРИЛОЖЕНИЯ
(рекомендуемое)

Листинг Б.1 – Файл mod_oesr.cpp

```

#include <QtWidgets>

#include "mod_oesr.h"

#include "ui_FormOesr.h"

#include "ui_DistanceDialog.h"

/** @brief дескриптор модуля для эмуляции ОЕСР*/
sModProps mod_props = {"ModuleInit", "ОЭСР", 2, "8,23,10"};

extern "C" mod_oesr* ModuleInit(QObject* parent, sModProps mod_props)
{
    return new mod_oesr(parent, mod_props);
}

mod_oesr::mod_oesr(QObject* parent, sModProps mod_props)
    : AbstractModule(parent, mod_props)
    , ui(new Ui::FormOesr)
    , dd(new Ui::DistanceDialog)
    //    , actionChannelTV(new QAction(QString::fromUtf8("ТВ"), this))
    //    , actionChannelTPV(new QAction(QString::fromUtf8("ТПВ"), this))
    //    , actionFieldNarrowTV(new QAction(QString::fromUtf8("УЗКОЕ"), this))
    //    , actionFieldWideTV(new QAction(QString::fromUtf8("ШИРОКОЕ"), this))
    //    , actionFieldWideTPV(new QAction(QString::fromUtf8("ШИРОКОЕ"), this))
    //    , actionFieldNarrowTPV(new QAction(QString::fromUtf8("УЗКОЕ"), this))
    //    , actionFieldNarrowX2TPV(new QAction(QString::fromUtf8("УЗКОЕ X2"),
this))
    //    , actionFieldSaveTPV(new QAction(QString::fromUtf8("ЭНЕПТОСБ"),
this))
    //    , actionFieldInitTPV(new QAction(QString::fromUtf8("ИНИЦИАЛ"), this))

```

```

//      , actionFieldOffTPV(new QAction(QString::fromUtf8("ВЫКЛЮЧЕН"), this))

//      , actionVerification(new QAction(QString::fromUtf8("ВЫБЕРКА"), this))

//      , actionVerificationTPV(new QAction(QString::fromUtf8("ВЫБЕРКА ПКТ"),
this))

//      , actionStandby(new QAction(QString::fromUtf8("ДЕЖУРНЫЙ"), this))

//      , actionVCHCorrOn(new QAction(QString::fromUtf8("ВКЛЮЧЕНО"), this))

//      , actionVCHCorrOff(new QAction(QString::fromUtf8("ВЫКЛЮЧЕНО"), this))

//      , actionAdjustLightTV(new QAction(QIcon(":/new/icons/bright.png"),
QString::fromUtf8("ОСВЕЩЕННОСТЬ"), this))

//      , actionAdjustBrightTV(new QAction(QIcon(":/new/icons/brightness.png"),
QString::fromUtf8("ЯРКОСТЬ
МАРКИ"), this))

//      , actionAdjustUpDownTV(new QAction(QIcon(":/new/icons/updown.png"),
QString::fromUtf8("ВВЕРХ-ВНИЗ"), this))

//      , actionAdjustLeftRightTV(new QAction(QIcon(":/new/icons/leftright.png"),
QString::fromUtf8("ВЛЕВО-ВПРАВО"),
this))

//      , actionAdjustFocusTPV(new QAction(QIcon(":/new/icons/focus.png"),
QString::fromUtf8("Фокусировка"), this))

//      , actionAdjustContrastTPV(new QAction(QIcon(":/new/icons/contrast.png"),
QString::fromUtf8("Контраст"),
this))

//      , actionAdjustBrightTPV(new QAction(QIcon(":/new/icons/brightness.png"),
QString::fromUtf8("ЯРКОСТЬ
МАРКИ"), this))

//      , actionAdjustUpDownTPV(new QAction(QIcon(":/new/icons/updown.png"),
QString::fromUtf8("ВВЕРХ-ВНИЗ"), this))

//      , actionAdjustLeftRightTPV(new QAction(QIcon(":/new/icons/leftright.png"),
QString::fromUtf8("ВЛЕВО-ВПРАВО"),
this))

//      , actionReadyLD(new QAction(QString::fromUtf8("ГОТОВ"), this))

//      , actionFailLD(new QAction(QString::fromUtf8("ОТКАЗ"), this))

//      , actionNoEmitLD(new QAction(QString::fromUtf8("НЕТ ИЗЛ."), this))

//      , actionOverheatOn(new QAction(QString::fromUtf8("ПЕРЕГРЕВ"), this))

//      , actionOverheatOff(new QAction(QString::fromUtf8("НЕТ ПЕРЕГРЕВА"),
this))

, actionSetDistance(new QAction(QString::fromUtf8("ВВОД ДАЛЬНОСТИ"), this))

```

```

, bcktv(new QImage(":/new/image/mihaylovka.png"))

, bcktpvpos(new QImage(":/new/image/mihaylovkatpv.png"))

, bcktpvneg(new QImage(":/new/image/mihaylovkatpv.png"))

, bckadjust(new QImage(":/new/image/adjustmark.png"))

, scene(new QGraphicsScene())

, markld(new MarkLD())

, marknarrowfield(new MarkNarrowField(20))

, markwidepkt(new MarkWidePKT())

, marknarrowpkt(new MarkNarrowPKT())

{

    ui->setupUi(widget);

    dd->setupUi(&ddw);


    ui->progressBarScale->reset();

    ui->progressBarScale->setStyle(QStyleFactory::create("Windows"));


    ui->graphicsView->setScene(scene);

    scene->setSceneRect(0, 0, 768, 576);

    bcktpvpos->invertPixels();


    background = new QGraphicsPixmapItem(QPixmap::fromImage(*bcktv)) ;

    // background->scaled(768, 384, Qt::KeepAspectRatioByExpanding);

    scene->addItem(background);

    scene->addItem(markld);

    scene->addItem(marknarrowfield);

    scene->addItem(markwidepkt);

    scene->addItem(marknarrowpkt);

```

```

background->setVisible(false);

markld->setVisible(false);

marknarrowfield->setVisible(false);

markwidepkt->setVisible(false);

marknarrowpkt->setVisible(false);


this->setActions();

this->setConnections();

this->setForm();

ui->frameTop->setVisible(oesrTV.indication.info);

ui->frameBottom->setVisible(oesrTV.indication.info);

}


mod_oesr::~~mod_oesr()

{

    delete ui;

    delete actionSetDistance;


    delete bcktv;

    delete bcktpvpos;

    delete bcktpvneg;


    delete markld;

    delete marknarrowfield;

    delete markwidepkt;

    delete marknarrowpkt;


    delete scene;

```

```

}

bool mod_oesr::MsgTrap(const CanMsg &msg)

{
    //qDebug() << QString::number(msg.hdr.id,16) << " " <<
    QString::number(msg.data[0],16)<< " " << QString::number(msg.data[1],16)<< " "
    << QString::number(msg.data[2],16)<< " " << QString::number(msg.data[3],16)<<
    " " << QString::number(msg.data[4],16)<< " " << QString::number(msg.data[5],16)
    << " " << QString::number(msg.data[6],16)<< " " <<
    QString::number(msg.data[7],16);

    // qDebug()<<msg.hdr.v2.rcv;

    switch (msg.hdr.v2.rcv)
    {

    case oesr_tv:    this->processCommandTV(msg);

        break;

    case oesr_tpv: processCommandTPV(msg);

        break;

    case oesr_ld:    processCommandLD(msg);

        break;

    }

    return true;
}

void mod_oesr::timerEvent(QTimerEvent *tEvent)

{
    qDebug() << "timerEvent\n";

    QObject::timerEvent(tEvent);
}

bool mod_oesr::eventFilter(QObject *obj, QEvent *event)

{

```

```

    qDebug() << "eventFilter\n";

    return QObject::eventFilter(obj, event);
}

void mod_oesr::setActions()

{

    ui->labelDistance->setContextMenuPolicy(Qt::ActionsContextMenu);

    ui->labelDistance->addAction(actionSetDistance);

}

void mod_oesr::setConnections()

{

    connect(actionSetDistance, SIGNAL(triggered()), this,
    SLOT(SetDistanceDialog()));

    connect(ui->pushButtonTV, SIGNAL(toggled(bool)), this, SLOT(setForm()));

    connect(ui->comboBoxSMODE, SIGNAL(activated(int)), this, SLOT(comboBoxChange(int)));

    connect(ui->comboBoxSAdjust, SIGNAL(activated(int)), this, SLOT(comboBoxChange(int)));

    connect(ui->comboBoxSAdjustPKT, SIGNAL(activated(int)), this, SLOT(comboBoxChange(int)));

    connect(ui->comboBoxSCurrParam, SIGNAL(activated(int)), this, SLOT(comboBoxChange(int)));

    connect(ui->comboBoxSCurrParamTPV, SIGNAL(activated(int)), this, SLOT(comboBoxChange(int)));
;

    connect(ui->comboBoxSServiceInfo, SIGNAL(activated(int)), this, SLOT(comboBoxChange(int)));

    connect(ui->comboBoxSCross, SIGNAL(activated(int)), this, SLOT(comboBoxChange(int)));

```

```

        connect(ui-
>comboBoxSScalePKT, SIGNAL(activated(int)), this, SLOT(comboBoxChange(int)));

        connect(ui-
>comboBoxSFirelock, SIGNAL(activated(int)), this, SLOT(comboBoxChange(int)));

        connect(ui-
>comboBoxSTvField, SIGNAL(activated(int)), this, SLOT(comboBoxChange(int)));

        connect(ui-
>comboBoxSModeTPV, SIGNAL(activated(int)), this, SLOT(comboBoxChange(int)));

        connect(ui-
>comboBoxSTPVField, SIGNAL(activated(int)), this, SLOT(comboBoxChange(int)));

        connect(ui-
>comboBoxSPosNeg, SIGNAL(activated(int)), this, SLOT(comboBoxChange(int)));

        connect(ui-
>comboBoxSCorrVCH, SIGNAL(activated(int)), this, SLOT(comboBoxChange(int)));

        connect(ui-
>comboBoxSCorrVCH, SIGNAL(activated(int)), this, SLOT(comboBoxChange(int)));
    }

void mod_oesr::processCommandTV(const CanMsg &msg)
{
    int ans = 1; // код ответного сообщения оэср: 0 -команда
    принята к исполнению, 1-команда не поддерживается, 2-команда не может быть
    выполнена прямо сейчас (абонент временно занят), 3-команда не может быть
    выполнена в данном режиме работы, 4-ошибка в параметрах команды, 99- ответное
    сообщение не требуется

    switch(msg.hdr.v2.dsr)
    {

        case oesr_set_mode: //< установить режим работы
(рабочий/дежурный) */
            ans=this->SetStandby(msg.data);

            break;

        case oesr_adjust_tv: //< включить режим выверки */
            ans=SetAdjust(msg.data);

            break;

        case oesr_adjust_ir: //< включить режим выверки ПКТ */

```

```

        ans=SetAdjustPKT(msg.data);

        break;

        case oesr_param_sel:                                /*!< выбрать регулируемый параметр ТВ для
изменения */

            ans=this->SelectCurrentParam();

            break;

        case oesr_param_inc:                                /*!< увеличить выбранный параметр ТВ */

            this->SetParamUp();

            ans=99;

            break;

        case oesr_param_dec:                                /*!< уменьшить выбранный параметр ТВ */

            this->SetParamDown();

            ans=99;

            break;

        case oesr_param_set:                                /*!< установить значение регулируемого параметра
ТВ */

            ans=SetCurrentParam(msg.data);

            break;

        case oesr_indic:                                    /*!< выбрать режим индикации шкалы
пулемёта/прицельной марки/зон запрета/служебной */

            ans=this->SetIndication(msg.data);

            break;

        case oesr_set_zoom:                                  /*!< переключить узкое/широкое поле ТВ канала
*/

            ans=this->SetFieldTV(msg.data);

            break;

        case oesr_diag:                                      /*!< проверить работоспособность */

            ans=WorkTest();

            break;

        case oesr_get_state:                                /*!< получить слово состояния ОЭСР */

```

```

    ans=StateTest();

    break;

    case oesr_set_var_fz:                                /*!< установить координаты и размеры
маркера переменной зоны запрета стрельбы */

        ans=VariableZoneBan(msg.data);

        break;

    case oesr_set_const_fz_0:                            /*!< установить координаты и размеры маркера
постоянной зоны запрета стрельбы (левая) */

        ans=ConstZoneBan1(msg.data);

        break;

    case oesr_set_const_fz_1:                            /*!< установить координаты и размеры маркера
постоянной зоны запрета стрельбы (правая) */

        ans=ConstZoneBan2(msg.data);

        break;

    case oesr_custom_info_0:                            /*!< вывести 8 символов пользовательской
информации */

        ans=UserInfo1(msg.data);

        break;

    case oesr_custom_info_1:                            /*!< вывести 8 символов пользовательской
информации */

        ans=UserInfo2(msg.data);

        break;

    case oesr_custom_info_2:                            /*!< вывести 8 символов пользовательской
информации */

        ans=UserInfo3(msg.data);

        break;

    case oesr_custom_info_3:                            /*!< вывести 8 символов пользовательской
информации */

        ans=UserInfo4(msg.data);

        break;

    case oesr_set_speed:                                /*!< установить скорость обмена по шине
CAN */

        ans=SetSpeed(msg.data);

```

```

        break;

    }

    switch(ans) // код ответного сообщения оэср: 0 -команда принята к
    исполнению, 1-команда не поддерживается, 2-команда не может быть выполнена прямо
    сейчас (абонент временно занят), 3-команда не может быть выполнена в данном
    режиме работы, 4-ошибка в параметрах команды, 99- ответное сообщение не
    требуется

    {

    case 0:

        sendCommandAccepted(msg, 8);

        break;

    case 1:

        sendCommanNotSupported(msg, 8);

        break;

    case 2:

        sendCommanNotExecuteNow(msg, 8);

        break;

    case 3:

        sendCommandNotExecuteMode(msg, 8);

        break;

    case 4:

        sendCommandOptionError(msg, 8);

        break;

    case 99:

        break;

    }

}

void mod_oesr::processCommandTPV(const CanMsg &msg)

{

