

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ  
федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий  
(наименование института полностью)

Кафедра «Прикладная математика и информатика»  
(наименование кафедры)

09.03.03 Прикладная информатика  
(код и наименование направления подготовки, специальности)

Бизнес-информатика  
(направленность (профиль)/специализация)

## БАКАЛАВРСКАЯ РАБОТА

на тему «Разработка информационной системы сбора и визуализации данных  
(на примере ООО Уралбумага)»

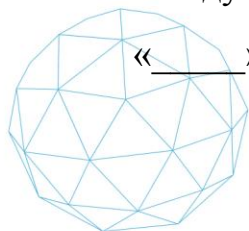
Студент А.С. Круподеров (И.О. Фамилия) \_\_\_\_\_ (личная подпись)

Руководитель Н.Н. Рогова (И.О. Фамилия) \_\_\_\_\_ (личная подпись)

**Допустить к защите**

Заведующий кафедрой к.т.н., доцент, А.В. Очеповский \_\_\_\_\_ (личная подпись)  
(ученая степень, звание, И.О. Фамилия)

« \_\_\_\_\_ » \_\_\_\_\_ 20 \_\_\_\_\_ г.



Тольятти 2019



**Росдистант**  
ВЫСШЕЕ ОБРАЗОВАНИЕ ДИСТАНЦИОННО

## АННОТАЦИЯ

Выпускная квалификационная работа посвящена теме «Разработка информационной системы сбора и визуализации данных (на примере ООО Уралбумага)».

Работа включает: 80 страниц, 5 таблиц, 32 рисунка, 5 приложений, количество библиографических источников – 28 (15 российских и 13 иностранных).

Выпускная квалификационная работа посвящена исследованию работы упаковочно-транспортной системы цеха по производству гофротары ООО «Уралбумага» группы предприятий «ПЦБК», а также ее объединению с системой верхнего уровня с целью получить статистические данные, необходимые для учета производительности и исключительных ситуаций.

Производство гофротары представляет собой совокупность производственных линий по выпуску картонных ящиков. Характеристики и производительность каждой производственной линии индивидуальны, при этом продукция со всех производственных линий поступает на общую систему упаковки через транспортную систему. Упаковочно-транспортная система является общим ресурсом для производства и эффективное использования ее ресурса во многом определяет объем выпуска готовой продукции.

Упаковочно-транспортная система является автономной и замкнутой. Возможность своевременного анализа позволит решить задачу эффективного использования упаковочно-транспортной системы. Основная часть выпускной квалификационной работы содержит этапы анализа и разработки транспорта данных, их применение и эффект на практике.

Результатом выпускной квалификационной работы является создание цифрового двойника упаковочно-транспортной линии цеха и базы статистических данных для анализа. Данная работа является отправной точкой к инновационному проекту создания цифрового двойника всего цеха гофротары и анализу данных BigData.

## ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ .....	5
Глава 1 АНАЛИЗ ДЕЯТЕЛЬНОСТИ ПРЕДПРИЯТИЯ ООО «УРАЛБУМАГА»	8
1.1 Техничко-экономическая характеристика упаковочно-транспортной линии ООО «Уралбумага» .....	8
1.1.1 Характеристика предприятия ООО «Уралбумага» .....	8
1.1.2 Функциональная модель предприятия ООО «Уралбумага» .....	11
Выводы по параграфу 1.1 .....	14
1.2 Концептуальное моделирование предметной области .....	14
1.3 Постановка задачи на разработку информационной системы .....	17
1.3.1 Цель и назначение автоматизированного варианта решения задачи .....	17
1.3.2 Общая характеристика организации решения задачи на ЭВМ .....	18
1.3.3 Формализация расчетов подзадач .....	19
Выводы по параграфу 1.3 .....	20
1.4 Анализ существующих разработок и обоснование выбора технологии проектирования .....	20
1.4.1 Определение критериев анализа .....	22
1.4.2 Сравнительная характеристика существующих разработок .....	22
Выводы по главе 1 .....	23
Глава 2 РАЗРАБОТКА ПРОЕКТНЫХ РЕШЕНИЙ .....	24
2.1 Архитектура информационной системы сбора и визуализации данных .....	24
2.2 Логическое моделирование информационной системы сбора и визуализации данных .....	26
2.2.1 Логическая модель и ее описание .....	26
2.2.2 Определение требований к информационной системе .....	31
2.3 Реализация шлюза данных .....	35
2.3.1 OPC протокол и его возможности .....	35
2.3.2 Структура программного решения шлюза данных .....	36
2.3.3 Реализация библиотеки сбора данных по протоколу OPC .....	37
2.3.4 Реализация интерфейса шлюза данных .....	40

