

Выпускная квалификационная работа

Прототип приложения для настройки и конфигурации устройств дуговой защиты

по программе профессиональной переподготовки:

«Разработчик прикладного программного обеспечения (Языки С и С++)»

Выполнила: Леонова Татьяна Александровна
Руководитель: Старший преподаватель ВИШ ИДО
 Брык Иван Юрьевич

2022

Цель работы:

Разработать прототип приложения для настройки и конфигурации устройств дуговой защиты

Актуальность:

Данное приложение призвано для увеличения скорости настройки и конфигурации устройства на производстве одного из предприятий, занимающихся дуговой защитой.

Предметная область

Дуговая защита – вид быстродействующей защиты от КЗ, основанный на регистрации спектра света открытой электрической дуги. При написании работы использовалось устройство на основе ЗДЗ волоконно-оптического типа. Данный тип ЗДЗ реагирует на световую вспышку от электрической дуги. В качестве датчика, реагирующего на световую вспышку от электрической дуги, используется волоконно-оптический датчик (ВОД).

Дуговая защита шин

По виду датчик ЗДЗ может быть разделена на три типа:

-клапанного типа



-фототиристорного типа

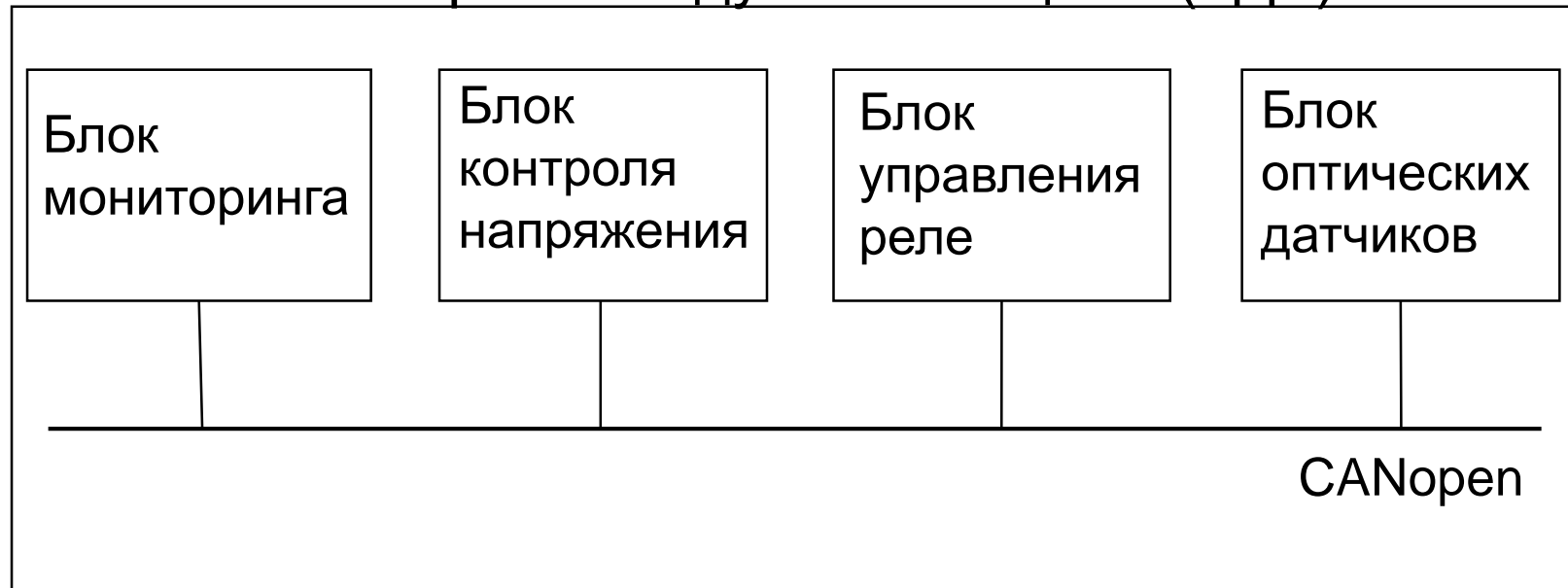


-оптоволоконного типа



Структура устройства

Устройство дуговой защиты (УДЗ)