```

```

int ans = 0;

switch (msg.hdr.v2.dsr)
{
case oesr_ir_set_state:          /*!< включить/выключить ТПВ */

    ans=this->OnOffTPV(msg.data);

    break;

case oesr_ir_param_sel:          /*!< выбрать регулируемый параметр ТПВ для
изменения */

    ans=this->SelectCurrentParamTPV();

    break;

case oesr_ir_param_inc:          /*!< увеличить выбранный параметр ТПВ */

    SetParamUpTPV();

    ans=99;

    break;

case oesr_ir_param_dec:          /*!< уменьшить выбранный параметр ТПВ */

    SetParamDownTPV();

    ans=99;

    break;

case oesr_ir_param_set:          /*!< установить значение регулируемого
параметра ТПВ */

    break;

case oesr_ir_pic_inv:            /*!< переключить позитив/негатив ТПВ */

    ans=this->PositivNegativeTPV(msg.data);

    break;

case oesr_ir_hf_fix:             /*!< включить/выключить ВЧ коррекцию */

    ans=HfCorrection(msg.data);

    break;

case oesr_ir_zoom:               /*!< переключить узкое/узкое*2/широкое поле
ТПВ канала */

    ans=SetFieldTPV(msg.data);

```

```

        break;

    }

    switch(ans) // код ответного сообщения оэср: 0 -команда принята к
    исполнению, 1-команда не поддерживается, 2-команда не может быть выполнена прямо
    сейчас (абонент временно занят), 3-команда не может быть выполнена в данном
    режиме работы, 4-ошибка в параметрах команды, 99- ответное сообщение не
    требуется

    {

    case 0:

        sendCommandAccepted(msg, 23);

        break;

    case 1:

        sendCommanNotSupported(msg, 23);

        break;

    case 2:

        sendCommanNotExecuteNow(msg, 23);

        break;

    case 3:

        sendCommandNotExecuteMode(msg, 23);

        break;

    case 4:

        sendCommandOptionError(msg, 23);

        break;

    case 99:

        break;

    }

}

void mod_oesr::processCommandLD(const CanMsg &msg)

{

```

```

//qDebug() <<msg.hdr.v2.rcv<<" "<<msg.hdr.v2.dsr;

int ans = 0;

switch(msg.hdr.v2.dsr)
{
case oesr_set_strob:                /*!< установить строб */

    ans=SetStrob(msg.data);

    break;

case oesr_measure:                 /*!< измерить дальность */

    ans=MeasureDistance();

    break;

case oesr_dist_select:             /*!< выбрать дальность */

    ans=SelectDistance();

    break;

case oesr_dist_get:                /*!< запросить дальность */

    ans=QueryDistance();

    break;

}

switch(ans) // код ответного сообщения оэср: 0 -команда принята к
исполнению, 1-команда не поддерживается, 2-команда не может быть выполнена прямо
сейчас (абонент временно занят), 3-команда не может быть выполнена в данном
режиме работы, 4-ошибка в параметрах команды, 99- ответное сообщение не
требуется

{

case 0:

    sendCommandAccepted(msg, 10);

    break;

case 1:

    sendCommanNotSupported(msg,10);

    break;

case 2:

```

```

        sendCommanNotExecuteNow(msg,10);

        break;

    case 3:

        sendCommandNotExecuteMode(msg,10);

        break;

    case 4:

        sendCommandOptionError(msg,10);

        break;

    case 99:

        break;

    }

}

void mod_oesr::sendCommandAccepted(const CanMsg &msg, int snd)
{

    quint16 data = msg.hdr.v2.dsr;

    CanMsg m(snd, 17,5, (quint8*)&data, sizeof(data));

    msgSend(&m);

}

void mod_oesr::sendCommandNotExecuteMode(const CanMsg &msg, int snd)
{

    quint16 data = msg.hdr.v2.dsr;

    CanMsg m(snd, 21,5, (quint8*)&data, sizeof(data));

    msgSend(&m);

}

void mod_oesr::sendCommanNotSupported(const CanMsg &msg, int snd)

```

```

{

    quint16 data = msg.hdr.v2.dsr;

    CanMsg m(snd, 19,5, (quint8*)&data, sizeof(data));

    msgSend(&m);

}

void mod_oesr::sendCommanNotExecuteNow(const CanMsg &msg, int snd)

{

    quint16 data = msg.hdr.v2.dsr;

    CanMsg m(snd, 20,5, (quint8*)&data, sizeof(data));

    msgSend(&m);

}

void mod_oesr::sendCommandOptionError(const CanMsg &msg, int snd)

{

    quint16 data = msg.hdr.v2.dsr;

    CanMsg m(snd, 22,5, (quint8*)&data, sizeof(data));

    msgSend(&m);

}

void mod_oesr::setForm()

{

    if (oesrTV.mode && !oesrTV.adjust && !oesrTV.adjustPKT)    // рабочий режим

    {

        if (ui->pushButtonTV->isChecked())    // отображать ТВ канал

        {

            background->setPixmap(QPixmap::fromImage(*bcktv));

            if(oesrTV.protectCover)    //защитная крышка открыта

```

```

{

    ui->labelCoverPos->setText (QString::fromUtf8 ("3К ОТКРЫТА"));

    background->setVisible (true);

    ui->labelDistCount->setText ("0");

    if (oesrLD.distance0)

        ui->labelDistCount->setText ("1");

    if (oesrLD.distance1)

        ui->labelDistCount->setText ("2");

    if (oesrLD.distance2)

        ui->labelDistCount->setText ("3");

    if (oesrLD.distance3)

        ui->labelDistCount->setText ("4");

    switch (oesrLD.currdist)
    {

    case 1:

        ui->labelDistance-
>setText (QString::number (oesrLD.distance0));

        ui->labelDistPos->setText ("1");

        break;

    case 2:

        ui->labelDistance-
>setText (QString::number (oesrLD.distance1));

        ui->labelDistPos->setText ("2");

        break;

    case 3:

        ui->labelDistance-
>setText (QString::number (oesrLD.distance2));

        ui->labelDistPos->setText ("3");

        break;

    case 4:

```

```

        ui->labelDistance-
>setText (QString::number (oesrLD.distance3));

        ui->labelDistPos->setText ("4");

        break;

    }

}

else          //защитная крышка закрыта

{

    ui->labelCoverPos->setText (QString::fromUtf8 ("ЗК ЗАКРЫТА"));

    background->setVisible (false);

}

ui->progressBarScale->reset ();

switch (oesrTV.currentParam)    //регулируемый параметр ТВ

{

case prm_brightness:

    ui->labelPic->setPixmap (QPixmap (":/new/icons/bright.png"));

    ui->progressBarScale->setMaximum (255);

    ui->progressBarScale->setValue (oesrTV.light);

    break;

case prm_mark_brightness:

    ui->labelPic-
>setPixmap (QPixmap (":/new/icons/brightness.png"));

    ui->progressBarScale->setMaximum (1);

    ui->progressBarScale->setValue (oesrTV.brightness);

    break;

case prm_mark_vert:

    ui->labelPic->setPixmap (QPixmap (":/new/icons/updown.png"));

    ui->progressBarScale->setMaximum (128);

    ui->progressBarScale->setValue (oesrTV.updown);

```

```

        break;

    case prm_mark_horz:

        ui->labelPic-
>setPixmap(QPixmap(":/new/icons/leftright.png"));

        ui->progressBarScale->setMaximum(128);

        ui->progressBarScale->setValue(oesrTV.leftright);

        break;

    }

    //индикация

    markld->setVisible(oesrTV.indication.cross);

    markld->setPos(oesrTV.leftright*6,oesrTV.updown*4.5);

    marknarrowpkt-
>setVisible(oesrTV.indication.scalepkt&&oesrTV.field);

    markwidepkt-
>setVisible(oesrTV.indication.scalepkt&&(!oesrTV.field));

    markwidepkt->setPos(oesrTV.leftrightPKT*3-
384,oesrTV.updownPKT*2.25/*-288*/);

    marknarrowpkt->setScale(1);

    marknarrowpkt->setPos(0,0);

    ui->frameTop->setVisible(oesrTV.indication.info);

    ui->frameBottom->setVisible(oesrTV.indication.info);

    //добавить индикацию зон запрета стрельбы

    //яркость марки

    if(oesrTV.brightness)        // белое

    {

        markld->setColor(Qt::white);

        marknarrowfield->setColor(Qt::white);

        markwidepkt->setColor(Qt::white);

        marknarrowpkt->setColor(Qt::white);

```

```

    }

    else //черное

    {

        markld->setColor(Qt::black);

        marknarrowfield->setColor(Qt::black);

        markwidepkt->setColor(Qt::black);

        marknarrowpkt->setColor(Qt::black);

    }

    // увеличить уменьшить поле ТВ

    //                                     ui->labelVisionField-
>removeAction(actionFieldWideTPV);

    //                                     ui->labelVisionField-
>removeAction(actionFieldNarrowTPV);

    //                                     ui->labelVisionField-
>removeAction(actionFieldNarrowX2TPV);

    //                                     ui->labelVisionField-
>removeAction(actionFieldSaveTPV);

    //                                     ui->labelVisionField-
>removeAction(actionFieldInitTPV);

    //          ui->labelVisionField->removeAction(actionFieldOffTPV);

    //          ui->labelVisionField->addAction(actionFieldWideTV);

    //          ui->labelVisionField->addAction(actionFieldNarrowTV);

    switch (oesrTV.field)

    {

    case 0:

        background->setScale(1);

        background->setPos(0,0);

        marknarrowfield->setVisible(oesrTV.indication.info);

        ui->labelVisionField->setText(QString::fromUtf8("ШИПОКОЕ"));

        //          actionFieldWideTV->setDisabled(true);

        //          actionFieldNarrowTV->setDisabled(false);

```

```

        break;

case 1:

    background->setScale(2);

    background->setPos(-400,-300);

    marknarrowfield->setVisible(false);

    ui->labelVisionField->setText(QString::fromUtf8("V3KOE"));

    //                actionFieldWideTV->setDisabled(false);

    //                actionFieldNarrowTV->setDisabled(true);

    break;

}


ui->labelChanType->setText(QString::fromUtf8("TB"));

//                actionChannelTV->setDisabled(true);

//                actionChannelTPV->setDisabled(false);

ui->labelMode->setText("                ");

ui->labelHFCorr->setText("                ");


//  способ

if(oesrLD.strob)

{

    ui->labelStrobValue->setText(QString::number(oesrLD.strob));

}

else

{

    ui->labelStrobValue->setText("0000");

}

}

```

```

else          // отображать ТПВ канал
{
    if(oesrTPV.state == tpv_work)          //ТПВ включен
    {
        ui->labelChanType->setText(QString::fromUtf8("ТПВ"));

        //          actionChannelTV->setDisabled(false);

        //          actionChannelTPV->setDisabled(true);

        if(oesrTV.protectCover)          //защитная крышка открыта
        {
            ui->labelCoverPos->setText(QString::fromUtf8("3К
ОТКРЫТА"));

            background->setVisible(true);

            if(oesrLD.distance0)

                ui->labelDistCount->setText("1");

            if(oesrLD.distance1)

                ui->labelDistCount->setText("2");

            if(oesrLD.distance2)

                ui->labelDistCount->setText("3");

            if(oesrLD.distance3)

                ui->labelDistCount->setText("4");

            switch(oesrLD.currdist)
            {

            case 1:

                ui->labelDistance-
>setText(QString::number(oesrLD.distance0));

                ui->labelDistPos->setText("1");

                break;

            case 2:

```

```

        ui->labelDistance-
>setText(QString::number(oesrLD.distance1));

        ui->labelDistPos->setText("2");

        break;

    case 3:

        ui->labelDistance-
>setText(QString::number(oesrLD.distance2));

        ui->labelDistPos->setText("3");

        break;

    case 4:

        ui->labelDistance-
>setText(QString::number(oesrLD.distance3));

        ui->labelDistPos->setText("4");

        break;

    }

}

else //защитная крышка закрыта

{

    ui->labelCoverPos->setText(QString::fromUtf8("3К
ЗАКРЫТА"));

    background->setVisible(false);

}

// ui->labelVisionField-
>setText(QString::fromUtf8("ШИРОКОЕ"));

// ui->labelVisionField-
>addAction(actionFieldWideTPV);

// ui->labelVisionField-
>addAction(actionFieldNarrowTPV);

// ui->labelVisionField-
>addAction(actionFieldNarrowX2TPV);

// ui->labelVisionField-
>addAction(actionFieldSaveTPV);

// ui->labelVisionField-
>addAction(actionFieldInitTPV);

```

```

//                                     ui->labelVisionField-
>addAction(actionFieldOffTPV);

//                                     ui->labelVisionField-
>removeAction(actionFieldWideTV);

//                                     ui->labelVisionField-
>removeAction(actionFieldNarrowTV);

//                                     actionFieldWideTPV->setDisabled(true);

if(oesrTPV.pict) //негатив
{
    background->setPixmap(QPixmap::fromImage(*bcktpvneg));
}

else //позитив
{
    background->setPixmap(QPixmap::fromImage(*bcktpvpos));
}

switch (oesrTPV.currentParam)
{
case pi_focus:
    ui->labelPic-
>setPixmap(QPixmap(":/new/icons/focus.png"));

    ui->progressBarScale->setMaximum(54);

    ui->progressBarScale->setValue(oesrTPV.focus);

    break;

case pi_contrast:
    ui->labelPic-
>setPixmap(QPixmap(":/new/icons/contrast.png"));

    ui->progressBarScale->setMaximum(8);

    ui->progressBarScale->setValue(oesrTPV.contrast);

```

```

        break;

    case pi_mark_brightness:

        ui->labelPic-
>setPixmap(QPixmap(":/new/icons/brightness.png"));

        ui->progressBarScale->setMaximum(1);

        ui->progressBarScale->setValue(oesrTPV.brightness);

        break;

    case pi_mark_vert:

        ui->labelPic-
>setPixmap(QPixmap(":/new/icons/updown.png"));

        ui->progressBarScale->setMaximum(128);

        ui->progressBarScale->setValue(oesrTV.updown);

        break;

    case pi_mark_horz:

        ui->labelPic-
>setPixmap(QPixmap(":/new/icons/leftright.png"));

        ui->progressBarScale->setMaximum(128);

        ui->progressBarScale->setValue(oesrTV.leftright);

        break;

}

//индикация

markld->setVisible(oesrTV.indication.cross);

markld->setPos(oesrTV.leftright*6,oesrTV.updown*4.5);

//
>setVisible(oesrTV.indication.scalepkt&&(!oesrTV.field));          markwidepkt-

ui->frameTop->setVisible(oesrTV.indication.info);

ui->frameBottom->setVisible(oesrTV.indication.info);