2.4 Реализация сервера или брокера данных.....	41
2.4.1 Протокол транспорта данных .....	42
2.4.2 Структура серверного решения .....	44
2.4.3 Реализация библиотек MQTT транспорта данных .....	45
2.4.4 Реализация библиотеки обращения к базе данных .....	47
2.4.5 Реализация интерфейса серверной части .....	47
Вывод по параграфу 2.4.....	49
2.5 Предоставление данных конечным пользователям.....	49
2.5.1 Предоставление отчетности.....	49
2.5.2 Визуализация данных по типу Цифрового двойника.....	54
2.5.3 Доступ к архивным данным.....	58
Вывод по параграфу 2.5.....	60
2.6 Возможности интеграции с другими системами .....	60
2.6.1 Связь информационной системы с внешними системами.....	61
2.6.2 Структура базы данных информационной системы .....	61
Выводы по главе 2.....	63
Глава 3 ОЦЕНКА И ОБОСНОВАНИЕ ЭКОНОМИЧЕСКОЙ ЭФФЕКТИВНОСТИ ПРОЕКТА .....	64
3.1 Выбор и обоснование методики расчета экономической эффективности....	64
3.2 Расчет показателей экономической эффективности проекта.....	65
Выводы по главе 3.....	69
ЗАКЛЮЧЕНИЕ .....	70
СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ .....	71
ПРИЛОЖЕНИЕ А Листинг структуры библиотеки DLL_OPС.....	74
ПРИЛОЖЕНИЕ Б Листинг структуры библиотеки dll_MQTT_Broker .....	76
ПРИЛОЖЕНИЕ В Листинг структуры библиотеки dll_MQTT_Client .....	77
ПРИЛОЖЕНИЕ Г Листинг структуры библиотеки dll_SQL .....	79
ПРИЛОЖЕНИЕ Д Пример отчета съема продукции .....	80

## ВВЕДЕНИЕ

Современное производство гофротары представляет собой совокупность высокоавтоматизированных производственных линий с минимальным количеством ручного труда.

Производство гофротары начинается с гофроагрегата, где подается ролевое сырье и производится листовой картон определенного профиля, в зависимости от требований клиента к упаковке. Листовой картон стекается на 8 перерабатывающих линий плоской или ротационной высечки, где происходит высечка, печать и формирование паллет гофропродукции, в соответствии со спецификацией заказа. Сформированные паллеты выходят на хвостовые наземные транспортеры линий переработки, откуда продукцию забирает автоматизированная телега упаковочно-транспортной системы. Телега передает паллеты на Паллетайзеры упаковочно-транспортной системы, где происходит окончательное формирование и упаковка паллет стрейч-пленкой. Упакованные паллеты по наземным транспортерам стекаются на выход упаковочно-транспортной системы, откуда их забирают и отвозят на склад погрузчики.

Производительность всего цеха во многом зависит от пропускной способности упаковочно-транспортной системы, надежности, а также алгоритма ее работы и профессионализма управляющего персонала.

Объектом исследования бакалаврской работы является деятельность упаковочно-транспортной системы цеха по производству гофротары ООО «Уралбумага».

Предмет исследования бакалаврской работы – автоматизированная информационная системы управления цеха по производству гофротары ООО «Уралбумага».

Целью выпускной квалификационной работы является разработка информационной системы сбора и визуализации данных (на примере ООО Уралбумага).

Для достижения цели необходимо решить следующие задачи:

- выполнить анализ предметной области, определить существующие проблемы и выявить пути решения;
- моделирование предметной области, разработка архитектуры программного обеспечения, а также реализация разработанных моделей на практике;
- оценить экономическую эффективность разработанной информационной системы.

Данная ВКР состоит из введения, 3 глав, заключения и списка литературы.

В первой главе описана организационная структура компании ООО Уралбумага, проведено исследование предметной области упаковочно-транспортной линии цеха производства гофротары, приведены требования и задачи разработки информационной системы, будет проведен анализ существующих решений, выбрана и обоснована стратегия автоматизации производственных процессов.

Во второй главе описана проектная часть, структура базы данных, описание программных модулей, интерфейс программы, осуществлена фактическая разработка проектных решений.

В третьей главе приведено экономическое обоснование разработанной информационной системы.

В заключении подведены результаты работы, описана практическая значимость разработанной информационной системы.

