МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования «Самарский национальный исследовательский университет имени академика С.П. Королева (Самарский университет)»

Институт информатики и кибернетики

Кафедра программных систем

**ОТЧЕТ**  
  
 по лабораторному практикуму по дисциплине

«Разработка современного пользовательского интерфейса

Вариант № 25

Обучающийся группы 6132-020402D А.В. Дьяконов

Обучающийся группы 6132-020402D М.Ю. Лазарев

Руководитель Л.С. Зеленко

Самара 2025

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования «Самарский национальный исследовательский университет имени академика С.П. Королева (Самарский университет)»

Институт информатики и кибернетики

Кафедра программных систем

ЗАДАНИЕ

на лабораторный практикум по дисциплине

«Разработка современного пользовательского интерфейса»

обучающимся в группе № 6132-020402D

А.В. Дьяконову

М.Ю. Лазареву

1. Задание**:** С помощью SCADA-системы WinCC OA 3.17 разработать программное обеспечение, моделирующее поведение работы нефтетехнологической установки, предусмотреть работу нефтетехнологической установки при возникновении нештатных ситуаций
2. Исходные данные к проекту**:** см. приложение к заданию
3. Перечень вопросов, подлежащих разработке:
   1. Произвести анализ предметной области: изучить технологический процесс производства товарной нефти, состав оборудования и его характеристики, а также поведение оборудования в режиме реального времени
   2. Разработать информационную модель реального объекта, определить точки данных, команды управления, аварийные ситуации
   3. Разработать прототипы экранных форм (мнемосхем)
   4. Разработать алгоритмы управления нефтетехнологической установки
   5. Реализовать скрипты для элементов управления, провести тестирование и отладку
   6. Оформить документацию
4. Перечень графических разработок:
   1. Мнемосхема системы
   2. Панель настроек
   3. Главная панель системы
5. Календарный план выполнения работ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № п/п | Содержание работы по этапам | Объем этапа в % к общему объему проекта | Срок  окончания | Фактическое выполнение |
| 1 | Оформление технического задания и его утверждение | 5 | 01.03.2025 |  |
| 2 | Описание и анализ предметной области | 15 | 01.03.2025 |  |
| 3 | Проектирование системы | 25 |  |  |
| 3.1 | Разработка информационной модели объекта | 10 | 15.03.2025 |  |
| 3.2 | Разработка прототипа интерфейса пользователя | 10 | 29.03.2025 |  |
| 4 | Реализация проекта, разработка контрольных примеров. | 40 | 17.05.2025 |  |
| Предъявление реализации руководителю | 5 | 24.05.2025 |  |
| 5 | Корректировка проекта и оформление документации проекта. Защита проекта с представлением презентации. | 10 | 31.05.2025 |  |

Задание принял  
 к исполнению 15.02.2025 А.В. Дьяконов

15.02.2025 М.Ю. Лазарев

ПРИЛОЖЕНИЕ  
к заданию на лабораторный практикум  
обучающимся в группе № 6132-020402D

А.В. Дьяконову

М.Ю. Лазареву

Исходные данные к проекту:

1. Характеристика объекта автоматизации:

## объект автоматизации: нефтетехнологическая установка;

## виды автоматизируемой деятельности:

* + процесс авторизации пользователей системы;
  + процесс моделирования работы оборудования нефтетехнологической установки;
  + процесс управления оборудованием нефтетехнологической установки при возникновении нештатных ситуаций;
  + процесс визуализации работы оборудования нефтетехнологической установки;
  + процесс обработки аварийных сообщений;

## количество ролей пользователей – 3;

## минимальная длина пароля – 4 (символа);

## максимальная длина пароля – 12 (символов);

## минимальная длина логина – 4 (символа);

## максимальная длина логина – 12 (символов);

## количество режимов управления работой оборудования – 2;

## количество типов элементов оборудования – 7;

## количество запорных клапанов – 4;

## количество регулируемых клапанов – 3;

## количество емкостей – 4;

## количество датчиков уровня жидкости – 5;

## количество датчиков температуры – 1;

## количество труб – 9;

## нагревательный элемент – 1;

## объем ёмкости Е0, Е1, Е2 – 200 м3;

## минимальное значение уровня в ёмкостях Е0, Е1, Е2 – 200 мм;

## максимальное значение уровня в ёмкостях Е0, Е1, Е2 – 3000 мм;

## минимальное значение объёма в ёмкостях Е0, Е1, Е2 – 12 м3;

## максимальное значение объёма в ёмкостях Е0, Е1, Е2 – 176 м3;

## минимальное значение скорости поступления продукта в ёмкости Е0, Е1, Е2 – 5 м3/час;

## максимальное значение скорости поступления продукта в ёмкости Е0, Е1, Е2 – 20 м3/час;

## объем ёмкости Е3 – 200 м3;

## минимальное значение уровня в ёмкости Е3 по воде – 200 мм;

## максимальное значение уровня в ёмкости Е3 по воде – 1000 мм;

## минимальное значение уровня в ёмкости Е3 по нефти – 1000 мм;

## максимальное значение уровня в ёмкости Е3 по нефти – 3000 мм;

## минимальное значение объёма в ёмкости Е3 по воде – 12 м3;

## максимальное значение объёма в ёмкости Е3 по воде – 58 м3;

## минимальное значение объёма в ёмкости Е3 по нефти – 58 м3;

## максимальное значение объёма в ёмкости Е3 по нефти – 176 м3;

## минимальное значение скорости поступления продукта в ёмкость Е3 – 15 м3/час;

## максимальное значение скорости поступления продукта в ёмкость Е3 – 60 м3/час;

## минимальная мощность нагревательного элемента - 0 кВт;

## максимальная мощность нагревательного элемента - 24 кВт;

## поддерживаемое значение температуры в ёмкости Е3 – 65 °С.

1. Требования к информационному обеспечению:
2. информационное обеспечение разрабатывается на основе следующих источников:
   * описание принципов работы установки подготовки нефти (УПН) [Электронный ресурс]. URL: https://neftegaz.ru/tech-library/oborudovanie-dlya-sbora-i-podgotovki-nefti-i-gaza/141662-ustanovka-podgotovki-nefti-upn/ (дата обращения: 16.02.2025);
   * описание принципов работы установки по сбросу воды (УПСВ) [Электронный ресурс]. URL: https://tdsarrz.ru/produktsiya/  
     oborudovanie\_dlya\_podgotovki\_nefti\_i\_gaza/ustanovki\_predvaritelnogo\_sbrosa\_vody.html (дата обращения: 18.02.2025);
3. структуры точек данных определяются в процессе проектирования;
4. структура мнемосхемы определяется в процессе проектирования.
5. Требования к техническому обеспечению:
6. тип ЭВМ – IBM PC совместимый;
7. монитор с разрешающей способностью не ниже 800 х 600;
8. манипулятор – мышь;
9. технические характеристики определяются в процессе выполнения проекта.
10. Требования к программному обеспечению:
11. тип операционной системы – Windows 7 и выше;
12. SCADA-система – WinCC OA 3.17;
13. язык программирования – CTRL;
14. среда программирования – Vision.
15. Общие требования к проектируемой системе:

5.1 Функции, реализуемые системой:

1. функции системы:
   * аутентификация пользователя в системе, настройка интерфейса пользователя на заданную роль;
   * моделирование работы оборудования в ручном/ автоматическом режиме:
   * запуск оборудования;
   * открытие/закрытие запорных клапанов;
   * открытие/закрытие/регулировка запорно-регулирующих клапанов;
   * включение/отключение/регулировка нагревательного элемента;
   * контроль уровня жидкости в емкостях;
   * контроль температуры продукта в емкости;
   * контроль технологического процесса;
   * логирование и регистрация действий;
   * моделирование возникновения нештатных ситуаций (отказов) в режиме реального времени и выдача аварийных сообщений;
   * визуализация процесса работы оборудования:
   * отображение штатной работы оборудования;
   * отображение аварийных сообщений;
   * выдача справочной информации;
2. функции администратора:
   * авторизация пользователя в системе (ввод логина/пароля);
   * настройка параметров оборудования (точек данных):
   * задать скорость подачи продуктов;
   * задать мощность нагревательного элемента;
   * задать вероятность отказа всех типов оборудования;
   * работа с оборудованием (ручной режим):
   * запустить работу оборудования;
   * остановить работу оборудования;
   * открыть запорный клапан;
   * закрыть запорный клапан;
   * отрегулировать открытие/закрытие запорно-регулирующего клапана;
   * включить нагревательный элемент;
   * выключить нагревательный элемент;
   * включить датчики;
   * выключить датчики;
   * квитировать аварийные сообщения;
   * просмотр архивных параметров за требуемый интервал времени;
   * просмотр справочной информации;
3. функции оператора:
   * авторизация пользователя в системе (ввод логина/пароля);
   * работа с оборудованием (ручной режим):
   * запустить работу емкостей;
   * остановить работу емкостей;
   * открыть запорный клапан;
   * закрыть запорный клапан;
   * отрегулировать открытие/закрытие запорно-регулирующего клапана;
   * включить нагревательный элемент;
   * выключить нагревательный элемент;
   * включить датчики;
   * выключить датчики;
   * квитировать аварийные сообщения;
   * просмотр архивных параметров за требуемый интервал времени;
   * просмотр справочной информации;
4. функции гостя:
   * наблюдение за ходом технологического процесса.

5.2 Технические требования к системе:

1. режим работы ‑ диалоговый;
2. система должна удовлетворять санитарным правилам и нормам  
    СанПин 2.2.2./2.4.2198-07;
3. условия работы средств вычислительной техники (содержание вредных веществ, пыли и подвижность воздуха) должны соответствовать ГОСТ 12.1.005, 12.01.007;
4. температура окружающего воздуха – 15-35°С;
5. влажность воздуха – 45-75%.

Руководитель   
проекта Л.С. Зеленко

Задание принял  
к исполнению 15.02.2025 А.В. Дьяконов

15.02.2025 М.Ю. Лазарев

РЕФЕРАТ

Отчет 118 с, 36 рисунков, 11 таблиц, 20 источников, 2 приложения.

НЕФТЕТЕХНОЛОГИЧЕСКАЯ УСТАНОВКА, НЕФТЬ, ОТСТОЙНИК НЕФТИ, ПЕРЕРАБОТКА НЕФТИ, ПОДГОТОВКА НЕФТИ, РАЗДЕЛЕНИЕ НЕФТИ, СЕПАРАЦИОННАЯ ЕМКОСТЬ, ТОВАРНАЯ НЕФТЬ.

Объектом автоматизации является нефтетехнологическая установка.

Во время лабораторного практикума разработана система моделирования работы нефте-технологической установки с помощью SCADA-системы WinCC OA 3.17. В системе реализованы три роли пользователей: администратор, оператор и гость. Администратор выбирает режим управления производственным процессом: ручной или автоматический, настраивает параметры емкостей, нагревательного элемента, датчиков уровня жидкости, запорных и запорно-регулирующих клапанов. Оператор наблюдает за ходом процесса и при возникновении аварийных ситуаций включается в процесс работы оборудования. Гость только наблюдает за ходом процесса.

В системе имеется возможность контроля уровня и нагрева жидкости в технологических емкостях, регулировка запорных и запорно-регулирующих клапанов, перевод из ручного в автоматический режим и обратно, настройка основных параметров оборудования, ввод оборудования в эксплуатацию, вывод оборудования из эксплуатации, настройка критических параметров, отображение и обработка аварийных ситуаций, формирование и хранение сообщений и отчетов по аварийным и критическим ситуациям.

Программное обеспечение разработано на языке CTRL в среде WinCC OA 3.17 и функционирует под управлением операционной системы Windows 7 и выше.

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 13](#_Toc192794429)

[1 Описание предметной области 15](#_Toc192794430)

[1.1 Основные понятия и определения 15](#_Toc192794431)

[1.2 Описание технологического процесса получения товарной нефти 16](#_Toc192794432)

[1.3 Описание оборудования, задействованного в работе нефте-технологической установки 17](#_Toc192794433)

[1.3.1 Описание отстойников нефти 17](#_Toc192794434)

[1.3.2 Описание сепарационной емкости 19](#_Toc192794435)

[1.3.3 Описание запорного и запорно-регулирующего клапана 21](#_Toc192794436)

[1.3.4 Описание нагревательного элемента 24](#_Toc192794437)

[1.3.5 Описание труб 26](#_Toc192794438)

[1.3.6 Описание датчика уровня жидкости 27](#_Toc192794439)

[1.4 Постановка задачи 29](#_Toc192794440)

[1.4.1 Функции системы 29](#_Toc192794441)

[1.4.2 Функции администратора 31](#_Toc192794442)

[1.4.3 Функции оператора 31](#_Toc192794443)

[1.4.4 Режим гостя 32](#_Toc192794444)

[2 Проектирование системы 35](#_Toc192794445)

[2.1 Разработка информационной модели объекта 35](#_Toc192794446)

[2.1.1 Описание точек данных (Data Points) 35](#_Toc192794447)

[2.2 Проектирование интерфейса пользователя 40](#_Toc192794448)

[3 Реализация системы 44](#_Toc192794449)

[3.1 Настройка элементов точек данных 44](#_Toc192794450)

[3.2 Настройка исходных значений 44](#_Toc192794451)

[3.3 Общие настройки объекта 45](#_Toc192794452)

[3.4 Обработка сообщений (alert) 45](#_Toc192794453)

[3.5 Разработка и описание графического интерфейса 46](#_Toc192794454)

[3.5.1 Описание графических элементов мнемосхемы 46](#_Toc192794455)

[3.5.2 Начало работы 48](#_Toc192794456)

[3.5.3 Режим администратора 48](#_Toc192794457)

[3.5.4 Режим оператора 50](#_Toc192794458)

[3.5.5 Режим гостя 50](#_Toc192794459)

[3.5.6 Обработка внештатных ситуаций 50](#_Toc192794460)

[3.5.7 Просмотр параметров оборудования 51](#_Toc192794461)

[Заключение 52](#_Toc192794462)

[Список использованных источников 53](#_Toc192794463)

[ПРИЛОЖЕНИЕ А Описание настроек точек данных 54](#_Toc192794464)

[ПРИЛОЖЕНИЕ Б Сценарии (скрипты управления) 55](#_Toc192794465)

ВВЕДЕНИЕ

SCADA (Supervisory Control and Data Acquisition) – это программно-аппаратные комплексы, предназначенные для мониторинга, управления и анализа технологических процессов в режиме реального времени [1].

Основные задачи SCADA-систем:

* мониторинг технологических процессов – сбор, отображение и анализ данных с датчиков и оборудования;
* управление в реальном времени – удалённое управление устройствами, настройка параметров работы;
* архивирование и анализ данных – хранение информации для последующего анализа, прогнозирования и оптимизации;
* оповещение и диагностика аварий – уведомления о критических ситуациях для быстрого реагирования;
* интеграция с другими системами – взаимодействие с системами планирования ресурсов предприятия (ERP – Entertainment Resource Planning), управления производственными процессами (MES - Manufacturing Execution System), интернета вещей (IoT – Internet of Things) и другими информационными платформами.

SCADA играет ключевую роль в автоматизированных системах управления технологическими процессами (АСУ ТП), обеспечивая:

* повышение эффективности – минимизация ручного труда, ускорение реакции на отклонения;
* безопасность – контроль аварийных ситуаций и предотвращение катастроф;
* гибкость и масштабируемость – возможность адаптации под разные производства (энергетика, нефтегаз, машиностроение и др.);
* оптимизацию ресурсов – снижение затрат на обслуживание и энергопотребление.

С развитием цифровизации SCADA-системы становятся всё более интеллектуальными, поддерживают облачные технологии, искусственный интеллект и IoT, что делает их ещё более востребованными в современных АСУ ТП.

Во время лабораторного практикума необходимо разработать систему моделирования производственного участка нефтетехнологической установки по нагреву водонефтяной эмульсии, которая позволит регулировать нагрев и объем отстоявшейся товарной нефти для дальнейшей транспортировки на магистральный трубопровод.

Система будет разрабатываться с помощью SCADA-системы SIMATIC WinCC OA 3.17, которая позволяет визуализировать основные технологические параметры, осуществлять мониторинг и управление процессом в реальном времени, гибко кастомизировать интерфейс, интегрировать и настроить оборудование для связи с датчиками и контроллерами, автоматизировать процесс с реализацией сценариев управления и реакций на изменение технологических параметров, а также архивировать и хранить данные для дальнейшего анализа и оптимизации.

1. Описание предметной области
   1. Основные понятия и определения

Нефть - природная маслянистая горючая жидкость со специфическим запахом, состоящая в основном из сложной смеси углеводородов различной молекулярной массы и некоторых других химических соединений — также нефть всегда содержит в качестве примеси небольшое количество воды и неорганических веществ [2].

Во время первичной переработки нефти на производстве сырая нефть доводится до состояния товарной нефти.

Сырая нефть, добытая из скважины, представляет собой водонефтяную эмульсию с большим содержанием природного газа и различных механических примесей (парафины, соли, асфальтены, смолы, различные металлы).

Товарная нефть представляет собой очищенную от механических примесей и готовую к транспортировке дегазированную нефть с околонулевым содержанием воды.

Деэмульгаторы нефти - поверхностно-активные вещества, способствующие предотвращению образования, а также разрушению уже образованных эмульсий нефти и воды. Иными словами, реагент позволяет отделить молекулы нефти от молекул воды [3].

При добавлении в нефтяные эмульсии деэмульгатор замещает содержащиеся в нефти природные эмульгаторы, препятствующие смешению воды и нефти. Различные механические примеси, содержащиеся в нефти, собираются на поверхности границы между каплями воды и нефтью. Деэмульгаторы способствуют удалению этих примесей из нефти вместе с водой. В настоящее время широкое распространение получили неионогенные деэмульгаторы на основе метанола (Дин-4, Decleave, СНПХ-4315 и т. п.)

Коагулянт – реагенты, применяемые в процессах подготовки нефти, включая обессоливание и деэмульгирование, для ускорения осаждения дисперсной фазы (воды или твёрдых частиц) [4]. В настоящее время самым распространенным коагулянтом является полиакриламид.

* 1. Описание технологического процесса получения товарной нефти

Для получения товарной нефти водонефтяная эмульсия проходит путь в несколько ступеней сепарации (дегазирования и разделения) с различных установок подготовки нефти с применением добавочных реактивов и реагентов. Как правило количество ступеней сепарации зависит от химического состава эмульсии, находящихся в ней примесей и применяемых реагентов из лабораторных расчетов.

Сырая нефть проходит через специальные сосуды под давлением – сепарационные емкости, где происходит непосредственно технологический процесс сепарации [5].

С последней ступени сепарации из сепарационных емкостей под крайне малым давлением максимально дегазированная сырая нефть с различными химическими добавками поступает в специальную емкость – отстойник нефти. В отстойнике нефти происходит процесс смешивания и отстаивания различных видов сырой нефти. Под действием силы тяжести происходит процесс отстаивания, который позволяет отделить более плотные компоненты (воду и осадок) от менее плотной нефти, которая поднимается вверх [6].

При нагреве отстойника нефти до 50-70°С снижается вязкость нефти, что обеспечивает дополнительное ускорение процесса разделения фаз. В итоге при нагреве и отстое сырая нефть разделяется на товарную нефть и воду, оставшиеся реагенты и ингибиторы растворяются в воде, а реактивы и механические примеси выпадают в осадок вместе с водой. Товарная нефть поступает далее в магистральный нефтепровод до нефтестабилизационного производства, где ведется разделение фракций и окончательная подготовка продукта. Осадок в виде воды с химическими и механическими примесями сливается в дренажный трубопровод и отправляется на установки поддержания пластового давления для дальнейшего заводнения скважин и увеличения добычи сырой нефти.

* 1. Описание оборудования, задействованного в работе нефте-технологической установки

В ходе технологического процесса производства товарной нефти будет задействовано следующее оборудование:

* датчик уровня жидкости (по товарной нефти и воде);
* сепарационные емкости E0, E1, Е2;
* отстойник нефти E3;
* датчик указателя температуры на Е3;
* нагревательный элемент на Е3;
* запорные клапана;
* запорно-регулирующие клапана;
* трубы.
  + 1. Описание отстойников нефти

Отстойники нефти широко применяются на нефтедобывающих и перерабатывающих предприятиях для подготовки нефти к дальнейшей транспортировке и переработке.

Процесс разделения основан на гравитационном отстаивании, при котором более тяжелые компоненты (вода и твердые частицы) оседают на дно резервуара, а нефть, обладающая меньшей плотностью, поднимается вверх. Для повышения эффективности разделения могут использоваться дополнительные методы, такие как подогрев эмульсии или введение химических реагентов. Подогрев снижает вязкость нефти, способствуя более быстрому разделению фаз [7].

Существуют различные конструкции отстойников нефти, основные из которых:

* горизонтальные отстойники нефти (ОГ): представляют собой цилиндрические емкости, расположенные горизонтально. Они оснащены внутренними перегородками, которые способствуют эффективному разделению фаз. Горизонтальные отстойники часто используются для обезвоживания нефти и удаления механических примесей [8];
* вертикальные отстойники нефти (ОН): имеют вертикальную цилиндрическую форму и занимают меньшую площадь на производственной площадке. Они применяются для разделения водонефтяной эмульсии и удаления растворенного газа [9];
* многоступенчатые отстойники: предназначены для поэтапного разделения нефти, воды и примесей, обеспечивая более глубокую очистку продукта.

Отстойники нефти оснащены различными элементами для повышения эффективности работы:

* внутренние перегородки: разделяют резервуар на секции, способствуя более эффективному разделению фаз [10];
* cистемы подогрева: используются для снижения вязкости нефти, что ускоряет процесс разделения;
* коалесцирующие устройства: способствуют объединению мелких капель воды в более крупные, облегчая их удаление из нефти;
* cистемы дренажа: предназначены для удаления осадка и воды, накопившихся на дне отстойника.

Современные отстойники могут оснащаться нагревательными элементами для снижения вязкости нефти, системами аэрации для улучшения разделения фаз, автоматическими датчиками уровня для контроля и регулирования процессов.

Использование отстойников нефти позволяет значительно повысить качество подготовленного продукта, снижая содержание воды и механических примесей, что важно для эффективной и безопасной транспортировки и переработки нефти.

На рисунке 1 приведены примеры отстойника нефти.

  
Рисунок 1 – Примеры отстойника нефти

В таблице 1 описаны технические характеристики отстойника нефти.

Таблица 1 ‒ Технические характеристики отстойников нефти

|  |  |  |
| --- | --- | --- |
| Наименование показателей | ОГН-200 | ОБН-3000 |
| Объем аппарата, м3 | 200 | 100 |
| Пропускная способность, м3/сут | 4000-8000 | 3000-6000 |
| Обводненность выходящей нефти не более, % | 0,2–0,5 | 0,5 |
| Рабочая среда | нефть (нефтепродукт), вода | нефть, пластовая вода |
| Рабочее давление, МПа | 0,6 | 0,6 |
| Температура среды, °С | до 100 | до 100 |
| Диаметр емкости, мм | 3400 | 3400 |

* + 1. Описание сепарационной емкости

Сепарационные ёмкости — это устройства, предназначенные для разделения многокомпонентных смесей на отдельные фазы, такие как газ, жидкость и твёрдые примеси. Они широко применяются в нефтегазовой промышленности для эффективного отделения нефти, газа и воды [11].

Основной функцией сепарационных емкостей является сепарация сырой нефти, которая включает в себя дегазацию (отделение газа от жидкости) и первичное разделение водонефтяной эмульсии с применением химических реагентов, ингибиторов и реактивов.

Сепарационные ёмкости классифицируются по ориентации на:

* горизонтальные сепараторы: имеют горизонтальную цилиндрическую форму и используются для разделения больших объёмов жидкости и газа;
* вертикальные сепараторы: представляют собой вертикальные цилиндры и применяются при ограниченном пространстве или для обработки смесей с высоким содержанием газа.

И по принципу действия на:

* гравитационные сепараторы: основаны на различии плотностей компонентов смеси, что приводит к их разделению под действием силы тяжести;
* центробежные сепараторы: используют центробежную силу для ускорения процесса разделения компонентов смеси.

Типичная сепарационная ёмкость состоит из следующих основных элементов:

* входной патрубок: обеспечивает равномерное распределение смеси внутри ёмкости;
* сепарационная камера: основное пространство, где происходит разделение фаз;
* перегородки и направляющие: способствуют эффективному разделению компонентов смеси;
* фильтры или перфорированные перегородки: улавливают инородные предметы и предотвращают их попадание в выходные патрубки
* отводы для удаления осадка: расположены в нижней части ёмкости и предназначены для периодического удаления накопившихся примесей.

Принцип работы сепарационной емкости заключается в том, что смесь поступает в сепарационную ёмкость через входной патрубок, где благодаря конструктивным особенностям и использованию физических принципов (гравитация, центробежная сила) происходит разделение на отдельные фазы. Газ поднимается вверх и отводится через верхний патрубок, жидкость собирается в нижней части ёмкости, а твёрдые примеси оседают на дно и удаляются через специальные отводы [11].

Использование сепарационных ёмкостей позволяет повысить эффективность технологических процессов, обеспечить качество конечного продукта и снизить износ оборудования за счёт удаления нежелательных примесей.

На рисунке 2 приведены примеры сепарационной емкости.

  
Рисунок 2 – Примеры сепарационной емкости

В таблице 2 описаны технические характеристики сепарационных емкостей.

Таблица 2 ‒ Технические характеристики сепарационных емкостей

|  |  |  |
| --- | --- | --- |
| Наименование показателей | РЖМАШ НГС-200 | СНМАШ НГС-3400 |
| Объем аппарата, м3 | 200 | 150 |
| Объемная производительность, м3/сут | до 3500 | до 2250 |
| Исполнение корпуса | сталь 09Г2С | сталь 09Г2С-6/8 |
| Рабочая среда | нефть, пластовая вода | нефть, пластовая вода |
| Рабочее давление, МПа | до 6,3 | до 4,0 |
| Температура среды, °С | от -60 до 200 | от -60 до 40 |
| Диаметр емкости, мм | 3400 | 3400 |

* + 1. Описание запорного и запорно-регулирующего клапана

При растущих требованиях к автоматизации производства более распространёнными стали электрозадвижки, которые представляют собой разновидность трубопроводной арматуры, оснащённой электрическим приводом, предназначенной для управления потоками рабочих сред в различных промышленных системах. Они подразделяются на два основных типа: запорные и запорно-регулирующие.

Запорные электрозадвижки предназначены для полного перекрытия или открытия потока рабочей среды, обеспечивая надёжную изоляцию участка трубопровода. Их основная функция заключается в обеспечении герметичности и предотвращении утечек. Ключевыми компонентами таких задвижек являются корпус, запорный элемент (клин или шибер) и электрический привод, обеспечивающий дистанционное управление [12].

Принцип работы запорной электрозадвижки заключается в том, что электропривод перемещает запорный элемент (клин или диск), который перекрывает проходное сечение трубы. При открытом состоянии не препятствует потоку и минимизирует гидравлические потери.