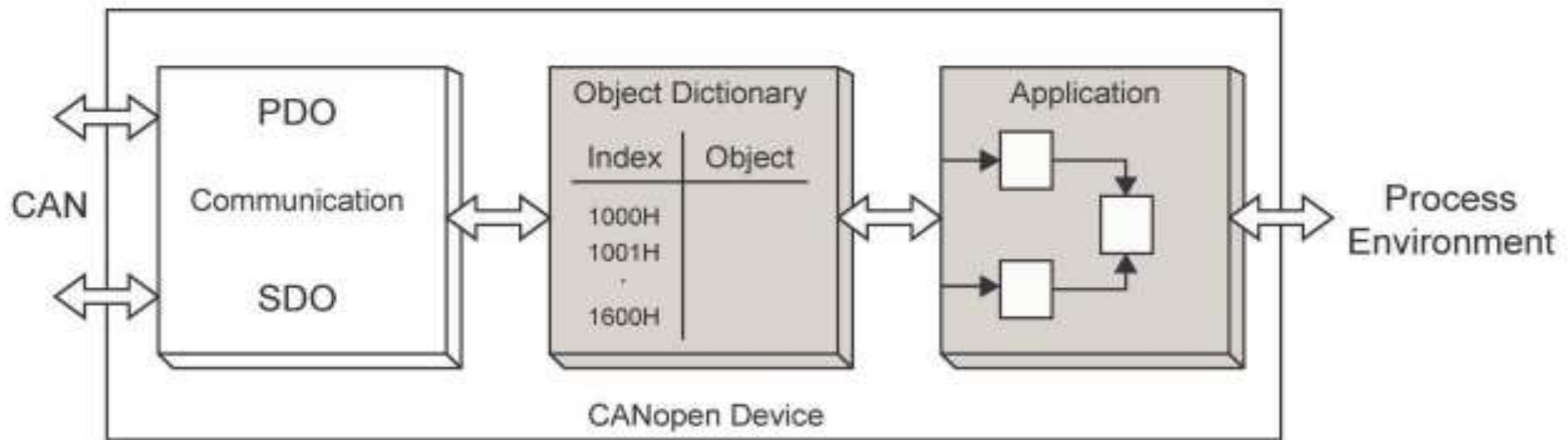


УДЗ ловит световое воздействие от дуги и воздействует на один из многих аварийных выключателей на подстанции.

ПО позволяет конфигурировать (задавать настройки из Словаря объектов) и настраивать (выбирать какие датчики включены или выключены в выбранном блоке) блоки отдельно от УДЗ и в его составе.

Описание протокола CANopen

CAN-шина – это магистральная шина для высоконадёжной передачи данных по последовательному каналу в широковещательном режиме. CAN использует короткие сообщения, максимум 94 бита.



Задачи:

- Разработка функциональных требований к ПО;
- Проработка документации на протокол обмена данными CANopen;
- Применение готовых библиотек Qt для работы с протоколом CAN;
- Разработка классов для работы с пользовательским интерфейсом;
- Разработка классов для работы с протоколом CANopen;
- Применение готовых библиотек для работы с XML файлами;
- Разработка классы для работы с датчиками;
- Проверка работоспособности приложения с помощью ручного тестирования;
- Анализ полученных результатов;

Функциональные требования:

- Программа должна иметь пользовательский интерфейс
 - Окно с запросом пути до файла конфигурации (*.ini);
 - Окно выбора типа блока;
- Программа должна предоставлять:
 - Подключение устройств к ПО по CANopen протоколу;
 - Обмен данными между ПО и устройствами по протоколу CANopen;
 - Индикацию датчиков;
 - Парсинг XML файлов: чтение данных из файла настроек;

Требования графическому интерфейсу:

Диалоговое окно с заголовком «?» и «X». В центре находится текстовое поле «Путь к файлу:» с кнопкой «Поиск» к его правому краю. В нижнем правом углу расположена кнопка «Ок».