Система управления упаковочно-транспортной системы автономна и самодостаточна, может функционировать без участия оператора, но, с точки зрения управления производством и учета, является «черной дырой» с большим влиянием человеческого фактора. В ходе работы будут введены понятия выработки и простоев упаковочно-транспортной линии, что позволит оценивать и корректировать действия персонала, повышая производительность линии и всего цеха в целом. Разработка цифрового двойника линии позволит начальникам цеха, участка и руководству своевременно принимать решения в

режиме реального времени. Также немаловажным является возможность анализа и корректировки алгоритма работы упаковочно-транспортной линии на основе статистических данных.

Границами работы будет зона влияния системы упаковочно-транспортной линии, от хвостовых наземных транспортеров перерабатывающих линий, до выхода готовых к отгрузке паллет с хвостов упаковочно-транспортной линии.

Тема объединения систем и визуализации данных является трендом настоящего времени, способным значительно повысить управляемость и эффективность оборудования и процессов производства. Многие производства в настоящий момент имеют автоматизированные, но устаревшие автономные системы, их объединение способно вывести предприятие на совершенно новый уровень, повышая производительность, не прибегая к увеличению парка оборудования. Анализируя вектор развития разработчиков серьезного производственного оборудования, отчетливо видно направление в сторону интерфейсов объединения систем, удаленной поддержки реального времени, дополненной реальности и т.п. Настоящая работа, несомненно, своевременна, охватывает наиболее критичную точку производства и позволит предприятию быть уверенным в завтрашнем дне.

В ходе работы будут встречены препятствия в виде отсутствия документации и поддержки со стороны производителя оборудования, отсутствие рабочих примеров в сети Интернет. Но, как показала практика, сложные задачи являются наиболее интересными, а девиз компании «вместе к успеху» и содействие серьезных специалистов АСУТП, развития и учета позволяет решать самые амбициозные задачи.

# **Глава 1 АНАЛИЗ ДЕЯТЕЛЬНОСТИ ПРЕДПРИЯТИЯ ООО «УРАЛБУМАГА»**

## **1.1 Техничко-экономическая характеристика упаковочно-транспортной линии ООО «Уралбумага»**

### **1.1.1 Характеристика предприятия ООО «Уралбумага»**

Группа предприятий ПЦБК основана в 1959 году и является крупнейшим предприятием гофр-упаковочной отрасли России. ПЦБК – вертикально интегрированная компания, оказывающая услуги широкого спектра: от переработки сырья и производства полуфабриката до выпуска и реализации готовой продукции. Полный цикл производства позволяет предлагать индивидуальные упаковочные решения для бизнеса, работающего в пищевой, химической, мебельной и других отраслях. Ассортимент востребованной продукции включает в себя четырехклапанные ящики, изделия сложной конфигурации и комплектующие материалы [12].

Деятельность цеха гофротары №2 направлена на выпуск продукции товарного картона и гофроупаковки. Номенклатура линий производства гофроупаковки очень широкая и насчитывает 15 единиц. Наиболее распространенным видом продукции являются четырехклапанные ящики для упаковки широкого спектра продукции. Кроме этого, методом плоской или ротационной высадки изготавливаются изделия лоткового типа, сложной конфигурации.

В 1951 году постановлением Совета Министров СССР принято решение о начале строительства Левшинского древесно-массного завода. Фундамент одного из крупнейших в стране целлюлозно-бумажных комбинатов был заложен на берегу реки Чусовой вблизи станции Голованово. Сейчас завод носит имя «Прикамский картон» и входит в Группу предприятий «ПЦБК».

Предприятие постоянно совершенствовалось и укрупнялось. В 2007 году торжественно открыт новый картонно-бумажный цех. Тогда же стартовал крупный инвестиционный проект по увеличению мощности цеха гофротары №2. Закуплено и установлено современное оборудование ведущих западных



фирм. По уровню оснащённости новое гофропроизводство стало первым в России.

В 2017 году ГП ПЦБК вошла в число первых шести предприятий по реализации федеральной программы повышения производительности труда. При активной поддержке государственной корпорации «Росатом» предлагаются и реализуются интереснейшие проекты, результат которых виден уже сейчас.

Предприятие разделено территориально, в частности, цех гофротары №2 располагается на отдельной площадке и подчиняется директору производства гофротары. Структура подчинения приведена на рисунке 1.1.