Запорно-регулирующие электрозадвижки совмещают функции как запорной, так и регулирующей арматуры. Они позволяют не только перекрывать поток, но и изменять его параметры, такие как расход или давление, посредством частичного открытия или закрытия проходного сечения. Это достигается благодаря особой конструкции запорного элемента и возможности точного позиционирования с помощью электропривода [13].

Принцип работы запорно-регулирующей электрозадвижки заключается в том, что электропривод может фиксировать положение запорного элемента на разных уровнях, тем самым изменяя расход среды. При этом позволяет плавно изменять расход рабочей среды и снижает ударные нагрузки за счет плавного закрытия.

Обе современные разновидности электрозадвижек состоят из следующих элементов:

* корпус: в нефтяной промышленности используется углеродистая или легированная сталь;
* запорный элемент (клин, диск, параллельные створки): в нефтяной промышленности обычно используется клин;
* шпиндель (выдвижной или невыдвижной);
* электропривод: асинхронный мотор для запорных операций, сервопривод для запорно-регулируемых операций;
* уплотнители (металлические или эластичные).

В нефтяной промышленности электроприводы имеют взрывозащищенное исполнение, дистанционное управление и систему автоматического контроля.

Электроприводы по принципу действия бывают [14]:

* многооборотные: позволяют регулировать напор при помощи частичного перекрытия потока;
* неполнооборотные: полностью перекрывают проходное сечение;
* прямоходные: при смене положения запорные детали движутся поступательно.

Преимущества использования электрозадвижек [15]:

* дистанционное управление: электроприводы позволяют оперативно и точно управлять состоянием арматуры на расстоянии, что особенно важно в автоматизированных системах;
* повышенная безопасность: возможность быстрого перекрытия или регулирования потока снижает риски аварийных ситуаций и обеспечивает защиту оборудования и персонала;
* точность регулирования: запорно-регулирующие задвижки обеспечивают плавное и точное изменение параметров потока, что важно для поддержания стабильности технологических процессов.

Правильный выбор и грамотная эксплуатация электрозадвижек обеспечивают надёжность и эффективность работы трубопроводных систем, снижая эксплуатационные затраты и повышая безопасность производственных процессов.

На рисунке 3 приведены примеры электрозадвижек.

  
Рисунок 3 – Примеры запорной и запорно-регулирующей электрозадвижек

В таблице 3 и 4 описаны технические характеристики электрозадвижек.

Таблица 3 ‒ Технические характеристики запорных электрозадвижек

|  |  |  |
| --- | --- | --- |
| Наименование показателей | DINAMIKA 30с941нж 200/25 RUSHWORK DC | ПромАрм 30с941нж3  200/16 RUSHWORK AC |
| Условный проход (Ду), мм | 200 | 200 |
| Условное давление среды (Ру), МПа | 2,5 | 1,6 |
| Исполнение | ХЛ1 сталь 20Л | ХЛ1 сталь 20Л |
| Электропривод | асинхронный, неполнооборотный | асинхронный, неполнооборотный |
| Тип управления | местный, удаленный | местный, удаленный |
| Средний ресурс, циклов в сутки | 3000 | 2500 |
| Наработка на отказ циклов в сутки | 1000 | 500 |
| Срок службы, лет | 30 | 20 |

Таблица 4 ‒ Технические характеристики запорно-регулирующих электрозадвижек

|  |  |  |
| --- | --- | --- |
| Наименование показателей | DINAMIKA 30лс941нж 200/16 СибЗТА ЭМБ | БЗТПА 30лс999нж 200/25 AUMA SQ |
| Условный проход (Ду), мм | 200 | 200 |
| Условное давление среды (Ру), МПа | 1,6 | 2,5 |
| Исполнение | ХЛ1 сталь 20ГЛ | ХЛ1 сталь 20ГЛ |
| Электропривод | сервопривод, многооборотный | сервопривод, многооборотный |
| Тип управления | местный, удаленный | местный, удаленный |
| Средний ресурс, циклов в сутки | 3000 | 2500 |
| Наработка на отказ, циклов в сутки | 1000 | 500 |
| Срок службы, лет | 30 | 10 |

* + 1. Описание нагревательного элемента

Электронагреватели в отстойниках нефти играют ключевую роль в процессе разделения сырой нефти на составляющие компоненты. Подогрев водонефтяной эмульсии снижает её вязкость, что способствует более эффективному и быстрому разделению фаз в отстойниках. Для удобства использования в отстойник нефти встраиваются трубчатые электронагреватели (ТЭНы), которые представляют собой герметичные трубки с размещённой внутри обогревательной спиралью [16].

Преимущества использования электронагревателей в отстойниках нефти:

* эффективность разделения: подогрев снижает вязкость эмульсии, ускоряя процесс разделения нефти и воды;
* стабильность процесса: обеспечивается постоянная температура, что способствует стабильной работе отстойников;
* автоматизация: возможность интеграции электронагревателей в системы автоматического контроля и управления технологическими процессами.

Для отстойников нефти широко используются встраиваемые ТЭНы [17]:

* врезного типа: врезаются в стенки емкостей, трубопроводов или отстойников через резьбовые соединения или фланцы, являются более безопасным решением так как не контактируют со средой напрямую;
* погружного типа: погружаются непосредственно в жидкость через предусмотренные для этого люки с фланцевым соединением, являются более эффективными так как непосредственно контактируют со средой.

ТЭНы врезного типа предпочтительны для стационарных емкостей и резервуаров, а погружного типа предпочтительны в передвижных емкостях и бойлерах.

Таким образом, применение электронагревателей в отстойниках нефти является неотъемлемой частью технологического процесса, направленного на повышение эффективности и качества разделения сырой нефти.

На рисунке 4 приведены примеры нагревательного элемента.

  
Рисунок 4 – Примеры нагревательного элемента

В таблице 5 описаны технические характеристики нагревательных элементов.

Таблица 5 ‒ Технические характеристики нагревательных элементов

|  |  |  |
| --- | --- | --- |
| Наименование показателей | Полимернагрев НП-8-1-3000 | Полимернагрев НВ-24-3000 |
| Тип | погружной | врезной |
| Нагревательный элемент | керамический ТЭН-12 | керамический ТЭН-12 |
| Длина, мм | 3000 | 3000 |
| Рабочая среда | нефтепродукты, мазут, вода | нефтепродукты, мазут, вода |
| Напряжение, В | 220 | 220, 380 |
| Мощность, Вт | 8000 | 24000 |
| Удельная мощность, Вт/см2 | 0,5-2,0 | 0,5-1,2 |

* + 1. Описание труб

На нефтетехнологических установках для переработки и перекачки нефти используются трубы из углеродной или легированной стали с антикоррозионным покрытием, что обеспечивает стойкость к механическим нагрузкам.

Основные требования к трубам [18]:

* механическая прочность (устойчивость к давлению и нагрузкам);
* химическая стойкость (защита от воздействия агрессивных сред);
* температурная устойчивость (особенно для подогретой нефти);
* герметичность и надежность соединений (в основном фланцевые).

Самый распространённый диаметр стандартного современного трубопровода для перекачки нефти 150 и 200 мм, исходя из спецификаций условного прохода запорной арматуры. В зависимости от агрессивности среды и внешних факторов применяется армирование трубопровода композитным материалом (стекловолокном, базальтом).

Использование качественных трубопроводов в системах подготовки нефти – залог эффективной и безопасной работы производственного участка.

На рисунке 5 приведены примеры трубопровода для перекачки нефти.

Изображение выглядит как небо, облако, труба, на открытом воздухе

Контент, сгенерированный ИИ, может содержать ошибки.Изображение выглядит как небо, на открытом воздухе, труба, забор

Контент, сгенерированный ИИ, может содержать ошибки.  
Рисунок 5 – Примеры трубопровода

В таблице 6 описаны технические характеристики трубопровода.

Таблица 6 ‒ Технические характеристики труб

|  |  |  |
| --- | --- | --- |
| Наименование показателей |  |  |
| Условный диаметр, мм | 150 | 200 |
| Толщина стенки, мм | 4,5 | 6,0 |
| Рабочая среда | нефть, пластовая вода | нефть, пластовая вода |

* + 1. Описание датчика уровня жидкости

В нефтегазовой промышленности принято использовать уровнемеры для измерения уровня жидкости в резервуарах, отстойниках, сепараторах и других технологических емкостях [19].

Они обеспечивают:

* контроль уровня нефти и воды в отстойниках и емкостях;
* предотвращение перелива жидкости;
* автоматизацию процессов подготовки нефти и водоотделения.

Самый распространенный – уровнемер буйкового или поплавкового типа. Буй (поплавок с высокой плотностью) частично погружен в жидкость. Когда уровень изменяется, выталкивающая сила Архимеда меняется, что фиксируется чувствительным элементом (например, пружиной, рычагом или датчиком веса).

Основные элементы буйкового уровнемера:

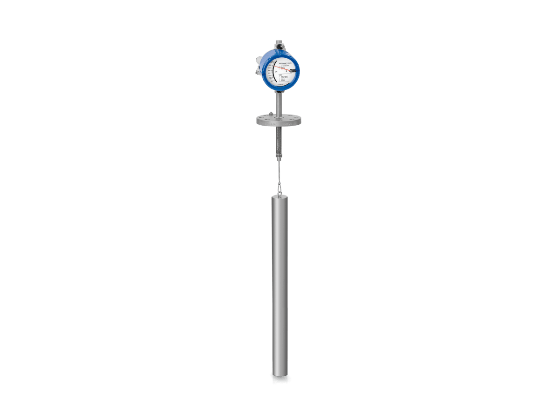
* буй (из нержавеющей стали, титана, ПВХ, керамики) – чувствительный элемент;
* трос/рычаговая система – передает движение буйка;
* измерительный механизм – преобразует изменение уровня в сигнал;
* электронный блок – передает данные в SCADA или АСУ ТП.

Основные виды буйковых уровнемеров:

* механические: буй соединен с противовесом и тросом, уровень измеряется по шкале;
* электронные: измеряют силу натяжения троса и преобразуют ее в цифровой сигнал;
* пневматические: используют давление газа для определения уровня жидкости.

Буйковый уровнемер – надежное и точное устройство для измерения уровня нефти и воды. Современные модели интегрируются в автоматизированные системы, что повышает эффективность контроля процессов подготовки нефти.

На рисунке 6 приведены примеры буйкового уровнемера.

  
Рисунок 6 – Примеры буйкового уровнемера

В таблице 7 описаны технические характеристики буйкового уровнемера.

Таблица 7 ‒ Технические характеристики буйкового уровнемера

|  |  |  |
| --- | --- | --- |
| Наименование показателей | BW-25 | Siemens SITRANS LG |
| Тип | механический буйковый | электронный буйковый |
| Принцип работы | Измерение силы натяжения буйка | Тензометрический датчик и микропроцессор |
| Диапазон измерения, м | 0,5-25 | 0,3-30 |
| Точность измерения | ±1,5% | ±0,5% |
| Температурный диапазон, °C | от -40 до 120 | от -196 до +450 |
| Давление в рабочей среде, МПа | до 1,6 | до 40 |
| Материал буйка | нержавеющая сталь, титан, ПВХ | нержавеющая сталь |
| Выходной сигнал | Подключение через датчик VEGA M или аналогичный | Интегрированный, 4-20 мА, HART, Profibus, Foundation Fieldbus |

* 1. Постановка задачи

Во время лабораторного практикума необходимо разработать систему моделирования работы нефтетехнологической установки с помощью SCADA-системы WinCC OA 3.17, которая будет управлять процессом производства товарной нефти, и с помощью которой пользователи смогут наблюдать за данным процессом посредством мнемосхемы, отображающей текущее состояние оборудования на разных этапах.

В системе должно быть реализовано три роли пользователей: администратор, оператор и гость.

Каждый пользователь должен авторизоваться в системе (ввести логин (от 4 до 12 символов) и пароль (от 4 до 12 символов)), система должна проверить учетные данные пользователей и настроить интерфейс пользователя на заданную роль.

* + 1. Функции системы

С различных установок по подготовке нефти водонефтяная эмульсия будет поступать по нефтепроводу до емкостей последней ступени сепарации нефти. Установка будет включать 4 ёмкости: Е0, Е1, Е2, Е3. Через запорный клапан V0 в сепарационную емкость E0 будет поступать сырая нефть с содержанием реагента-деэмульгатора Дин-4, где эмульсия начнет разрушаться. Через запорный клапан V1 в сепарационную емкость E1 будет поступать сырая нефть с содержанием реактива-коагулянта полиакриламида, где будут усиливаться процессы осаждения воды и механических примесей. Через запорный клапан V2 в сепарационную емкость E2 будет поступать сырая нефть с небольшим содержанием пресной воды, где будет происходить вымывание солей.

Из данных емкостей по нефтепроводу через запорно-регулирующие клапана V3, V4 и V5 продукты будут поступать в отстойник нефти E3, где они будут смешиваться естественным путем. В отстойнике нефти Е3 посредством повышения температуры при помощи встроенного нагревательного элемента вязкость нефти будет снижаться, а под действием гравитационных сил будет усиливаться разделение фаз на нефть и воду. При взаимодействии в отстойнике реагента, реактива, пресной воды, нагрева и гравитации, будет получаться слой товарной нефти, которая с отстойника нефти Е3 через запорный клапан V6 будет поступать в магистральный нефтепровод. В осадок будут выпадать остатки воды, солей и механических примесей, которые будут сливаться через запорный клапан V7 в дренажный трубопровод.

Система должна моделировать работу оборудования в автоматическом режиме (открывать/закрывать запорные и запорно-регулирующие клапаны, обеспечивать поступление сырой нефти с примесями в емкости Е0, Е1, Е2, Е3, включать/выключать нагревательный элемент в емкости Е3, контролировать температуру в емкости Е3, контролировать уровни заполнения всех емкостей, контролировать работу оборудования (обнаруживать отказы), восстанавливать работоспособность оборудования после ремонта, обеспечивать поступление товарной нефти в магистральный нефтепровод), моделировать возникновение нештатных ситуаций (отказов) в режиме реального времени и выдавать аварийные сообщения, визуализировать процесс работы оборудования (отображать штатную работу оборудования, отображать аварийные сообщения), выдавать справочную информацию.

Необходимо предусмотреть возможность экстренного закрытия клапанов и выключения нагревательных элементов при возникновении аварийных ситуаций.

Все технологические процессы должны быть визуализированы, на мнемосхеме должно отображаться состояния и параметры всего оборудования, на главной панели должны отображаться аварийные сообщения и другие элементы управления работой системы.

* + 1. Функции администратора

Основная задача администратора – настройка параметров оборудования, задействованного в технологическом процессе, а именно:

* задать скорость потока жидкости;
* задать процент открытия каждого регулируемого клапана;
* задать мощность нагревательного элемента;
* задать вероятность отказа оборудования.

Система должна обеспечить проверку корректности всех введенных администратором данных.

В системе должен быть реализован ручной режим управления оборудованием, а именно:

* запустить работу оборудования;
* остановить работу оборудования;
* включить нагревательный элемент;
* отрегулировать мощность нагревательного элемента;
* отключить нагревательный элемент;
* открыть запорный клапан;
* закрыть запорный клапан;
* открыть запорно-регулирующий клапан;
* отрегулировать запорно-регулирующий клапан;
* закрыть запорно-регулирующий клапан;
* квитировать аварийные сообщения;
* просматривать архивные параметры за заданный интервал времени;
* просматривать справочную информацию и текущие состояния оборудования;
* контролировать работу оборудования в режиме реального времени;
* отслеживать изменения параметров и реагировать на нештатные ситуации.
  + 1. Функции оператора

В системе должен быть реализован ручной режим управления оборудованием, а именно:

* запустить работу оборудования;
* остановить работу оборудования;
* включить нагревательный элемент;
* отрегулировать мощность нагревательного элемента;
* отключить нагревательный элемент;
* открыть запорный клапан;
* закрыть запорный клапан;
* открыть запорно-регулирующий клапан;
* отрегулировать запорно-регулирующий клапан;
* закрыть запорно-регулирующий клапан;
* квитировать аварийные сообщения;
* просматривать архивные параметры за заданный интервал времени;
* просматривать справочную информацию и текущие состояния оборудования;
* контролировать работу оборудования в режиме реального времени;
* отслеживать изменения параметров и реагировать на нештатные ситуации.
  + 1. Режим гостя

Гость сможет только наблюдать за работой системы (за ходом технологического процесса).

Администратор и оператор должны иметь возможность просмотра справочной информации о возможностях системы.

Таким образом, в системе должны быть реализованы следующие функции:

1. функции системы:
   * аутентификация пользователя в системе, настройка интерфейса пользователя на заданную роль;
   * моделирование работы оборудования в автоматическом режиме:
   * открытие/закрытие запорных и регулирующих клапанов;
   * обеспечение поступления сока в емкости;
   * включение/выключение нагревательных элементов в емкостях;
   * контроль температуры в емкостях;
   * контроль уровня заполнения емкостей;
   * контроль работы оборудования (обнаружение отказов);
   * восстановление работоспособности оборудования после ремонта;
   * обеспечение поступления товарной нефти в магистральный нефтепровод;
   * моделирование возникновения нештатных ситуаций (отказов) в режиме реального времени и выдача аварийных сообщений;
   * визуализация процесса работы оборудования:
   * отображение штатной работы оборудования;
   * отображение аварийных сообщений;
   * выдача справочной информации;
2. функции администратора:
   * авторизация пользователя в системе (ввод логина и пароля);
   * настройка параметров оборудования (точек данных):
   * задать скорость потока жидкости;
   * задать процент открытия каждого регулируемого клапана;
   * задать мощность нагревательного элемента;
   * задать вероятность отказа оборудования;
   * работа с оборудованием (ручной режим):
   * запустить работу оборудования;
   * остановить работу оборудования;
   * включить нагревательный элемент;
   * отрегулировать мощность нагревательного элемента;
   * отключить нагревательный элемент;
   * открыть запорный клапан;
   * закрыть запорный клапан;
   * открыть запорно-регулирующий клапан;
   * отрегулировать запорно-регулирующий клапан;
   * закрыть запорно-регулирующий клапан;
   * квитировать аварийные сообщения;
   * ремонт оборудования:
   * отправить оборудование в ремонт;
   * вернуть оборудование из ремонта;
   * просмотр архивных параметров за требуемый интервал времени;
   * просмотр справочной информации;
3. функции оператора:
   * авторизация пользователя в системе (ввод логина и пароля);
   * работа с оборудованием (ручной режим):
   * запустить работу оборудования;
   * остановить работу оборудования;
   * включить нагревательный элемент;
   * отрегулировать мощность нагревательного элемента;
   * отключить нагревательный элемент;
   * открыть запорный клапан;
   * закрыть запорный клапан;
   * открыть запорно-регулирующий клапан;
   * отрегулировать запорно-регулирующий клапан;
   * закрыть запорно-регулирующий клапан;
   * квитировать аварийные сообщения;
   * просмотр архивных параметров за требуемый интервал времени;
   * просмотр справочной информации;
4. функции гостя:
   * наблюдение за ходом технологического процесса.

1. Проектирование системы
   1. Разработка информационной модели объекта

Отправной точкой при создании проекта в WinCC OA является разработка информационной модели будущей системы, включающей иерархию и структуры данных, алармов и их свойств. Основой для формирования информационной модели может служить (в зависимости от назначения и масштаба системы) техническая структура объектов, включенных в SCADA-систему (аналогия «элемент – устройство – агрегат – …»), организационная модель объекта, например иерархия диспетчерских пунктов (местный уровень – региональный уровень – центральный уровень), или их сочетание. Информационная модель выражается в виде совокупности типов точек данных и созданных на их основе структур для хранения конкретных переменных процесса.

* + 1. Описание точек данных (Data Points)

Ключевым элементом при построении информационной модели в WinCC OA является понятие точки данных, связывающей переменные (теги), относящиеся к некоторому устройству или объекту, в единую древовидную структуру с возможностью создания произвольных подуровней. На практике в точку данных, относящуюся к одному устройству, объединяются примерно от 4 до 30 переменных.

В соответствии с идеологией WinCC OA для создания точек данных должен быть определен соответствующий тип точки данных, который будет выполнять роль шаблона при формировании множества точек данных с такой структурой.

Тип точки данных (DPT – data point type) – это описание структуры разнородных типизированных элементов, объединенных по признаку принадлежности к одному физическому объекту. Тип точки данных определяет формат представления сущности и показывает, в каком виде информация о процессе будет храниться в точке данных, какими элементами и какого типа будет описан реальный объект. Тип не хранит значение сущностей, конкретные значения будут храниться в соответствующей точке данных – экземпляре типа ТД. Точка данных (ТД, DP – data point) – это переменная, характеризующая состояние объекта, состоящая из элементов ТД, строго определенных структурой ТТД.

Элемент точка данных (ЭТД, DPE – data point element) – это динамическая переменная, обозначающая состояние объекта или внутренние состояния системы строго определенного типа данных.

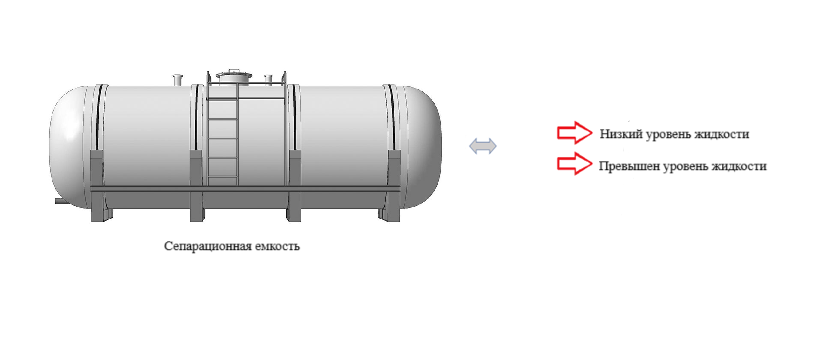
На рисунках 7-11 представлены информационные модели оборудования разрабатываемой системы, здесь DP – Data Point – Сигналы состояния, CMD – Command – Команды управления, AL – Alert – Аварийные сообщения.

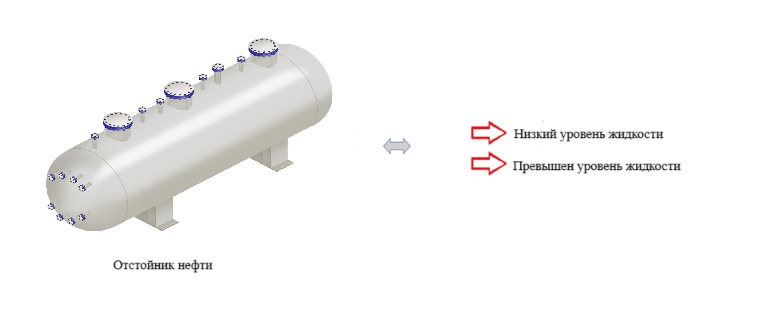


Рисунок 7 – Информационная модель запорного и   
запорно-регулирующего клапана

  
Рисунок 8 – Информационная модель датчика уровня

  
Рисунок 9 – Информационная модель нагревательного элемента

  
Рисунок 10 – Информационная модель сепарационной емкости

  
Рисунок 11 – Информационная модель отстойника нефти

В таблицах 8 и 9 представлено описание параметров оборудования, участвующих в построении информационной модели системы.

В системе должно быть предусмотрено:

* выдача ошибок при нарушении технологического процесса;
* остановка нагрева при нарушении технологического процесса;
* аварийное закрытие запорных и запорно-регулирующих клапанов при нарушении технологического процесса;
* аварийное ручное отключение.

Таблица 8 – Описание параметров оборудования

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Параметр | Диапазон значений | Тип параметра |
| Запорно-регулирующий клапан | Степень открытия | [0; 20; 40; 60; 80; 100] % | настраиваемый |
| Состояние | [открыт; закрыт] | логический |
| Запорный клапан | Состояние | [открыт; закрыт] | логический |
| Датчик уровня | Текущий уровень заполнения емкости | [0-3000] мм | измеряемый |
| Состояние | [работает; не работает] | логический |
| Нагревательный элемент | Мощность | [0-24] кВт | настраиваемый |
| Состояние | [работает; не работает] | логический |
| Сепарационная емкость | Уровень заполнения емкости | [0-3000] мм | измеряемый |
| Состояние | [работает; не работает] | логический |
| Отстойник нефти | Уровень заполнения емкости:  - по воде;  - по нефти. | [0-1000] мм  [1000-3000] мм | измеряемый |
| Температура | [0; 65] °С | измеряемый |
| Состояние | [работает; не работает] | логический |
| Труба | Скорость потока жидкости | [5-20] м3/час | настраиваемый |

Таблица 9 – Описание точек данных

| Точка данных | Команды | Состояния | Неисправности/Аварийные сообщения |
| --- | --- | --- | --- |
| Запорный и запорно-регулирующий клапан | открыть,  закрыть,  отправить в ремонт. | открыт,  закрыт,  в ремонте. | * не открылся; * не закрылся; * износ привода. |
| Датчик уровня | включить,  выключить,  отправить в ремонт. | включен, выключен,  отказал,  в ремонте. | * не включился; * не выключился. |
| Нагревательный элемент | включить,  выключить,  остудить,  нагреть. | включен,  выключен.  . | * недогрев, * перегрев. |
| Сепарационная емкость | – | – | * низкий уровень жидкости, * превышен уровень жидкости. |
| Отстойник нефти | – | – | * низкий уровень жидкости, * высокий уровень жидкости. |

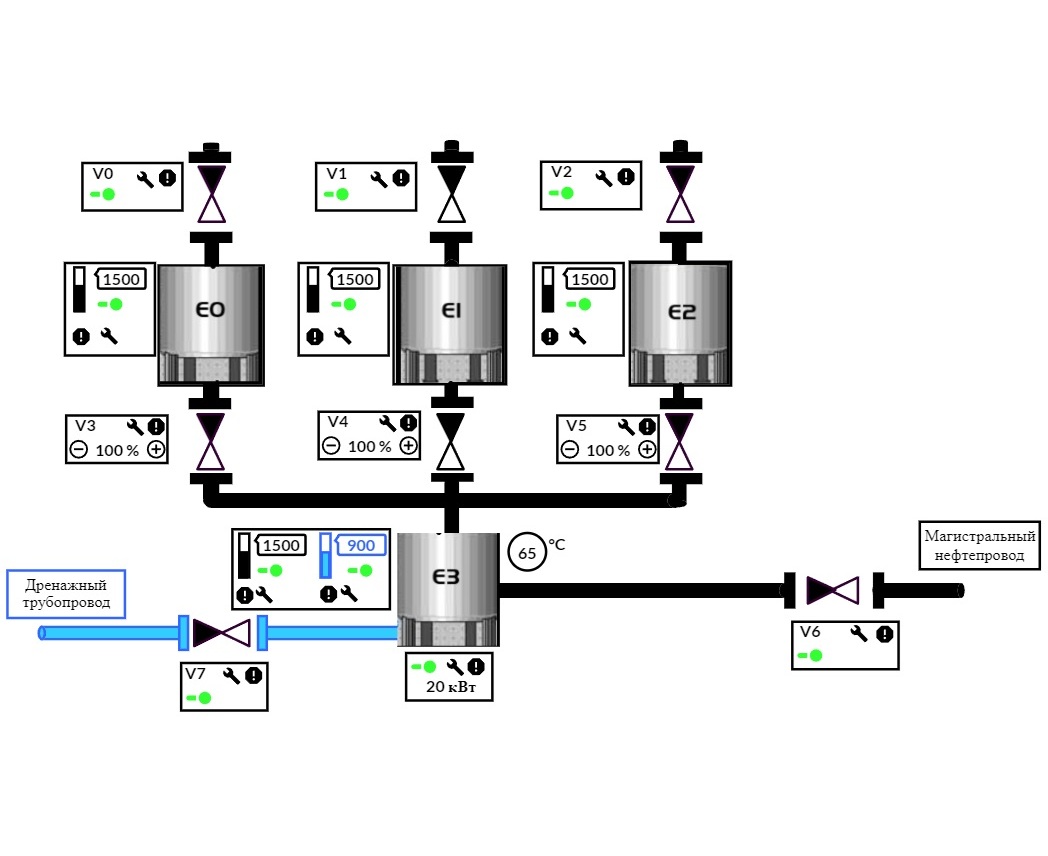
* 2. Проектирование интерфейса пользователя

Проектирование пользовательского интерфейса (ПИ) сегодня является очень важным аспектом разработки и требует от разработчиков специальных знаний о том, как разработать хороший интерфейс. Качество пользовательского интерфейса является самостоятельной характеристикой программного продукта, сопоставимо по значимости с такими его показателями, как надежность и эффективность использования вычислительных ресурсов.