```

```

// TODO добавить индикацию зон запрета стрельбы

//яркость марки

if(oesrTPV.brightness)          // белое
{
    markld->setColor(Qt::white);

    marknarrowfield->setColor(Qt::white);

    markwidepkt->setColor(Qt::white);

    marknarrowpkt->setColor(Qt::white);
}

else                             //черное
{
    markld->setColor(Qt::black);

    marknarrowfield->setColor(Qt::black);

    markwidepkt->setColor(Qt::black);

    marknarrowpkt->setColor(Qt::black);
}

//вч коррекция

if(oesrTPV.hffix)                //включена
{
    ui->labelHFCorr->setText(QString::fromUtf8("КОИТYP"));
}

else
{
    ui->labelHFCorr->setText(QString::fromUtf8(""));
}

```

```

// увеличить уменьшить поле тпв

switch (oesrTPV.field)
{
case tpv_wide:

    background->setScale(1);

    background->setPos(0,0);

    marknarrowfield->setVisible(oesrTV.indication.info);

    markwidepkt->setVisible(oesrTV.indication.scalepkt);

    markwidepkt->setPos(oesrTV.leftrightPKT*3-
384,oesrTV.updownPKT*2.25-288);

    marknarrowpkt->setVisible(false);

    marknarrowpkt->setScale(1);

    marknarrowpkt->setPos(0,0);

    ui->labelVisionField-
>setText(QString::fromUtf8("ШИПОКОЕ"));

        //                actionFieldWideTPV->setDisabled(true);

        //                actionFieldNarrowTPV-
>setDisabled(false);

        //                actionFieldNarrowX2TPV-
>setDisabled(false);

    break;

case tpv_narrow:

    background->setScale(2);

    background->setPos(-400,-300);

    marknarrowfield->setVisible(oesrTV.indication.info);

    markwidepkt->setVisible(false);

    marknarrowpkt->setVisible(oesrTV.indication.scalepkt);

    marknarrowpkt->setScale(1);

```

```

        marknarrowpkt->setPos(0,0);

        ui->labelVisionField-
>setText(QString::fromUtf8("Y3KOE"));

        //                actionFieldWideTPV->setDisabled(false);

        //                actionFieldNarrowTPV->setDisabled(true);

        //                actionFieldNarrowX2TPV-
>setDisabled(false);

        break;

    case tpv_narrowx2:

        background->setScale(4);

        background->setPos(-1150,-900);

        marknarrowfield->setVisible(false);

        markwidepkt->setVisible(false);

        marknarrowpkt->setVisible(oesrTV.indication.scalepkt);

        marknarrowpkt->setScale(1.8);

        marknarrowpkt->setPos(-300,-250);

        ui->labelVisionField->setText(QString::fromUtf8("Y3KOE
X2"));

        //                actionFieldWideTPV->setDisabled(false);

        //                actionFieldNarrowTPV-
>setDisabled(false);

        //                actionFieldNarrowX2TPV-
>setDisabled(true);

        break;

    case tpv_notready:

        break;

}

//  cтpоб

if(oesrLD.strob)

{

```

```

        ui->labelStrobValue->setText(QString::number(oesrLD.strob));
    }

    else

    {

        ui->labelStrobValue->setText("0000");

    }

    background->setVisible(true);

}

else //ТПВ ВЫКЛЮЧЕН

{

    ui->labelVisionField->setText(QString::fromUtf8("ВЫКЛЮЧЕН"));

    background->setVisible(false);

    marknarrowfield->setVisible(false);

    markld->setVisible(false);

    marknarrowpkt->setVisible(false);

    markwidepkt->setVisible(false);

    ui->labelHFCorr->setText("          ");

}

}

}

else if (oesrTV.mode && oesrTV.adjust && !oesrTV.adjustPKT) // выверка

{

    if (ui->pushButtonTV->isChecked()) // отображать ТВ канал

    {

        if(oesrTV.field) // ТВДУ

        {

            oesrTV.protectCover = 0;

            background->setPixmap(QPixmap::fromImage(*bckadjust));

```

```

        ui->labelCoverPos->setText(QString::fromUtf8("3К ЗАКРЫТА"));

        background->setVisible(true);

        marknarrowfield->setVisible(false);

    }

    else // ТВДШ

    {

        oesrTV.protectCover = 1;

        background->setPixmap(QPixmap::fromImage(*bcktv));

        ui->labelCoverPos->setText(QString::fromUtf8("3К ОТКРЫТА"));

        background->setVisible(true);

        marknarrowfield->setVisible(oesrTV.indication.info);

    }

    ui->progressBarScale->reset();

    switch (oesrTV.currentParam) //регулируемый параметр ТВ

    {

    case prm_brightness:

        ui->labelPic->setPixmap(QPixmap(":/new/icons/bright.png"));

        ui->progressBarScale->setMaximum(255);

        ui->progressBarScale->setValue(oesrTV.light);

        break;

    case prm_mark_brightness:

        ui->labelPic-
>setPixmap(QPixmap(":/new/icons/brightness.png"));

        ui->progressBarScale->setMaximum(1);

        ui->progressBarScale->setValue(oesrTV.brightness);

        break;

    case prm_mark_vert:

```

```

        ui->labelPic->setPixmap(QPixmap(":/new/icons/updown.png"));

        ui->progressBarScale->setMaximum(128);

        ui->progressBarScale->setValue(oesrTV.updown);

        break;

    case prm_mark_horz:

        ui->labelPic-
>setPixmap(QPixmap(":/new/icons/leftright.png"));

        ui->progressBarScale->setMaximum(128);

        ui->progressBarScale->setValue(oesrTV.leftright);

        break;

    }

    //индикация

    markld->setVisible(true);

    markld->setPos(oesrTV.leftright*6,oesrTV.updown*4.5);


    marknarrowpkt->setVisible(false);

    markwidepkt->setVisible(false);

    markwidepkt->setPos(oesrTV.leftrightPKT*3-
384,oesrTV.updownPKT*2.25-288);

    marknarrowpkt->setScale(1);

    marknarrowpkt->setPos(0,0);

    ui->frameTop->setVisible(oesrTV.indication.info);

    ui->frameBottom->setVisible(oesrTV.indication.info);

    // TODO добавить индикацию зон запрета стрельбы


    //яркость марки

    if(oesrTV.brightness||oesrTV.field)        // белое

    {

        markld->setColor(Qt::white);

```

```

        marknarrowfield->setColor(Qt::white);

        markwidepkt->setColor(Qt::white);

        marknarrowpkt->setColor(Qt::white);
    }

    else                                     //черное

    {

        markld->setColor(Qt::black);

        marknarrowfield->setColor(Qt::black);

        markwidepkt->setColor(Qt::black);

        marknarrowpkt->setColor(Qt::black);
    }


    ui->labelChanType->setText(QString::fromUtf8("TB"));

    //                actionChannelTV->setDisabled(true);

    //                actionChannelTPV->setDisabled(false);

    ui->labelMode->setText(QString::fromUtf8("ВЫБЕРКА"));

    ui->labelHFCorr->setText("                ");

}

else                                     // отображать ТПВ канал

{

    if(oesrTPV.state == tpv_work)         //ТПВ включен

    {

        ui->labelChanType->setText(QString::fromUtf8("ТПВ"));

        //                actionChannelTV->setDisabled(false);

        //                actionChannelTPV->setDisabled(true);

        ui->labelMode->setText(QString::fromUtf8("ВЫБЕРКА"));


        ui->labelVisionField->setText(QString::fromUtf8("ШИРОКОЕ"));
    }
}

```

```

        if (oesrTPV.pict)           //негатив
        {
            background->setPixmap(QPixmap::fromImage(*bcktpvneg));
        }

        else //позитив
        {
            background->setPixmap(QPixmap::fromImage(*bcktpvpos));
        }

        ui->progressBarScale->reset();

        switch (oesrTPV.currentParam)
        {
            case pi_focus:
                ui->labelPic-
>setPixmap(QPixmap(":/new/icons/focus.png"));

                ui->progressBarScale->setMaximum(54);

                ui->progressBarScale->setValue(oesrTPV.focus);

                break;

            case pi_contrast:
                ui->labelPic-
>setPixmap(QPixmap(":/new/icons/contrast.png"));

                ui->progressBarScale->setMaximum(8);

                ui->progressBarScale->setValue(oesrTPV.contrast);

                break;

            case pi_mark_brightness:
                ui->labelPic-
>setPixmap(QPixmap(":/new/icons/brightness.png"));

                ui->progressBarScale->setMaximum(1);

                ui->progressBarScale->setValue(oesrTPV.brightness);

```

```

        break;

    case pi_mark_vert:

        ui->labelPic-
>setPixmap(QPixmap(":/new/icons/updown.png"));

        ui->progressBarScale->setMaximum(128);

        ui->progressBarScale->setValue(oesrTV.updown);

        break;

    case pi_mark_horz:

        ui->labelPic-
>setPixmap(QPixmap(":/new/icons/leftright.png"));

        ui->progressBarScale->setMaximum(128);

        ui->progressBarScale->setValue(oesrTV.leftright);

        break;

}

```

```

markld->setVisible(true);

markld->setPos(oesrTV.leftright*6,oesrTV.updown*4.5);

marknarrowpkt->setVisible(false);

markwidepkt->setVisible(false);

ui->frameTop->setVisible(oesrTV.indication.info);

ui->frameBottom->setVisible(oesrTV.indication.info);

marknarrowfield->setVisible(false);

```

```

//яркость марки

```

```

if(oesrTPV.brightness)          // белое

```

```

{

```

```

    markld->setColor(Qt::white);

```

```

        marknarrowfield->setColor(Qt::white);

        markwidepkt->setColor(Qt::white);

        marknarrowpkt->setColor(Qt::white);

    }

    else                                     //черное

    {

        markld->setColor(Qt::black);

        marknarrowfield->setColor(Qt::black);

        markwidepkt->setColor(Qt::black);

        marknarrowpkt->setColor(Qt::black);

    }


    //вч коррекция

    if (oesrTPV.hffix)                     //включена

    {

        ui->labelHFCorr->setText(QString::fromUtf8("КОИТУР"));

    }

    else

    {

        ui->labelHFCorr->setText(QString::fromUtf8("      "));

    }

    background->setVisible(true);

}

else                                     //ТПВ ВЫКЛЮЧЕН

{

    ui->labelVisionField->setText(QString::fromUtf8("ВЫКЛЮЧЕН"));

    background->setVisible(false);

    marknarrowfield->setVisible(false);

```

```

        markld->setVisible(false);

        marknarrowpkt->setVisible(false);

        markwidepkt->setVisible(false);

        ui->labelHFCorr->setText("                ");

    }

}

else if(oesrTV.mode && !oesrTV.adjust && oesrTV.adjustPKT)    //выверка
ПКТ
{

    if (ui->pushButtonTV->isChecked())    // отображать ТВ канал

    {

        oesrTV.protectCover = 1;

        background->setPixmap(QPixmap::fromImage(*bcktv));

        ui->labelCoverPos->setText(QString::fromUtf8("ЗК ОТКРЫТА"));

        background->setVisible(true);

        marknarrowfield->setVisible(false);

        ui->progressBarScale->reset();

        switch (oesrTV.currentParam)    //регулируемый параметр ТВ

        {

        case prm_brightness:

            ui->labelPic->setPixmap(QPixmap(":/new/icons/bright.png"));

            ui->progressBarScale->setMaximum(255);

            ui->progressBarScale->setValue(oesrTV.light);

            break;

        case prm_mark_brightness:

            ui->labelPic-
>setPixmap(QPixmap(":/new/icons/brightness.png"));

            ui->progressBarScale->setMaximum(1);

```

```

        ui->progressBarScale->setValue(oesrTV.brightness);

        break;

    case prm_mark_vert:

        ui->labelPic->setPixmap(QPixmap(":/new/icons/updown.png"));

        ui->progressBarScale->setMaximum(255);

        ui->progressBarScale->setValue(oesrTV.updownPKT);

        break;

    case prm_mark_horz:

        ui->labelPic-
>setPixmap(QPixmap(":/new/icons/leftright.png"));

        ui->progressBarScale->setMaximum(255);

        ui->progressBarScale->setValue(oesrTV.leftrightPKT);

        break;

    }

    //индикация

    markld->setVisible(false);

    markld->setPos(oesrTV.leftright*6,oesrTV.updown*4.5);


    marknarrowpkt->setVisible(false);

    markwidepkt->setVisible(true);

    markwidepkt->setPos(oesrTV.leftrightPKT*3-
384,oesrTV.updownPKT*2.25);

    marknarrowpkt->setScale(1);

    marknarrowpkt->setPos(0,0);

    ui->frameTop->setVisible(oesrTV.indication.info);

    ui->frameBottom->setVisible(oesrTV.indication.info);


    //яркость марки

    if(oesrTV.brightness||oesrTV.field)        // белое

```

```

        {

            markld->setColor(Qt::white);

            marknarrowfield->setColor(Qt::white);

            markwidepkt->setColor(Qt::white);

            marknarrowpkt->setColor(Qt::white);

        }

        else                                     //черное

        {

            markld->setColor(Qt::black);

            marknarrowfield->setColor(Qt::black);

            markwidepkt->setColor(Qt::black);

            marknarrowpkt->setColor(Qt::black);

        }

//яркость марки

if(oesrTPV.brightness)                         // белое

{

    markld->setColor(Qt::white);

    marknarrowfield->setColor(Qt::white);

    markwidepkt->setColor(Qt::white);

    marknarrowpkt->setColor(Qt::white);

}

else                                     //черное

{

    markld->setColor(Qt::black);

    marknarrowfield->setColor(Qt::black);

    markwidepkt->setColor(Qt::black);

    marknarrowpkt->setColor(Qt::black);

}