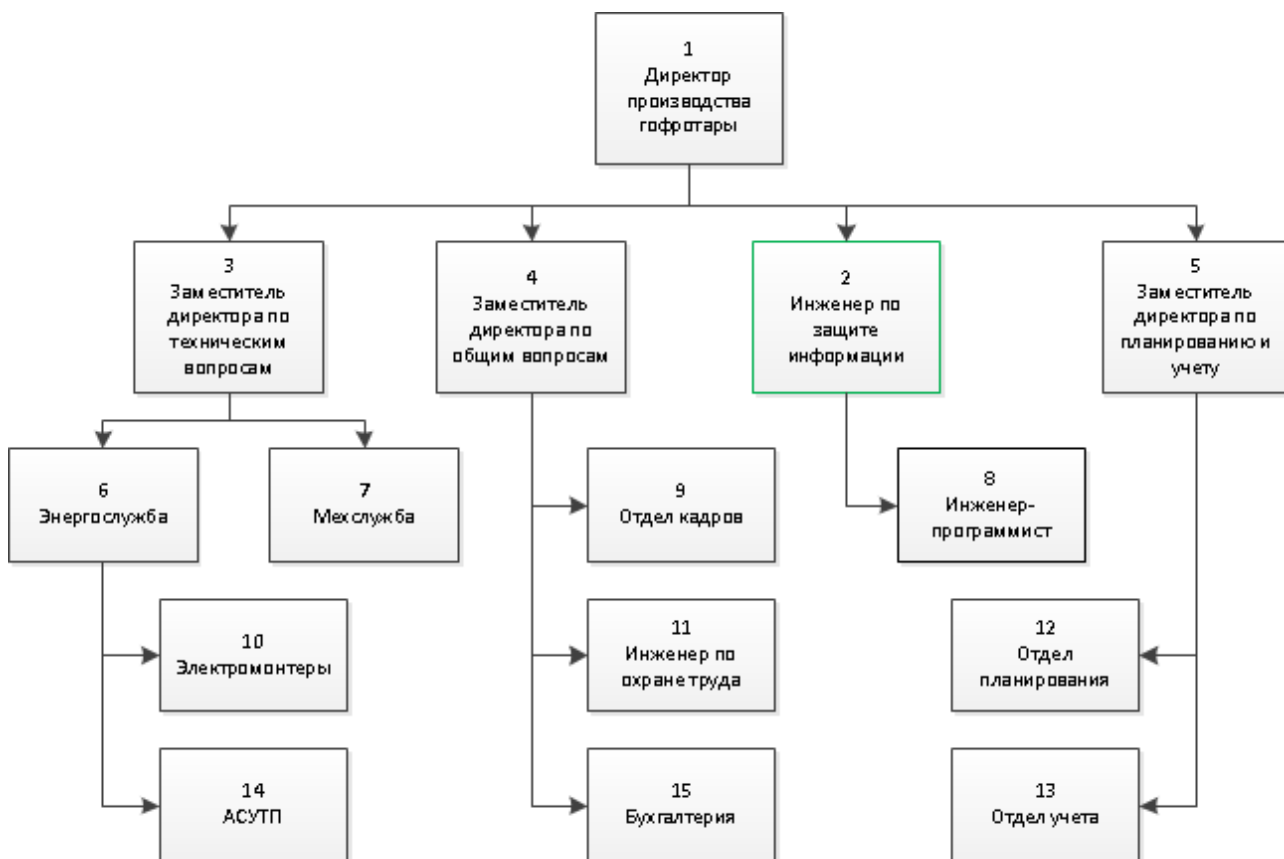


Рисунок 1.1 – Схема подчинения

Настоящая работа разрабатывалась с рабочего места (должности) инженера по защите информации, подчиненной напрямую директору производства. Это обусловлено тем, что выполняемые задачи тесно связаны как с производством, так и с планированием и учетом. Также должность

подразумевает распределение и контроль информации с точки зрения безопасности.

Предметной областью исследования является упаковочно-транспортная система цеха и ее зона влияния и автоматизации.

Упаковочно-транспортная система Logitek компании Emmeri Group установлена в цехе производства гофротары в 2011 году при массовом дооснащении оборудования и производственных мощностей. Потребность в производительной упаковочно-транспортной линии возникла в условиях приобретения и монтажа нового основного гофрировального агрегата и пяти новых перерабатывающих линий плоской и ротационной высадки. Компания Emmeri Group зарубежная, точнее Итальянская, производит оборудование по формированию, обвязке и упаковке для производств гофропродукции [18].