Эффективный ПИ должен обеспечивать удобство, надежность, безопасность, обеспечивать интуитивное управление оборудованием, оперативную обратную связь, визуализацию процессов и доступ к справочной информации.

Отличительной особенностью интерфейса систем реального времени является наличие ограничения времени его реакции на внешние воздействия. В человеко-машинных системах реального времени в процессе управления участвует человек. Он может просто следить за тем, чтобы не случилось каких-нибудь аварийных ситуаций. Если они происходят, он должен предпринимать меры, помогая системе.

На рисунке 12 изображен прототип мнемосхемы разрабатываемой системы. Система будет включать 4 ёмкости: E0, Е1, Е2 и Е3. На вход сепарационной ёмкости Е0 необходимо подавать сырую нефть с реагентом-деэмульгатором Дин-4, на вход сепарационной ёмкости Е1 – сырую нефть с реактивом-коагулянтом полиакриламидом, на вход E2 – сырую нефть с содержанием пресной воды для вымывания солей и осадков. Поступление продуктов должно регулироваться запорными электрозадвижками V0, V1 и V2 соответственно. Из ёмкостей Е0, Е1 и Е2 продукты по трубопроводам должны поступать в отстойник нефти E3, где будет происходить их смешивание. Поступление продуктов в Е3 должно регулироваться запорно-регулирующими электрозадвижками V3, V4 и V5. На выходе емкости Е3 должна быть установлена запорная электрозадвижка V6, контролирующая отпуск готовой товарной нефти дальше на магистральный нефтепровод. Так же с E3 будет установлена запорная электрозадвижка V7 для сброса воды, механических примесей и осадков, получившихся в результате смешивания. В отстойнике нефти E3 должен быть установлен нагревательный элемент, который будет доводить температуру смеси до заданной температуры.

  
Рисунок 12 – Прототип мнемосхемы системы

На рисунке 13 представлен прототип панели настройки оборудования. Здесь будет можно регулировать мощность нагревательного элемента, указывать скорость потока жидкости, управлять открытием/закрытием регулируемых электрозадвижек, задавать вероятность отказа каждого типа оборудования.

На рисунке 14 представлен прототип главной панели приложения в автоматическом режиме для роли администратора. В системе будет предусмотрено переключение режима работы процесса выбором соответствующего пункта, кнопки  и  для управления запуском и остановкой процесса.

Изображение выглядит как текст, снимок экрана, число, Шрифт

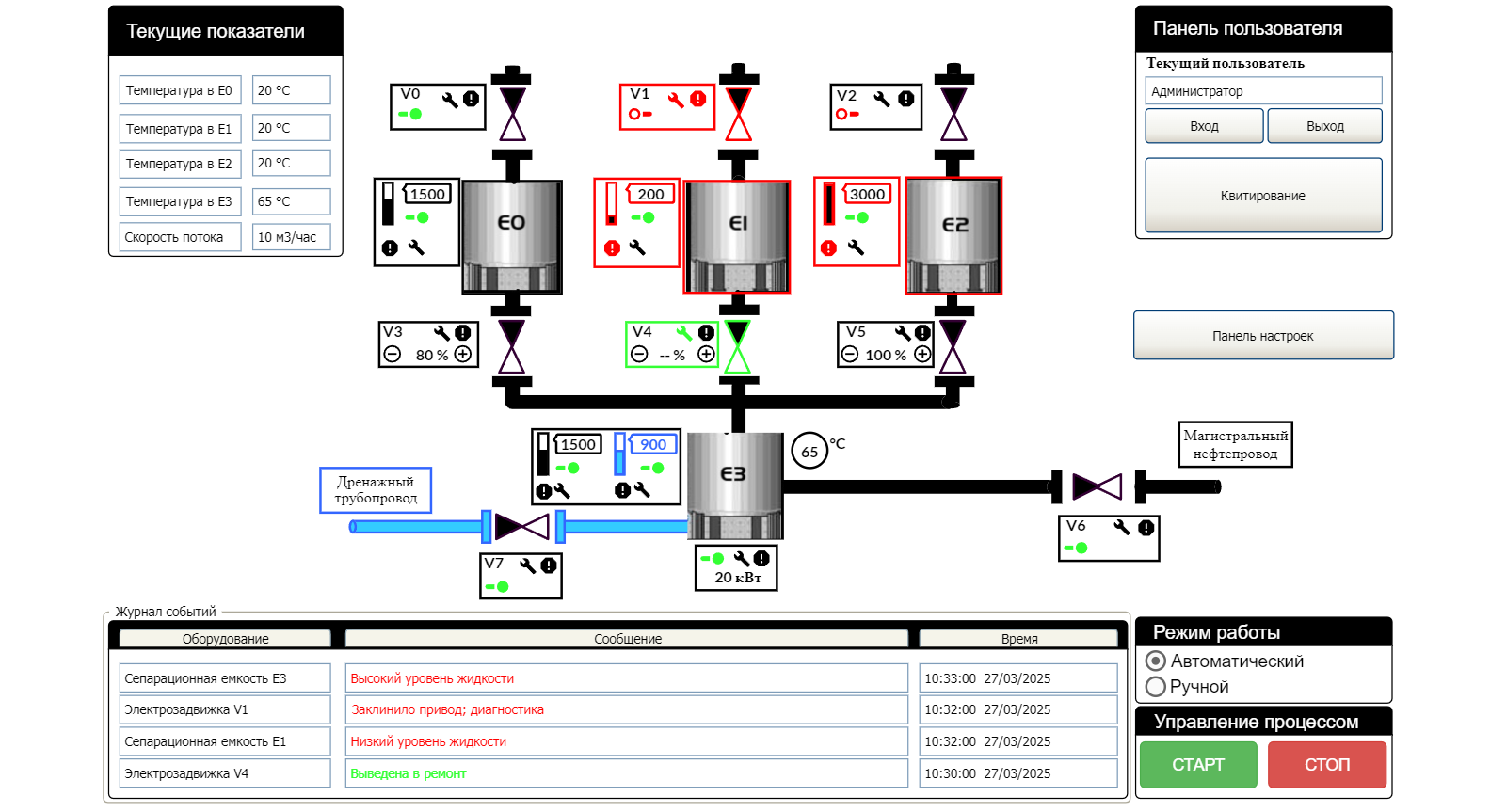
Контент, сгенерированный ИИ, может содержать ошибки.  
Рисунок 13 – Прототип панели настройки оборудования

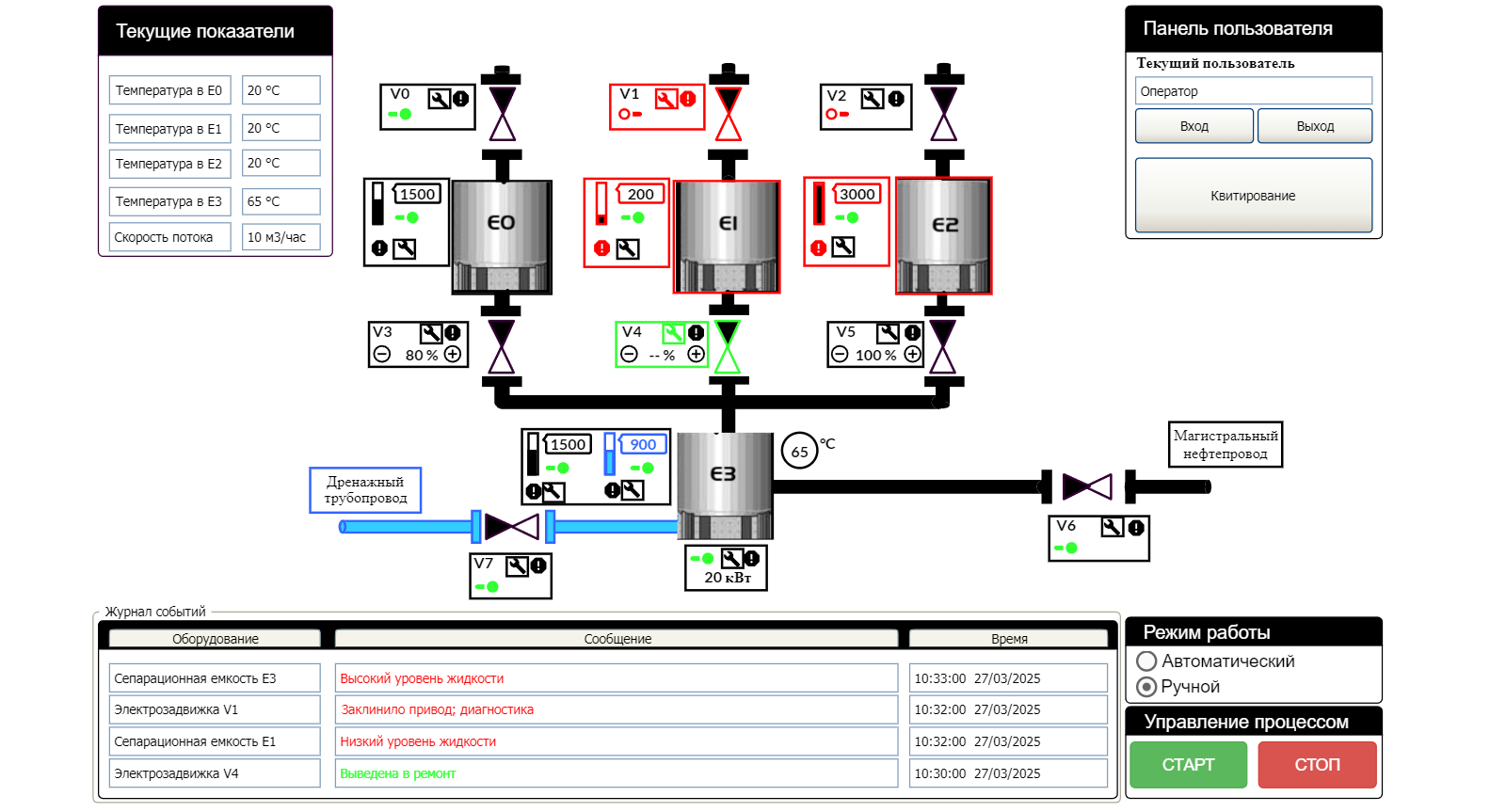
Для администратора будет предусмотрена панель настроек оборудования, нажав соответствующую кнопку Изображение выглядит как текст, Шрифт, снимок экрана, белый

Контент, сгенерированный ИИ, может содержать ошибки. для получения доступа. Процесс будет проходить согласно точкам данных, которые будет задавать сам администратор.

На рисунке 15 представлен прототип главной панели приложения в ручном режиме для роли оператора. В системе будет предусмотрено внесение изменений в процесс. С помощью соответствующих кнопок и переключателей будет можно открывать и закрывать запорные электрозадвижки, регулировать степень открытия/закрытия запорно-регулирующих электрозадвижек, включать и отключать уровнемеры и нагревательный элемент, а также выводить оборудование в ремонт при помощи иконки . При штатной работе оборудования индикация минимальна, при различных ситуациях будет предусмотрена цветовая индикация. Например, при сообщении об аварии или неисправности иконка  будет отображаться и окрашиваться красным цветом, как и силуэт оборудования. Иконка оборудования будет окрашиваться зеленым цветом, если оборудование находится в ремонте.

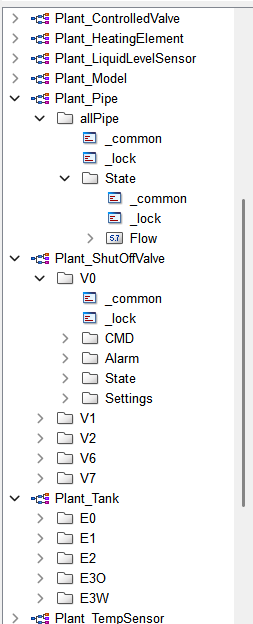
В системе будет предусмотрено визуальное отображение уровня заполнения емкостей, текущие показатели, общее квитирование ошибок, авторизация пользователя и журнал событий, в который будут включены оборудование-источник, текст сообщения и время получения.

  
Рисунок 14 – Прототип главной панели приложения (автоматический режим) для роли администратора

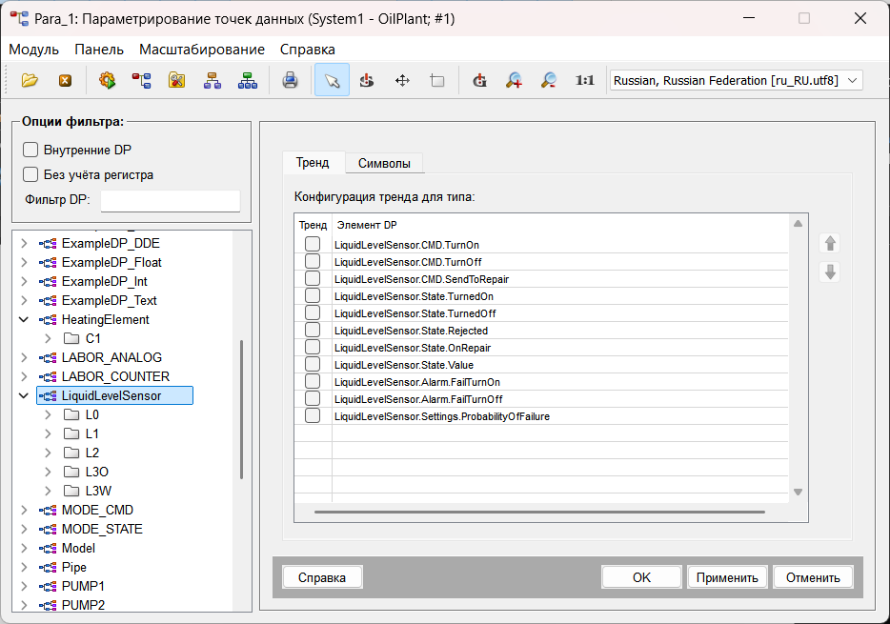
  
Рисунок 15 - Прототип главной панели приложения (ручной режим) для роли оператор

1. Реализация системы
   1. **Настройка элементов точек данных**

Для точной работы системы требуется организовать точки данных для элементов интерфейса. В WinCC OA точки данных представлены в виде дерева (рисунок 16).

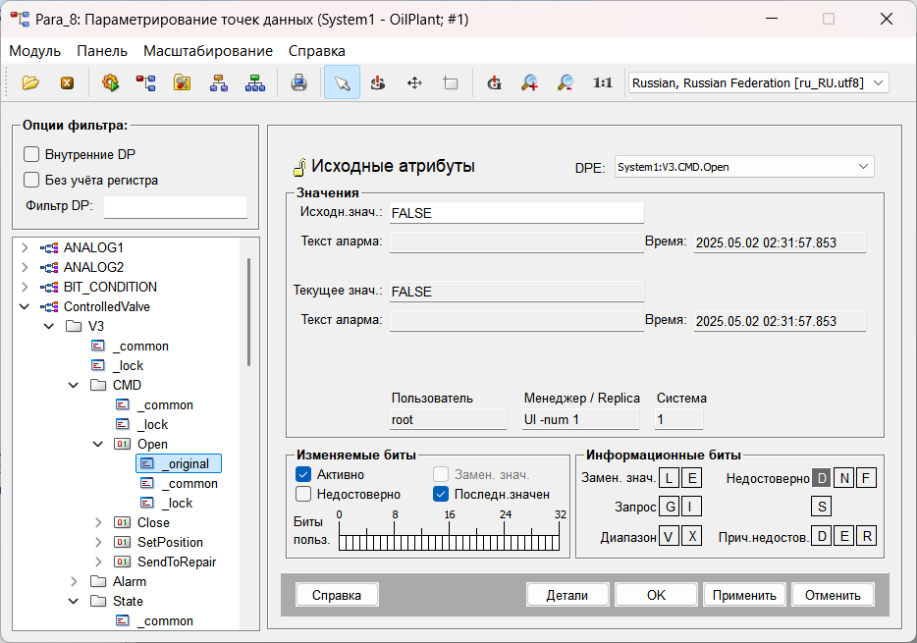
Рисунок 16 – Общая структура точек данных проекта (дерево ТД)

В соответствии с информационной моделью системы, разработанной в п. 2.1.1, авторами были внесены с систему необходимые точки данных (описание приведено в приложении А), на рисунке 17 приведен пример ТД «Датчик уровня».

 **Рисунок 17 – Описание ТД «Датчик уровня»**

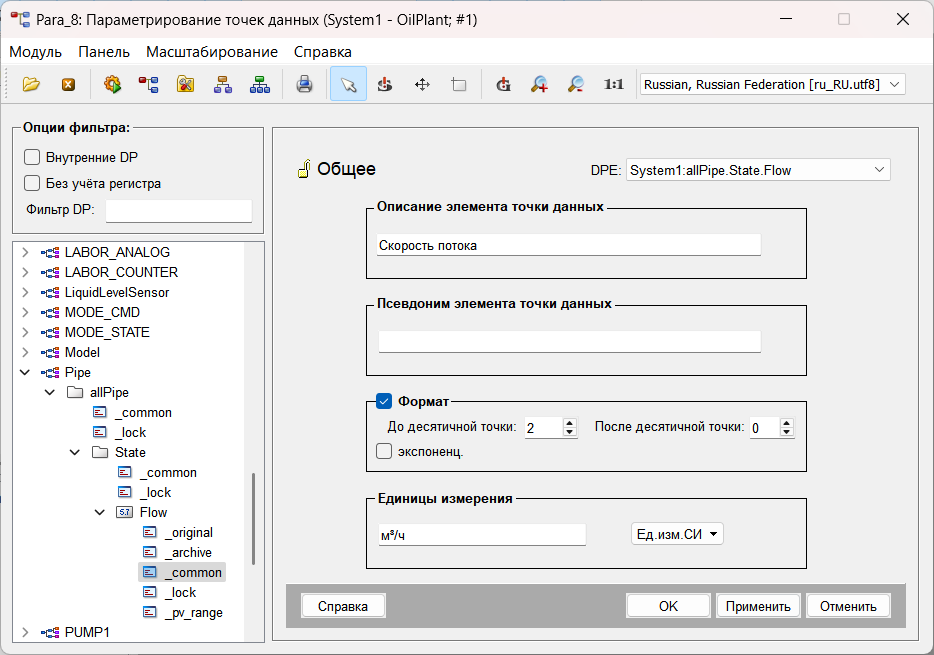
* 1. **Настройка исходных значений**

В зависимости от типа настраивается начальное значение точки (рисунок 18).

Рисунок 18 – Настройка \_original для ТД «Регулируемая электрозадвижка»

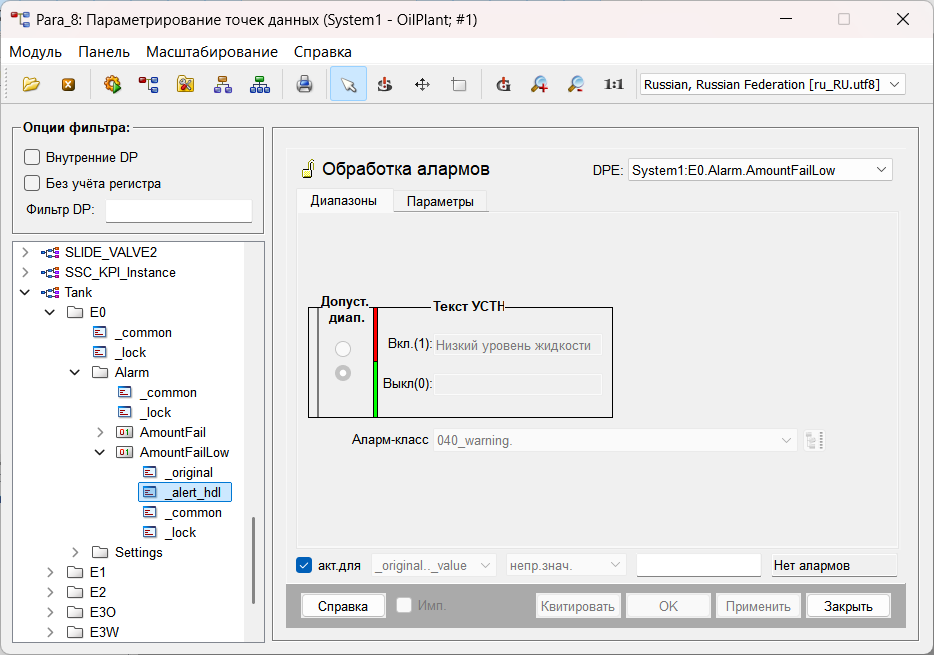
* 1. **Общие настройки объекта**

**Существует возможность также совершить общие настройки. Например, добавить описание или ограничить количество знаков после запятой для соответствующих типов точек данных (рисунок 19).**

  
Рисунок 19 – Настройка \_common для ТД «Трубопровод»

* 1. **Обработка сообщений (alert)**

Если значение точки данных превышает некое заданное значение, то можно настроить вывод соответствующего сообщения (Рисунок 20).

  
Рисунок 20 – Настройка \_alert\_hdl для ТД «Емкость»

* 1. **Разработка и описание графического интерфейса**

Графические пользовательские интерфейсы (ГПИ) играют ключевую роль в человеко-машинных интерфейсах (ЧМИ) систем реального времени (СРВ), обеспечивая эффективное взаимодействие операторов с технологическими процессами.

Основное назначение ГПИ в ЧМИ СРВ включает в себя:

* визуализацию данных - представление текущих параметров, состояний оборудования и сигналов тревоги;
* управление процессами - предоставление оператору средств для взаимодействия с системой, таких как кнопки и переключатели;
* обеспечение интуитивного интерфейса - создание понятного и логичного пользовательского интерфейса для быстрого реагирования на изменения в системе.

Развитый графический интерфейс, включающий использование цвета, анимации и других визуальных элементов, является основным достоинством ЧМИ, непосредственно влияющим на производительность и восприимчивость оператора [20].

При разработке ГПИ для ЧМИ СРВ учитываются следующие особенности:

* производительность - оптимизация количества и сложности графических элементов для обеспечения быстрого отклика интерфейса;
* масштабируемость - возможность адаптации интерфейса к различным разрешениям экранов и условиям эксплуатации;
* интерактивность - обеспечение обратной связи при взаимодействии с элементами управления;
* стандартизация - соблюдение единых стандартов и стилей оформления для повышения узнаваемости и удобства использования.

Программное обеспечение должно выполнять функции настройки параметров, запуска процедур сбора данных, записи данных с отображением характера регистрируемых сигналов и временного изменения параметров на экране дисплея, а также реализацию вычислительных алгоритмов цифровой обработки сигналов с отображением результатов комбинированными средствами представления информации [20].

* + 1. Описание графических элементов мнемосхемы

В разрабатываемую систему входят элементы, помогающие оператору эффективно использовать возможности управления технологическим процессом на объекте подготовки нефти, их описание приведено в таблицах 10 и 11.

Таблица 10 – Описание кнопок для управления

|  |  |
| --- | --- |
| Кнопка | Назначение |
| 1 | 2 |
|  | Кнопки изменения состояния регулирующей электрозадвижки |
|  | Вывод справочной информации |
|  | Переключатель состояния оборудования с индикацией (красный цвет – закрыто/отключено; зеленый цвет – открыто/включено) |
|  | Переключатель нагревательного элемента (включить/выключить) |
|  | Кнопка наладки (ремонта) элементов мнемосхемы (электрозадвижки V1-V7, ТЭН, датчики) при возникновении нештатной ситуации |

Продолжение таблицы 10

|  |  |
| --- | --- |
| 1 | 2 |
|  | Кнопки управления процессом |
|  | Переключатель режима работы оборудования.  Доступно администратору и оператору. |
|  | Кнопка общего квитирования неисправностей |
|  | Кнопка настроек оборудования. Доступна только администратору. |
|  | Меню переключения панелей между трендами и оборудованием |

Таблица 11 – Описание пиктограмм

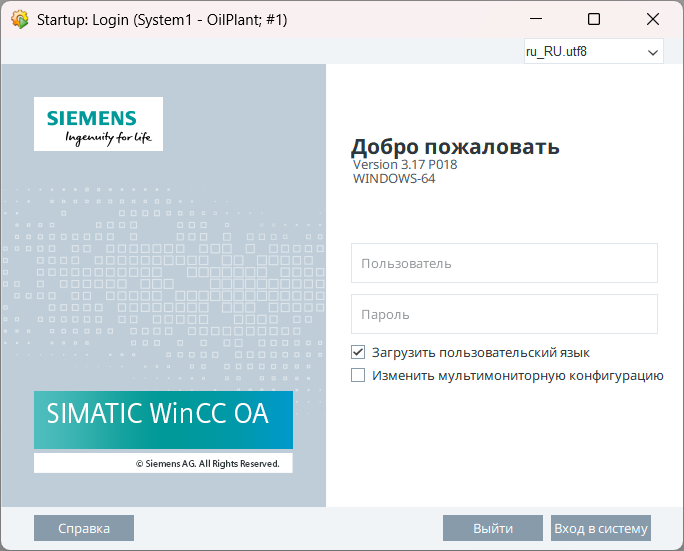
|  |  |
| --- | --- |
| Пиктограмма | Назначение |
| 1 | 2 |
|  | Индикация электрозадвижки на мнемосхеме (серый – закрыта, белый – открыта, зелёный – в ремонте, красный – авария) |
|  | Индикатор внештатной или аварийной ситуации |
|  | Визуальный индикатор уровня жидкости (желтый – низкий уровень, чёрный/синий для нефти/воды – нормальный, красный – высокий уровень/аварийная ситуация) |

Продолжение таблицы 11

|  |  |
| --- | --- |
| 1 | 2 |
|  | Отображение значений характеристик оборудования |
|  | Наименование соответствующего оборудования |
|  | Направление жидкости с установки |

* + 1. Начало работы

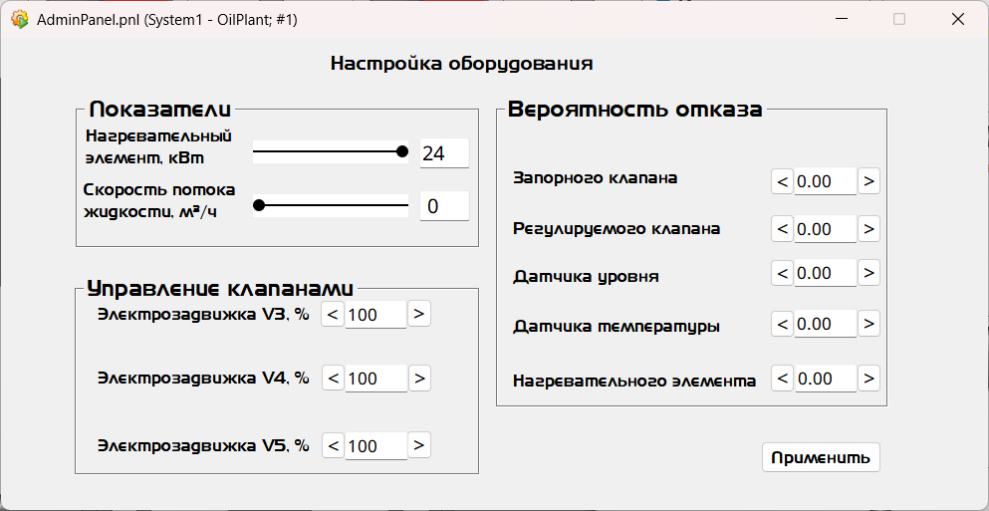
В системе предусмотрено три роли пользователей: администратор, оператор и гость. Перед началом работы с системой пользователи должны авторизоваться. На рисунке 21 приведено окно авторизации в системе, которое отображается при запуске системы.