```

```

//вч коррекция

if (oesrTPV.hffix)          //включена

{

ui->labelHFCorr->setText (QString::fromUtf8 ("КОНТУР"));

}

else

{

ui->labelHFCorr->setText (QString::fromUtf8 ("          "));

}

background->setVisible (true);

}

else          //ТПВ ВЫКЛЮЧЕН

{

ui->labelVisionField->setText (QString::fromUtf8 ("ВЫКЛЮЧЕН"));

background->setVisible (false);

marknarrowfield->setVisible (false);

markld->setVisible (false);

marknarrowpkt->setVisible (false);

markwidepkt->setVisible (false);

ui->labelHFCorr->setText ("          ");

}

}

}

else if (!oesrTV.mode && !oesrTV.adjust && !oesrTV.adjustPKT)          //дежурный

{

ui->labelChanType->setText (QString::fromUtf8 ("    "));

ui->labelVisionField->setText (QString::fromUtf8 ("    "));

ui->labelMode->setText (QString::fromUtf8 ("ДЕЖУРНЫЙ"));

```

```

ui->labelCoverPos->setText(QString::fromUtf8("3К ЗАКРЫТА"));

ui->labelHFCorr->setText("          ");

ui->labelPic->setPixmap(QPixmap(":/new/icons/bright.png"));

ui->progressBarScale->setMaximum(255);

ui->progressBarScale->setValue(oesrTV.light);

background->setVisible(false);

markld->setVisible(false);

marknarrowfield->setVisible(false);

markwidepkt->setVisible(false);

marknarrowpkt->setVisible(false);

}

else

{

qDebug()<< QString::fromUtf8("Выключено все") <<oesrTV.mode <<oesrTV.adjust
<<oesrTV.adjustPKT ;

}

setState();

}

void mod_oesr::setState()

{

ui->comboBoxSMODE->setCurrentIndex(oesrTV.mode);

ui->comboBoxSAdjust->setCurrentIndex(oesrTV.adjust);

ui->comboBoxSAdjustPKT->setCurrentIndex(oesrTV.adjustPKT);

switch (oesrTV.currentParam)

{

case prm_brightness:

ui->comboBoxSCurrParam->setCurrentIndex(0);

break;

```

```

case prm_mark_brightness:

ui->comboBoxSCurrParam->setCurrentIndex(1);

break;

case prm_mark_vert:

ui->comboBoxSCurrParam->setCurrentIndex(2);

break;

case prm_mark_horz:

ui->comboBoxSCurrParam->setCurrentIndex(3);

break;

}

switch (oesrTPV.currentParam)

{

case pi_focus:

ui->comboBoxSCurrParamTPV->setCurrentIndex(0);

break;

case pi_contrast:

ui->comboBoxSCurrParamTPV->setCurrentIndex(1);

break;

case pi_mark_brightness:

ui->comboBoxSCurrParamTPV->setCurrentIndex(2);

break;

case pi_mark_vert:

ui->comboBoxSCurrParamTPV->setCurrentIndex(3);

break;

case pi_mark_horz:

ui->comboBoxSCurrParamTPV->setCurrentIndex(4);

break;

```

```

}

ui->comboBoxSServiceInfo->setCurrentIndex(oesrTV.indication.info);

ui->comboBoxSCross->setCurrentIndex(oesrTV.indication.cross);

ui->comboBoxSScalePKT->setCurrentIndex(oesrTV.indication.scalepkt);

ui->comboBoxSFirelock->setCurrentIndex(oesrTV.indication.firelock);

ui->comboBoxSTvField->setCurrentIndex(oesrTV.field);

ui->comboBoxSModeTPV->setCurrentIndex(oesrTPV.state);

ui->comboBoxSTPVField->setCurrentIndex(oesrTPV.field);

ui->comboBoxSPosNeg->setCurrentIndex(oesrTPV.pict);

ui->comboBoxSCorrVCH->setCurrentIndex(oesrTPV.hffix);

}

void mod_oesr::comboBoxChange(int index)

{

if(sender()==ui->comboBoxSMode)

{

if (oesrTV.mode != index)

{

oesrTV.mode = index;

if(index == 1)                // переходим в рабочий режим

{

oesrTV.protectCover = 1;      //открываем защитную крышку

oesrTV.adjust = 0;

oesrTV.adjustPKT = 0;

background->setVisible(true);

}

else if(index == 0)                // переходим в дежурный режим

```

```

{
    oesrTV.protectCover = 0;

    oesrTV.adjust = 0;

    oesrTV.adjustPKT = 0;

    background->setVisible(false);

}

}

}

else if (sender() == ui->comboBoxSAdjust)
{
    oesrTV.adjust = index;
}

else if (sender() == ui->comboBoxSAdjustPKT)
{
    switch(index)
    {
        case 0:
            oesrTV.adjustPKT = 0;

            break;

        case 1:
            oesrTV.adjustPKT = 1;

            oesrTV.adjust = 0;

            break;

    }

}

else if (sender() == ui->comboBoxSCurrParam)
{
    switch (index)

```

```

{

case 0:

oesrTV.currentParam = prm_brightness;

break;

case 1:

oesrTV.currentParam = prm_mark_brightness;

case 2:

if(oesrTV.adjust||oesrTV.adjustPKT)

oesrTV.currentParam = prm_mark_vert;

else

ui->comboBoxSCurrParam->setCurrentIndex(1);

break;

case 3:

if(oesrTV.adjust||oesrTV.adjustPKT)

oesrTV.currentParam = prm_mark_horz;

else

ui->comboBoxSCurrParam->setCurrentIndex(1);

break;

}

}

else if(sender()==ui->comboBoxSCurrParamTPV)

{

switch (index)

{

case 0:

oesrTPV.currentParam = pi_focus;

break;

case 1:

```

```

oesrTPV.currentParam = pi_contrast;

break;

case 2:

oesrTPV.currentParam = pi_mark_brightness;

break;

case 3:

if (oesrTV.adjust || oesrTV.adjustPKT)

oesrTPV.currentParam = pi_mark_vert;

else

ui->comboBoxSCurrParamTPV->setCurrentIndex(1);

break;

case 4:

if (oesrTV.adjust || oesrTV.adjustPKT)

oesrTPV.currentParam = pi_mark_horz;

else

ui->comboBoxSCurrParamTPV->setCurrentIndex(1);

break;

}

}

else if (sender() == ui->comboBoxSServiceInfo)

{

oesrTV.indication.info = index;

}

else if (sender() == ui->comboBoxSCross)

{

oesrTV.indication.cross = index;

}

else if (sender() == ui->comboBoxSScalePKT)

```

```

{
oesrTV.indication.scalepkt = index;
}

else if(sender()==ui->comboBoxSFirelock)
{
oesrTV.indication.firelock = index;
}

else if(sender()==ui->comboBoxSTvField)
{
oesrTV.field = index;
}

else if(sender()==ui->comboBoxSMoDeTPV)
{
switch (index)
{
case 0:
oesrTPV.state = tpv_off;
oesrTPV.hffix = 0;
break;

case 1:
oesrTPV.state = tpv_safemode;
oesrTPV.hffix = 0;
break;

case 2:
oesrTPV.state = tpv_init;
oesrTPV.hffix = 0;
break;

case 3:

```

```

oesrTPV.state = tpv_work;

oesrTPV.hffix = 0;

break;

}

oesrTPV.currentParam = pi_focus;

}

else if(sender() == ui->comboBoxSTPVField)

{

switch (index)

{

case 1:// широкое

oesrTPV.field = tpv_wide;

break;

case 2: //узкое

oesrTPV.field = tpv_narrow;

break;

case 3:

oesrTPV.field = tpv_narrowx2;

break;

}

}

else if(sender() == ui->comboBoxSPosNeg)

{

oesrTPV.pict = index;

}

else if(sender() == ui->comboBoxSCorrVCH)

{

oesrTPV.hffix = index;

```

```

}

this->setForm();

}

int mod_oesr::SetStandby(const quint8 *data)
{
    bool setmode = data[0];

    if (oesrTV.mode != setmode)
    {
        oesrTV.mode = setmode;

        if(setmode == 1)                // переходим в рабочий режим
        {
            oesrTV.protectCover = 1;    //открываем защитную крышку
            oesrTV.adjust = 0;
            oesrTV.adjustPKT = 0;
            background->setVisible(true);
        }

        else if(setmode == 0)           // переходим в дежурный режим
        {
            oesrTV.protectCover = 0;
            oesrTV.adjust = 0;
            oesrTV.adjustPKT = 0;
            background->setVisible(false);
        }

        else
            return 4;
    }

    this->setForm();
}

```

```
return 0;
```

```
}
```

```
int mod_oesr::SetAdjust(const quint8 *data)
```

```
{
```

```
if(oesrTV.mode ==0)
```

```
return 3;
```

```
switch(data[0])
```

```
{
```

```
case 0:
```

```
oesrTV.adjust = 0;
```

```
break;
```

```
case 1:
```

```
oesrTV.adjust = 1;
```

```
break;
```

```
default:
```

```
return 4;
```

```
}
```

```
this->setForm();
```

```
return 0;
```

```
}
```

```
int mod_oesr::SetAdjustPKT(const quint8 *data)
```

```
{
```

```
if(oesrTV.mode ==0)
```

```
return 3;
```

```
switch(data[0])
```

```
{
```

```

case 0:

oesrTV.adjustPKT = 0;

break;

case 1:

oesrTV.adjustPKT = 1;

oesrTV.adjust = 0;

break;

default:

return 4;

}

this->setForm();

return 0;

}

int mod_oesr::SelectCurrentParam()

{

if(oesrTV.mode ==0)

return 3;

switch (oesrTV.currentParam)

{

case prm_brightness:

oesrTV.currentParam = prm_mark_brightness;

break;

case prm_mark_brightness:

if(oesrTV.adjust||oesrTV.adjustPKT)

oesrTV.currentParam = prm_mark_vert;

else

oesrTV.currentParam = prm_brightness;

```

```

break;

case prm_mark_vert:

oesrTV.currentParam = prm_mark_horz;

break;

case prm_mark_horz:

oesrTV.currentParam = prm_brightness;

break;

}

this->setForm();

return 0;

}

int mod_oesr::SetParamUp()

{

if(oesrTV.mode ==0)

return 3;

switch (oesrTV.currentParam)

{

case prm_brightness:

if(oesrTV.light<255)

oesrTV.light++;

break;

case prm_mark_brightness:

oesrTV.brightness = !oesrTV.brightness;

break;

case prm_mark_vert:

if(oesrTV.adjustPKT)

{

```

```

if(oesrTV.updownPKT > 0)

oesrTV.updownPKT--;

}

else

if(oesrTV.updown > 0)

oesrTV.updown--;

break;

case prm_mark_horz:

if(oesrTV.adjustPKT)

{

if(oesrTV.leftrightPKT < 255)

oesrTV.leftrightPKT++;

}

else

if(oesrTV.leftright < 128)

oesrTV.leftright++;

break;

}

this->setForm();

return 99;

}

int mod_oesr::SetParamDown()

{

if(oesrTV.mode ==0)

return 3;

switch (oesrTV.currentParam)

{

```

```

case prm_brightness:

if(oesrTV.light>0)

oesrTV.light--;

break;

case prm_mark_brightness:

oesrTV.brightness = !oesrTV.brightness;

break;

case prm_mark_vert:

if(oesrTV.adjustPKT)

{

if(oesrTV.updownPKT < 255)

oesrTV.updownPKT++;

}

else

if(oesrTV.updown < 128)

oesrTV.updown++;

break;

case prm_mark_horz:

if(oesrTV.adjustPKT)

{

if(oesrTV.leftrightPKT > 0)

oesrTV.leftrightPKT--;

}

else

if(oesrTV.leftright > 0)

oesrTV.leftright--;

break;

}

```

```

this->setForm();

return 99;

}

int mod_oesr::SetCurrentParam(const quint8 *data)

{

if(oesrTV.mode ==0)

return 3;

switch (data[0])

{

case prm_brightness:

oesrTV.light=data[1];

break;

case prm_mark_brightness:

oesrTV.brightness =data[1];

break;

case prm_mark_vert:

if(oesrTV.adjustPKT)

{

oesrTV.updownPKT=data[1];

}

else if(oesrTV.adjust)

oesrTV.updown=data[1];

else

return 3;

break;

case prm_mark_horz:

if(oesrTV.adjustPKT)

```

```

{
    oesrTV.leftrightPKT=data[1];
}

else if(oesrTV.adjust)

    oesrTV.leftright=data[1];

else

    return 3;

break;

default:

    return 4;

}

this->setForm();

return 0;

}

int mod_oesr::SetIndication(const quint8 *data)

{

    if(oesrTV.mode ==0)

        return 3;

    oesrTV.indication.info = data[0]&1;

    oesrTV.indication.cross = ((data[0]&2)>>1);

    oesrTV.indication.scalepkt = ((data[0]&4)>>2);

    oesrTV.indication.firelock = ((data[0]&8)>>3);

    this->setForm();

    return 0;

}

int mod_oesr::SetFieldTV(const quint8 *data)

```

```

{

if(oesrTV.adjustPKT || (oesrTV.mode ==0))

return 3;

switch (data[0])

{

case 1:

oesrTV.field = 0;           // широкое

break;

case 2:

oesrTV.field = 1;           //узкое

break;

default:

return 4;

}

this->setForm();

return 0;

}

int mod_oesr::StateTest()

{

int mod_oesr::VariableZoneBan(const quint8 *data)

{

if(oesrTV.adjustPKT|| (oesrTV.mode ==0))

return 3;

oesrTV.varzone.x1 = (data[0] << 9) + data[1];

oesrTV.varzone.y1 =(data[2] << 9) + data[3];

oesrTV.varzone.x2 = (data[4] << 9) + data[5];

```

```

oesrTV.varzone.y2 =(data[6] << 9) + data[7];

return 0;

}

```

```

int mod_oesr::ConstZoneBan1(const quint8 *data)

{

if(oesrTV.adjustPKT || (oesrTV.mode ==0))

return 3;

oesrTV.constzone1.x1 = (data[0] << 9) + data[1];

oesrTV.constzone1.y1 =(data[2] << 9) + data[3];

oesrTV.constzone1.x2 = (data[4] << 9) + data[5];

oesrTV.constzone1.y2 =(data[6] << 9) + data[7];

return 0;

}