Упаковочно-транспортная линия состоит из наземных транспортеров, транспортной телеги и двух Паллетайзеров. Продукция, выпускаемая перерабатывающими линиями, оседает на хвостовых наземных транспортерах линий, выполняющих роль так называемой «подушки» между производительностью перерабатывающей линии и скорости забора продукции телегой транспортно-упаковочной линией. Телега по заданному алгоритму движется и забирает сформированные паллеты с хвостовых наземных транспортеров перерабатывающих линий и передает их на вход линии упаковки. На линии упаковки паллеты, в соответствии с требованиями заказчика, закрепляются ПВХ-лентой и обертываются стрейч-пленкой [20]. Параметры и требования к упаковке паллет задаются в системе упаковочно-транспортной линии в начале заказа для каждой перерабатывающей линии отдельно. Упакованные в соответствии со всеми требованиями паллеты продукции выходит с линии упаковки и оседает на хвостовых наземных транспортерах, откуда они забираются и транспортируются на склад для хранения погрузчиками.

С точки зрения упаковочно-транспортной системы, ее системы управления, линия оснащена автоматизированными наземными

транспортерами, поворотными и обвязочными механизмами и транспортировочной телегой. Управляет упаковочно-транспортной линией контроллер. Для оператора также доступен пульт управления со SCADA системой для возможности ручного вмешательства в процесс. SCADA - часть АСУ ТП, предназначенная для обеспечения работы в реальном времени систем сбора, обработки, отображения и архивирования информации об объекте мониторинга или управления. Стоит отметить, что контроллер способен действовать самостоятельно, без вмешательства оператора и SCADA системы. Данная SCADA система способна предоставить данные о перемещении продукции, состоянии оборудования и аварийных ситуациях средствами протокола OPC. Данные от SCADA системы будут являться источником для разрабатываемой ИС настоящей работы.

### 1.1.2 Функциональная модель предприятия ООО «Уралбумага»

Целью работы является предоставление технологической информации во вне для анализа и оперативного принятия решений. Влияние человеческого фактора должно быть минимально и подконтрольно, ошибки довольно дороги и могут сказаться на производительности всего цеха. В текущем состоянии система автономна, влиять на процесс транспортировки и упаковки продукции может только оператор, что изображено на рисунке 1.2 контекстной диаграммы «КАК ЕСТЬ».

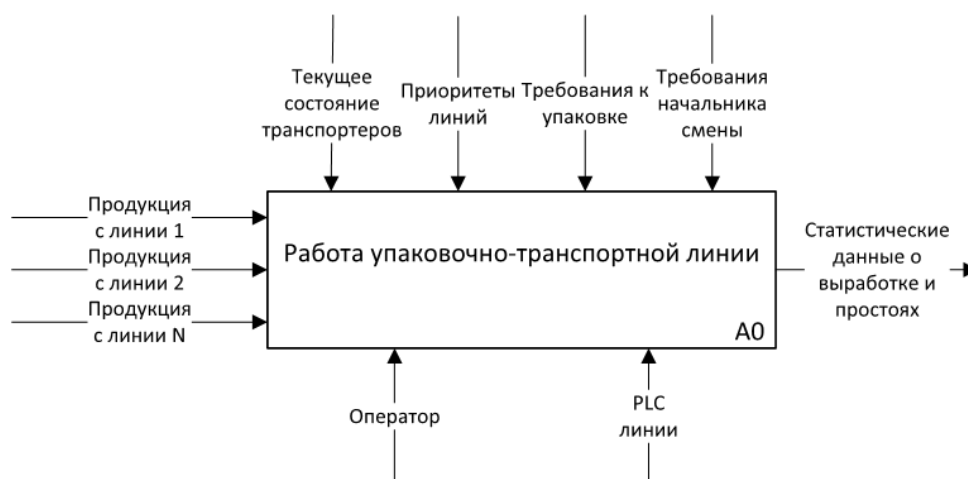


Рисунок 1.2 – Контекстная диаграмма «КАК ЕСТЬ»

На рисунке 1.3 представлена диаграмма «КАК ЕСТЬ» декомпозиции бизнес процесса «работы упаковочно-транспортной линии» контекстно диаграммы.