  
Рисунок 21 – Окно «Вход в систему»

В центральной части окна расположены поля для ввода логина и пароля. В нижней части окна расположены кнопки «Вход в систему» и «Выйти». При нажатии кнопки «Выйти» окно авторизации закрывается. При нажатии кнопки «Вход в систему», если логин и пароль введены верно, окно для входа в систему закроется и откроется новое окно для работы с системой.

* + 1. Режим администратора

После входа в систему администратору открывается рабочее окно, в котором присутствует основная мнемосхема, панели уведомлений, текущих показателей, режима работы оборудования и управления процессом. Администратор начинает работу с нажатия на «Панель администратора» для настройки параметров системы (рисунок 22).

  
Рисунок 22 – Окно «Панель администратора»

В данном окне задается мощность нагревательного элемента, скорость потока жидкости, допустимый процент открытия электрозадвижек и вероятность отказа оборудования. Нажав «Применить», администратор закрывает настройки оборудования и может, нажимая соответствующие пункты, кнопки и переключатели:

- запускать и останавливать оборудование;

- включать/отключать датчики уровня жидкости E0-E3;

- открывать/закрывать электрозадвижки V0-V2, V6, V7;

- регулировать электрозадвижки V3-V5;

- включать/отключать нагревательный элемент E3 и регулировать мощность непосредственно на мнемосхеме;

- просматривать справочную информацию;

- квитировать ошибки и аварии;

- переключать режим работы оборудования;

- просматривать текущие показатели, внештатные события и тренды;

- выводить оборудование в ремонт после внештатной ситуации;

- визуально отслеживать основные технологические параметры оборудования (уровень жидкости в E0-E3, температура в E3, процент открытия V3-V5).

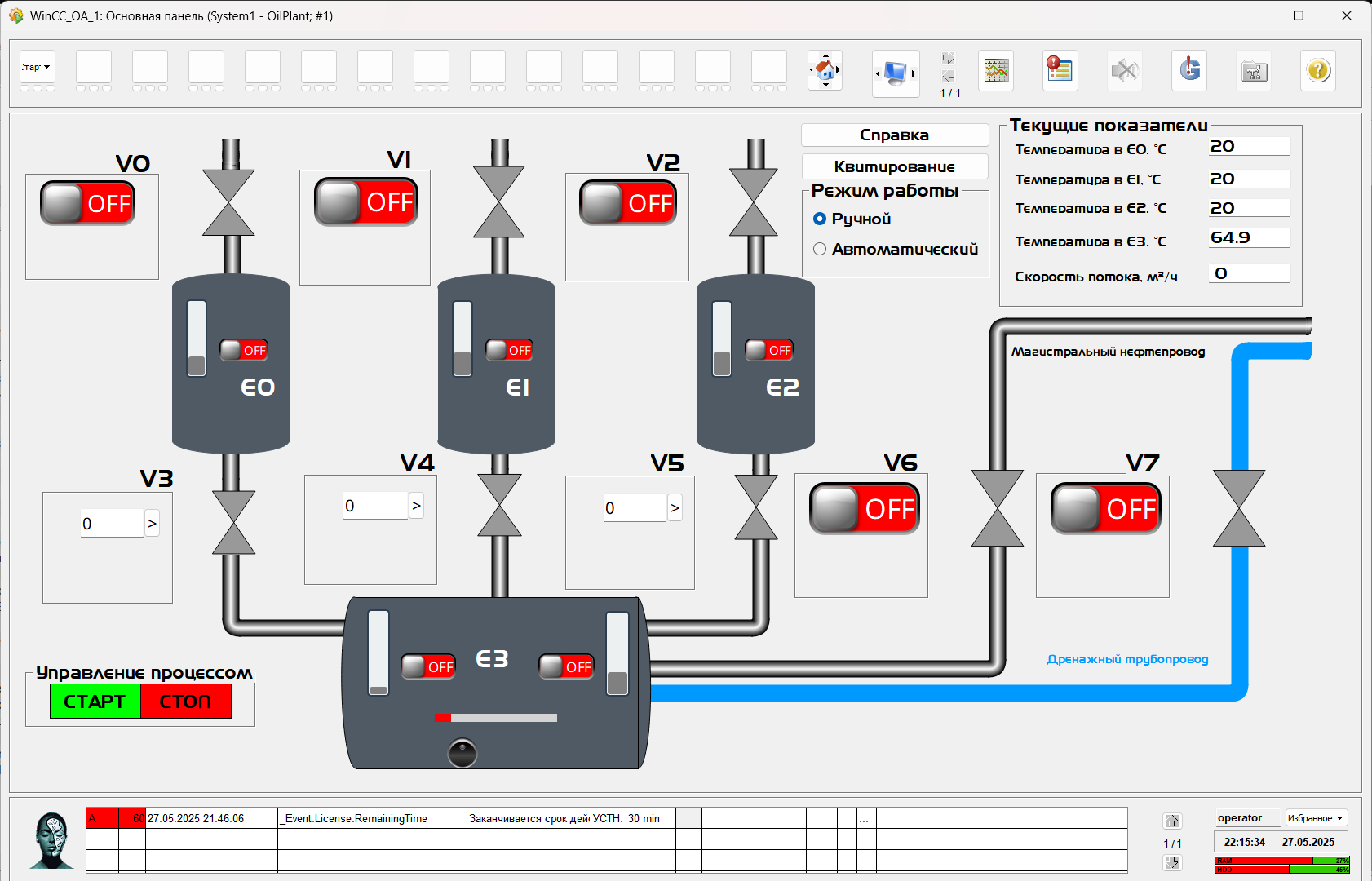
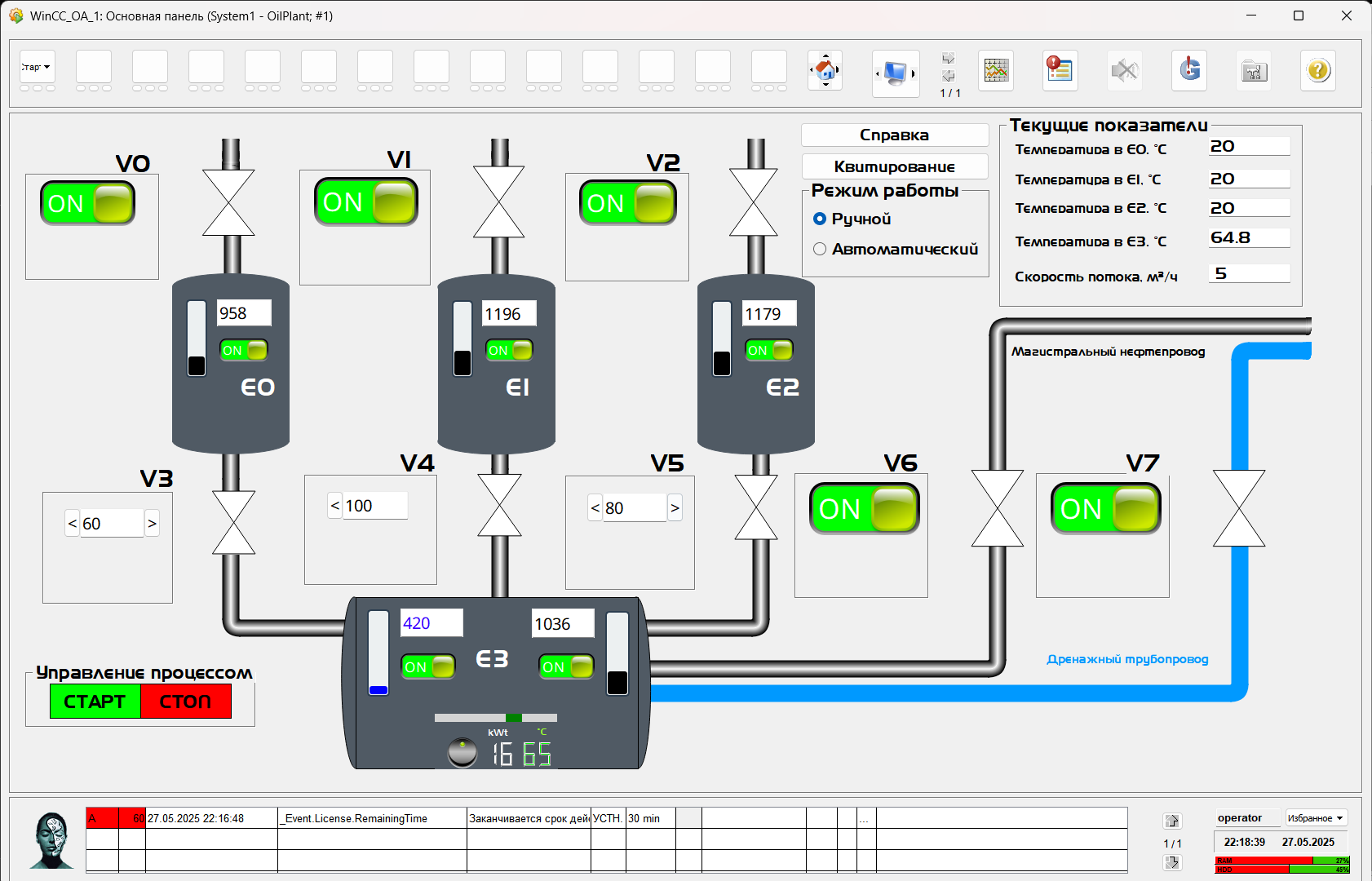
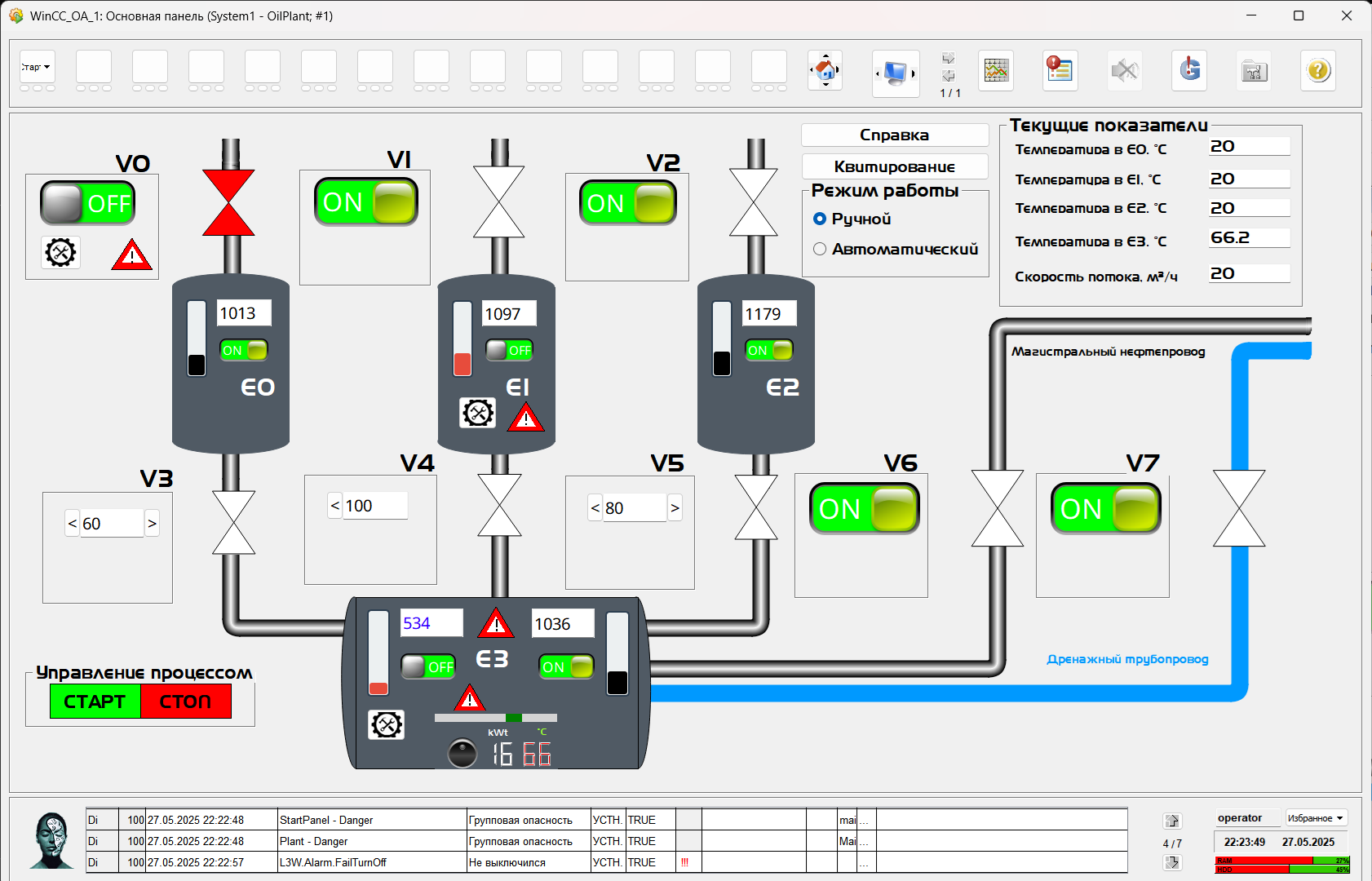
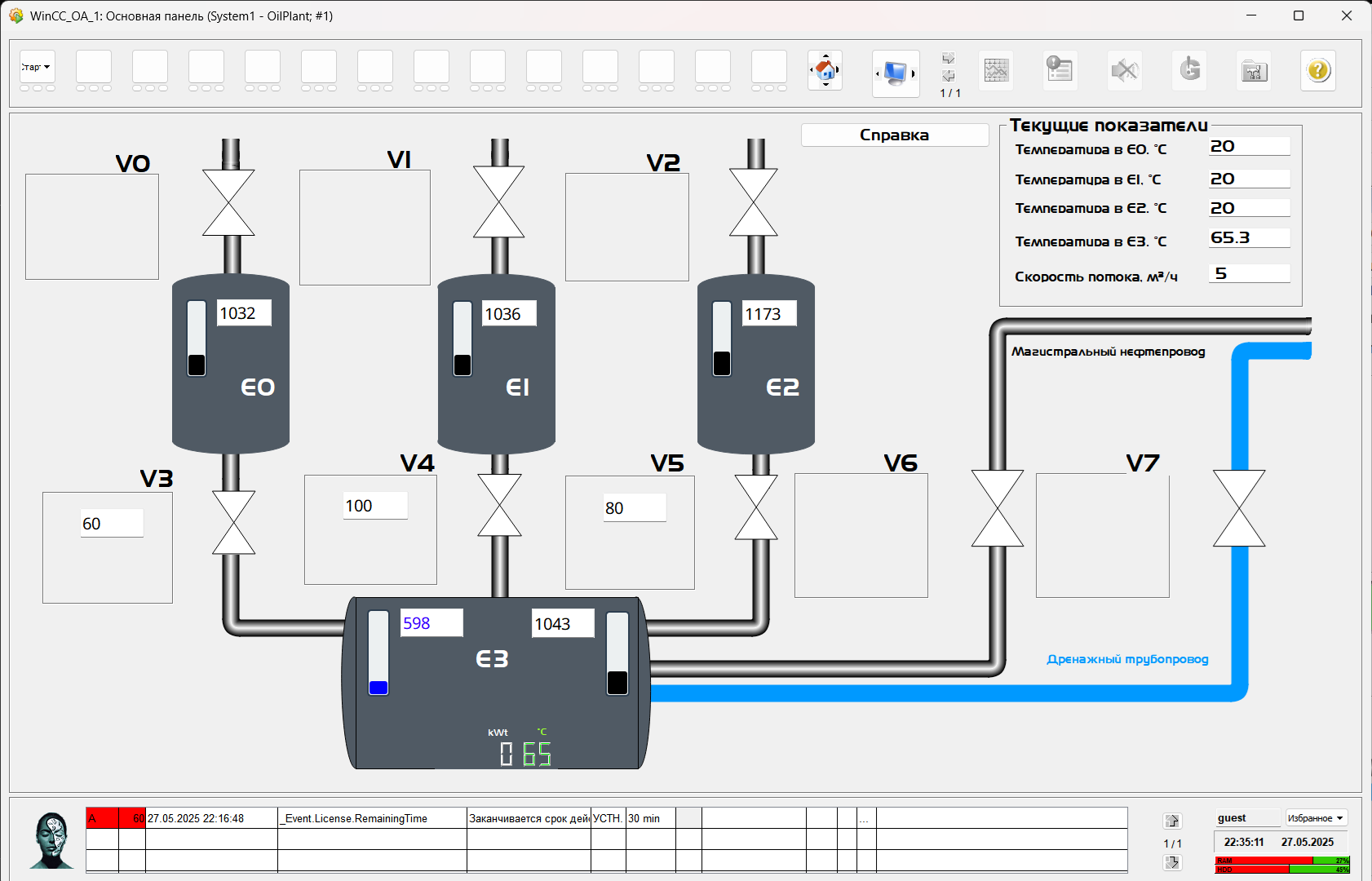
* + 1. Режим оператора

Оператор начинает работу с основного рабочего окна.

На рисунке 23 система приведена в начальное состояние, в котором отключено все оборудование и была проведена настройка администратором. Оператор может управлять всем доступным ему оборудованием, в том числе запускает систему кнопкой «СТАРТ» и включает поочередно оборудование, в том числе датчики уровня E0-E3, нагревательный элемент на E3, открывает необходимые электрозадвижки V0-V2, V6, V7 и устанавливает доступный процент открытия регулируемых электрозадвижек V3-V5.

На рисунке 24 приведено штатное состояние работы системы, где оператор непосредственно отслеживает состояние технологических показателей, а также может включить автоматический режим работы. В данном состоянии происходит непосредственно регулировка оборудования и отслеживание основных технологических показателей согласно основным регламентам установки в режиме реального времени.

**На рисунке 25 приведено аварийное состояние системы, при котором появляются соответствующие индикаторы и опознавательные цвета аварийных или внештатных ситуаций, привязанных к конкретному оборудованию, а также отображается соответствующая информация внизу панели. При этом появляются кнопки ремонта рядом с соответствующим оборудованием, нажав на которую оператор выводит в ремонт данное оборудование. После завершения ремонта оборудование вновь может быть введено в работу.**

  
Рисунок 23 – Основная панель приложения в начальном состоянии  
Рисунок 24 – Основная панель приложения в штатном состоянии  
Рисунок 25 – Основная панель приложения в аварийном состоянии  
Рисунок 26 – Основная панель приложения в гостевом режиме

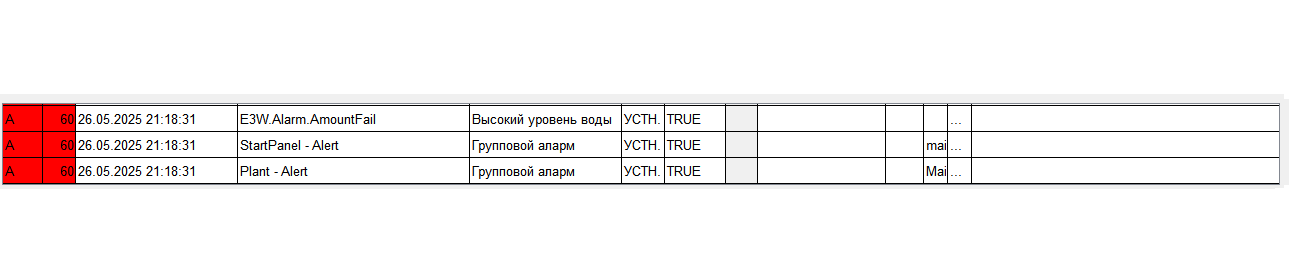
* + 1. Режим гостя

**На рисунке 26 приведено состояние оборудования в гостевом режиме, где для гостя недоступно и не предусмотрено управление оборудованием. В данном режиме предусмотрена только основная визуализация, в том числе визуальное отслеживание технологических параметров, и доступ к справочной информации.**

* + 1. **Обработка внештатных ситуаций**

**В системе предусмотрено возникновение внештатных ситуаций (alarm). Внештатная ситуация может быть связана с отказом оборудования или с превышением/снижением допустимых значений установленных технологических показателей в соответствии с технологическим процессом.**

**Аварийные сообщения о возникновении внештатной ситуации отображаются в нижней части рабочего окна. В списке выводятся время возникновения критической ситуации, панель аварии и описание проблемы (рисунок 27).**

**  
Рисунок 27 –** Сообщение о внештатной ситуации   
«Высокий уровень жидкости»

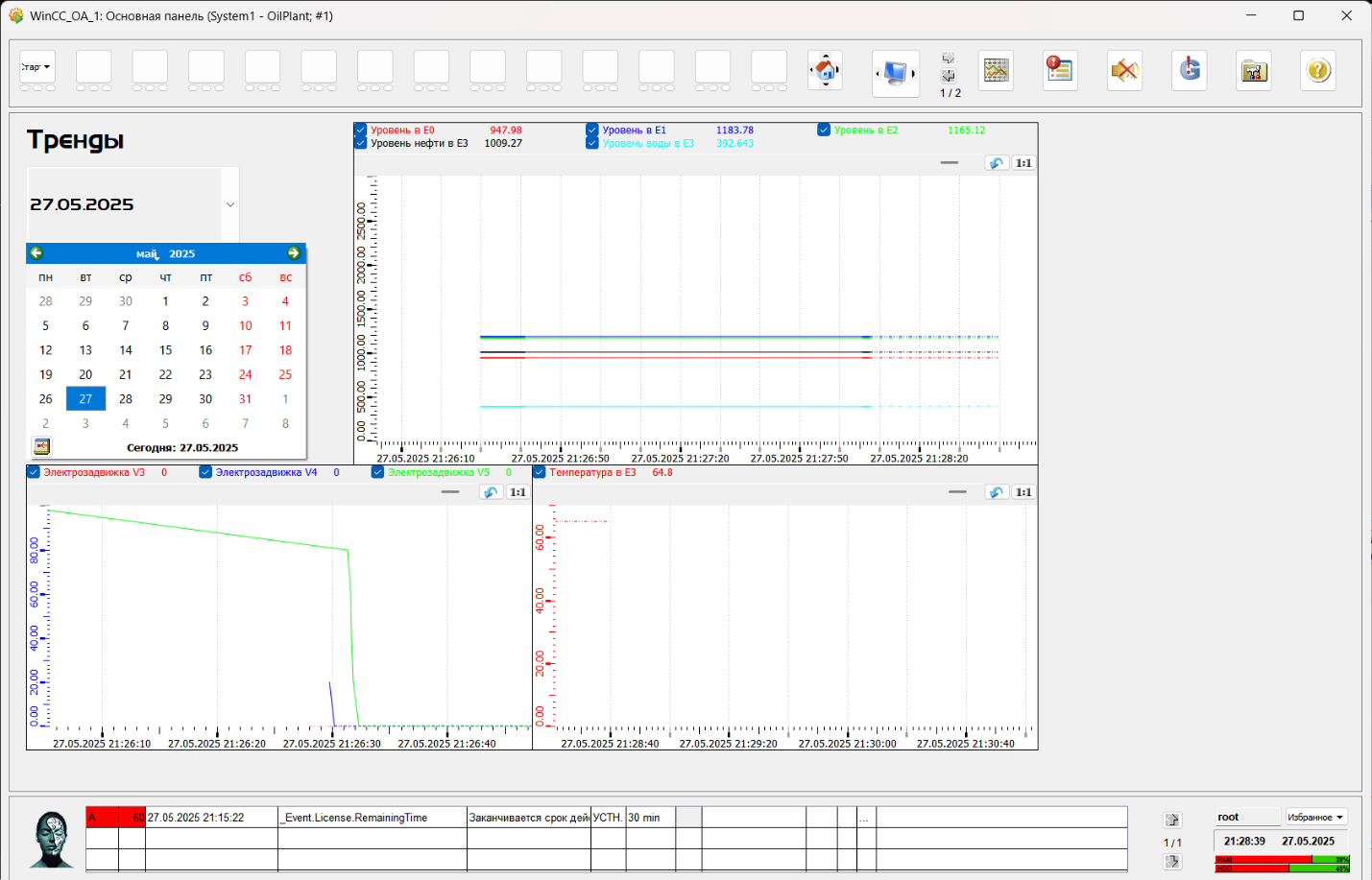
Для устранения отказа оборудования необходимо квитировать отказ и нажать на кнопку «Ремонт», которая непосредственно выводит оборудование в ремонт и окрашивает иконку оборудования в зелёный цвет в течение времени имитации ремонтных работ. Затем оборудование снова доступно для использования и находится в закрытом/выключенном положении.

При нарушении допустимых значений показателей технологического процесса (например, превышение верхнего порога жидкости) аварийное сообщение не квитируемо и автоматически убирается по достижению допустимых для технологического процесса значений.