```

```

int mod_oesr::ConstZoneBan2(const quint8 *data)

{

if(oesrTV.adjustPKT || (oesrTV.mode ==0))

return 3;

oesrTV.constzone2.x1 = (data[0] << 9) + data[1];

oesrTV.constzone2.y1 =(data[2] << 9) + data[3];

oesrTV.constzone2.x2 = (data[4] << 9) + data[5];

oesrTV.constzone2.y2 =(data[6] << 9) + data[7];

return 0;

}

```

```

int mod_oesr::UserInfo1(const quint8 *data)

{

```

```
if(oesrTV.mode ==0)

return 3;

return 0;

}
```

```
int mod_oesr::UserInfo2(const quint8 *data)

{

if(oesrTV.mode ==0)

return 3;

return 0;

}
```

```
int mod_oesr::UserInfo3(const quint8 *data)

{

if(oesrTV.mode ==0)

return 3;

return 0;

}
```

```
int mod_oesr::UserInfo4(const quint8 *data)

{

if(oesrTV.mode ==0)

return 3;

return 0;

}
```

```
int mod_oesr::SetStrob(const quint8 *data)

{
```

```

if(oesrTV.adjust || oesrTV.adjustPKT || (oesrTV.mode ==0))

return 3;

oesrLD.strob = (data[1] << 8) + data[0];

//qDebug()<< "SetStrob " << oesrLD.strob;

this->setForm();

return 0;

}

```

```

int mod_oesr::MeasureDistance()

{

if(oesrTV.adjust || oesrTV.adjustPKT || (oesrTV.mode ==0))

return 3;

oesrLD.distance0 = dd->spinBoxDistance1->value();

oesrLD.distance1 = dd->spinBoxDistance2->value();

oesrLD.distance2 = dd->spinBoxDistance3->value();

oesrLD.distance3 = dd->spinBoxDistance4->value();

if (oesrLD.distance0)

oesrLD.currdist = 1;

ui->comboBoxSDDataAvail->setCurrentIndex(1);

this->setForm();

return 0;

}

```

```

int mod_oesr::SelectDistance()

{

if(oesrTV.adjust || oesrTV.adjustPKT || (oesrTV.mode ==0))

return 3;

switch (oesrLD.currdist)

```

```

{

case 1:

if(oesrLD.distance0>0)

oesrLD.currdist = 2;

break;

case 2:

if(oesrLD.distance1>0)

oesrLD.currdist = 3;

else

oesrLD.currdist = 1;

break;

case 3:

if(oesrLD.distance2>0)

oesrLD.currdist = 4;

else

oesrLD.currdist = 1;

break;

case 4:

oesrLD.currdist = 1;

break;

}

return 0;

}

int mod_oesr::QueryDistance()

{

if(oesrTV.adjust || oesrTV.adjustPKT || (oesrTV.mode ==0))

return 3;

```

```

quint8 data[8];

data[1] = oesrLD.distance0 >> 8;

data[0] = oesrLD.distance0&255;

data[3] = oesrLD.distance1 >> 8;

data[2] = oesrLD.distance1&255;

data[5] = oesrLD.distance2 >> 8;

data[4] = oesrLD.distance2&255;

data[7] = oesrLD.distance3 >> 8;

data[6] = oesrLD.distance3&255;


CanMsg msg(oesr_ld, 804,3, data, sizeof(data));

msgSend(&msg);

return 99;

}


int mod_oesr::OnOffTPV(const quint8 *data)

{

if(oesrTV.mode ==0)

return 3;

switch (data[0])

{

case 0:

oesrTPV.state = tpv_off;

oesrTPV.hffix = 0;

break;

case 1:

oesrTPV.state = tpv_work;

oesrTPV.hffix = 0;

```

```

break;

default:

return 4;

}

oesrTPV.currentParam = pi_focus;

this->setForm();

return 0;

}


int mod_oesr::SelectCurrentParamTPV()

{

if(oesrTV.mode ==0)

return 3;

switch (oesrTPV.currentParam)

{

case pi_focus:

oesrTPV.currentParam = pi_contrast;

break;

case pi_contrast:

oesrTPV.currentParam = pi_mark_brightness;

break;

case pi_mark_brightness:

if(oesrTV.adjust||oesrTV.adjustPKT)

oesrTPV.currentParam = pi_mark_vert;

else

oesrTPV.currentParam = pi_focus;

break;

case pi_mark_vert:

```

```

oesrTPV.currentParam = pi_mark_horz;

break;

case pi_mark_horz:

oesrTPV.currentParam = pi_focus;

break;

}

this->setForm();

return 0;

}

```

Листинг Б.2 – Файл mark_common.cpp

```

#include "mark_common.h"
#include <QDebug>

MarkCommon::MarkCommon() :
    QGraphicsItemGroup()
{
    lines = new QList<QGraphicsLineItem *>();
    texts = new QList<QGraphicsSimpleTextItem *>();
}

MarkCommon::~MarkCommon()
{
    QListIterator<QGraphicsLineItem *> i(*lines);
    while (i.hasNext())
    {
        delete i.next();
    }
    lines->clear();
    delete lines;

    foreach (QGraphicsItem* obj, childItems()) {
        // qDebug() << obj->data(0).toString() << obj;
        delete obj;
    }
    texts->clear();
    delete texts;
}

void MarkCommon::setColor(QColor color)
{
    QListIterator<QGraphicsLineItem *> i(*lines);
    while (i.hasNext())
    {
        i.next()->setPen(QPen(QBrush(QColor(color)), Qt::SolidPattern), 2));
    }
    QListIterator<QGraphicsSimpleTextItem *> t(*texts);
    while (t.hasNext())

```

```

        {
            t.next()->setPen(QPen(QBrush(QColor(color), Qt::SolidPattern), 2));
        }
    }

void MarkCommon::addItem(QGraphicsLineItem *line, QPen pen)
{
    lines->append(line);
    lines->last()->setPen(pen);
    this->addToGroup(lines->last());
}

void MarkCommon::addItem(QGraphicsSimpleTextItem *text, int x, int y, QPen
pen)
{
    texts->append(text);
    texts->last()->setPen(pen);
    texts->last()->setPos(x, y);

    this->addToGroup(texts->last());
}

```

Листинг Б.3 – Файл mark_id.cpp

```

#include "mark_id.h"

MarkLD::MarkLD() :
    MarkCommon()
{
    addItem( new QGraphicsLineItem(0,-50,0,-4,this),QPen(QBrush(Qt::black,
Qt::SolidPattern), 2));
    addItem(new QGraphicsLineItem(4,0,50,0,this),QPen(QBrush(Qt::black,
Qt::SolidPattern), 2));
    addItem(new QGraphicsLineItem(0,4,0,50,this),QPen(QBrush(Qt::black,
Qt::SolidPattern), 2));
    addItem( new QGraphicsLineItem(-50,0,-4,0,this),QPen(QBrush(Qt::black,
Qt::SolidPattern), 2));
}

MarkLD::~~MarkLD()
{
}

```

Листинг Б.4 – Файл mark_narrowfield.cpp

```

#include "mark_narrowfield.h"

MarkNarrowField::MarkNarrowField(int lenght) :
    MarkCommon()
{
    ln = lenght;

    addItem( new
QGraphicsLineItem(192,144+ln,192,144,this),QPen(QBrush(Qt::black,
Qt::SolidPattern), 2));
    addItem( new
QGraphicsLineItem(192,144,192+ln,144,this),QPen(QBrush(Qt::black,
Qt::SolidPattern), 2));
}

```

```

        addItem(new QGraphicsLineItem(576-
ln,144,576,144,this),QPen(QBrush(Qt::black, Qt::SolidPattern), 2));
        addItem(new
QGraphicsLineItem(576,144,576,144+ln,this),QPen(QBrush(Qt::black,
Qt::SolidPattern), 2));

        addItem(new QGraphicsLineItem(576,432-
ln,576,432,this),QPen(QBrush(Qt::black, Qt::SolidPattern), 2));
        addItem(new QGraphicsLineItem(576,432,576-
ln,432,this),QPen(QBrush(Qt::black, Qt::SolidPattern), 2));

        addItem(new
QGraphicsLineItem(192+ln,432,192,432,this),QPen(QBrush(Qt::black,
Qt::SolidPattern), 2));
        addItem(new QGraphicsLineItem(192,432,192,432-
ln,this),QPen(QBrush(Qt::black, Qt::SolidPattern), 2));

    }

MarkNarrowField::~MarkNarrowField()
{

}

```

Листинг Б.5 – Файл mark_narrowpkt.cpp

```

#include "mark_narrowpkt.h"

MarkNarrowPKT::MarkNarrowPKT()
{
    int gx = 420;
    int gy = 65;

    addItem( new QGraphicsLineItem(gx,gy,gx-
10,gy+15,this),QPen(QBrush(Qt::black, Qt::SolidPattern), 2));
    addItem( new
QGraphicsLineItem(gx,gy,gx+10,gy+15,this),QPen(QBrush(Qt::black,
Qt::SolidPattern), 2));
    addItem(new
QGraphicsSimpleTextItem("10",this),gx+15,gy+3,QPen(QBrush(Qt::black,
Qt::SolidPattern), 1));
    gy += 100;
    addItem( new QGraphicsLineItem(gx,gy,gx-
10,gy+15,this),QPen(QBrush(Qt::black, Qt::SolidPattern), 2));
    addItem( new
QGraphicsLineItem(gx,gy,gx+10,gy+15,this),QPen(QBrush(Qt::black,
Qt::SolidPattern), 2));
    addItem(new
QGraphicsSimpleTextItem("12",this),gx+15,gy+3,QPen(QBrush(Qt::black,
Qt::SolidPattern), 1));
    gy += 100;
    addItem( new QGraphicsLineItem(gx,gy,gx-
10,gy+15,this),QPen(QBrush(Qt::black, Qt::SolidPattern), 2));
    addItem( new
QGraphicsLineItem(gx,gy,gx+10,gy+15,this),QPen(QBrush(Qt::black,
Qt::SolidPattern), 2));
    addItem(new
QGraphicsSimpleTextItem("14",this),gx+15,gy+3,QPen(QBrush(Qt::black,
Qt::SolidPattern), 1));
}

```

```

        gy += 150;
        addItem( new QGraphicsLineItem(gx,gy,gx-
10,gy+15,this),QPen(QBrush(Qt::black, Qt::SolidPattern), 2));
        addItem( new
QGraphicsLineItem(gx,gy,gx+10,gy+15,this),QPen(QBrush(Qt::black,
Qt::SolidPattern), 2));
        addItem(new
QGraphicsSimpleTextItem("16",this),gx+15,gy+3,QPen(QBrush(Qt::black,
Qt::SolidPattern), 1));
    }

```

Листинг Б.6 – Файл mark_widepkt.cpp

```

#include "mark_widepkt.h"

MarkWidePKT::MarkWidePKT() :
    MarkCommon()
{
    int gx = 400;
    int sy = 45;
    int dy = 8;
    int ws = 40;
    int ns = 30;
    // вертикальная линия
    addItem(new QGraphicsLineItem(gx,0,gx,576,this),QPen(QBrush(Qt::black,
Qt::SolidPattern), 2));
    // квадратики сверху :)
    addItem( new QGraphicsLineItem(gx-
140,45,gx+140,45,this),QPen(QBrush(Qt::black, Qt::SolidPattern), 2));
    addItem(new QGraphicsLineItem(gx-35,10,gx-
35,45,this),QPen(QBrush(Qt::black, Qt::SolidPattern), 2));
    addItem(new QGraphicsLineItem(gx-70,10,gx-
70,45,this),QPen(QBrush(Qt::black, Qt::SolidPattern), 2));
    addItem(new QGraphicsLineItem(gx-105,10,gx-
105,45,this),QPen(QBrush(Qt::black, Qt::SolidPattern), 2));
    addItem(new QGraphicsLineItem(gx-140,10,gx-
140,45,this),QPen(QBrush(Qt::black, Qt::SolidPattern), 2));

    addItem(new
QGraphicsLineItem(gx+35,10,gx+35,45,this),QPen(QBrush(Qt::black,
Qt::SolidPattern), 2));
    addItem(new
QGraphicsLineItem(gx+70,10,gx+70,45,this),QPen(QBrush(Qt::black,
Qt::SolidPattern), 2));
    addItem(new
QGraphicsLineItem(gx+105,10,gx+105,45,this),QPen(QBrush(Qt::black,
Qt::SolidPattern), 2));
    addItem(new
QGraphicsLineItem(gx+140,10,gx+140,45,this),QPen(QBrush(Qt::black,
Qt::SolidPattern), 2));

    // горизонтальные линии
    addItem(new QGraphicsLineItem(gx-
ns,sy+dy,gx+ns,sy+dy,this),QPen(QBrush(Qt::black, Qt::SolidPattern), 2));
    addItem(new QGraphicsLineItem(gx-
ws,sy+dy*3,gx+ws,sy+dy*3,this),QPen(QBrush(Qt::black, Qt::SolidPattern), 2));
    addItem(new QGraphicsLineItem(gx-
ns,sy+dy*6,gx+ns,sy+dy*6,this),QPen(QBrush(Qt::black, Qt::SolidPattern), 2));
    addItem(new QGraphicsLineItem(gx-
ws,sy+dy*10,gx+ws,sy+dy*10,this),QPen(QBrush(Qt::black, Qt::SolidPattern),
2));