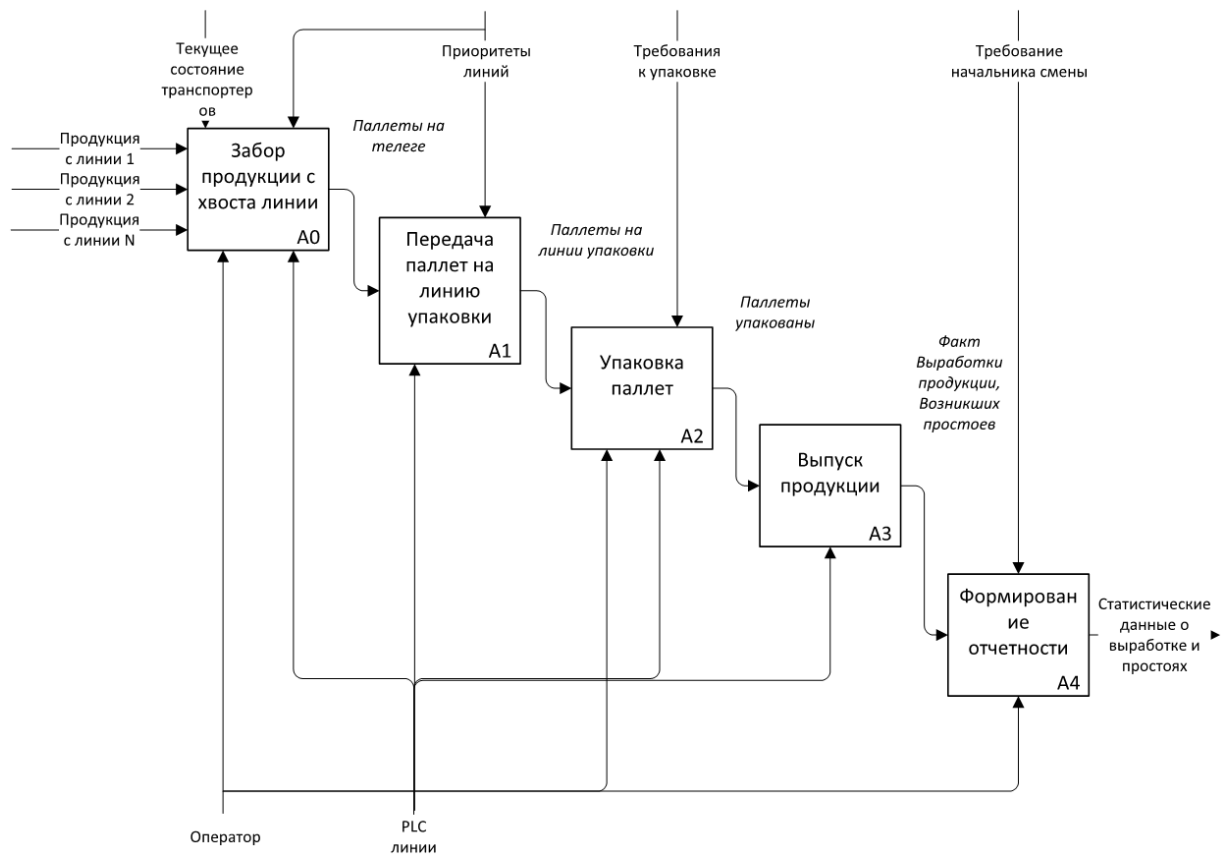


Рисунок 1.3 – Диаграмма «КАК ЕСТЬ» декомпозиции бизнес процесса «работы упаковочно-транспортной линии»

Паллеты готовой продукции забираются с хвостовых транспортеров перерабатывающих подвижной транспортной телегой и перемещаются на линию упаковки. По линии упаковки паллеты продукции, двигаясь друг за другом, упаковываются и стекаются на хвостовые транспортеры линии упаковки, откуда продукция забирается автопогрузчиком.

Также на рисунке 1.4 представим UML диаграмму типа Use-Case описывающую взаимодействие с системой упаковочно-транспортной линии.