Основное окно приложения с заголовком «_ □ X». Меню включает: «Настройки», «Индикация состояния блоков», «Конфигурирование блоков», «Служебная информация» и «О программе». В основной области расположены следующие элементы управления:

- «Плагин» — выпадающий список
- «Имя интерфейса» — выпадающий список
- «Имя USB-CAN преобразователя» — текстовое поле
- «Серия» — текстовое поле
- «Имя канала» — текстовое поле
- «Скорость BitRate» — выпадающий список

В нижней панели расположены кнопки: «Соединение», «Отключение», «Выбор устройства» и «Выход».

Требования графическому интерфейсу:

Объект	Вкл/Выкл	Забл/Разбл	Объект	Вкл/Выкл	Забл/Разбл	Объект	Вкл/Выкл	Забл/Разбл	Объект	Вкл/Выкл	Забл/Разбл
Датчик1	<input type="checkbox"/>	<input type="checkbox"/>	Выход1	<input type="checkbox"/>	<input type="checkbox"/>	Выход9	<input type="checkbox"/>	<input type="checkbox"/>	Выход1	<input type="checkbox"/>	<input type="checkbox"/>
Датчик2	<input type="checkbox"/>	<input type="checkbox"/>	Выход2	<input type="checkbox"/>	<input type="checkbox"/>	Выход10	<input type="checkbox"/>	<input type="checkbox"/>	Выход2	<input type="checkbox"/>	<input type="checkbox"/>
Датчик3	<input type="checkbox"/>	<input type="checkbox"/>	Выход3	<input type="checkbox"/>	<input type="checkbox"/>	Выход11	<input type="checkbox"/>	<input type="checkbox"/>	Выход3	<input type="checkbox"/>	<input type="checkbox"/>
Датчик4	<input type="checkbox"/>	<input type="checkbox"/>	Выход4	<input type="checkbox"/>	<input type="checkbox"/>	Выход12	<input type="checkbox"/>	<input type="checkbox"/>	Выход4	<input type="checkbox"/>	<input type="checkbox"/>
Датчик5	<input type="checkbox"/>	<input type="checkbox"/>	Выход5	<input type="checkbox"/>	<input type="checkbox"/>	Выход13	<input type="checkbox"/>	<input type="checkbox"/>	Выход5	<input type="checkbox"/>	<input type="checkbox"/>
Датчик6	<input type="checkbox"/>	<input type="checkbox"/>	Выход6	<input type="checkbox"/>	<input type="checkbox"/>	Выход14	<input type="checkbox"/>	<input type="checkbox"/>	Выход6	<input type="checkbox"/>	<input type="checkbox"/>
Датчик7	<input type="checkbox"/>	<input type="checkbox"/>	Выход7	<input type="checkbox"/>	<input type="checkbox"/>	Выход15	<input type="checkbox"/>	<input type="checkbox"/>	Выход7	<input type="checkbox"/>	<input type="checkbox"/>
Датчик8	<input type="checkbox"/>	<input type="checkbox"/>	Выход8	<input type="checkbox"/>	<input type="checkbox"/>	Выход16	<input type="checkbox"/>	<input type="checkbox"/>	Выход8	<input type="checkbox"/>	<input type="checkbox"/>

Объект	Вкл/Выкл	Забл/Разбл	Объект	Вкл/Выкл	Забл/Разбл	Объект	Вкл/Выкл	Забл/Разбл
Вход1	<input type="checkbox"/>	<input type="checkbox"/>	Вход6	<input type="checkbox"/>	<input type="checkbox"/>	Выход5	<input type="checkbox"/>	<input type="checkbox"/>
Вход2	<input type="checkbox"/>	<input type="checkbox"/>	Выход1	<input type="checkbox"/>	<input type="checkbox"/>			
Вход3	<input type="checkbox"/>	<input type="checkbox"/>	Выход2	<input type="checkbox"/>	<input type="checkbox"/>			
Вход4	<input type="checkbox"/>	<input type="checkbox"/>	Выход3	<input type="checkbox"/>	<input type="checkbox"/>			
Вход5	<input type="checkbox"/>	<input type="checkbox"/>	Выход4	<input type="checkbox"/>	<input type="checkbox"/>			