```

```

        addItem(new QGraphicsLineItem(gx-
ns,sy+dy*15,gx+ns,sy+dy*15,this),QPen(QBrush(Qt::black, Qt::SolidPattern),
2));
        addItem(new QGraphicsLineItem(gx-
ws,sy+dy*21,gx+ws,sy+dy*21,this),QPen(QBrush(Qt::black, Qt::SolidPattern),
2));
        addItem(new QGraphicsLineItem(gx-
ns,sy+dy*28,gx+ns,sy+dy*28,this),QPen(QBrush(Qt::black, Qt::SolidPattern),
2));
        addItem(new QGraphicsLineItem(gx-
ws,sy+dy*36,gx+ws,sy+dy*36,this),QPen(QBrush(Qt::black, Qt::SolidPattern),
2));
        addItem(new QGraphicsLineItem(gx-
ns,sy+dy*45,gx+ns,sy+dy*45,this),QPen(QBrush(Qt::black, Qt::SolidPattern),
2));
        addItem(new QGraphicsLineItem(gx-
ws,sy+dy*55,gx+ws,sy+dy*55,this),QPen(QBrush(Qt::black, Qt::SolidPattern),
2));

        //цифирки
        addItem(new QGraphicsSimpleTextItem("4",this),gx-ws-20,sy-
8+dy*3,QPen(QBrush(Qt::black, Qt::SolidPattern), 1));
        addItem(new QGraphicsSimpleTextItem("8",this),gx+ws+10,sy-
8+dy*10,QPen(QBrush(Qt::black, Qt::SolidPattern), 1));
        addItem(new QGraphicsSimpleTextItem("12",this),gx-ws-20,sy-
8+dy*21,QPen(QBrush(Qt::black, Qt::SolidPattern), 1));
        addItem(new QGraphicsSimpleTextItem("16",this),gx+ws+10,sy-
8+dy*36,QPen(QBrush(Qt::black, Qt::SolidPattern), 1));
        addItem(new QGraphicsSimpleTextItem("20",this),gx-ws-20,sy-
8+dy*55,QPen(QBrush(Qt::black, Qt::SolidPattern), 1));
    }

MarkWidePKT::~MarkWidePKT()
{

}

```

Приложение В
РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ
(рекомендуемое)

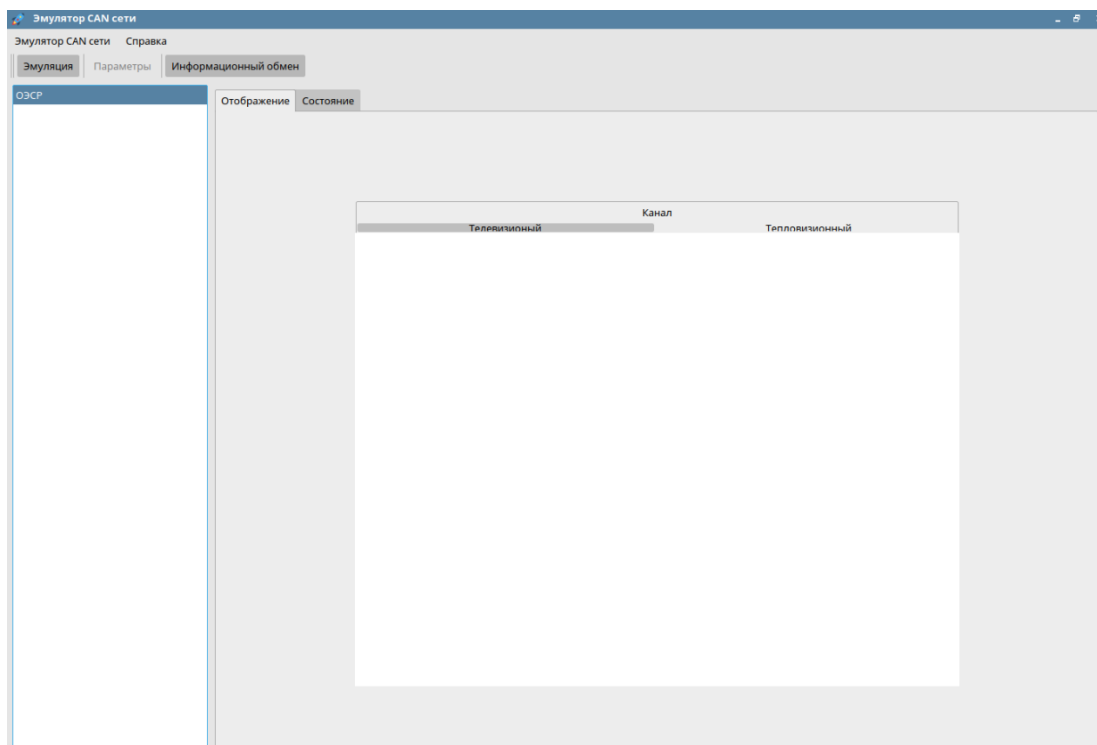


Рисунок В.1 – Состояние «включен»

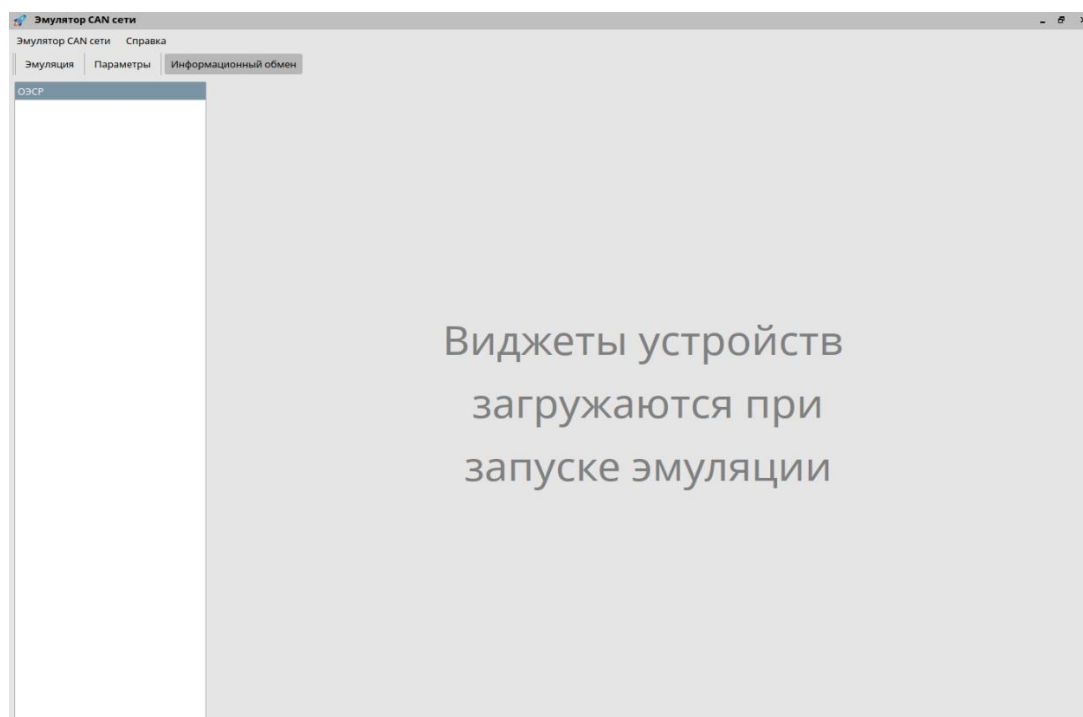


Рисунок В.2 – Состояние «выключен»

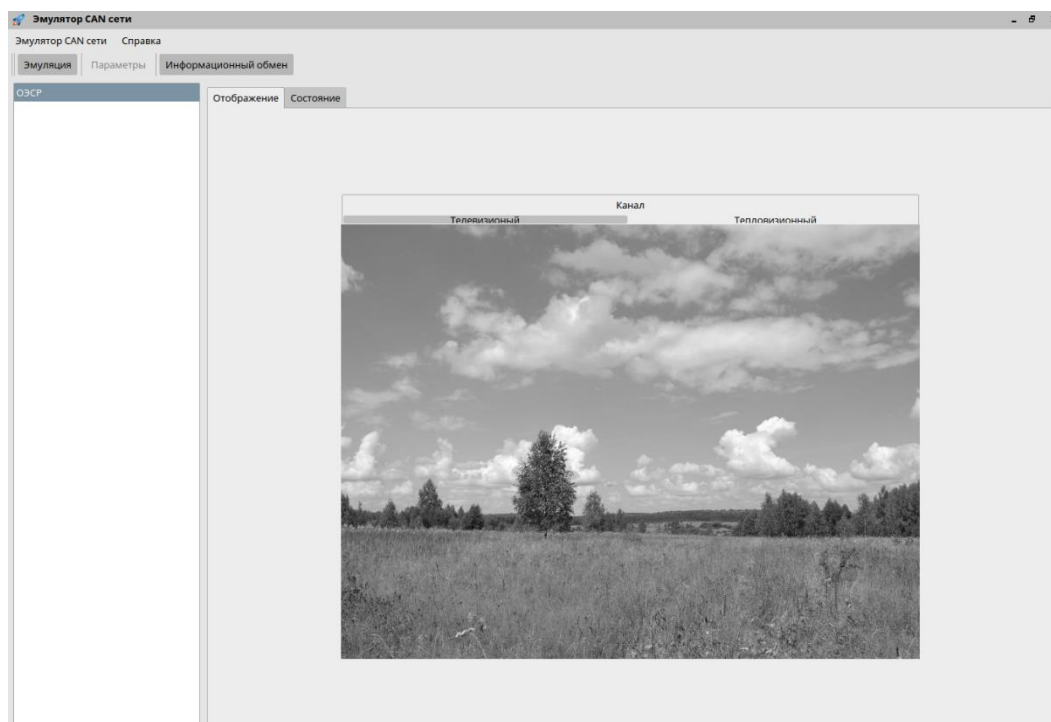


Рисунок В.3 – Состояние «Рабочий»

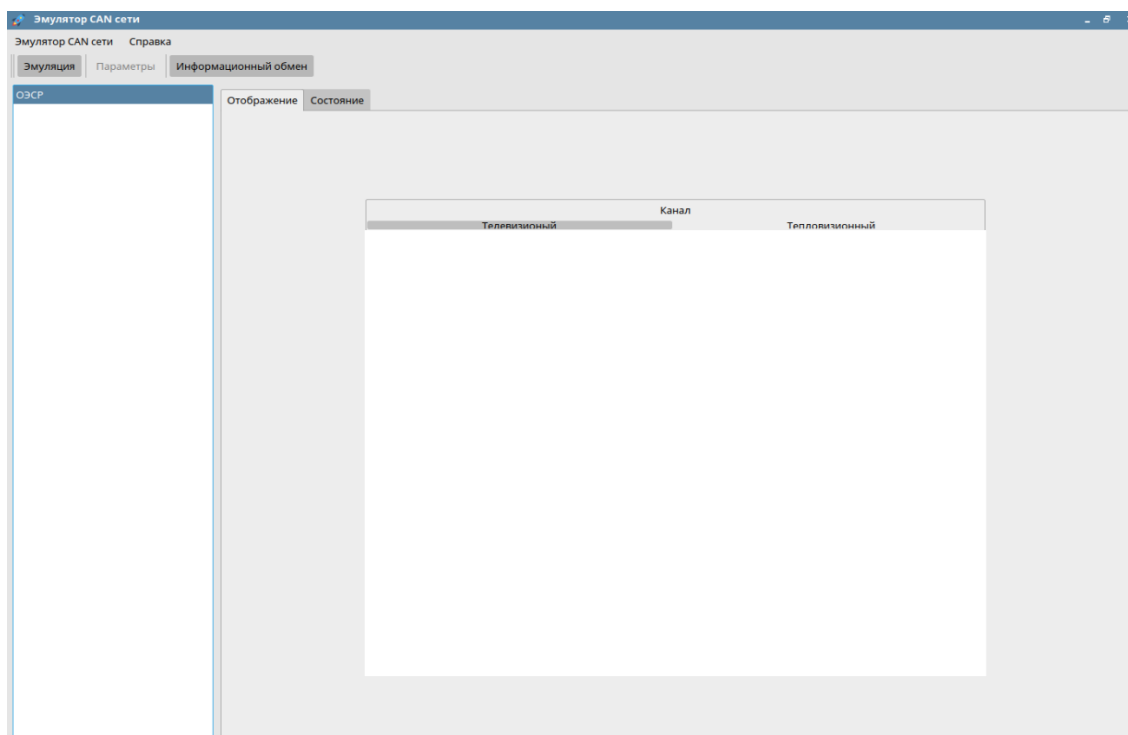


Рисунок В.4 – Состояние «Дежурный»

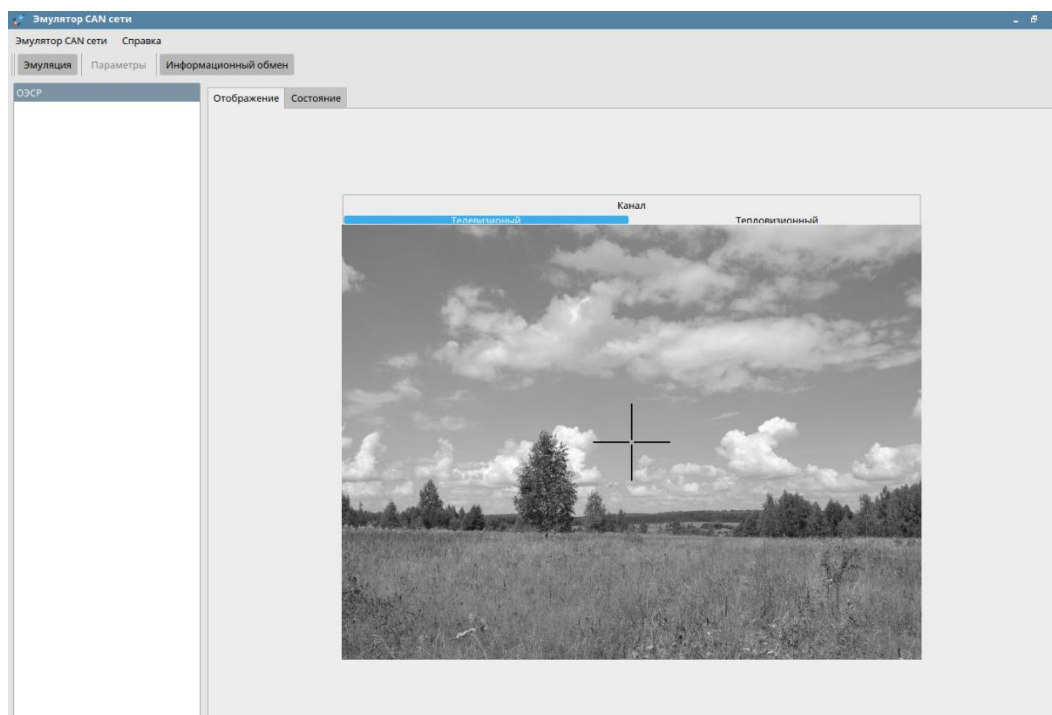


Рисунок В.5 – Состояние «Выверка»