* + 1. **Просмотр параметров оборудования**

**Для отображения архива значений параметров необходимо выбрать в меню панель трендов и указать в календаре интересующую дату. После этого на панели отобразятся тренды – визуализация** зависимости основных технических величин от времени. Таким образом можно отследить изменение значений параметра в определенный период времени, выбирая левой кнопкой мыши необходимый период в часах и минутах непосредственно на графиках трендов. Панель трендов включает в себя основную технологическую информацию о температуре в Е3, значениях открытия запорно-регулирующих электрозадвижек V3-V5 и уровней жидкости в емкостях E0-E3.

**Панель трендов приведена на рисунке 28.**

**Рисунок 28 – Панель трендов**

**Заключение**

**В ходе выполнения лабораторного практикума были изучены возможности SCADA-системы WinCC OA 3.17, с ее помощью была разработана система моделирования нефтетехнологической установки по сбросу воду и получения товарной нефти.**

**В первой главе приведены основные понятия предметной области, рассмотрены принципы работы нефтетехнологической установки, технологический процесс получения товарной нефти и основное оборудование и средства, применяемые в нефтяной промышленности на установках подготовки нефти. На основании данной информации были определены основные функции разрабатываемой системы и требования к ней.**

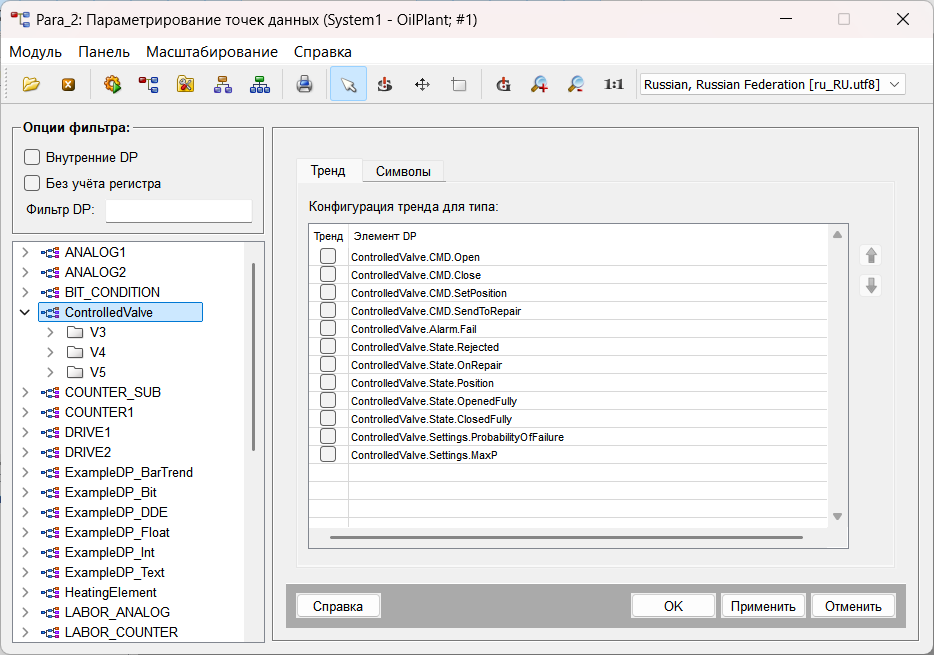
**Во второй главе приведены результаты проектирования системы: разработана информационная модель системы, определены основные точки данных для оборудования,** описан прототип интерфейса системы.

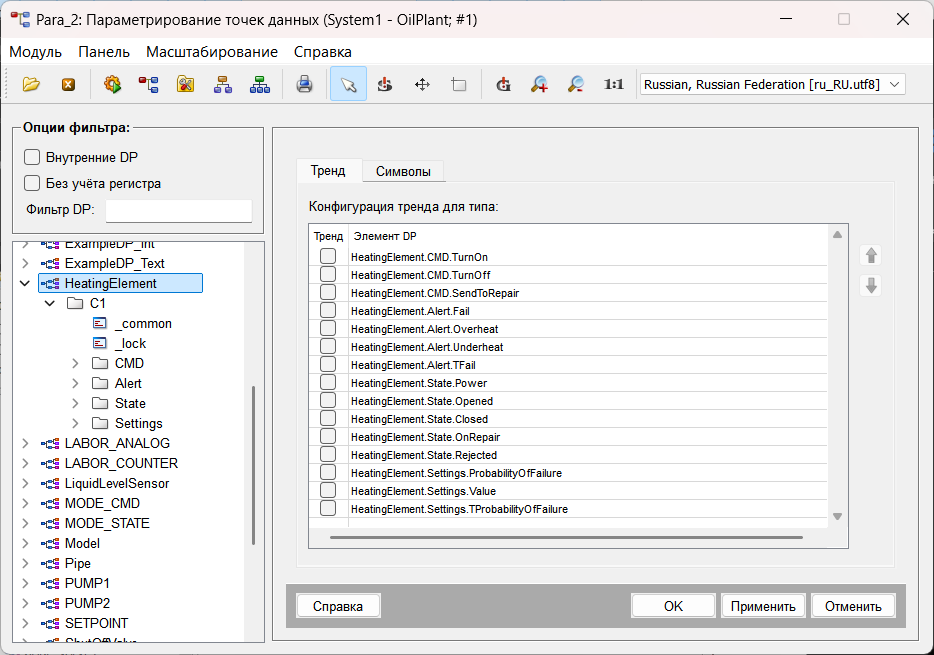
**В третьей главе описаны основные возможности разработанной системы: интерфейс пользователя с соответствием с мнемосхемой, созданы точки данных, написаны скрипты для реализации работы системы (см. приложение Б) и ее взаимодействия с пользователем, при нарушении допустимых норм в журнал сообщений приходит предупреждение.**

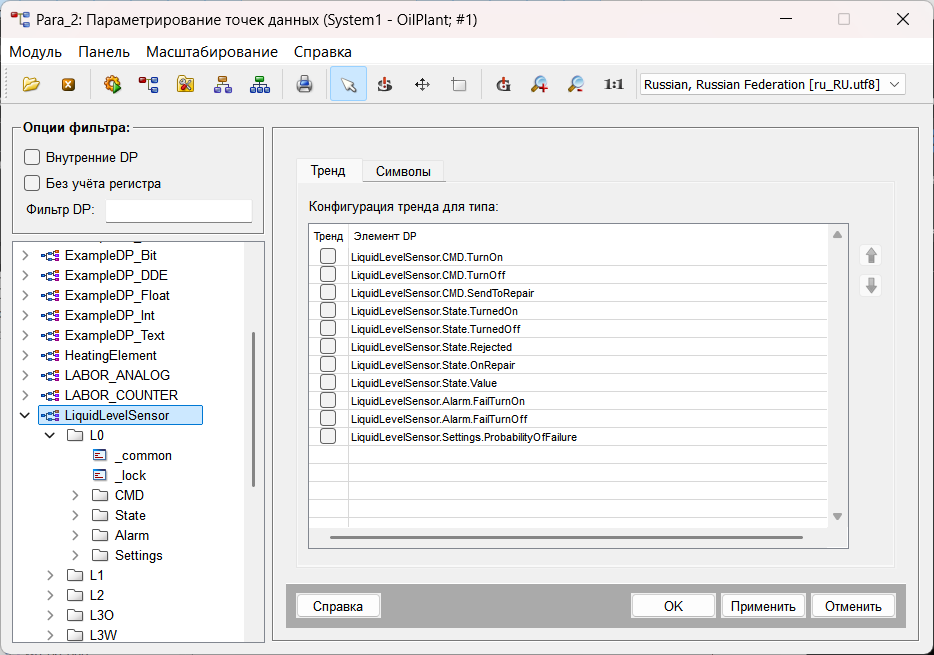
**Список использованных источников**

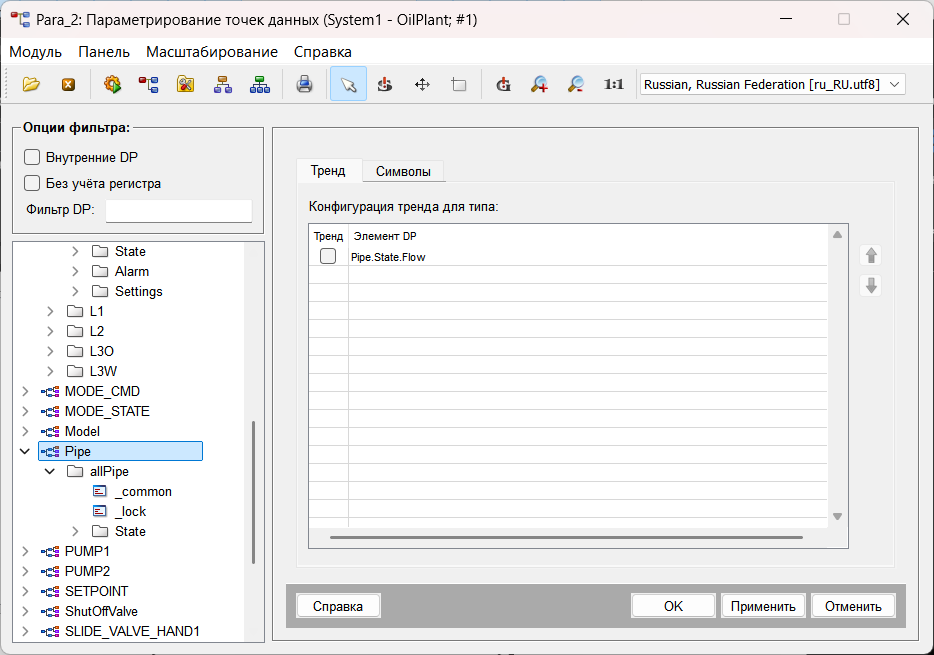
1. SCADA [Электронный ресурс] // URL: https://ru.wikipedia.org/  
   wiki/SCADA (дата обращения 28.02.2025).
2. Нефть [Электронный ресурс] // URL: https://ru.wikipedia.org/  
   wiki/Нефть (дата обращения 28.02.2025).
3. Деэмульгаторы нефти марки Decleave [Электронный ресурс]. / Группа Компаний Миррико. Официальный сайт // URL: https://mirrico.ru/products/demulsifiers-decleave/ (дата обращения 28.02.2025).
4. Справочник "Химия и технология нефти и газа" 3-е издание / Капустин В.М., Гуреева А.А. // Москва, Химия, 2006, стр. 245 (дата обращения 28.02.2025).
5. Сосуд под давлением [Электронный ресурс] // URL: https://ru.wikipedia.org/wiki/Сосуд\_под\_давлением (дата обращения 28.02.2025).
6. Справочник "Технология переработки нефти и газа"/ Позднышев Г.Н. // Москва, Недра, 1982, стр. 134–136 (дата обращения 2.03.2025).
7. Установка предварительного сброса воды [Электронный ресурс] // URL: https://ru.wikipedia.org/wiki/установка\_предварительного\_сброса\_воды (дата обращения 2.03.2025).
8. Горизонтальный отстойник нефти ОГ [Электронный ресурс]. / Саратовский резервуарный завод САРРЗ. Официальный сайт // URL: https://sarrz.ru/produkciya/separatory\_otstojniki/otstojnik\_og.html (дата обращения 5.03.2025).
9. Отстойник нефти ОН [Электронный ресурс]. / Торговый Дом Саратовского резервуарного завода. Официальный сайт // URL: https://tdsarrz.ru/produktsiya/separatory\_i\_otstoyniki\_dlya\_nefti\_gaza\_i\_vody/otstojnik\_nefti\_on.html (дата обращения 5.03.2025).
10. Отстойники нефти типа ОГН-П [Электронный ресурс]. / ООО ТД «ПЗЭМ». Пензенский Завод Энергетического Машиностроения. Оффициальный сайт // URL: https://pzem.ru/catalog/separatory-i-otstoyniki/otstoyniki/otstoynik-nefti-tipa-ogn-p/ (дата обращения 5.03.2025).
11. Нефтегазовые сепараторы: назначение, типы и особенности выбора [Электронный ресурс]. / Завод «ПензЭнергоМаш». Официальный сайт // URL: https://zavodpem.ru/company/articles/neftegazovye-separatory-naznachenie-tipy-osobennosti-vybora/ (дата обращения 5.03.2025).
12. Запорно-регулирующая арматура задвижки [Электронный ресурс]. / Промышленный портал Краснодарского края. Официальный сайт // URL: https://promkuban.ru/press/poleznaya-informaciya/zaporno-reguliruiushhaia-armatura-zadvizki-2024-25-iiulia-12-17 (дата обращения 9.03.2025).
13. Задвижки запорно-регулирующие [Электронный ресурс]. / Компания МКТ-АСДМ. Группа Компаний «Новые Технологии». Официальный сайт // URL: https://asdm.nt-rt.ru/images/manuals/  
    zadv\_zap\_regul.pdf (дата обращения 9.03.2025).
14. Типы электроприводов задвижек [Электронный ресурс]. / «Челябинская арматура». Официальный сайт // URL: https://www.chelarm.ru/articles/tipy-elektroprivodov-zadvizhek/ (дата обращения 9.03.2025).
15. Электроприводы для запорной арматуры, классификация и принцип работы [Электронный ресурс]. / Компания "ТЕХМАРКЕТ". Официальный сайт // URL: https://www.techmarcet.ru/informatsiya/stati/ehlektroprivody-dlja-zapornojj-armatury (дата обращения 9.03.2025).
16. Электрические нагреватели в процессах сепарации нефти и газа [Электронный ресурс]. / Компания ООО "ТД Электронагрев". Официальный сайт // URL: https://electro-nagrev.ru/primenenie/promyshlennyy-nagrev/electric-heaters-in%20oil-and-gas-separation-processes/ (дата обращения 11.03.2025).
17. Нагреватели для нефтепродуктов [Электронный ресурс]. / ООО «Полимернагрев». Официальный сайт // URL: https://polymernagrev.ru/  
    catalog/nagrevateli-dlya-nefteproduktov/ (дата обращения 11.03.2025).
18. ГОСТ 8732–78 (СТ СЭВ 1481–78). / Государственный стандарт союза ССР. / Трубы стальные бесшовные горячедеформированные (дата обращения 15.03.2025).
19. ГОСТ 8.563–2009. / Национальный стандарт Российской Федерации. / Методики (методы) измерений (дата обращения 15.03.2025).
20. Золотарев С., Фрейдман А., АСУ ТП на базе операционных систем QNX и Windows NT для нефтегазовых предприятий [Электронный ресурс]. / Компания SWD Software Ltd. Официальный сайт // URL: https://www.swd.ru/files/pdf/branches/Nautilus\_oil%26gas.pdf (дата обращения 5.05.2025).

ПРИЛОЖЕНИЕ А  
Описание настроек точек данных

   
Рисунок А.1 – Точка данных «Запорно-регулирующая электрозадвижка»

   
Рисунок А.2 – Точка данных «Нагревательный элемент»

   
Рисунок А.3 – Точка данных «Датчик уровня»

   
Рисунок А.4 – Точка данных «Трубопровод»

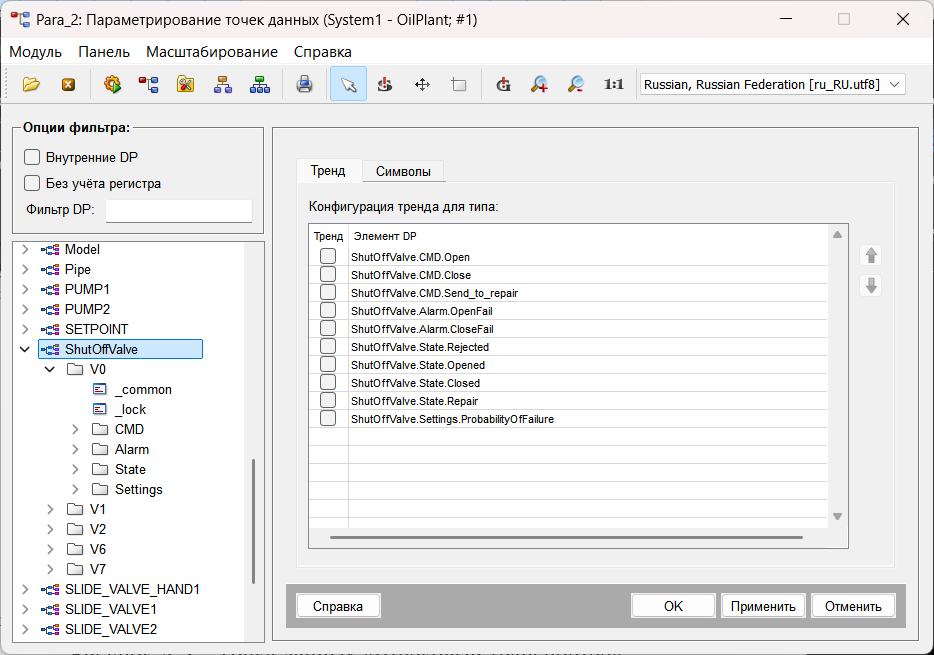
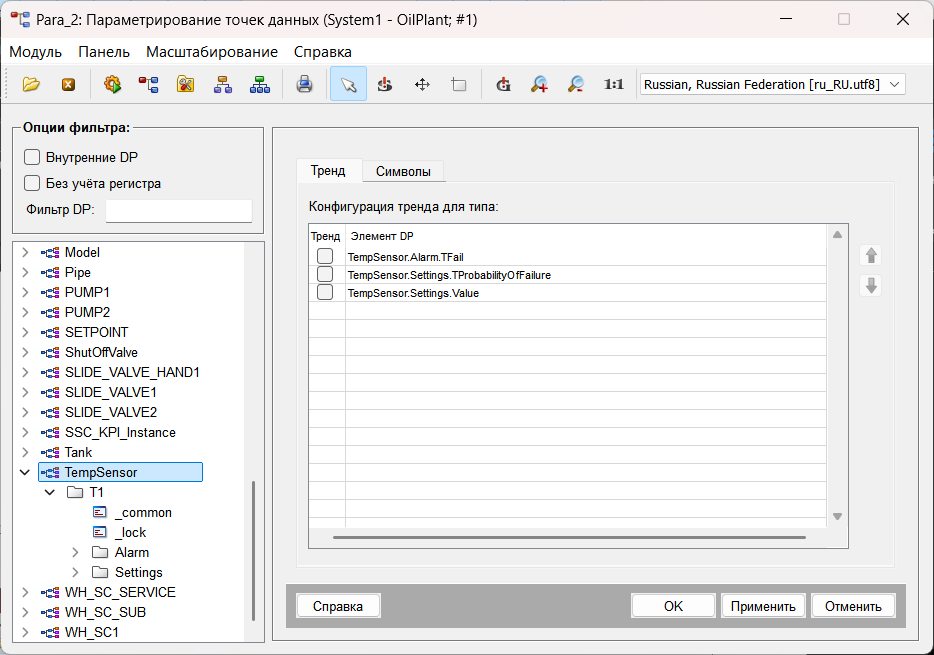
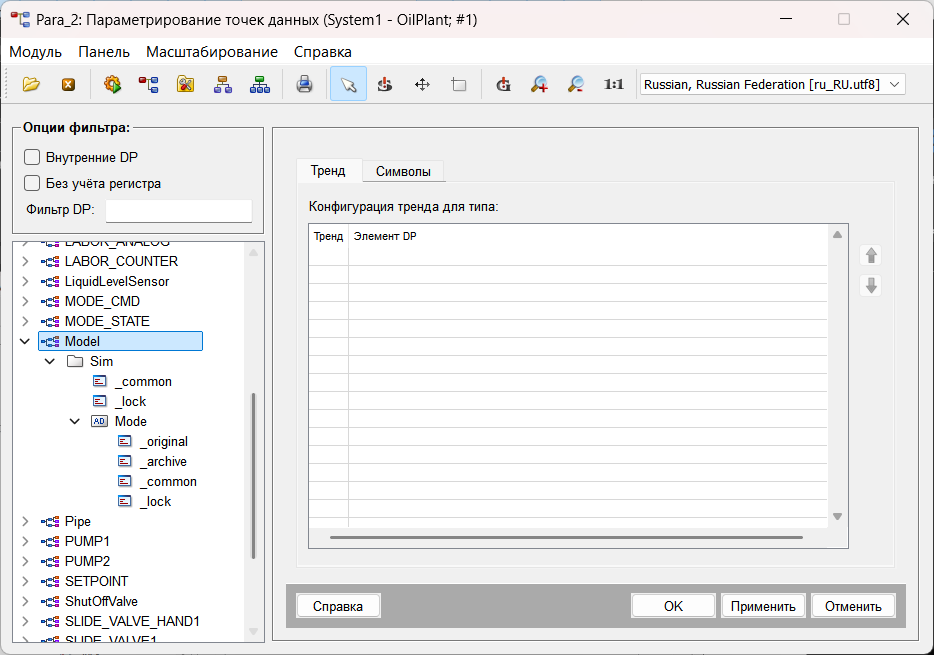
   
Рисунок А.5 – Точка данных «Запорная электрозадвижка»

   
Рисунок А.6 – Точка данных «Емкость»

   
Рисунок А.7 – Точка данных «Датчик температуры»

   
Рисунок А.8 – Точка данных «Режим работы»

ПРИЛОЖЕНИЕ Б  
Сценарии (скрипты управления)

[POLYGON1] [1] - [Initialize]

main()

{

EP\_setBackColor();

EP\_setChangeState();

EP\_RepairLogic();

EP\_AutoReject();

}

void EP\_AutoReject()

{

dpConnect("EP\_AutoRejectCB",

"System1:"+$ShuttOffValve+".Alarm.OpenFail:\_online..\_value",

"System1:"+$ShuttOffValve+".Alarm.CloseFail:\_online..\_value"

);

}

void EP\_AutoRejectCB(string dpOpenFail, bool isOpenFail,

string dpCloseFail, bool isCloseFail)

{

bool shouldReject = isOpenFail || isCloseFail;

dpSet("System1:"+$ShuttOffValve+".State.Rejected", shouldReject);

DebugTN("Rejected установлен в", shouldReject ? "true" : "false");

}

void EP\_setChangeState()

{

dpConnect("EP\_setChangeStateCB",

"System1:"+$ShuttOffValve+".CMD.Open:\_online..\_value",

"System1:"+$ShuttOffValve+".CMD.Close:\_online..\_value");

}

void EP\_setChangeStateCB(string dpOpen, bool cmdOpen, string dpClose, bool cmdClose)

{

float probFailure;

int result = dpGet("System1:"+$ShuttOffValve+".Settings.ProbabilityOfFailure", probFailure);

if(result != 0) return;

if(cmdOpen)

{

float random = (float)rand()/32767;

DebugN("Проверка открытия. Случайное значение:", random, "Вероятность отказа:", probFailure);

if(random <= probFailure) {

dpSet("System1:"+$ShuttOffValve+".Alarm.OpenFail", true);

} else {

dpSet("System1:"+$ShuttOffValve+".State.Opened", true,

"System1:"+$ShuttOffValve+".State.Closed", false);

}

dpSet("System1:"+$ShuttOffValve+".CMD.Open", false);

}

// Обработка команды CLOSE

if(cmdClose)

{

float random = (float)rand()/32767;

DebugN("Проверка закрытия. Случайное значение:", random, "Вероятность отказа:", probFailure);

if(random <= probFailure) {

dpSet("System1:"+$ShuttOffValve+".Alarm.CloseFail", true);

} else {

dpSet("System1:"+$ShuttOffValve+".State.Opened", false,

"System1:"+$ShuttOffValve+".State.Closed", true);

}

dpSet("System1:"+$ShuttOffValve+".CMD.Close", false);

}

}

void EP\_setBackColor()

{

dpConnect("EP\_setBackColorCB",

"System1:"+$ShuttOffValve+".State.Repair:\_online..\_value",

"System1:"+$ShuttOffValve+".State.Rejected:\_online..\_value",

"System1:"+$ShuttOffValve+".State.Opened:\_online..\_value");

}

void EP\_setBackColorCB(string dpRepair, bool isRepair,

string dpRejected, bool isRejected,

string dpOpened, bool isOpened)

{

if(isRepair == true) {

this.backCol = "Green";

DebugTN("Цвет: Зеленый (Ремонт)");

}

else if(isRejected == true) {

this.backCol = "Red";

DebugTN("Цвет: Красный (Авария)");

}

else if (isOpened == true) {

this.backCol = "White";

DebugTN("Цвет: Белый (Норма)");

}

else {

this.backCol = "Grey";

DebugTN("Цвет: Серый (В простое)");

}

}

void EP\_RepairLogic()

{

dpConnect("EP\_RepairLogicCB",

"System1:"+$ShuttOffValve+".CMD.Send\_to\_repair:\_online..\_value");

}

void EP\_RepairLogicCB(string dpRepair, bool cmdRepair)

{

if(cmdRepair)

{

DebugTN("Начало ремонта...");

dpSet("System1:"+$ShuttOffValve+".State.Repair", true);

delay(3);

dpSet(

"System1:"+$ShuttOffValve+".Alarm.OpenFail", false,

"System1:"+$ShuttOffValve+".Alarm.CloseFail", false,

"System1:"+$ShuttOffValve+".State.Rejected", false,

"System1:"+$ShuttOffValve+".State.Repair", false,

"System1:"+$ShuttOffValve+".CMD.Send\_to\_repair", false,

"System1:"+$ShuttOffValve+".CMD.Close", true,

"System1:"+$ShuttOffValve+".CMD.Open", false,

"System1:"+$ShuttOffValve+".State.Closed", true,

"System1:"+$ShuttOffValve+".State.Opened", false

);

DebugTN("Ремонт завершен. Все состояния сброшены.");

}

}

[FRAME1] [2] - [Initialize]

main()

{

EP\_setBackColor();

}

void EP\_setBackColor(string dpRepair, bool isRepair,

string dpRejected, bool isRejected,

string dpOpened, bool isOpened)

{

if(isRepair == true) {

this.borderColor = "Green";

DebugTN("Цвет: Зеленый (Ремонт)");

}

else if(isRejected == true) {

this.borderColor = "Red";

DebugTN("Цвет: Красный (Авария)");

}

else if (isOpened == true) {

this.borderColor = "White";

DebugTN("Цвет: Белый (Норма)");

}

else {

this.borderColor = "Grey";

DebugTN("Цвет: Серый (В простое)");

}

}

[Polyline1] [3] - [Initialize]

main()

{

EP\_setVisible();

}

void EP\_setVisible()

{

dyn\_errClass err;

if( !dpExists($ShuttOffValve+".State.Rejected"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_setVisibleCB",

$ShuttOffValve+".State.Rejected");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_setVisibleCB(string dp, bool boNewValue)

{

setValue("", "visible", boNewValue);

}

// SimpleCtrlScript {EP\_setVisible}

// DP {$dpe\_alarm}

// DPConfig {:\_alert\_hdl..\_act\_state\_color}

// DPTyp {string}

// Active {0}

// Operator {equal}

// Value {}

// SimpleCtrlScriptEnd {EP\_setVisible}

[Polyline2] [4] - [Initialize]