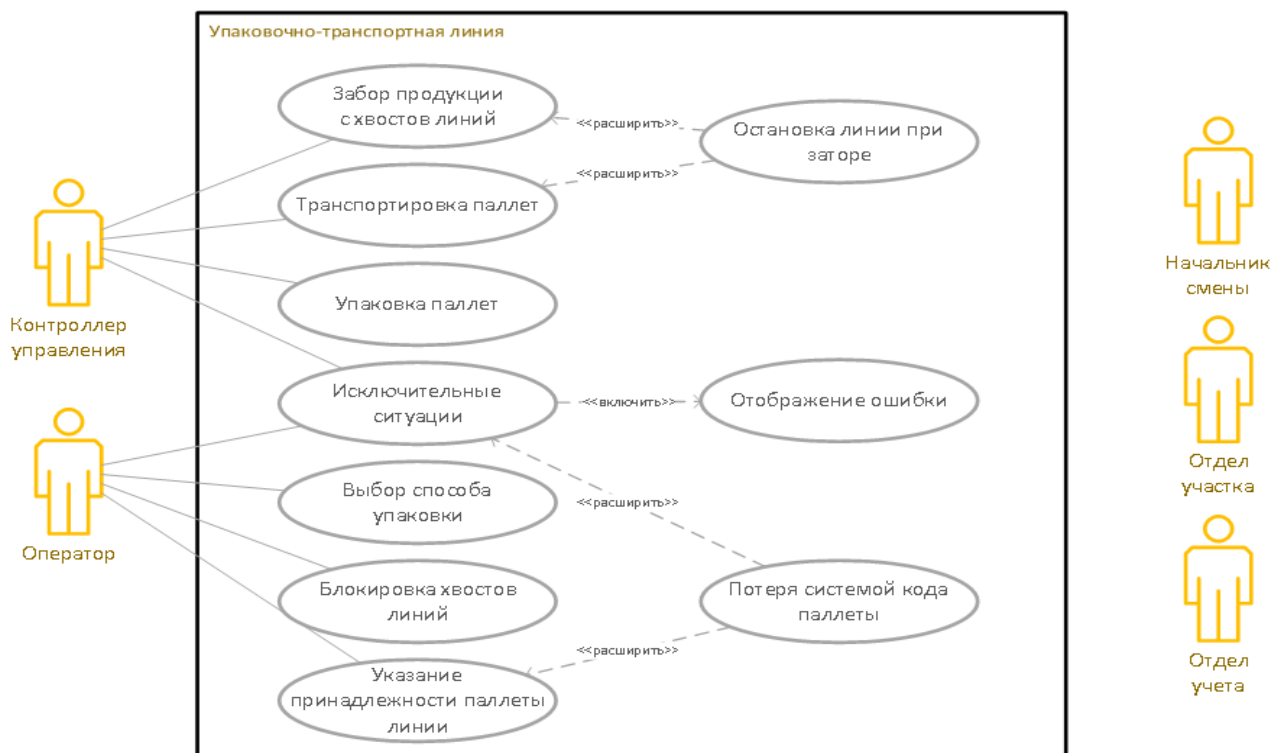


Рисунок 1.4 – Диаграмма использования системы упаковочно-транспортной линии

Как видно из схемы IDEF0 и диаграммы использования UML, начальники смены и участка, высшее руководство не оказывают никакого влияния и не контролируют работу упаковочно-транспортной системы. При текущем подходе многое зависит от действий оператора, а данные выработки и простоев передаются на словах и имеют низкую степень доверия.

Должность инженера по защите информации, на практике, подразумевает контроль над всеми производственными системами и серверами, их данными. Это дает возможности анализировать информационные потоки, видеть узкие места и возможности к автоматизации и повышению эффективности работы цеха и выравниванию информационных потоков. Такие возможности требуют от сотрудника постоянного участия в проектах развития систем управления предприятия. Плотное сотрудничество со службами автоматизации, развития, учета и грамотное руководство со стороны директора производства позволяет реализовывать самые амбициозные проекты.

Информационное развитие предприятий в последнее время движется большими шагами, что обусловлено пониманием и подстегиванием со стороны правительства Пермского края и государства.

### **Выводы по параграфу 1.1**

Цифровизация производства – необратимый путь развития российского предприятия. Более эффективная организация производственных процессов напрямую зависит от объема и доверия к техническим и технологическим данным, на базе которых возможно более точное принятие управленческих решений.

Задачей данной работы является интеграция компьютеризированных систем, детализация и визуализация процессов наиболее критичного узла производства, от которого зависит производительность и выполнение плана всего цеха. Предоставление инструмента контроля процессов упаковочно-транспортной линии позволит свести к минимуму влияние человеческого фактора и своевременно принимать решения по повышению производительности.

### **1.2 Концептуальное моделирование предметной области**

Система управления упаковочно-транспортной линией способна действовать автономно по жестко заданному алгоритму, либо управляться с помощью команд оператора через HMI интерфейс SCADA системы. Но интеграция с внешними системами и вывод информации во вне не предусмотрен разработчиками изначально. Проанализировав поведение упаковочно-транспортной системы и пообщавшись с операторами на местах, стало ясно – система не безупречна, многое зависит от действий оператора.

В настоящее время информация по учету на упаковочно-транспортной линии доводится оператором до начальника смены в бумажном виде, в заполненной от руки таблице. С определенным интервалом оператор подготавливает и передает следующую информацию:

- выработка, или количество упакованных паллет по каждому заказу по каждой линии переработки;
- простои в произвольной форме и примерным временем начала и продолжительности.