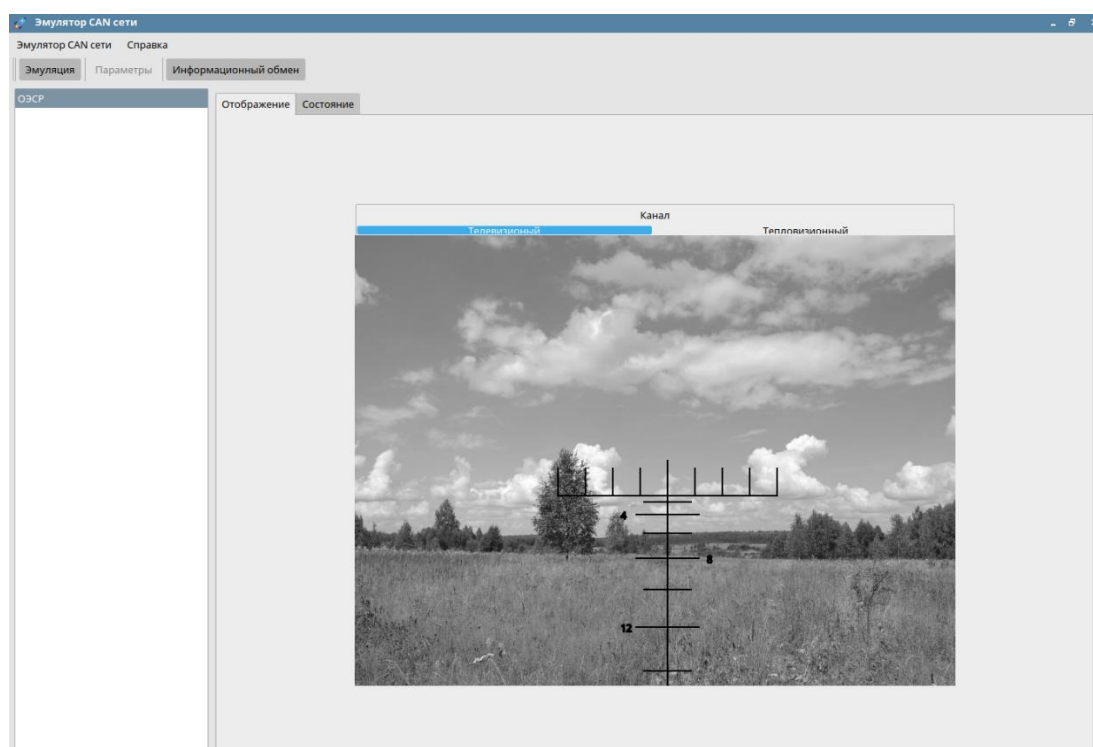


Рисунок В.6 – Состояние «Выверка ПКТ»

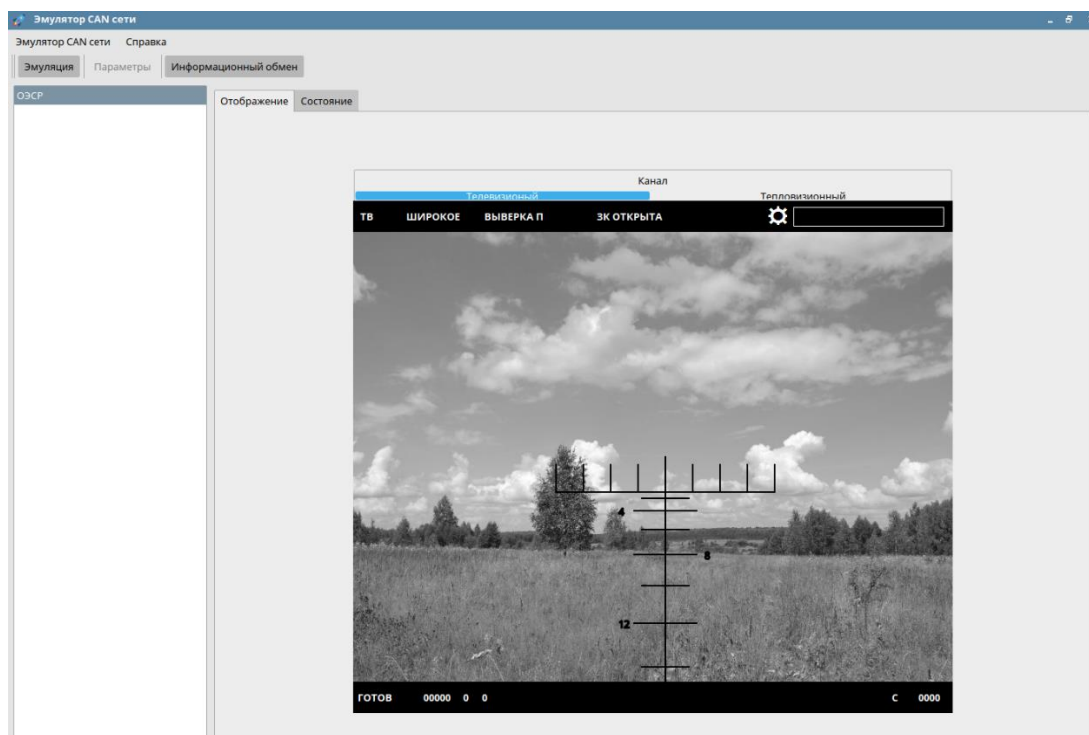


Рисунок В.7 – Состояние «Индексация служебной информации»

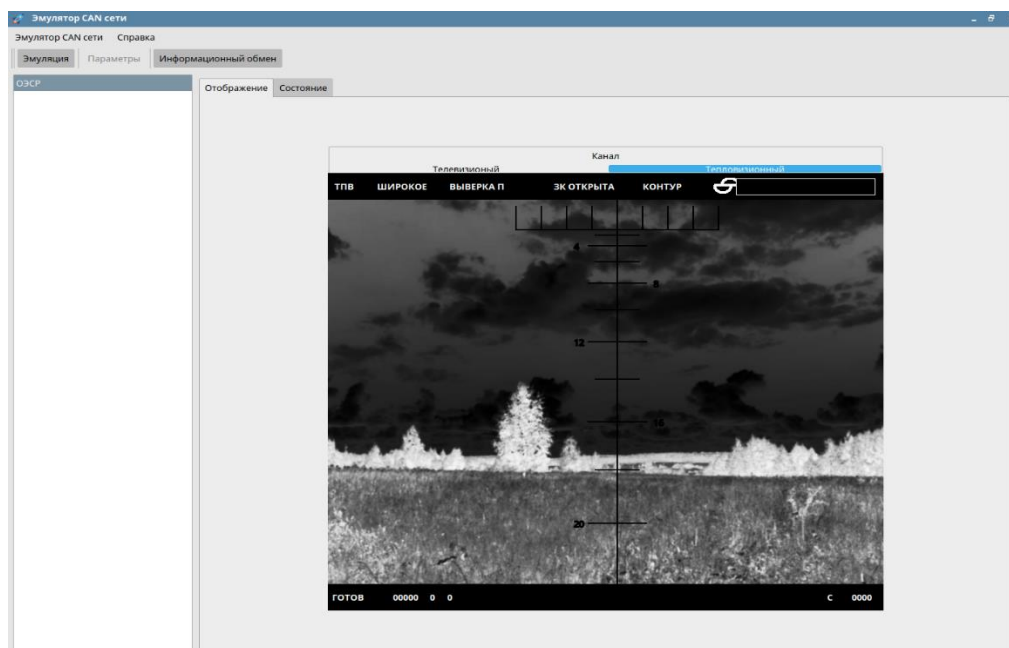


Рисунок В.8 – Состояние «Режим ТПВ»

Приложение Г
СТРАНИЦЫ ИНТЕРФЕЙСА
(рекомендуемое)