// SimpleCtrlScriptStart {valid}

main()

{

EP\_setVisible();

}

void EP\_setVisible()

{

dyn\_errClass err;

if( !dpExists($ShuttOffValve+".State.Rejected"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_setVisibleCB",

$ShuttOffValve+".State.Rejected");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_setVisibleCB(string dp, bool boNewValue)

{

setValue("", "visible", boNewValue);

}

[Line1] [5] - [Initialize]

// SimpleCtrlScriptStart {valid}

main()

{

EP\_setVisible();

}

void EP\_setVisible()

{

dyn\_errClass err;

if( !dpExists($ShuttOffValve+".State.Rejected"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_setVisibleCB",

$ShuttOffValve+".State.Rejected");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_setVisibleCB(string dp, bool boNewValue)

{

setValue("", "visible", boNewValue);

}

[Line2] [6] - [Initialize]

// SimpleCtrlScriptStart {valid}

main()

{

EP\_setVisible();

}

void EP\_setVisible()

{

dyn\_errClass err;

if( !dpExists($ShuttOffValve+".State.Rejected"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_setVisibleCB",

$ShuttOffValve+".State.Rejected");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_setVisibleCB(string dp, bool boNewValue)

{

setValue("", "visible", boNewValue);

}

[PUSH\_BUTTON1] [7] - [Clicked]

main()

{

EP\_RepairButton();

}

void EP\_RepairButton()

{

if(!dpExists("System1:"+$ShuttOffValve+".CMD.Send\_to\_repair:\_original..\_value") ||

!dpExists("System1:"+$ShuttOffValve+".State.Repair:\_original..\_value"))

{

DebugTN("Ошибка: отсутствуют необходимые точки данных");

return;

}

dpSetWait("System1:"+$ShuttOffValve+".CMD.Send\_to\_repair:\_original..\_value", true);

dpSetWait("System1:"+$ShuttOffValve+".State.Repair:\_original..\_value", true);

dpConnect("EP\_RepairCompleteCB",

"System1:"+$ShuttOffValve+".CMD.Send\_to\_repair:\_online..\_value");

}

void EP\_RepairCompleteCB(string dpRepair, bool cmdRepair)

{

if(cmdRepair)

{

DebugTN("Начало процедуры ремонта...");

delay(3);

dpSet(

"System1:"+$ShuttOffValve+".Alarm.OpenFail", false,

"System1:"+$ShuttOffValve+".Alarm.CloseFail", false,

"System1:"+$ShuttOffValve+".State.Rejected", false,

"System1:"+$ShuttOffValve+".State.Repair", false,

"System1:"+$ShuttOffValve+".CMD.Send\_to\_repair", false,

"System1:"+$ShuttOffValve+".CMD.Close", true,

"System1:"+$ShuttOffValve+".CMD.Open", false,

"System1:"+$ShuttOffValve+".State.Closed", true,

"System1:"+$ShuttOffValve+".State.Opened", false

);

DebugTN("Ремонт завершен. Все аварийные состояния сброшены.");

}

}

[PUSH\_BUTTON1] [7] - [Initialize]

main()

{

EP\_setVisibility();

}

void EP\_setVisibility()

{

dpConnect("EP\_setVisibilityCB",

"System1:"+$ShuttOffValve+".State.Rejected:\_online..\_value",

"System1:"+$ShuttOffValve+".State.Repair:\_online..\_value");

}

void EP\_setVisibilityCB(string dpRejected, bool isRejected,

string dpRepair, bool isRepair)

{

if(isRejected == 1 || isRepair == 1)

{

this.visible = true;

this.backCol = "Green";

}

else

{

this.visible = false;

}

}

[ToggleSwitch\_ewo1] [8] - [toggled]

toggled(bool on)

{

dpSet(

"System1:"+$ShuttOffValve+".CMD.Close", !on,

"System1:"+$ShuttOffValve+".CMD.Open", on

);

}

[ToggleSwitch\_ewo1] [8] - [Initialize]

main()

{

EP\_setBackColor();

EP\_SetVisible();

}

void EP\_setBackColor()

{

dpConnect("EP\_setBackColorCB",

"System1:"+$ShuttOffValve+".State.Closed:\_online..\_value");

}

void EP\_setBackColorCB(string dpSource, bool boNewValue)

{

if (!boNewValue){

setValue("", "backCol", "green");

this.checked = true;

}

else {

setValue("", "backCol", "red");

this.checked = false;

}

}

void EP\_SetVisible()

{

if (getUserPermission(2)) {

this.visible = true;}

else {

this.visible = false;

}

}

[PUSH\_BUTTON+] [1] - [Clicked]

main(mapping event)

{

int value;

int maxvalue;

dpConnect($ControlledValve+".State.OpenedFully",

$ControlledValve+".State.ClosedFully");

dpGet($ControlledValve+".State.Position", value,

$ControlledValve+".Settings.MaxP", maxvalue);

TF\_Position\_V3.text = value;

if (value < 90)

{

value = value + 20;

if (value > maxvalue) value = maxvalue;

dpSet($ControlledValve+".State.Position", value);

}

else if (value == maxvalue & getUserPermission(3))

{

dpSet($ControlledValve+".State.Position", value,

$ControlledValve+".Settings.MaxP", value);

}

else if (value == maxvalue & getUserPermission(7))

{

ChildPanelOnCentral("No.pnl",

"",

makeDynString(""));

}

}

[PUSH\_BUTTON+] [1] - [Initialize]

main()

{

EP\_setVisibility();

}

void EP\_setVisibility()

{

dpConnect("EP\_setVisibilityCB",

"System1:"+$ControlledValve+".State.OpenedFully:\_online..\_value");

}

void EP\_setVisibilityCB(string OpenedFully, bool isOpenedFully)

{

if (isOpenedFully == true)

{

this.visible = false;

} else if (getUserPermission(2))

{

this.visible = true;

}

else

{

this.visible = false;

}

}

[PUSH\_BUTTON-] [2] - [Clicked]

main(mapping event)

{

int value;

dpConnect($ControlledValve+".State.OpenedFully",

$ControlledValve+".State.ClosedFully");

dpGet($ControlledValve+".State.Position", value);

TF\_Position\_V3.text = value;

if (value > 0)

{

value = value - 20;

if (value < 0) value = 0;

dpSet($ControlledValve+".State.Position", value);

}

}

[PUSH\_BUTTON-] [2] - [Initialize]

main()

{

EP\_setVisibility();

EP\_SetVisible();

}

void EP\_setVisibility()

{

dpConnect("EP\_setVisibilityCB",

"System1:"+$ControlledValve+".State.ClosedFully:\_online..\_value");

}

void EP\_setVisibilityCB(string ClosedFully, bool isClosedFully)

{

if (isClosedFully == true)

{

this.visible = false;

} else if (getUserPermission(2))

{

this.visible = true;

}

else

{

this.visible = false;

}

}

[TEXT\_FIELD1] [3] - [Initialize]

main()

{

EP\_textFieldIn();

}

void EP\_textFieldIn()

{

dyn\_errClass err;

if( !dpExists( "System1:"+$ControlledValve+".State.Position:\_online..\_value"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_textFieldInCB",

"System1:"+$ControlledValve+".State.Position:\_online..\_value");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_textFieldInCB(string dp, float fNewValue)

{

setValue("", "text", dpValToString(dp, fNewValue, TRUE) );

}

[POLYGON1] [4] - [Initialize]

main()

{

EP\_setBackColor(); //

EP\_setChangeState(); //

EP\_RepairLogic(); //

EP\_AutoRejectLogic();

EP\_AutoClosedOpened();

}

void EP\_AutoRejectLogic()

{

dpConnect("EP\_AutoRejectCB",

"System1:"+$ControlledValve+".Alarm.Fail:\_online..\_value",

"System1:"+$ControlledValve+".State.Rejected:\_online..\_value");

}

void EP\_AutoRejectCB(string dpAlarmFail, bool isAlarmFail,

string dpRejected, bool isRejected)

{

if(isAlarmFail != isRejected)

{

dpSet("System1:"+$ControlledValve+".State.Rejected", isAlarmFail);

}

}

void EP\_setChangeState()

{

dpConnect("EP\_setChangeStateCB",

"System1:"+$ControlledValve+".CMD.Open:\_online..\_value",

"System1:"+$ControlledValve+".CMD.Close:\_online..\_value");

}

void EP\_setChangeStateCB(string dpOpen, bool cmdOpen, string dpClose, bool cmdClose)

{

float probFailure;

int result = dpGet("System1:"+$ControlledValve+".Settings.ProbabilityOfFailure", probFailure);

if(result != 0) return;

if(cmdOpen)

{

float random = (float)rand()/32767;

if(random <= probFailure) {

dpSet("System1:"+$ControlledValve+".Alarm.Fail", true);

} else {

dpSet(

"System1:"+$ControlledValve+".State.OpenedFully", false,

"System1:"+$ControlledValve+".State.ClosedFully", false

);

}

dpSet("System1:"+$ControlledValve+".CMD.Open", false);

}

if(cmdClose)

{

float random = (float)rand()/32767;

if(random <= probFailure) {

dpSet("System1:"+$ControlledValve+".Alarm.Fail", true);

} else {

dpSet(

"System1:"+$ControlledValve+".State.OpenedFully", false,

"System1:"+$ControlledValve+".State.ClosedFully", false

);

}

dpSet("System1:"+$ControlledValve+".CMD.Close", false);

}

}

void EP\_setBackColor()

{

dpConnect("EP\_setBackColorCB",

"System1:"+$ControlledValve+".State.OnRepair:\_online..\_value",

"System1:"+$ControlledValve+".State.Rejected:\_online..\_value",

"System1:"+$ControlledValve+".Alarm.Fail:\_online..\_value",

"System1:"+$ControlledValve+".State.Position:\_online..\_value");

}

void EP\_setBackColorCB(string dpOnRepair, bool isOnRepair,

string dpRejected, bool isRejected,

string dpAlarmFail, bool isAlarmFail,

string dpPosition, int position)

{

if(isOnRepair) {

setValue("", "backCol", "Green"); // Ремонт → зелёный

}

else if(isRejected || isAlarmFail) {

setValue("", "backCol", "Red"); // Авария → красный

}

else if(position == 0) {

setValue("", "backCol", "Grey"); // В простое → серый

}

else {

setValue("", "backCol", "White"); // Норма → белый

}

}

void EP\_RepairLogic()

{

dpConnect("EP\_RepairLogicCB",

"System1:"+$ControlledValve+".CMD.SendToRepair:\_online..\_value");

}

void EP\_RepairLogicCB(string dpRepair, bool cmdRepair)

{

if(cmdRepair)

{

dpSet("System1:"+$ControlledValve+".State.OnRepair", true);

delay(3);

dpSet(

"System1:"+$ControlledValve+".Alarm.Fail", false,

"System1:"+$ControlledValve+".State.Rejected", false,

"System1:"+$ControlledValve+".State.OnRepair", false,

"System1:"+$ControlledValve+".CMD.SendToRepair", false

);

}

}

void EP\_AutoClosedOpened()

{

dpConnect("EP\_AutoClosedOpenedCB",

"System1:"+$ControlledValve+".State.Position:\_online..\_value",

"System1:"+$ControlledValve+".State.OpenedFully:\_online..\_value",

"System1:"+$ControlledValve+".State.ClosedFully:\_online..\_value");

}

void EP\_AutoClosedOpenedCB(string dpPosition, int position,

string dpOpened, bool isOpened,

string dpClosed, bool isClosed)

{

if (position == 0)

{

dpSet(

"System1:"+$ControlledValve+".State.ClosedFully", true,

"System1:"+$ControlledValve+".State.OpenedFully", false

);

}

else if (position == 100)

{

dpSet(

"System1:"+$ControlledValve+".State.OpenedFully", true,

"System1:"+$ControlledValve+".State.ClosedFully", false

);

}

else

{

if (isOpened || isClosed)

{

dpSet(

"System1:"+$ControlledValve+".State.OpenedFully", false,

"System1:"+$ControlledValve+".State.ClosedFully", false

);

}

}

}

[FRAME1] [5] - [Initialize]

main()

{

EP\_setBackColor();

}

void EP\_setBackColor()

{

dpConnect("EP\_setBackColorCB",

"System1:"+$ControlledValve+".State.OnRepair:\_online..\_value",

"System1:"+$ControlledValve+".State.Rejected:\_online..\_value",

"System1:"+$ControlledValve+".Alarm.Fail:\_online..\_value");

}

void EP\_setBackColorCB(string dpOnRepair, bool isOnRepair,

string dpRejected, bool isRejected,

string dpAlarmFail, bool isAlarmFail)

{

if(isOnRepair) {

this.borderColor = "Green"; // Ремонт → зелёный

}

else if(isRejected || isAlarmFail) {

this.borderColor = "Red"; // Авария → красный

}

else {

this.borderColor = "White"; // Норма → белый

}

}

[Polyline1] [6] - [Initialize]

main()

{

EP\_setVisible();

}

void EP\_setVisible()

{

dyn\_errClass err;

if( !dpExists($ControlledValve+".State.Rejected"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_setVisibleCB",

$ControlledValve+".State.Rejected");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_setVisibleCB(string dp, bool boNewValue)

{

setValue("", "visible", boNewValue);

}

[Polyline2] [7] - [Initialize]

main()

{

EP\_setVisible();

}

void EP\_setVisible()

{

dyn\_errClass err;

if( !dpExists($ControlledValve+".State.Rejected"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_setVisibleCB",

$ControlledValve+".State.Rejected");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_setVisibleCB(string dp, bool boNewValue)

{

setValue("", "visible", boNewValue);

}

[Line1] [8] - [Initialize]

main()

{

EP\_setVisible();

}

void EP\_setVisible()

{

dyn\_errClass err;

if( !dpExists($ControlledValve+".State.Rejected"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_setVisibleCB",

$ControlledValve+".State.Rejected");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_setVisibleCB(string dp, bool boNewValue)

{

setValue("", "visible", boNewValue);

}

[Line2] [9] - [Initialize]

main()

{

EP\_setVisible();

}

void EP\_setVisible()

{

dyn\_errClass err;

if( !dpExists($ControlledValve+".State.Rejected"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_setVisibleCB",

$ControlledValve+".State.Rejected");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_setVisibleCB(string dp, bool boNewValue)

{

setValue("", "visible", boNewValue);

}

[PUSH\_BUTTON1] [10] - [Clicked]

main()

{

EP\_RepairButton();

}

void EP\_RepairButton()

{

if(!dpExists("System1:"+$ControlledValve+".CMD.SendToRepair:\_original..\_value") ||

!dpExists("System1:"+$ControlledValve+".State.OnRepair:\_original..\_value"))

{

return;

}

dpSetWait("System1:"+$ControlledValve+".CMD.SendToRepair:\_original..\_value", true);

dpSetWait("System1:"+$ControlledValve+".State.OnRepair:\_original..\_value", true);

dpConnect("EP\_RepairCompleteCB",

"System1:"+$ControlledValve+".CMD.SendToRepair:\_online..\_value");

}

void EP\_RepairCompleteCB(string dpRepair, bool cmdRepair)

{

if(cmdRepair)

{

delay(3);

dpSet(

"System1:"+$ControlledValve+".State.OnRepair", false,

"System1:"+$ControlledValve+".CMD.SendToRepair", false,

"System1:"+$ControlledValve+".State.Rejected", false,

"System1:"+$ControlledValve+".Alarm.Fail", false,

"System1:"+$ControlledValve+".State.OpenedFully", false,

"System1:"+$ControlledValve+".State.ClosedFully", true,

"System1:"+$ControlledValve+".State.Position", 0

);

}

}

[PUSH\_BUTTON1] [10] - [Initialize]

main()

{

EP\_setVisibility();

}

void EP\_setVisibility()

{

dpConnect("EP\_setVisibilityCB",

"System1:"+$ControlledValve+".State.Rejected:\_online..\_value",

"System1:"+$ControlledValve+".State.OnRepair:\_online..\_value");

}

void EP\_setVisibilityCB(string dpRejected, bool isRejected,

string dpRepair, bool isRepair)

{

if(isRejected == 1 || isRepair == 1)

{

this.visible = true;

this.borderColor = "Green";

}

else

{

this.visible = false;

}

}

[RECTANGLE1] [2] - [Initialize]

const float MM\_PER\_M3 = 3400.0 / 200.0; // 17 мм на 1 м³

const float UPDATE\_INTERVAL = 1.0; // Интервал обновления (сек)

main()

{

EP\_AutoReject();

dpConnect("updateSystem",

"System1:allPipe.State.Flow",

$ShuttOffValve+".State.Opened",

$TankE+".Settings.Capacity",

$TankE+".Settings.CapacityBoundary",

$TankE+".Settings.CapacityBoundaryLow",

$LiquidSensorL+".State.Rejected",

$ControledValve+".State.Position"

);

dpConnect("sensorControl",

$LiquidSensorL+".CMD.TurnOn:\_online..\_value",

$LiquidSensorL+".CMD.TurnOff:\_online..\_value",

$LiquidSensorL+".CMD.SendToRepair:\_online..\_value",

$LiquidSensorL+".Settings.ProbabilityOfFailure:\_online..\_value",

$LiquidSensorL+".State.TurnedOn:\_online..\_value"

);

}

void EP\_AutoReject()

{

dpConnect("EP\_AutoRejectCB",

"System1:"+$LiquidSensorL+".Alarm.FailTurnOn:\_online..\_value",

"System1:"+$LiquidSensorL+".Alarm.FailTurnOff:\_online..\_value"

);

}

void EP\_AutoRejectCB(string dpOnFail, bool isOnFail,

string dpOffFail, bool isOffFail)

{

bool shouldReject = isOnFail || isOffFail;

dpSet($LiquidSensorL+".State.Rejected", shouldReject);

DebugTN("Rejected установлен в", shouldReject ? "true" : "false");

}

void updateSystem(

string dpFlow, float flowM3H,

string dpValve, bool isValveOpened,

string dpCapacity, float volumeM3,

string dpUpperBound, float upperBoundMM,

string dpLowerBound, float lowerBoundMM,

string dpRejected, bool isRejected,

string dpCVPosition, int positionCV

)

{

float inflow = 0.0;

float outflow = 0.0;

if (isValveOpened && !isRejected) {

inflow = flowM3H / 3600.0 \* UPDATE\_INTERVAL;

}

if (positionCV > 0) {

float flowCV = flowM3H \* positionCV / 80.0;

outflow = -flowCV / 3600.0 \* UPDATE\_INTERVAL;

}

float newVolume = volumeM3 + inflow + outflow;

float levelMM = newVolume \* MM\_PER\_M3;

dpSet(

$TankE+".Settings.Capacity", newVolume,

$LiquidSensorL+".State.Value", levelMM

);

if (levelMM > lowerBoundMM && levelMM < upperBoundMM) {

dpSet(

$TankE+".Alarm.AmountFail:\_original..\_value", false,

$TankE+".Alarm.AmountFailLow:\_original..\_value", false

);

}

checkLevelAlarms(levelMM, upperBoundMM, lowerBoundMM);

}

void checkLevelAlarms(float levelMM, float upperBoundMM, float lowerBoundMM)

{

if(levelMM > upperBoundMM){

dpSet(

$TankE+".Alarm.AmountFail:\_original..\_value", 1

);

}

else if (levelMM < lowerBoundMM){

dpSet(

$TankE+".Alarm.AmountFailLow:\_original..\_value", 1

);

}

else {

dpSet($TankE+".Alarm.AmountFail:\_original..\_value", 0,

$TankE+".Alarm.AmountFailLow:\_original..\_value", 0);

}

}

void sensorControl(

string dpTurnOn, bool cmdTurnOn,

string dpTurnOff, bool cmdTurnOff,

string dpRepair, bool cmdRepair,

string dpProbFail, float probFailure,

string dpTurnedOn, bool isTurnedOn

)

{

if (cmdRepair) {

dpSet(

$LiquidSensorL+".State.OnRepair", true,

$LiquidSensorL+".CMD.SendToRepair", false

);

delay(3);

dpSet(

$LiquidSensorL+".Alarm.FailTurnOn", false,

$LiquidSensorL+".Alarm.FailTurnOff", false,

$LiquidSensorL+".State.Rejected", false,

$LiquidSensorL+".State.OnRepair", false

);

return;

}

if (cmdTurnOn) processTurnCommand(true, probFailure, isTurnedOn);

if (cmdTurnOff) processTurnCommand(false, probFailure, isTurnedOn);

}

void processTurnCommand(bool turnOn, float probFailure, bool isTurnedOn)

{

if ((turnOn && isTurnedOn) || (!turnOn && !isTurnedOn)) {

dpSet($LiquidSensorL+".CMD." + (turnOn ? "TurnOn" : "TurnOff"), false);

return;

}

float random = (float)rand() / 32767.0;

string cmdType = turnOn ? "TurnOn" : "TurnOff";

if (random <= probFailure) {

dpSet(

$LiquidSensorL+".Alarm.Fail" + cmdType, true,

$LiquidSensorL+".State.Rejected", true

);

DebugTN("Ошибка " + cmdType + "! Вероятность: " + (probFailure \* 100) + "%");

} else {

dpSet(

$LiquidSensorL+".State.Turned" + (turnOn ? "On" : "Off"), true,

$LiquidSensorL+".State.Turned" + (turnOn ? "Off" : "On"), false,

$LiquidSensorL+".Alarm.Fail" + cmdType, false,

$LiquidSensorL+".State.Rejected", false

);

}

dpSet($LiquidSensorL+".CMD." + cmdType, false);

}

[ToggleSwitch\_ewo1] [4] - [toggled]

toggled(bool on)

{

dpSet(

$LiquidSensorL+".CMD.TurnOff", !on,

$LiquidSensorL+".CMD.TurnOn", on

);

}

[ToggleSwitch\_ewo1] [4] - [Initialize]

main()

{

EP\_setBackColor();

EP\_SetVisible();

}

void EP\_setBackColor()

{

dpConnect("EP\_setBackColorCB",

$LiquidSensorL+".State.TurnedOff");

}

void EP\_setBackColorCB(string dpSource, bool boNewValue)

{

if (!boNewValue){

setValue("", "backCol", "green");

this.checked = true;

}

else {

setValue("", "backCol", "red");

this.checked = false;

}

}

void EP\_SetVisible()

{

if (getUserPermission(2)) {

this.visible = true;}

else {

this.visible = false;

}

}

[PUSH\_BUTTON1] [5] - [Clicked]

main()

{

EP\_RepairButton();

}

void EP\_RepairButton()

{

dpSetWait($LiquidSensorL+".CMD.SendToRepair:\_original..\_value", true);

dpSetWait($LiquidSensorL+".State.OnRepair:\_original..\_value", true);

dpConnect("EP\_RepairCompleteCB",

$LiquidSensorL+".CMD.SendToRepair:\_online..\_value");

}

void EP\_RepairCompleteCB(string dpRepair, bool cmdRepair)

{

if(cmdRepair)

{

delay(3);

dpSet(

$LiquidSensorL+".State.OnRepair", false,

$LiquidSensorL+".CMD.SendToRepair", false,

$LiquidSensorL+".State.Rejected", false,

$LiquidSensorL+".Alarm.FailTurnOn", false,

$LiquidSensorL+".Alarm.FailTurnOff", false,

$LiquidSensorL+".State.TurnedOff", true

);

DebugTN("Ремонт завершен");

}

}

[PUSH\_BUTTON1] [5] - [Initialize]

main()

{

EP\_setVisibility();

}

void EP\_setVisibility()

{

dpConnect("EP\_setVisibilityCB",

$LiquidSensorL+".State.Rejected:\_online..\_value",

$LiquidSensorL+".State.OnRepair:\_online..\_value");

}

void EP\_setVisibilityCB(string dpRejected, bool isRejected,

string dpRepair, bool isRepair)

{

if(isRejected == 1 || isRepair == 1)

{

this.visible = true; // Показываем кнопку

this.backCol = "Green"; // Зелёный цвет

}

else

{

this.visible = false; // Скрываем кнопку

}

}

[Polyline1] [6] - [Initialize]

main()

{

EP\_setVisible();

}

void EP\_setVisible()

{

dyn\_errClass err;

if( !dpExists($LiquidSensorL+".State.Rejected",

$TankE+".Alarm.AmountFail",

$TankE+".Alarm.AmountFailLow"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_setVisibleCB",

$LiquidSensorL+".State.Rejected",

$TankE+".Alarm.AmountFail",

$TankE+".Alarm.AmountFailLow");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_setVisibleCB(string dpRejected, bool isRejected,

string dpAmountFail, bool isAmountFail,

string dpAmountFailLow, bool isAmountFailLow)

{

bool visible = isRejected || isAmountFail || isAmountFailLow;

setValue("", "visible", visible);

}

[Polyline2] [7] - [Initialize]

main()

{

EP\_setVisible();

}

void EP\_setVisible()

{

dyn\_errClass err;

if( !dpExists($LiquidSensorL+".State.Rejected",

$TankE+".Alarm.AmountFail",

$TankE+".Alarm.AmountFailLow"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_setVisibleCB",

$LiquidSensorL+".State.Rejected",

$TankE+".Alarm.AmountFail",

$TankE+".Alarm.AmountFailLow");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_setVisibleCB(string dpRejected, bool isRejected,

string dpAmountFail, bool isAmountFail,

string dpAmountFailLow, bool isAmountFailLow)

{

bool visible = isRejected || isAmountFail || isAmountFailLow;

setValue("", "visible", visible);

}

[Line1] [8] - [Initialize]

main()

{

EP\_setVisible();

}

void EP\_setVisible()

{

dyn\_errClass err;

if( !dpExists($LiquidSensorL+".State.Rejected",

$TankE+".Alarm.AmountFail",

$TankE+".Alarm.AmountFailLow"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_setVisibleCB",

$LiquidSensorL+".State.Rejected",

$TankE+".Alarm.AmountFail",

$TankE+".Alarm.AmountFailLow");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_setVisibleCB(string dpRejected, bool isRejected,

string dpAmountFail, bool isAmountFail,

string dpAmountFailLow, bool isAmountFailLow)

{

bool visible = isRejected || isAmountFail || isAmountFailLow;

setValue("", "visible", visible);

}

[Line2] [9] - [Initialize]

main()

{

EP\_setVisible();

}

void EP\_setVisible()

{

dyn\_errClass err;

if( !dpExists($LiquidSensorL+".State.Rejected",

$TankE+".Alarm.AmountFail",

$TankE+".Alarm.AmountFailLow"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_setVisibleCB",

$LiquidSensorL+".State.Rejected",

$TankE+".Alarm.AmountFail",

$TankE+".Alarm.AmountFailLow");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_setVisibleCB(string dpRejected, bool isRejected,

string dpAmountFail, bool isAmountFail,

string dpAmountFailLow, bool isAmountFailLow)

{

bool visible = isRejected || isAmountFail || isAmountFailLow;

setValue("", "visible", visible);