Соединение Отключение Выбор устройства Выход

Версия HW ВОД6 Забл/Разбл ВОД6

Версия FW Тестирование ВОД6 Забл/Разбл ВОД7

ВОД1 ВОД7 Забл/Разбл ВОД8

Тестирование ВОД1 Тестирование ВОД7

ВОД2 ВОД8

Тестирование ВОД2 Тестирование ВОД8

ВОД3 Базовый адрес

Тестирование ВОД3 Забл/Разбл ВОД1

ВОД4 Забл/Разбл ВОД2

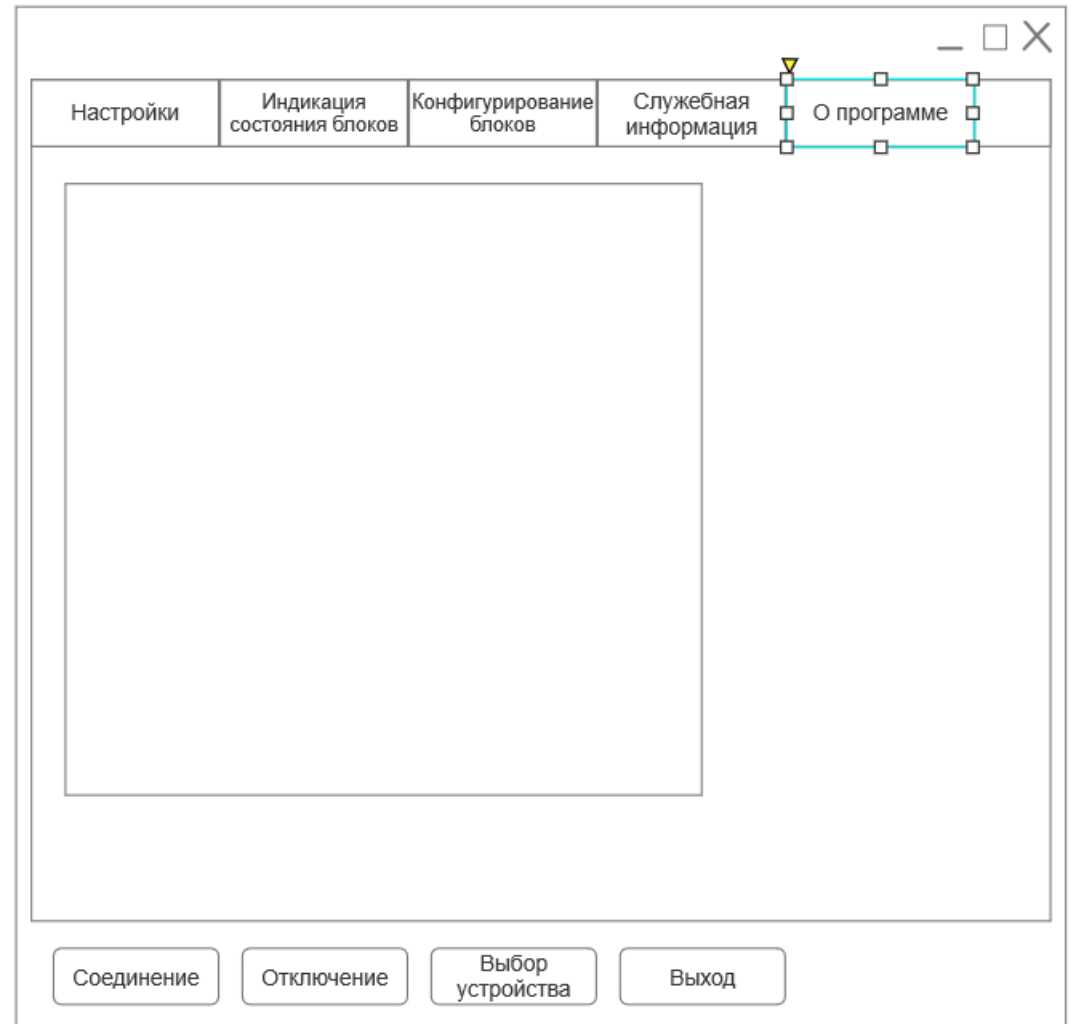
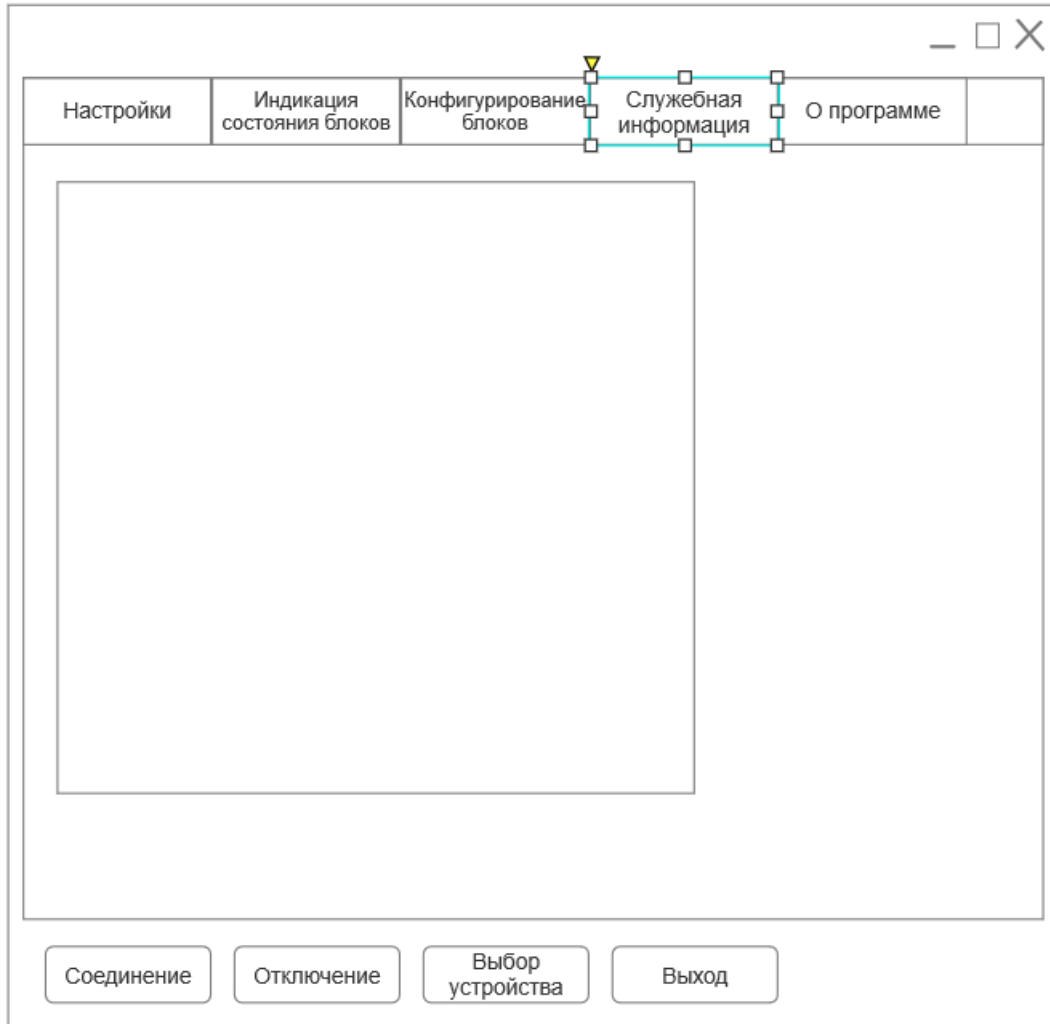
Тестирование ВОД4 Забл/Разбл ВОД3