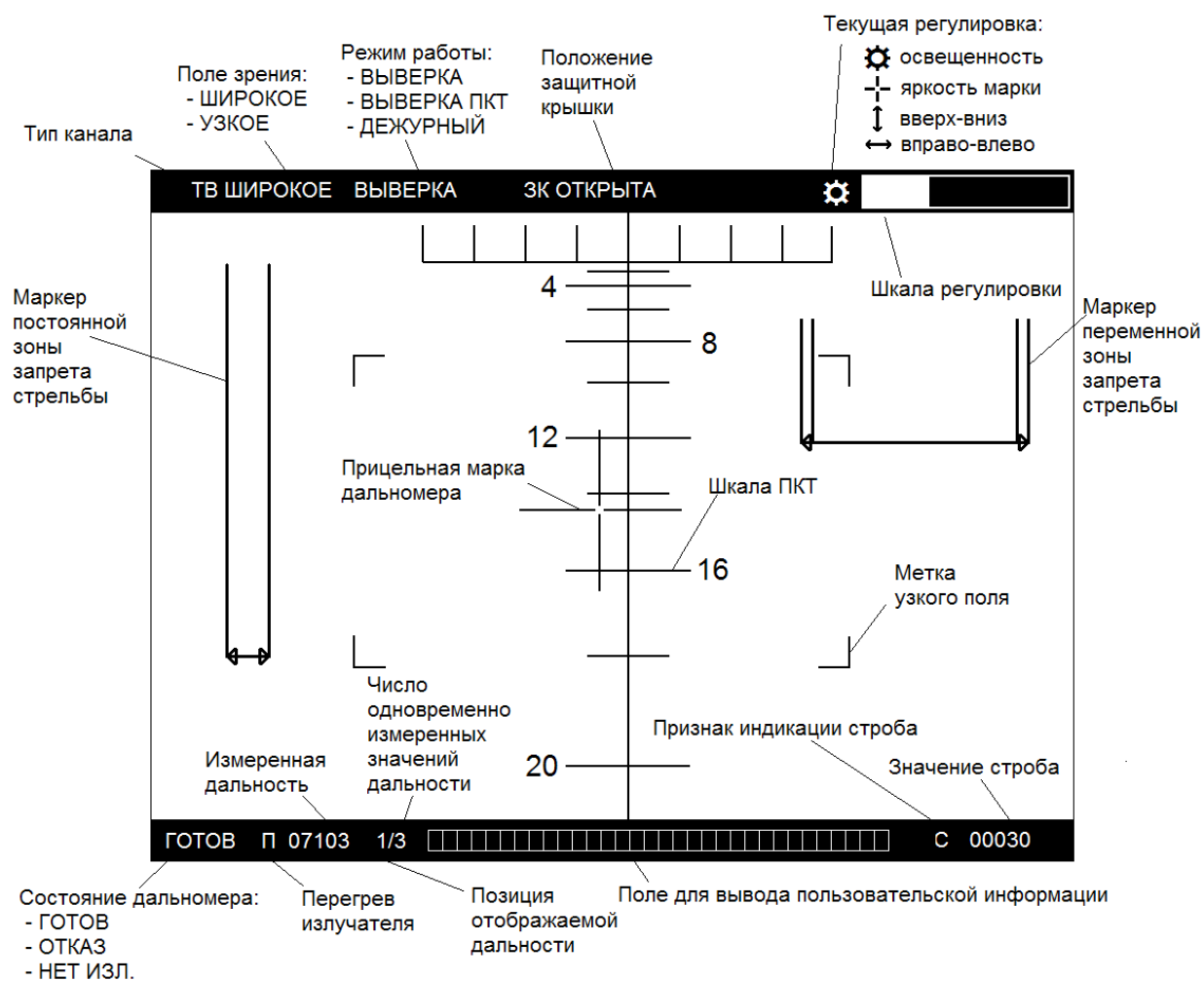


Рисунок Г.1 – Вид поля зрения
телевизионного дневного широкопольного канала

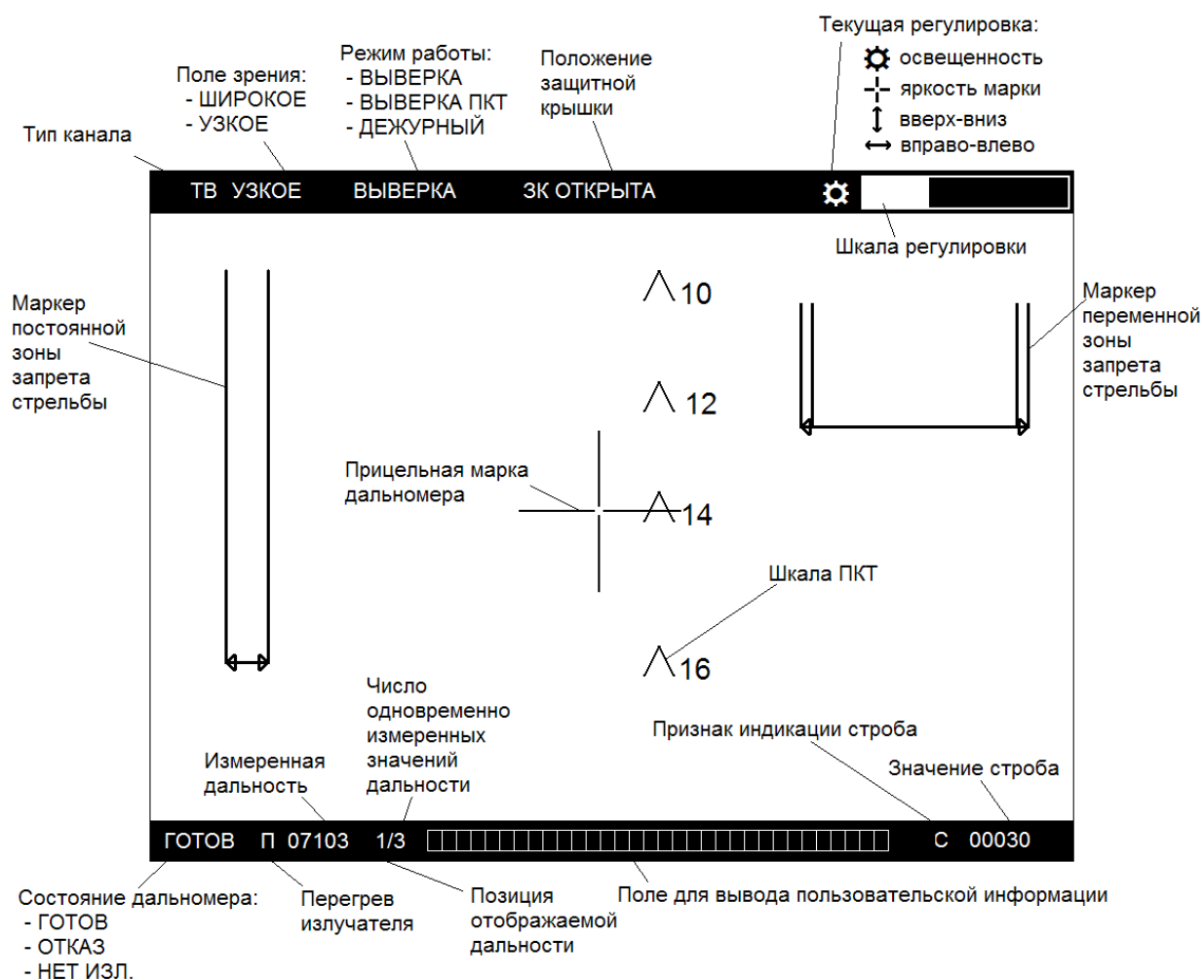


Рисунок Г.2 – Вид поля зрения
телевизионного дневного узкопольного канала

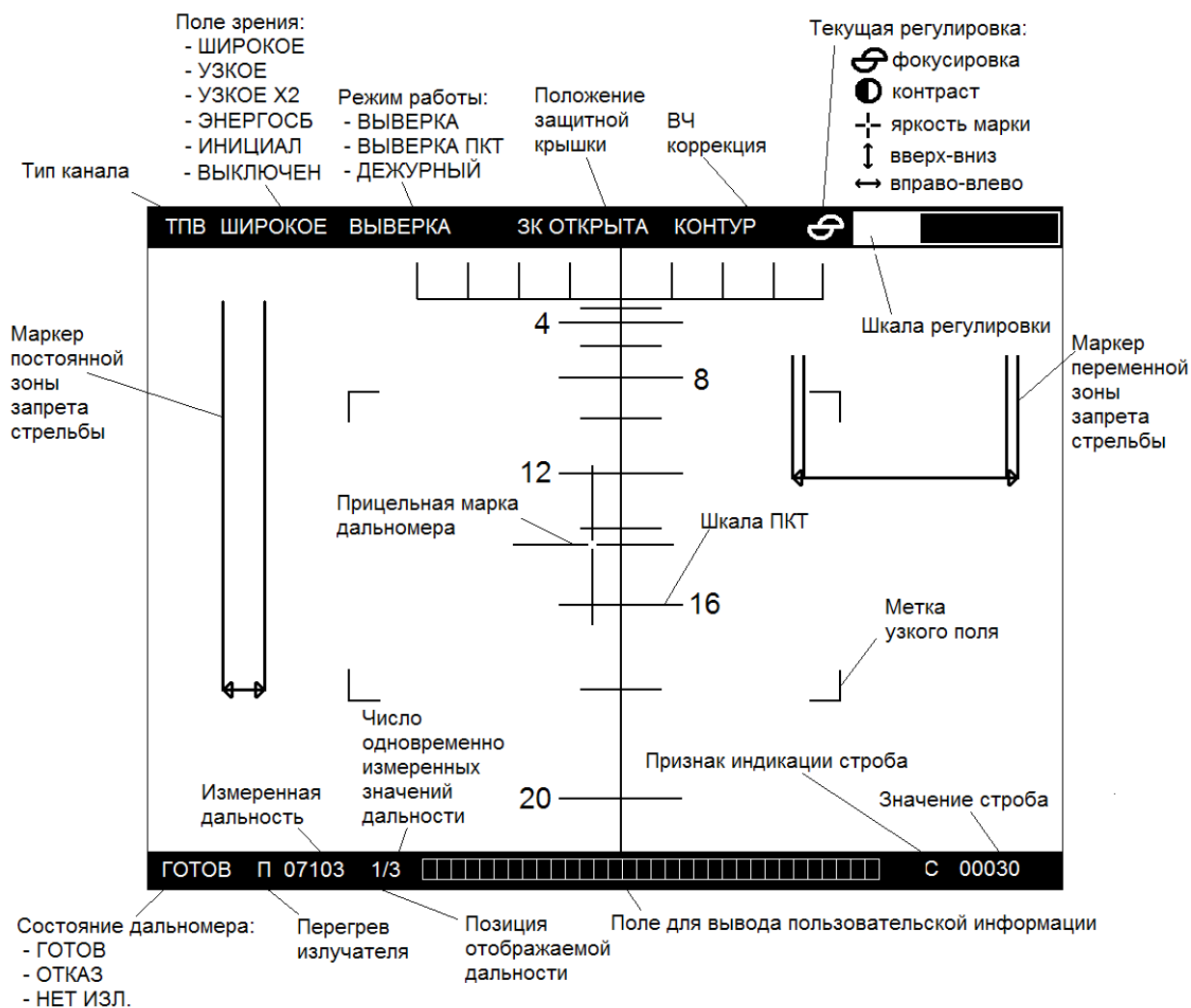


Рисунок Г.3 – Вид поля зрения
тепловизионного широкопольного канала

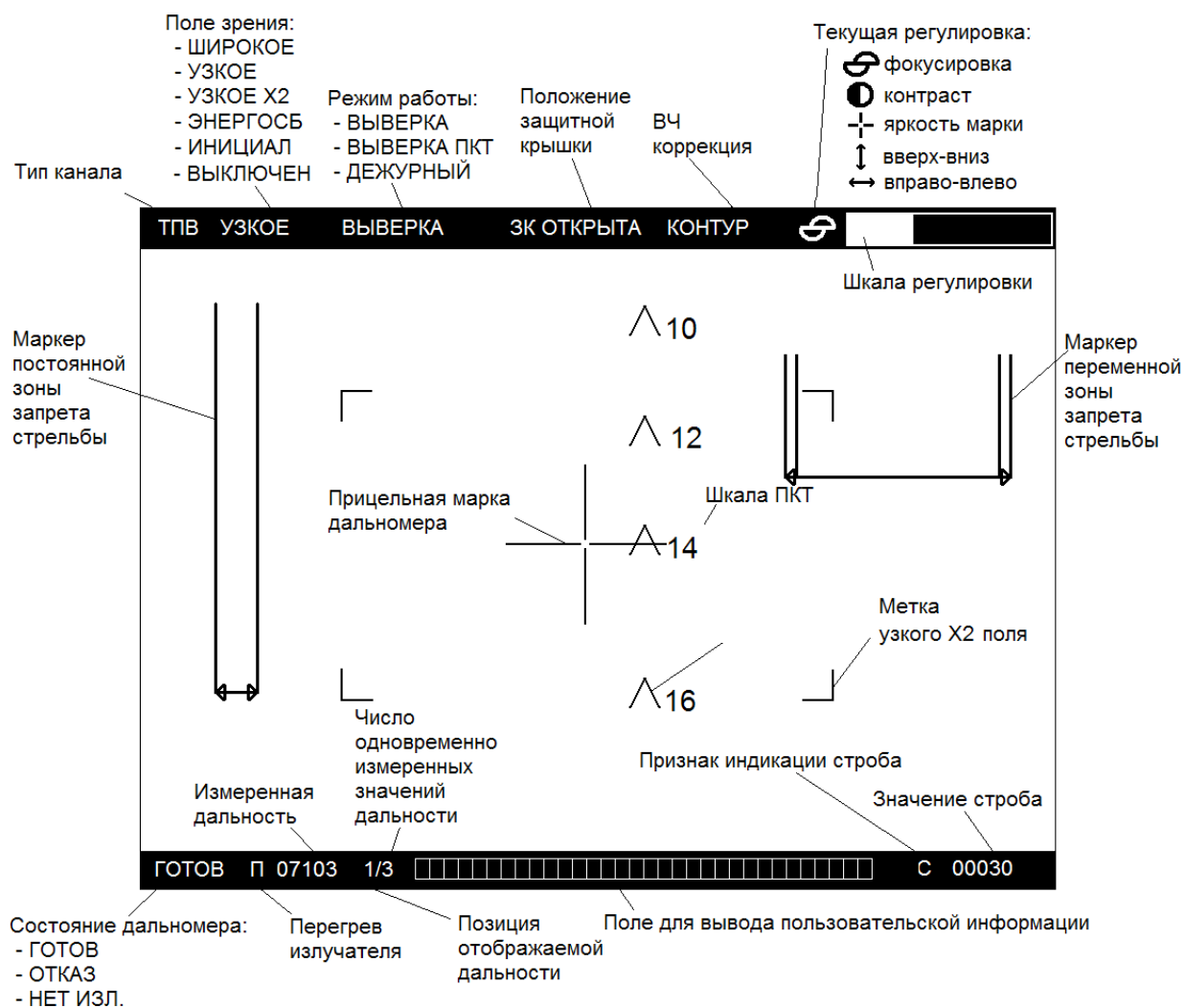


Рисунок Г.4 – Вид поля зрения
тепловизионного узкопольного канала

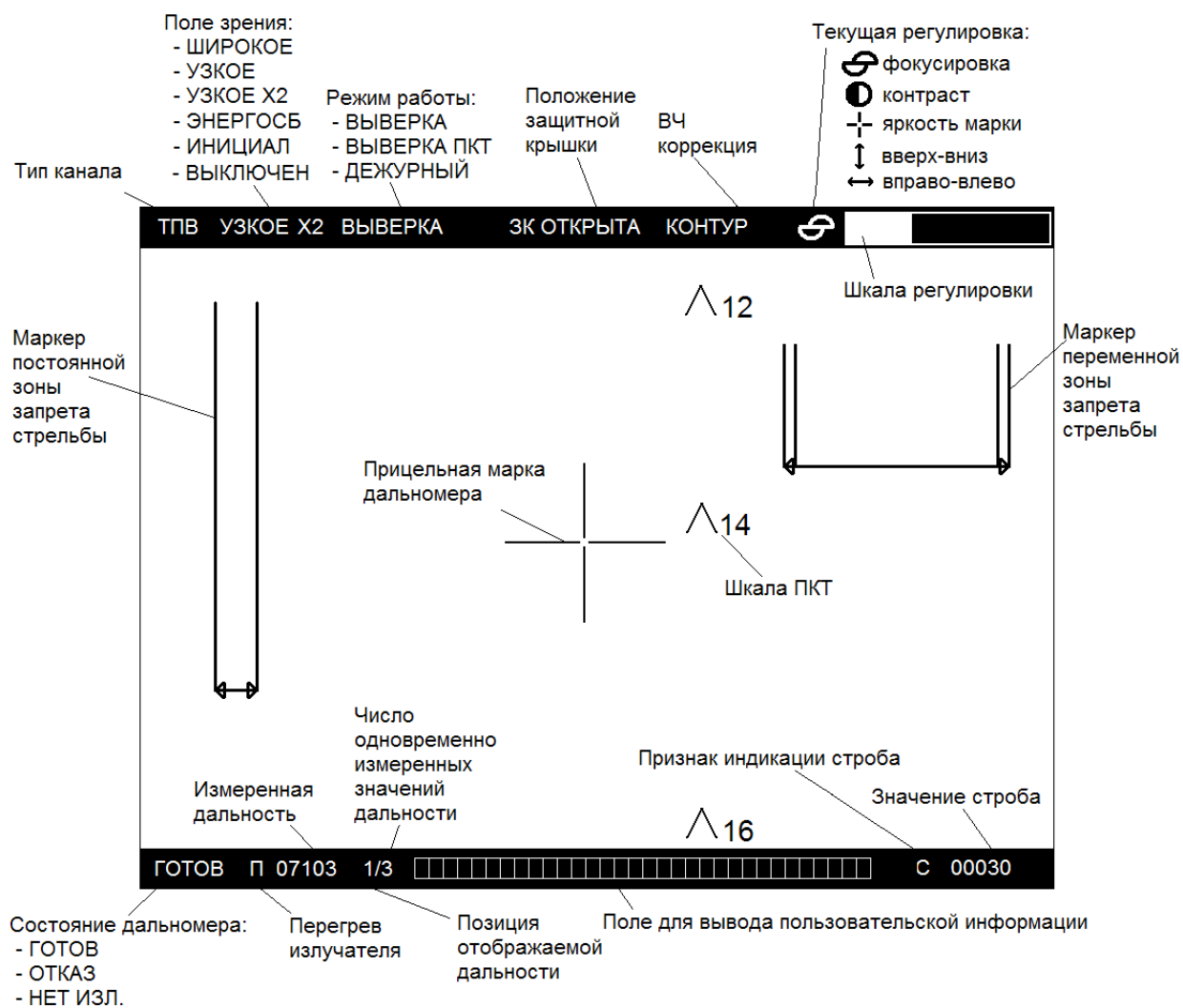


Рисунок Г.5 – Вид поля зрения
тепловизионного узкопольного канала
с электронным увеличением (×2)

Приложение Д
Изображения аналогов
(рекомендуемое)



Рисунок Д.1 – Комплекс Адунок



Рисунок Д.2 – Комплекс Ирония

Приложение Е
АКТ О ВНЕДРЕНИИ РЕЗУЛЬТАТОВ
(рекомендуемое)

УТВЕРЖДАЮ
Заместитель генерального директора АО
«НПП «Рубин» по науке

к.т.н., доцент И.В. Сериков

«___» _____ 2020 г.

АКТ

о внедрении основных результатов дипломной работы
Варакина Андрея Алексеевича выполненной по теме: «Программный
имитатор оптико-электронных средств наблюдения»

Комиссия в составе: председателя комиссии – заместитель начальника
отделения Ганина А.А., членов комиссии: начальника отдела 09120 Рожкова П.В.,
главного специалиста отдела 09120 Череватенко М.В., начальника сектора 09121
Питикова Е.А. подтверждает, что результаты дипломной работы Варакина А.А.
реализованы при разработке технологического программного обеспечения средств
управления применяемого для отладки и стендового тестирования ПО ОКР
«Завет».

Результаты научной работы	Внедрение
Модуль программной имитации изделия «ОЭСР» ИПЦЮ.201219.012	Технологическое ПО, используемое в составе стенда изделия 83т289-1.5

Председатель комиссии:

заместитель начальника отделения, д.т.н., доцент

А.А. Ганин

Члены комиссии:

начальник отдела

П.В. Рожков

главный специалист отдела

М.В. Череватенко

начальник сектора

Е.А. Питиков