}

[TEXT\_FIELD1] [10] - [Initialize]

main()

{

EP\_textFieldIn();

EP\_setVisible();

}

void EP\_textFieldIn()

{

dyn\_errClass err;

if( !dpExists( $LiquidSensorL+".State.Value:\_online..\_value"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_textFieldInCB",

$LiquidSensorL+".State.Value:\_online..\_value");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_textFieldInCB(string dp, int iNewValue)

{

setValue("", "text", dpValToString(dp, iNewValue, TRUE) );

}

void EP\_setVisible()

{

dyn\_errClass err;

if( !dpExists( $LiquidSensorL+".State.TurnedOff:\_online..\_value"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_setVisibleCB",

$LiquidSensorL+".State.TurnedOff:\_online..\_value");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_setVisibleCB(string dp, bool boNewValue)

{

setValue("", "visible", !boNewValue);

}

[PROGRESS\_BAR1] [11] - [Initialize]

main()

{

dpConnect("updateProgressBar",

$LiquidSensorL+".State.Value:\_online..\_value",

$LiquidSensorL+".State.TurnedOff:\_online..\_value",

$LiquidSensorL+".State.TurnedOn:\_online..\_value",

$LiquidSensorL+".State.Rejected:\_online..\_value",

$LiquidSensorL+".State.OnRepair:\_online..\_value");

}

void updateProgressBar(string dpValue, float value,

string dpTurnedOff, bool isTurnedOff,

string dpTurnedOn, bool isTurnedOn,

string dpRejected, bool isRejected,

string dpOnRepair, bool isOnRepair)

{

this.value = value;

if (isOnRepair) {

setProgressBarStyle("#008000"); // Зеленый

}

else if (isRejected) {

setProgressBarStyle("#e74c3c"); // Красный

}

else if (!isTurnedOn || isTurnedOff) {

setProgressBarStyle("#808080"); // Серый

}

else if (value >= 3000) {

setProgressBarStyle("#e74c3c"); // Красный

}

else if (value <= 200) {

setProgressBarStyle("#ffff00"); // Желтый

}

else {

setProgressBarStyle("#000000"); // Черный

}

}

void setProgressBarStyle(string color)

{

string style = "QProgressBar {"

+ "border: 2px solid #2c3e50;"

+ "border-radius: 5px;"

+ "background: #ecf0f1;"

+ "}"

+ "QProgressBar::chunk {"

+ "background: " + color + ";"

+ "border-radius: 3px;"

+ "margin: 1px;"

+ "}";

this.styleSheet = style;

}

[(Panel)] [0] - [Initialize]

main()

{

EP\_setChangeState();

EP\_RepairLogic();

EP\_AutoReject();

EP\_OnAlert();

}

void EP\_OnAlert()

{

dpConnect("EP\_OnAlertCB",

"C1.Settings.Value:\_original..\_value",

"C1.Alert.Underheat:\_original..\_value",

"C1.Alert.Overheat:\_original..\_value");

}

void EP\_OnAlertCB(string dpTemp, float temp,

string dpUAlert, bool isUAlert,

string dpOAlert, bool isOAlert)

{

if(temp < 64.6) {

dpSet(dpUAlert, true);

dpSet(dpOAlert, false);

}

else if(temp > 65.6) {

dpSet(dpOAlert, true);

dpSet(dpUAlert, false);

}

else {

dpSet(dpUAlert, false,

dpOAlert, false);

}

}

void EP\_AutoReject()

{

dpConnect("EP\_AutoRejectCB",

"System1:C1.Alert.Fail:\_online..\_value"

);

}

void EP\_AutoRejectCB(string dpFail, bool isFail)

{

bool shouldReject = isFail;

dpSet("System1:C1.State.Rejected", shouldReject);

DebugTN("Rejected установлен в", shouldReject ? "true" : "false");

}

void EP\_setChangeState()

{

dpConnect("EP\_setChangeStateCB",

"System1:C1.CMD.TurnOn:\_online..\_value",

"System1:C1.CMD.TurnOff:\_online..\_value");

}

void EP\_setChangeStateCB(string dpOpen, bool cmdOpen, string dpClose, bool cmdClose)

{

float probFailure;

int result = dpGet("System1:C1.Settings.ProbabilityOfFailure", probFailure);

if(result != 0) return;

if(cmdOpen)

{

float random = (float)rand()/32767;

DebugN("Проверка открытия. Случайное значение:", random, "Вероятность отказа:", probFailure);

if(random <= probFailure) {

dpSet("System1:C1.Alert.Fail", true);

} else {

dpSet("System1:C1.State.Opened", true,

"System1:C1.State.Closed", false);

}

dpSet("System1:C1.CMD.TurnOn", false);

}

if(cmdClose)

{

float random = (float)rand()/32767;

DebugN("Проверка закрытия. Случайное значение:", random, "Вероятность отказа:", probFailure);

if(random <= probFailure) {

dpSet("System1:C1.Alert.Fail", true);

} else {

dpSet("System1:C1.State.Opened", false,

"System1:C1.State.Closed", true);

}

dpSet("System1:C1.CMD.TurnOff", false);

}

}

void EP\_RepairLogic()

{

dpConnect("EP\_RepairLogicCB",

"System1:C1.CMD.SendToRepair:\_online..\_value");

}

void EP\_RepairLogicCB(string dpRepair, bool cmdRepair)

{

if(cmdRepair)

{

DebugTN("Начало ремонта...");

dpSet("System1:C1.State.OnRepair", true);

delay(3);

dpSet(

"System1:C1.Alert.Fail", false,

"System1:C1.State.Rejected", false,

"System1:C1.State.OnRepair", false,

"System1:C1.CMD.SendToRepair", false,

"System1:C1.CMD.TurnOff", true,

"System1:C1.CMD.TurnOn", false,

"System1:C1.State.Closed", true,

"System1:C1.State.Opened", false

);

DebugTN("Ремонт завершен. Все состояния сброшены.");

}

}

[LCD1] [1] - [Initialize]

main()

{

dpConnect("updateLCD", "C1.Settings.Value:\_original..\_value");

EP\_setVisible();

}

void updateLCD(string dpTemp, float valueTemp)

{

this.value = valueTemp;

string color;

if(valueTemp < 64.5) {

color = "yellow";

}

else if(valueTemp > 65.5) {

color = "red";

}

else {

color = "green";

}

this.foreCol = color;

}

void EP\_setVisible()

{

dyn\_errClass err;

if( !dpExists( "System1:C1.State.Closed:\_online..\_value"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_setVisibleCB",

"System1:C1.State.Closed:\_online..\_value");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_setVisibleCB(string dp, bool boNewValue)

{

setValue("", "visible", !boNewValue);

}

[LCD3] [2] - [Initialize]

main()

{

dpConnect("updateLCD", "C1.State.Power:\_original..\_value");

EP\_setVisible();

}

void updateLCD (string dpPower, int valuePower)

{

this.value = valuePower;

}

void EP\_setVisible()

{

dyn\_errClass err;

if( !dpExists( "System1:C1.State.Closed:\_online..\_value"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_setVisibleCB",

"System1:C1.State.Closed:\_online..\_value");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_setVisibleCB(string dp, bool boNewValue)

{

setValue("", "visible", !boNewValue);

}

[ToggleSwitch\_ewo3] [3] - [toggled]

toggled(bool on)

{

dpSet(

"System1:C1.CMD.TurnOff", !on,

"System1:C1.CMD.TurnOn", on

);

}

[ToggleSwitch\_ewo3] [3] - [Initialize]

main()

{

EP\_SetVisible();

EP\_SetOC();

}

void EP\_SetVisible()

{

if (getUserPermission(2)) {

this.visible = true;}

else {

this.visible = false;

}

}

void EP\_SetOC(string dpSource, bool boNewValue)

{

dpConnect("EP\_setOCCB",

"System1:C1.State.Closed");

}

void EP\_setOCCB()

{

if (!boNewValue){

this.checked = true;

}

else {

this.checked = false;

}

}

[SLIDER1] [4] - [Slide]

Slide(int value)

{

dpSet("C1.State.Power", value);

}

[SLIDER1] [4] - [Initialize]

main()

{

EP\_SetVisible();

dpConnect("onClosedChanged", "C1.State.Closed:\_original..\_value");

dpConnect("updateSlider", "C1.State.Power:\_original..\_value");

dpConnect("onPowerChange", "C1.State.Power", "C1.Settings.Value:\_original..\_value");

setTimer(1.0); // Обновление раз в секунду

}

void updateSlider(string dp, int value)

{

this.sliderPosition = value;

}

void onClosedChanged(string dpOff, bool isOff)

{

string style;

if(isOff) {

style = "QSlider::groove:horizontal {background: lightgray; height: 10px;}"

"QSlider::handle:horizontal {background: red; width: 20px;}";

dpSet("C1.State.Power", 0);

}

else {

style = "QSlider::groove:horizontal {background: lightgray; height: 10px;}"

"QSlider::handle:horizontal {background: green; width: 20px;}";

}

this.styleSheet(style);

}

void onPowerChange(string dpPower, int power, string dpTemp, float temp)

{

float change = 0;

if(power == 0) {

change = -0.001;

}

else if(temp >= 64.8) {

change = power \* 0.000005;

}

else {

change = power \* 0.0001;

}

temp += change;

if(temp < 0) temp = 0;

if(temp > 70) temp = 70;

dpSet(dpTemp, temp);

}

void timer()

{

int power = dpGet("C1.State.Power");

float temp = dpGet("C1.Settings.Value:\_original..\_value");

onPowerChange("C1.State.Power", power, "C1.Settings.Value", temp);

}

void EP\_SetVisible()

{

if (getUserPermission(2)) {

this.visible = true;}

else {

this.visible = false;

}

}

[LINE2] [5] - [Initialize]

main()

{

EP\_setVisible();

}

void EP\_setVisible()

{

dyn\_errClass err;

if( !dpExists("C1.State.Rejected",

"C1.Alert.Underheat",

"C1.Alert.Overheat"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_setVisibleCB",

"C1.State.Rejected",

"C1.Alert.Underheat",

"C1.Alert.Overheat");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_setVisibleCB(string dpRejected, bool isRejected,

string dpUnderheat, bool isUnderheat,

string dpOverheat, bool isOverheat)

{

bool visible = isRejected == 1 || isUnderheat == 1 || isOverheat == 1;

setValue("", "visible", visible);

}

[POLYGON5] [6] - [Initialize]

main()

{

EP\_setVisible();

}

void EP\_setVisible()

{

dyn\_errClass err;

if( !dpExists("C1.State.Rejected",

"C1.Alert.Underheat",

"C1.Alert.Overheat"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_setVisibleCB",

"C1.State.Rejected",

"C1.Alert.Underheat",

"C1.Alert.Overheat");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_setVisibleCB(string dpRejected, bool isRejected,

string dpUnderheat, bool isUnderheat,

string dpOverheat, bool isOverheat)

{

bool visible = isRejected == 1 || isUnderheat == 1 || isOverheat == 1;

setValue("", "visible", visible);

}

[POLYGON6] [7] - [Initialize]

main()

{

EP\_setVisible();

}

void EP\_setVisible()

{

dyn\_errClass err;

if( !dpExists("C1.State.Rejected",

"C1.Alert.Underheat",

"C1.Alert.Overheat"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_setVisibleCB",

"C1.State.Rejected",

"C1.Alert.Underheat",

"C1.Alert.Overheat");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_setVisibleCB(string dpRejected, bool isRejected,

string dpUnderheat, bool isUnderheat,

string dpOverheat, bool isOverheat)

{

bool visible = isRejected == 1 || isUnderheat == 1 || isOverheat == 1;

setValue("", "visible", visible);

}

[LINE5] [8] - [Initialize]

main()

{

EP\_setVisible();

}

void EP\_setVisible()

{

dyn\_errClass err;

if( !dpExists("C1.State.Rejected",

"C1.Alert.Underheat",

"C1.Alert.Overheat"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_setVisibleCB",

"C1.State.Rejected",

"C1.Alert.Underheat",

"C1.Alert.Overheat");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_setVisibleCB(string dpRejected, bool isRejected,

string dpUnderheat, bool isUnderheat,

string dpOverheat, bool isOverheat)

{

bool visible = isRejected == 1 || isUnderheat == 1 || isOverheat == 1;

setValue("", "visible", visible);

}

[LINE6] [9] - [Initialize]

main()

{

EP\_setVisible();

}

void EP\_setVisible()

{

dyn\_errClass err;

if( !dpExists("C1.State.Rejected",

"C1.Alert.Underheat",

"C1.Alert.Overheat"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_setVisibleCB",

"C1.State.Rejected",

"C1.Alert.Underheat",

"C1.Alert.Overheat");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_setVisibleCB(string dpRejected, bool isRejected,

string dpUnderheat, bool isUnderheat,

string dpOverheat, bool isOverheat)

{

bool visible = isRejected == 1 || isUnderheat == 1 || isOverheat == 1;

setValue("", "visible", visible);

}

[PUSH\_BUTTON2] [10] - [Clicked]

main()

{

EP\_RepairButton();

}

void EP\_RepairButton()

{

dpSetWait("C1.CMD.SendToRepair:\_original..\_value", true);

dpSetWait("C1.State.OnRepair:\_original..\_value", true);

dpConnect("EP\_RepairCompleteCB",

"C1.CMD.SendToRepair:\_online..\_value");

}

void EP\_RepairCompleteCB(string dpRepair, bool cmdRepair)

{

if(cmdRepair)

{

delay(3);

dpSet(

"System1:C1.Alert.Fail", false,

"System1:C1.State.Rejected", false,

"System1:C1.State.OnRepair", false,

"System1:C1.CMD.SendToRepair", false,

"System1:C1.CMD.TurnOff", true,

"System1:C1.CMD.TurnOn", false,

"System1:C1.State.Closed", true,

"System1:C1.State.Opened", false

);

DebugTN("Ремонт завершен");

}

}

[PUSH\_BUTTON2] [10] - [Initialize]

main()

{

EP\_setVisibility();

}

void EP\_setVisibility()

{

dpConnect("EP\_setVisibilityCB",

"C1.State.Rejected:\_online..\_value",

"C1.State.OnRepair:\_online..\_value");

}

void EP\_setVisibilityCB(string dpRejected, bool isRejected,

string dpRepair, bool isRepair)

{

if(isRejected == 1 || isRepair == 1)

{

this.visible = true;

this.backCol = "Green";

}

else

{

this.visible = false;

}

}

[PRIMITIVE\_TEXT1] [11] - [Initialize]

main()

{

EP\_setVisible();

}

void EP\_setVisible()

{

dyn\_errClass err;

if( !dpExists( "System1:C1.State.Closed:\_online..\_value"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_setVisibleCB",

"System1:C1.State.Closed:\_online..\_value");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_setVisibleCB(string dp, bool boNewValue)

{

setValue("", "visible", !boNewValue);

}

[PRIMITIVE\_TEXT2] [12] - [Initialize]

main()

{

EP\_setVisible();

}

void EP\_setVisible()

{

dyn\_errClass err;

if( !dpExists( "System1:C1.State.Closed:\_online..\_value"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_setVisibleCB",

"System1:C1.State.Closed:\_online..\_value");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_setVisibleCB(string dp, bool boNewValue)

{

setValue("", "visible", !boNewValue);

}

[ELLIPSE2] [1] - [Initialize]

const float MM\_PER\_M3 = 3400.0 / 200.0;

const float UPDATE\_INTERVAL = 1.0;

main()

{

EP\_AutoReject();

dpConnect("updateSystem",

"System1:allPipe.State.Flow",

"System1:V7.State.Opened",

"System1:E3W.Settings.Capacity",

"System1:E3W.Settings.CapacityBoundary",

"System1:E3W.Settings.CapacityBoundaryLow",

"System1:L3W.State.Rejected",

"System1:V3.State.Position",

"System1:V4.State.Position",

"System1:V5.State.Position"

);

dpConnect("sensorControl",

"L3W.CMD.TurnOn:\_online..\_value",

"L3W.CMD.TurnOff:\_online..\_value",

"L3W.CMD.SendToRepair:\_online..\_value",

"L3W.Settings.ProbabilityOfFailure:\_online..\_value",

"L3W.State.TurnedOn:\_online..\_value"

);

}

void EP\_AutoReject()

{

dpConnect("EP\_AutoRejectCB",

"System1:L3W.Alarm.FailTurnOn:\_online..\_value",

"System1:L3W.Alarm.FailTurnOff:\_online..\_value"

);

}

void EP\_AutoRejectCB(string dpOnFail, bool isOnFail,

string dpOffFail, bool isOffFail)

{

bool shouldReject = isOnFail || isOffFail;

dpSet("L3W.State.Rejected", shouldReject);

DebugTN("Rejected установлен в", shouldReject ? "true" : "false");

}

void updateSystem(

string dpFlow, float flowM3H,

string dpValve, bool isValveOpened,

string dpCapacity, float volumeM3,

string dpUpperBound, float upperBoundMM,

string dpLowerBound, float lowerBoundMM,

string dpRejected, bool isRejected,

string dpCV3Position, int positionCV3,

string dpCV4Position, int positionCV4,

string dpCV5Position, int positionCV5

)

{

float inflowCV3 = 0.0;

float inflowCV4 = 0.0;

float inflowCV5 = 0.0;

float outflow = 0.0;

if (positionCV3 > 0) {

float flowCV3 = flowM3H \* positionCV3 / 80.0;

inflowCV3 = flowCV3 / 3600.0 \* UPDATE\_INTERVAL;

}

if (positionCV4 > 0) {

float flowCV4 = flowM3H \* positionCV4 / 80.0;

inflowCV4 = flowCV4 / 3600.0 \* UPDATE\_INTERVAL;

}

if (positionCV5 > 0) {

float flowCV5 = flowM3H \* positionCV5 / 80.0;

inflowCV5 = flowCV5 / 3600.0 \* UPDATE\_INTERVAL;

}

if (isValveOpened && !isRejected) {

outflow = - flowM3H \* 3 / 3600.0 \* UPDATE\_INTERVAL;

}

float newVolume = volumeM3 + inflowCV3 + inflowCV4 + inflowCV5 + outflow;

float levelMM = newVolume \* MM\_PER\_M3;

dpSet(

"E3W.Settings.Capacity", newVolume,

"L3W.State.Value", levelMM

);

if (levelMM > lowerBoundMM && levelMM < upperBoundMM) {

dpSet(

"E3W.Alarm.AmountFail:\_original..\_value", false,

"E3W.Alarm.AmountFailLow:\_original..\_value", false

);

}

checkLevelAlarms(levelMM, upperBoundMM, lowerBoundMM);

}

void checkLevelAlarms(float levelMM, float upperBoundMM, float lowerBoundMM)

{

if(levelMM > upperBoundMM){

dpSet(

"E3W.Alarm.AmountFail:\_original..\_value", 1

);

}

else if (levelMM < lowerBoundMM){

dpSet(

"E3W.Alarm.AmountFailLow:\_original..\_value", 1

);

}

else {

dpSet("E3W.Alarm.AmountFail:\_original..\_value", 0,

"E3W.Alarm.AmountFailLow:\_original..\_value", 0);

}

}

void sensorControl(

string dpTurnOn, bool cmdTurnOn,

string dpTurnOff, bool cmdTurnOff,

string dpRepair, bool cmdRepair,

string dpProbFail, float probFailure,

string dpTurnedOn, bool isTurnedOn

)

{

if (cmdRepair) {

dpSet(

"L3W.State.OnRepair", true,

"L3W.CMD.SendToRepair", false

);

delay(3);

dpSet(

"L3W.Alarm.FailTurnOn", false,

"L3W.Alarm.FailTurnOff", false,

"L3W.State.Rejected", false,

"L3W.State.OnRepair", false

);

return;

}

if (cmdTurnOn) processTurnCommand(true, probFailure, isTurnedOn);

if (cmdTurnOff) processTurnCommand(false, probFailure, isTurnedOn);

}

void processTurnCommand(bool turnOn, float probFailure, bool isTurnedOn)

{

if ((turnOn && isTurnedOn) || (!turnOn && !isTurnedOn)) {

dpSet("L3W.CMD." + (turnOn ? "TurnOn" : "TurnOff"), false);

return;

}

float random = (float)rand() / 32767.0;

string cmdType = turnOn ? "TurnOn" : "TurnOff";

if (random <= probFailure) {

dpSet(

"L3W.Alarm.Fail" + cmdType, true,

"L3W.State.Rejected", true

);

DebugTN("Ошибка " + cmdType + "! Вероятность: " + (probFailure \* 100) + "%");

} else {

dpSet(

"L3W.State.Turned" + (turnOn ? "On" : "Off"), true,

"L3W.State.Turned" + (turnOn ? "Off" : "On"), false,

"L3W.Alarm.Fail" + cmdType, false,

"L3W.State.Rejected", false

);

}

dpSet("L3W.CMD." + cmdType, false);

}

[ELLIPSE1] [2] - [Initialize]

const float MM\_PER\_M3 = 3400.0 / 200.0;

const float UPDATE\_INTERVAL = 1.0;

main()

{

EP\_AutoReject();

dpConnect("updateSystem",

"System1:allPipe.State.Flow",

"System1:V6.State.Opened",

"System1:V7.State.Opened",

"System1:E3O.Settings.Capacity",

"System1:E3O.Settings.CapacityBoundary",

"System1:E3O.Settings.CapacityBoundaryLow",

"System1:L3O.State.Rejected",

"System1:V3.State.Position",

"System1:V4.State.Position",

"System1:V5.State.Position"

);

dpConnect("sensorControl",

"L3O.CMD.TurnOn:\_online..\_value",

"L3O.CMD.TurnOff:\_online..\_value",

"L3O.CMD.SendToRepair:\_online..\_value",

"L3O.Settings.ProbabilityOfFailure:\_online..\_value",

"L3O.State.TurnedOn:\_online..\_value"

);

}

void EP\_AutoReject()

{

dpConnect("EP\_AutoRejectCB",

"System1:L3O.Alarm.FailTurnOn:\_online..\_value",

"System1:L3O.Alarm.FailTurnOff:\_online..\_value"

);

}

void EP\_AutoRejectCB(string dpOnFail, bool isOnFail,

string dpOffFail, bool isOffFail)

{

bool shouldReject = isOnFail || isOffFail;

dpSet("L3O.State.Rejected", shouldReject);

DebugTN("Rejected установлен в", shouldReject ? "true" : "false");

}

void updateSystem(

string dpFlow, float flowM3H,

string dpV6, bool isV6Opened,

string dpV7, bool isV7Opened,

string dpCapacity, float volumeM3,

string dpUpperBound, float upperBoundMM,

string dpLowerBound, float lowerBoundMM,

string dpRejected, bool isRejected,

string dpCV3Position, int positionCV3,

string dpCV4Position, int positionCV4,

string dpCV5Position, int positionCV5

)

{

float inflowCV3 = 0.0;

float inflowCV4 = 0.0;

float inflowCV5 = 0.0;

float outflowV6 = 0.0;

float outflowV7 = 0.0;

if (positionCV3 > 0) {

float flowCV3 = flowM3H \* positionCV3 / 80.0;

inflowCV3 = flowCV3 / 3600.0 \* UPDATE\_INTERVAL;

}

if (positionCV4 > 0) {

float flowCV4 = flowM3H \* positionCV4 / 80.0;

inflowCV4 = flowCV4 / 3600.0 \* UPDATE\_INTERVAL;

}

if (positionCV5 > 0) {

float flowCV5 = flowM3H \* positionCV5 / 80.0;

inflowCV5 = flowCV5 / 3600.0 \* UPDATE\_INTERVAL;

}

if (isV6Opened && !isRejected) {

outflowV6 = - flowM3H \* 2 / 3600.0 \* UPDATE\_INTERVAL;

}

if (isV7Opened && !isRejected) {

outflowV7 = - flowM3H / 3600.0 \* UPDATE\_INTERVAL;

}

float newVolume = volumeM3 + inflowCV3 + inflowCV4 + inflowCV5 + outflowV6 + outflowV7;

float levelMM = newVolume \* MM\_PER\_M3;

dpSet(

"E3O.Settings.Capacity", newVolume,

"L3O.State.Value", levelMM

);

if (levelMM > lowerBoundMM && levelMM < upperBoundMM) {

dpSet(

"E3O.Alarm.AmountFail:\_original..\_value", false,

"E3O.Alarm.AmountFailLow:\_original..\_value", false

);

}

checkLevelAlarms(levelMM, upperBoundMM, lowerBoundMM);

}

void checkLevelAlarms(float levelMM, float upperBoundMM, float lowerBoundMM)

{

if(levelMM > upperBoundMM){

dpSet(

"E3O.Alarm.AmountFail:\_original..\_value", 1

);

}

else if (levelMM < lowerBoundMM){

dpSet(

"E3O.Alarm.AmountFailLow:\_original..\_value", 1

);

}

else {

dpSet("E3O.Alarm.AmountFail:\_original..\_value", 0,

"E3O.Alarm.AmountFailLow:\_original..\_value", 0);

}

}

void sensorControl(

string dpTurnOn, bool cmdTurnOn,

string dpTurnOff, bool cmdTurnOff,

string dpRepair, bool cmdRepair,

string dpProbFail, float probFailure,

string dpTurnedOn, bool isTurnedOn

)

{

if (cmdRepair) {

dpSet(

"L3O.State.OnRepair", true,

"L3O.CMD.SendToRepair", false

);

delay(3);

dpSet(

"L3O.Alarm.FailTurnOn", false,

"L3O.Alarm.FailTurnOff", false,

"L3O.State.Rejected", false,

"L3O.State.OnRepair", false

);

return;

}

if (cmdTurnOn) processTurnCommand(true, probFailure, isTurnedOn);

if (cmdTurnOff) processTurnCommand(false, probFailure, isTurnedOn);

}

void processTurnCommand(bool turnOn, float probFailure, bool isTurnedOn)

{

if ((turnOn && isTurnedOn) || (!turnOn && !isTurnedOn)) {

dpSet("L3O.CMD." + (turnOn ? "TurnOn" : "TurnOff"), false);

return;

}

float random = (float)rand() / 32767.0;

string cmdType = turnOn ? "TurnOn" : "TurnOff";

if (random <= probFailure) {

dpSet(

"L3O.Alarm.Fail" + cmdType, true,

"L3O.State.Rejected", true

);

DebugTN("Ошибка " + cmdType + "! Вероятность: " + (probFailure \* 100) + "%");

} else {

dpSet(

"L3O.State.Turned" + (turnOn ? "On" : "Off"), true,

"L3O.State.Turned" + (turnOn ? "Off" : "On"), false,

"L3O.Alarm.Fail" + cmdType, false,

"L3O.State.Rejected", false

);

}

dpSet("L3O.CMD." + cmdType, false);

}

[TEXT\_FIELD1] [4] - [Initialize]

main()

{

EP\_textFieldIn();

EP\_setVisible();

}

void EP\_textFieldIn()

{

dyn\_errClass err;

if( !dpExists( "L3W.State.Value:\_online..\_value"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_textFieldInCB",