ВОД5 Забл/Разбл ВОД4

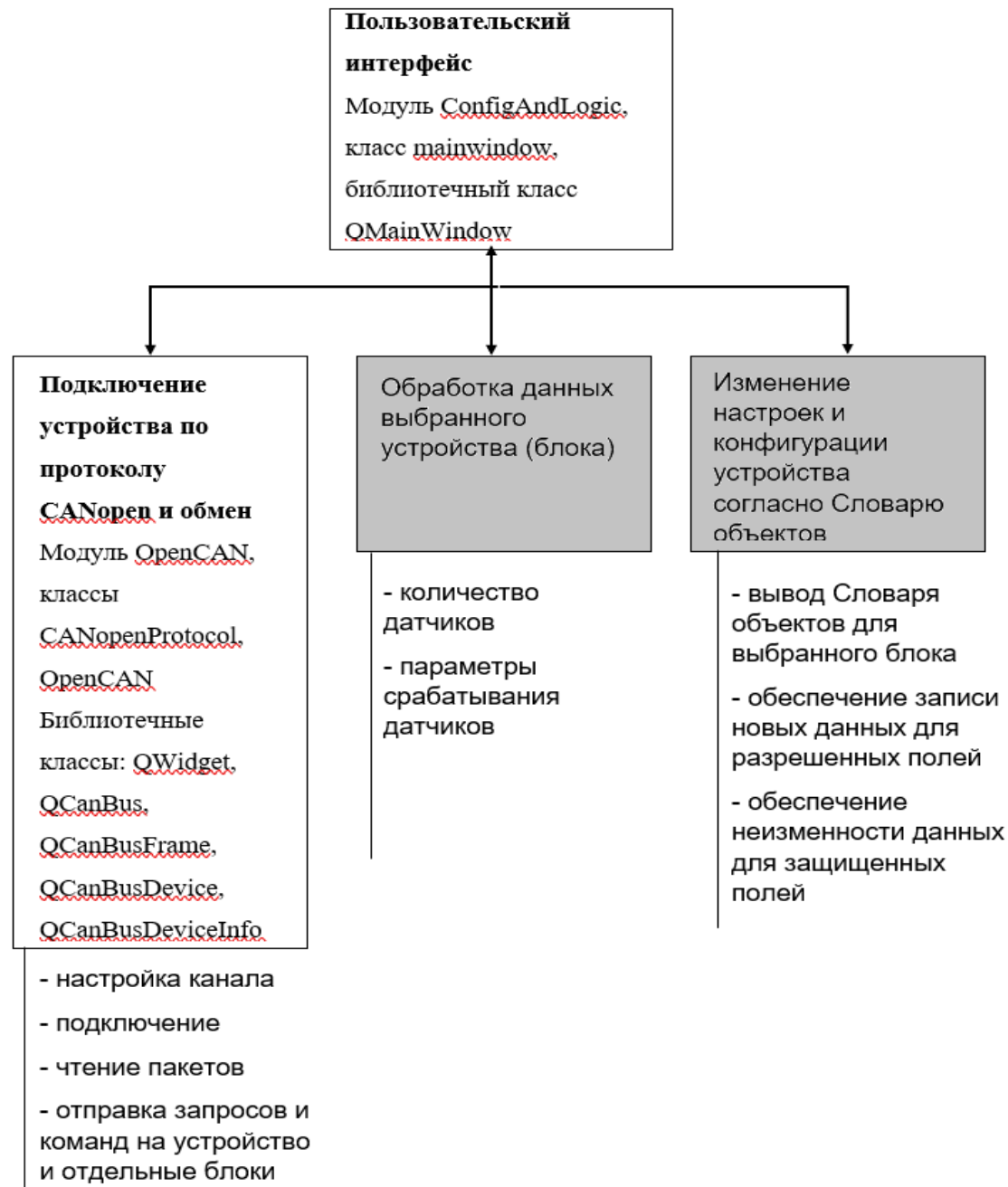
Тестирование ВОД5 Забл/Разбл ВОД5

Соединение Отключение Выбор устройства Выход

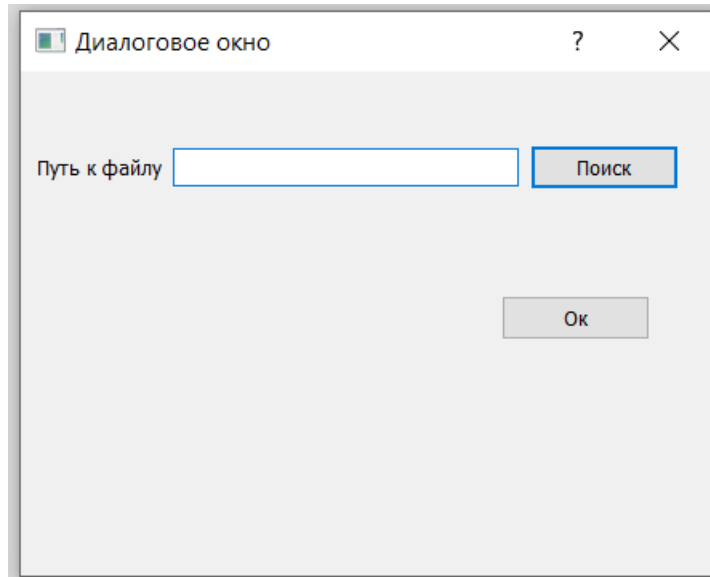
Требования графическому интерфейсу:



Архитектура программы



Реализация графического интерфейса



Реализация графического интерфейса

Form

Plugin	passthruCAN
Interface Name	
Name USB-CAN invertor	
Serial number	
Channel	
BitRate	125000

Главное окно

Настройки | Индикация состояния блока | Конфигурирование блока | Служебная информация | О программе

Плагин	passthruCAN
Название интерфейса	
Название USB-CAN преобразователя	
Серийный номер	
Канал	
BitRate	125000

Соединение | Отключение | Запись | Выбор устройства | Выход

Классы, необходимые для работы с протоколом CANopen

Имя класса	Описание класса	Методы класса
class CANopenProtocol	Обеспечение чтения и записи данных по протоколу CANopen	<pre>CANopenProtocol(); ~CANopenProtocol(); QCanBusFrame writeFrame(uint& id, QString& data);</pre>
class OpenCAN	Обеспечение чтения и записи данных по протоколу CANopen	<pre>explicit OpenCAN(QWidget *parent = nullptr); ~OpenCAN(); void ChangeSettings(); Set settings(); void Connect(); void Disconnect(); void writeMessage(); bool getFlag(); QList<QString> text(); QList<QString> getText(); QList<QString> getList(); private slots: void pluginChanged(QString name); void interfaceChanged(QString name); void on_Stop_clicked(); public slots: void read(); void RequestProductCode(); void createListS(); void getResponse();</pre>