"L3W.State.Value:\_online..\_value");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_textFieldInCB(string dp, int iNewValue)

{

setValue("", "text", dpValToString(dp, iNewValue, TRUE) );

}

void EP\_setVisible()

{

dyn\_errClass err;

if( !dpExists( "L3W.State.TurnedOff:\_online..\_value"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_setVisibleCB",

"L3W.State.TurnedOff:\_online..\_value");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_setVisibleCB(string dp, bool boNewValue)

{

setValue("", "visible", !boNewValue);

}

[TEXT\_FIELD2] [5] - [Initialize]

main()

{

EP\_textFieldIn();

EP\_setVisible();

}

void EP\_textFieldIn()

{

dyn\_errClass err;

if( !dpExists( "L3O.State.Value:\_online..\_value"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_textFieldInCB",

"L3O.State.Value:\_online..\_value");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_textFieldInCB(string dp, int iNewValue)

{

setValue("", "text", dpValToString(dp, iNewValue, TRUE) );

}

void EP\_setVisible()

{

dyn\_errClass err;

if( !dpExists( "L3O.State.TurnedOff:\_online..\_value"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_setVisibleCB",

"L3O.State.TurnedOff:\_online..\_value");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_setVisibleCB(string dp, bool boNewValue)

{

setValue("", "visible", !boNewValue);

}

[PROGRESS\_BAR1] [6] - [Initialize]

main()

{

dpConnect("updateProgressBar",

"L3W.State.Value:\_online..\_value",

"L3W.State.TurnedOff:\_online..\_value",

"L3W.State.TurnedOn:\_online..\_value",

"L3W.State.Rejected:\_online..\_value",

"L3W.State.OnRepair:\_online..\_value");

}

void updateProgressBar(string dpValue, float value,

string dpTurnedOff, bool isTurnedOff,

string dpTurnedOn, bool isTurnedOn,

string dpRejected, bool isRejected,

string dpOnRepair, bool isOnRepair)

{

this.value = value;

if (isOnRepair) {

setProgressBarStyle("#008000");

}

else if (isRejected) {

setProgressBarStyle("#e74c3c");

}

else if (!isTurnedOn || isTurnedOff) {

setProgressBarStyle("#808080");

}

else if (value >= 1000) {

setProgressBarStyle("#e74c3c");

}

else if (value <= 200) {

setProgressBarStyle("#ffff00");

}

else {

setProgressBarStyle("#0000FF");

}

}

void setProgressBarStyle(string color)

{

string style = "QProgressBar {"

+ "border: 2px solid #2c3e50;"

+ "border-radius: 5px;"

+ "background: #ecf0f1;"

+ "}"

+ "QProgressBar::chunk {"

+ "background: " + color + ";"

+ "border-radius: 3px;"

+ "margin: 1px;"

+ "}";

this.styleSheet = style;

}

[PROGRESS\_BAR2] [7] - [Initialize]

main()

{

dpConnect("updateProgressBar",

"L3O.State.Value:\_online..\_value",

"L3O.State.TurnedOff:\_online..\_value",

"L3O.State.TurnedOn:\_online..\_value",

"L3O.State.Rejected:\_online..\_value",

"L3O.State.OnRepair:\_online..\_value");

}

void updateProgressBar(string dpValue, float value,

string dpTurnedOff, bool isTurnedOff,

string dpTurnedOn, bool isTurnedOn,

string dpRejected, bool isRejected,

string dpOnRepair, bool isOnRepair)

{

this.value = value;

if (isOnRepair) {

setProgressBarStyle("#008000"); // Зеленый

}

else if (isRejected) {

setProgressBarStyle("#e74c3c"); // Красный

}

else if (!isTurnedOn || isTurnedOff) {

setProgressBarStyle("#808080"); // Серый

}

else if (value >= 3000) {

setProgressBarStyle("#e74c3c"); // Красный

}

else if (value <= 1000) {

setProgressBarStyle("#ffff00"); // Желтый

}

else {

setProgressBarStyle("#000000"); // Черный

}

}

void setProgressBarStyle(string color)

{

string style = "QProgressBar {"

+ "border: 2px solid #2c3e50;"

+ "border-radius: 5px;"

+ "background: #ecf0f1;"

+ "}"

+ "QProgressBar::chunk {"

+ "background: " + color + ";"

+ "border-radius: 3px;"

+ "margin: 1px;"

+ "}";

this.styleSheet = style;

}

[ToggleSwitch\_ewo1] [8] - [toggled]

toggled(bool on)

{

dpSet(

"L3W.CMD.TurnOff", !on,

"L3W.CMD.TurnOn", on

);

}

[ToggleSwitch\_ewo1] [8] - [Initialize]

main()

{

EP\_setBackColor();

EP\_SetVisible();

}

void EP\_setBackColor()

{

dpConnect("EP\_setBackColorCB",

"L3W.State.TurnedOff");

}

void EP\_setBackColorCB(string dpSource, bool boNewValue)

{

if (!boNewValue){

setValue("", "backCol", "green");

this.checked = true;

}

else {

setValue("", "backCol", "red");

this.checked = false;

}

}

void EP\_SetVisible()

{

if (getUserPermission(2)) {

this.visible = true;}

else {

this.visible = false;

}

}

[ToggleSwitch\_ewo4] [9] - [toggled]

toggled(bool on)

{

dpSet(

"L3O.CMD.TurnOff", !on,

"L3O.CMD.TurnOn", on

);

}

[ToggleSwitch\_ewo4] [9] - [Initialize]

main()

{

EP\_setBackColor();

EP\_SetVisible();

}

void EP\_setBackColor()

{

dpConnect("EP\_setBackColorCB",

"L3O.State.TurnedOff");

}

void EP\_setBackColorCB(string dpSource, bool boNewValue)

{

if (!boNewValue){

setValue("", "backCol", "green");

this.checked = true;

}

else {

setValue("", "backCol", "red");

this.checked = false;

}

}

void EP\_SetVisible()

{

if (getUserPermission(2)) {

this.visible = true;}

else {

this.visible = false;

}

}

[Polyline1] [10] - [Initialize]

main()

{

EP\_setVisible();

}

void EP\_setVisible()

{

dyn\_errClass err;

if( !dpExists("L3W.State.Rejected",

"E3W.Alarm.AmountFail",

"E3W.Alarm.AmountFailLow",

"L3O.State.Rejected",

"E3O.Alarm.AmountFail",

"E3O.Alarm.AmountFailLow"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_setVisibleCB",

"L3W.State.Rejected",

"E3W.Alarm.AmountFail",

"E3W.Alarm.AmountFailLow",

"L3O.State.Rejected",

"E3O.Alarm.AmountFail",

"E3O.Alarm.AmountFailLow");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_setVisibleCB(string dpWRejected, bool isWRejected,

string dpWAmountFail, bool isWAmountFail,

string dpWAmountFailLow, bool isWAmountFailLow,

string dpORejected, bool isORejected,

string dpOAmountFail, bool isOAmountFail,

string dpOAmountFailLow, bool isOAmountFailLow)

{

bool visible = isWRejected || isWAmountFail || isWAmountFailLow || isORejected || isOAmountFail || isOAmountFailLow;

setValue("", "visible", visible);

}

[Polyline2] [11] - [Initialize]

main()

{

EP\_setVisible();

}

void EP\_setVisible()

{

dyn\_errClass err;

if( !dpExists("L3W.State.Rejected",

"E3W.Alarm.AmountFail",

"E3W.Alarm.AmountFailLow",

"L3O.State.Rejected",

"E3O.Alarm.AmountFail",

"E3O.Alarm.AmountFailLow"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_setVisibleCB",

"L3W.State.Rejected",

"E3W.Alarm.AmountFail",

"E3W.Alarm.AmountFailLow",

"L3O.State.Rejected",

"E3O.Alarm.AmountFail",

"E3O.Alarm.AmountFailLow");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_setVisibleCB(string dpWRejected, bool isWRejected,

string dpWAmountFail, bool isWAmountFail,

string dpWAmountFailLow, bool isWAmountFailLow,

string dpORejected, bool isORejected,

string dpOAmountFail, bool isOAmountFail,

string dpOAmountFailLow, bool isOAmountFailLow)

{

bool visible = isWRejected || isWAmountFail || isWAmountFailLow || isORejected || isOAmountFail || isOAmountFailLow;

setValue("", "visible", visible);

}

[Line1] [12] - [Initialize]

main()

{

EP\_setVisible();

}

void EP\_setVisible()

{

dyn\_errClass err;

if( !dpExists("L3W.State.Rejected",

"E3W.Alarm.AmountFail",

"E3W.Alarm.AmountFailLow",

"L3O.State.Rejected",

"E3O.Alarm.AmountFail",

"E3O.Alarm.AmountFailLow"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_setVisibleCB",

"L3W.State.Rejected",

"E3W.Alarm.AmountFail",

"E3W.Alarm.AmountFailLow",

"L3O.State.Rejected",

"E3O.Alarm.AmountFail",

"E3O.Alarm.AmountFailLow");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_setVisibleCB(string dpWRejected, bool isWRejected,

string dpWAmountFail, bool isWAmountFail,

string dpWAmountFailLow, bool isWAmountFailLow,

string dpORejected, bool isORejected,

string dpOAmountFail, bool isOAmountFail,

string dpOAmountFailLow, bool isOAmountFailLow)

{

bool visible = isWRejected || isWAmountFail || isWAmountFailLow || isORejected || isOAmountFail || isOAmountFailLow;

setValue("", "visible", visible);

}

[Line2] [13] - [Initialize]

main()

{

EP\_setVisible();

}

void EP\_setVisible()

{

dyn\_errClass err;

if( !dpExists("L3W.State.Rejected",

"E3W.Alarm.AmountFail",

"E3W.Alarm.AmountFailLow",

"L3O.State.Rejected",

"E3O.Alarm.AmountFail",

"E3O.Alarm.AmountFailLow"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_setVisibleCB",

"L3W.State.Rejected",

"E3W.Alarm.AmountFail",

"E3W.Alarm.AmountFailLow",

"L3O.State.Rejected",

"E3O.Alarm.AmountFail",

"E3O.Alarm.AmountFailLow");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_setVisibleCB(string dpWRejected, bool isWRejected,

string dpWAmountFail, bool isWAmountFail,

string dpWAmountFailLow, bool isWAmountFailLow,

string dpORejected, bool isORejected,

string dpOAmountFail, bool isOAmountFail,

string dpOAmountFailLow, bool isOAmountFailLow)

{

bool visible = isWRejected || isWAmountFail || isWAmountFailLow || isORejected || isOAmountFail || isOAmountFailLow;

setValue("", "visible", visible);

}

[PUSH\_BUTTON1] [14] - [Clicked]

main()

{

EP\_RepairButton();

}

void EP\_RepairButton()

{

dpSetWait("L3W.CMD.SendToRepair:\_original..\_value", true);

dpSetWait("L3W.State.OnRepair:\_original..\_value", true);

dpConnect("EP\_RepairCompleteCB",

"L3W.CMD.SendToRepair:\_online..\_value");

}

void EP\_RepairCompleteCB(string dpRepair, bool cmdRepair)

{

if(cmdRepair)

{

delay(3);

dpSet(

"L3W.State.OnRepair", false,

"L3W.CMD.SendToRepair", false,

"L3W.State.Rejected", false,

"L3W.Alarm.FailTurnOn", false,

"L3W.Alarm.FailTurnOff", false

);

DebugTN("Ремонт завершен");

}

}

[PUSH\_BUTTON1] [14] - [Initialize]

main()

{

EP\_setVisibility();

}

void EP\_setVisibility()

{

dpConnect("EP\_setVisibilityCB",

"L3W.State.Rejected:\_online..\_value",

"L3W.State.OnRepair:\_online..\_value");

}

void EP\_setVisibilityCB(string dpRejected, bool isRejected,

string dpRepair, bool isRepair)

{

if(isRejected == 1 || isRepair == 1)

{

this.visible = true;

this.backCol = "Green";

}

else

{

this.visible = false;

}

}

[PUSH\_BUTTON2] [15] - [Clicked]

main()

{

EP\_RepairButton();

}

void EP\_RepairButton()

{

dpSetWait("L3O.CMD.SendToRepair:\_original..\_value", true);

dpSetWait("L3O.State.OnRepair:\_original..\_value", true);

dpConnect("EP\_RepairCompleteCB",

"L3O.CMD.SendToRepair:\_online..\_value");

}

void EP\_RepairCompleteCB(string dpRepair, bool cmdRepair)

{

if(cmdRepair)

{

delay(3);

dpSet(

"L3O.State.OnRepair", false,

"L3O.CMD.SendToRepair", false,

"L3O.State.Rejected", false,

"L3O.Alarm.FailTurnOn", false,

"L3O.Alarm.FailTurnOff", false

);

DebugTN("Ремонт завершен");

}

}

[PUSH\_BUTTON2] [15] - [Initialize]

main()

{

EP\_setVisibility();

}

void EP\_setVisibility()

{

dpConnect("EP\_setVisibilityCB",

"L3O.State.Rejected:\_online..\_value",

"L3O.State.OnRepair:\_online..\_value");

}

void EP\_setVisibilityCB(string dpRejected, bool isRejected,

string dpRepair, bool isRepair)

{

if(isRejected == 1 || isRepair == 1)

{

this.visible = true;

this.backCol = "Green";

}

else

{

this.visible = false;

}

}

[Применить] [10] - [Clicked]

main(mapping event)

{

PanelOff();

}

[PUSH\_BUTTON+] [11] - [Clicked]

main(mapping event)

{

int value;

dpGet("V3.Settings.MaxP", value);

TF\_Position\_V3.text = value;

if (value < 90)

{

value = value + 20;

if (value > 100) value = 100;

dpSet("V3.Settings.MaxP", value);

}

}

[PUSH\_BUTTON-] [12] - [Clicked]

main(mapping event)

{

int value;

dpGet("V3.Settings.MaxP", value);

TF\_Position\_V3.text = value;

if (value > 0)

{

value = value - 20;

if (value < 0) value = 0;

dpSet("V3.Settings.MaxP", value);

}

}

[TEXT\_FIELD1] [13] - [Initialize]

main()

{

EP\_textFieldIn();

}

void EP\_textFieldIn()

{

dyn\_errClass err;

if( !dpExists( "System1:V3.Settings.MaxP:\_online..\_value"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_textFieldInCB",

"System1:V3.Settings.MaxP:\_online..\_value");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_textFieldInCB(string dp, float fNewValue)

{

setValue("", "text", dpValToString(dp, fNewValue, TRUE) );

}

[PUSH\_BUTTON3] [14] - [Clicked]

main(mapping event)

{

int value;

dpGet("V4.Settings.MaxP", value);

TF\_Position\_V4.text = value;

if (value < 90)

{

value = value + 20;

if (value > 100) value = 100;

dpSet("V4.Settings.MaxP", value);

}

}

[PUSH\_BUTTON4] [15] - [Clicked]

main(mapping event)

{

int value;

dpGet("V4.Settings.MaxP", value);

TF\_Position\_V4.text = value;

if (value > 0)

{

value = value - 20;

if (value < 0) value = 0;

dpSet("V4.Settings.MaxP", value);

}

}

[TEXT\_FIELD3] [16] - [Initialize]

main()

{

EP\_textFieldIn();

}

void EP\_textFieldIn()

{

dyn\_errClass err;

if( !dpExists( "System1:V4.Settings.MaxP:\_online..\_value"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_textFieldInCB",

"System1:V4.Settings.MaxP:\_online..\_value");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_textFieldInCB(string dp, float fNewValue)

{

setValue("", "text", dpValToString(dp, fNewValue, TRUE) );

}

[PUSH\_BUTTON5] [17] - [Clicked]

main(mapping event)

{

int value;

dpGet("V5.Settings.MaxP", value);

TF\_Position\_V5.text = value;

if (value < 90)

{

value = value + 20;

if (value > 100) value = 100;

dpSet("V5.Settings.MaxP", value);

}

}

[PUSH\_BUTTON6] [18] - [Clicked]

main(mapping event)

{

int value;

dpGet("V5.Settings.MaxP", value);

TF\_Position\_V5.text = value;

if (value > 0)

{

value = value - 20;

if (value < 0) value = 0;

dpSet("V5.Settings.MaxP", value);

}

}

[TEXT\_FIELD4] [19] - [Initialize]

main()

{

EP\_textFieldIn();

}

void EP\_textFieldIn()

{

dyn\_errClass err;

if( !dpExists( "System1:V5.Settings.MaxP:\_online..\_value"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_textFieldInCB",

"System1:V5.Settings.MaxP:\_online..\_value");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_textFieldInCB(string dp, float fNewValue)

{

setValue("", "text", dpValToString(dp, fNewValue, TRUE) );

}

[PUSH\_BUTTON7] [21] - [Clicked]

main(mapping event)

{

float value;

dpGet("V0.Settings.ProbabilityOfFailure", value);

TF\_ProbabilityOfFailure\_V0.text = value;

if (value < 0.99)

{

value = value + 0.01;

if (value > 1.00) value = 1.00;

dpSet("V0.Settings.ProbabilityOfFailure", value,

"V1.Settings.ProbabilityOfFailure", value,

"V2.Settings.ProbabilityOfFailure", value,

"V6.Settings.ProbabilityOfFailure", value,

"V7.Settings.ProbabilityOfFailure", value);

}

}

[PUSH\_BUTTON8] [22] - [Clicked]

main(mapping event)

{

float value;

dpGet("V0.Settings.ProbabilityOfFailure", value);

TF\_ProbabilityOfFailure\_V0.text = value;

if (value > 0.00)

{

value = value - 0.01;

if (value < 0.00) value = 0.00;

dpSet("V0.Settings.ProbabilityOfFailure", value,

"V1.Settings.ProbabilityOfFailure", value,

"V2.Settings.ProbabilityOfFailure", value,

"V6.Settings.ProbabilityOfFailure", value,

"V7.Settings.ProbabilityOfFailure", value);

}

}

[TEXT\_FIELD5] [23] - [Initialize]

main()

{

EP\_textFieldIn();

}

void EP\_textFieldIn()

{

dyn\_errClass err;

if( !dpExists( "System1:V0.Settings.ProbabilityOfFailure:\_online..\_value"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_textFieldInCB",

"System1:V0.Settings.ProbabilityOfFailure:\_online..\_value");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_textFieldInCB(string dp, float fNewValue)

{

setValue("", "text", dpValToString(dp, fNewValue, TRUE) );

}

[PUSH\_BUTTON9] [24] - [Clicked]

main(mapping event)

{

float value;

dpGet("V3.Settings.ProbabilityOfFailure", value);

TF\_ProbabilityOfFailure\_V0.text = value;

if (value < 0.99)

{

value = value + 0.01;

if (value > 1.00) value = 1.00;

dpSet("V3.Settings.ProbabilityOfFailure", value,

"V4.Settings.ProbabilityOfFailure", value,

"V5.Settings.ProbabilityOfFailure", value);

}

}

[PUSH\_BUTTON10] [25] - [Clicked]

main(mapping event)

{

float value;

dpGet("V3.Settings.ProbabilityOfFailure", value);

TF\_ProbabilityOfFailure\_V0.text = value;

if (value > 0.00)

{

value = value - 0.01;

if (value < 0.00) value = 0.00;

dpSet("V3.Settings.ProbabilityOfFailure", value,

"V4.Settings.ProbabilityOfFailure", value,

"V5.Settings.ProbabilityOfFailure", value);

}

}

[TEXT\_FIELD6] [26] - [Initialize]

main()

{

EP\_textFieldIn();

}

void EP\_textFieldIn()

{

dyn\_errClass err;

if( !dpExists( "System1:V3.Settings.ProbabilityOfFailure:\_online..\_value"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_textFieldInCB",

"System1:V3.Settings.ProbabilityOfFailure:\_online..\_value");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_textFieldInCB(string dp, float fNewValue)

{

setValue("", "text", dpValToString(dp, fNewValue, TRUE) );

}

[PUSH\_BUTTON11] [27] - [Clicked]

main(mapping event)

{

float value;

dpGet("L0.Settings.ProbabilityOfFailure", value);

TF\_ProbabilityOfFailure\_V0.text = value;

if (value < 0.99)

{

value = value + 0.01;

if (value > 1.00) value = 1.00;

dpSet("L0.Settings.ProbabilityOfFailure", value,

"L1.Settings.ProbabilityOfFailure", value,

"L2.Settings.ProbabilityOfFailure", value,

"L3O.Settings.ProbabilityOfFailure", value,

"L3W.Settings.ProbabilityOfFailure", value);

}

}

[PUSH\_BUTTON12] [28] - [Clicked]

main(mapping event)

{

float value;

dpGet("L0.Settings.ProbabilityOfFailure", value);

TF\_ProbabilityOfFailure\_V0.text = value;

if (value > 0.00)

{

value = value - 0.01;

if (value < 0.00) value = 0.00;

dpSet("L0.Settings.ProbabilityOfFailure", value,

"L1.Settings.ProbabilityOfFailure", value,

"L2.Settings.ProbabilityOfFailure", value,

"L3O.Settings.ProbabilityOfFailure", value,

"L3W.Settings.ProbabilityOfFailure", value);

}

}

[TEXT\_FIELD7] [29] - [Initialize]

main()

{

EP\_textFieldIn();

}

void EP\_textFieldIn()

{

dyn\_errClass err;

if( !dpExists( "System1:L0.Settings.ProbabilityOfFailure:\_online..\_value"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_textFieldInCB",

"System1:L0.Settings.ProbabilityOfFailure:\_online..\_value");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_textFieldInCB(string dp, float fNewValue)

{

setValue("", "text", dpValToString(dp, fNewValue, TRUE) );

}

[PUSH\_BUTTON13] [30] - [Clicked]

main(mapping event)

{

float value;

dpGet("C1.Settings.TProbabilityOfFailure", value);

TF\_ProbabilityOfFailure\_V0.text = value;

if (value < 0.99)

{

value = value + 0.01;

if (value > 1.00) value = 1.00;

dpSet("C1.Settings.TProbabilityOfFailure", value);

}

}

[PUSH\_BUTTON14] [31] - [Clicked]

main(mapping event)

{

float value;

dpGet("C1.Settings.TProbabilityOfFailure", value);

TF\_ProbabilityOfFailure\_V0.text = value;

if (value > 0.00)

{

value = value - 0.01;

if (value < 0.00) value = 0.00;

dpSet("C1.Settings.TProbabilityOfFailure", value);

}

}

[TEXT\_FIELD8] [32] - [Initialize]

main()

{

EP\_textFieldIn();

}

void EP\_textFieldIn()

{

dyn\_errClass err;

if( !dpExists( "System1:C1.Settings.TProbabilityOfFailure:\_online..\_value"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_textFieldInCB",

"System1:C1.Settings.TProbabilityOfFailure:\_online..\_value");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_textFieldInCB(string dp, float fNewValue)

{

setValue("", "text", dpValToString(dp, fNewValue, TRUE) );

}

[PUSH\_BUTTON15] [33] - [Clicked]

main(mapping event)

{

float value;

dpGet("C1.Settings.ProbabilityOfFailure", value);

TF\_ProbabilityOfFailure\_V0.text = value;

if (value < 0.99)

{

value = value + 0.01;

if (value > 1.00) value = 1.00;

dpSet("C1.Settings.ProbabilityOfFailure", value);

}

}

[PUSH\_BUTTON16] [34] - [Clicked]

main(mapping event)

{

float value;

dpGet("C1.Settings.ProbabilityOfFailure", value);

TF\_ProbabilityOfFailure\_V0.text = value;

if (value > 0.00)

{

value = value - 0.01;

if (value < 0.00) value = 0.00;

dpSet("C1.Settings.ProbabilityOfFailure", value);

}

}

[TEXT\_FIELD9] [35] - [Initialize]

main()

{

EP\_textFieldIn();

}

void EP\_textFieldIn()

{

dyn\_errClass err;

if( !dpExists( "System1:C1.Settings.ProbabilityOfFailure:\_online..\_value"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_textFieldInCB",

"System1:C1.Settings.ProbabilityOfFailure:\_online..\_value");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_textFieldInCB(string dp, float fNewValue)

{

setValue("", "text", dpValToString(dp, fNewValue, TRUE) );

}

[SLIDERKWT] [36] - [Slide]

Slide(int value)

{

dpSet("C1.State.Power", value);

}

[SLIDERKWT] [36] - [Initialize]

main()

{

dpConnect("onClosedChanged", "C1.State.Closed:\_original..\_value");

dpConnect("updateSlider", "C1.State.Power:\_original..\_value");

initSliderStyle();

}

void initSliderStyle()

{

string sliderStyle =

"QSlider {"

" min-width: 100px;"

"}"

"QSlider::groove:horizontal {"

" background: black;"

" height: 2px;"

"}"

"QSlider::handle:horizontal {"

" background: black;"

" width: 12px;"

" height: 12px;"

" margin: -5px 0;"

" border-radius: 6px;"

"}";

this.styleSheet = sliderStyle;

}

void updateSlider(string dp, int value)

{

this.sliderPosition = value;

}

void onClosedChanged(string dpOff, bool isOff)

{

if(isOff) {

dpSet("C1.State.Power", 0);

}

}

[SLIDERM3H] [37] - [Slide]

Slide(int value)

{

dpSet("allPipe.State.Flow", value);

}

[SLIDERM3H] [37] - [Initialize]

main()

{

dpConnect("updateSlider", "allPipe.State.Flow:\_original..\_value");

initSliderStyle();

}

void initSliderStyle()

{

string sliderStyle =

"QSlider::groove:horizontal {"

" background: black;"

" height: 2px;"

"}"

"QSlider::handle:horizontal {"

" background: black;"

" width: 12px;"

" height: 12px;"

" margin: -5px 0;"

" border-radius: 6px;"

"}";

this.styleSheet = sliderStyle;

}

void updateSlider(string dp, int value)

{

if(value == 0){

this.sliderPosition = 0;

}

else {

this.sliderPosition = value;

}

}

[TEXT\_FIELD10] [41] - [Initialize]

main()

{

EP\_textFieldIn();

}

void EP\_textFieldIn()

{

dyn\_errClass err;

if( !dpExists( "System1:C1.State.Power:\_online..\_value"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_textFieldInCB",

"System1:C1.State.Power:\_online..\_value");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_textFieldInCB(string dp, int iNewValue)

{

setValue("", "text", dpValToString(dp, iNewValue, TRUE) );

}

[TEXT\_FIELD11] [42] - [Initialize]

main()

{

EP\_textFieldIn();

}

void EP\_textFieldIn()

{

dyn\_errClass err;

if( !dpExists( "System1:allPipe.State.Flow:\_online..\_value"))

{

setValue("", "color", "\_dpdoesnotexist");

return;

}

dpConnect("EP\_textFieldInCB",

"System1:allPipe.State.Flow:\_online..\_value");

err = getLastError();

if (dynlen(err) > 0)

setValue("", "color", "\_dpdoesnotexist");

}

void EP\_textFieldInCB(string dp, float fNewValue)

{

setValue("", "text", dpValToString(dp, fNewValue, TRUE) );

}