Проверка работоспособности программы

№	Сценарий проверки	Результат
1	Запуск программы	Проведено 50 проверок, количество ошибок 0
2	Поиск *.ini файла на компьютере	Проведено проверок 50, ошибок 0, файл запускается из любой директории
3	Выбор плагина	Проведено проверок 50, если выбрать неверный плагин, программа завершается. При выборе верного плагина программа работает корректно
4	Проверка соединения с устройством	Проведено проверок 50, устройство подключается корректно, полученные пакеты выводятся во вкладке «Служебная информация»
5	Проверка отключения устройства	Проведено проверок 50, устройство отключается корректно без ошибок
6	Проверка отправки запроса на сброс устройства	Проведено проверок 50, ошибок 0, после отправки запроса, устройство посылает новые пакеты данных с другими Id, что можно проследить во вкладке «Служебная информация»
7	Проверка выхода из приложения	Проведено проверок 50, программа закрывается без ошибок, при этом устройство отключается корректно, если оно не было отключено ранее, кнопкой «Отключение»

Заключение

- Разработаны функциональные требования
- Проработана документация для работы с протоколом CANopen
- Проработаны библиотечные классы для работы с протоколом CAN
- Разработан графический интерфейс
- Разработаны классы для подключения устройства и обмена данными.
- Проведено ручное тестирование во время отладки для подтверждения корректности работы программы.