Не говоря о задержках и возможных искажениях передаваемой информации, беглый анализ работы упаковочно-транспортной линии на месте позволил выявить потерю важной с точки зрения анализа и управления информации:

- объем прохождения брака по транспортной линии, который отнимает определенную долю производительности;
- несвоевременный съём продукции с хвостовых транспортеров линий переработки, что приводит к их снижению скорости или остановке;
- факты падения паллет продукции с наземных транспортеров или транспортной телеги;
- нерациональное распределение нагрузки между двумя Паллетайзерами.

Также замечен человеческий фактор в виде смещения приоритетов и съема продукции в первую очередь с определенных производственных линий, где сотрудники имеют дружеские отношения.

В целом выделены следующие недостатки текущей реализации системы и ее процессов для дальнейшей проработки:

- информация по учету выработки и простоев линии, необходимая для анализа и принятия управленческих решений, минимальна, отсутствуют важные для анализа события;
- трудоемкость сбора и обработки информации по выработке и простоям, передаваемой в бумажном или устном варианте от оператора до начальника смены;
- задержка при передаче информации и невозможность оперативного контроля и вмешательства со стороны начальника смены, интервал как правило 4 часа;

- риск искажения информации при передаче от оператора до начальника смены, при занесении данных с бумажного носителя в электронный Excel рапорт начальника смены;
- снижение производительности линии из-за отсутствия контроля и влияния человеческого фактора, отсутствие информации по прохождению брака по транспортной линии.

Как было сказано ранее, целью работы не является изменение процесса работы упаковочно-транспортной линии. Более того, вмешательство в конструктив линии и логику работы контроллера управления недопустимо. Контекстная диаграмма «КАК ДОЛЖНО БЫТЬ», приведенная на рисунке 1.5 отличается возможностью влияния и автоматизацией сбора производственной информации выработки и простоев каждого этапа производственного процесса.

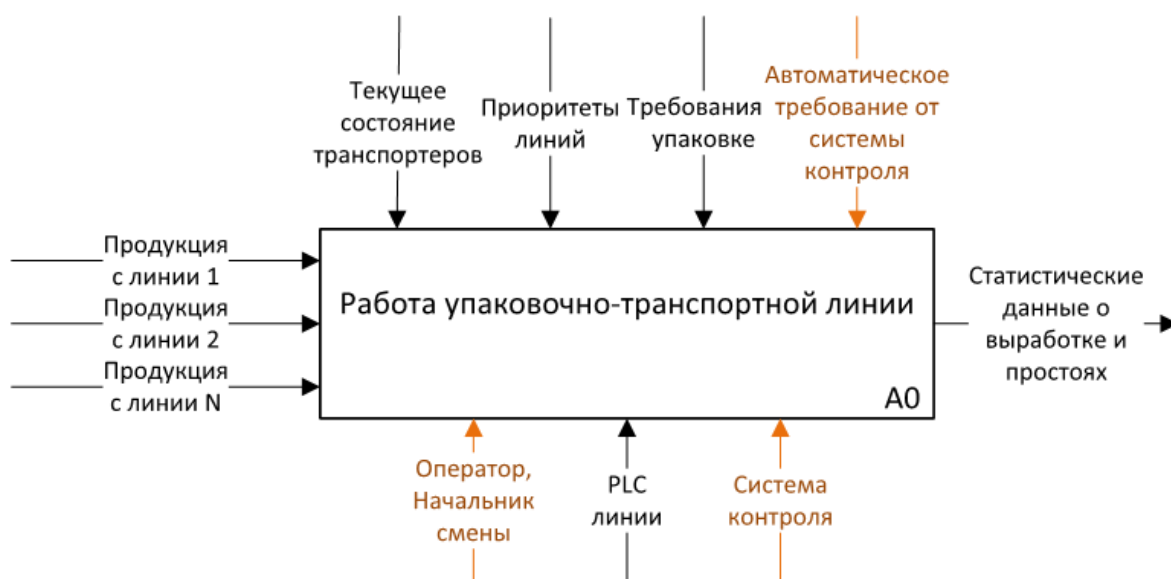


Рисунок 1.5 – Контекстная диаграмма «КАК ДОЛЖНО БЫТЬ»

На рисунке 1.6 представлена диаграмма «КАК ДОЛЖНО БЫТЬ» декомпозиции бизнес процесса «работы упаковочно-транспортной линии» контекстно диаграммы.



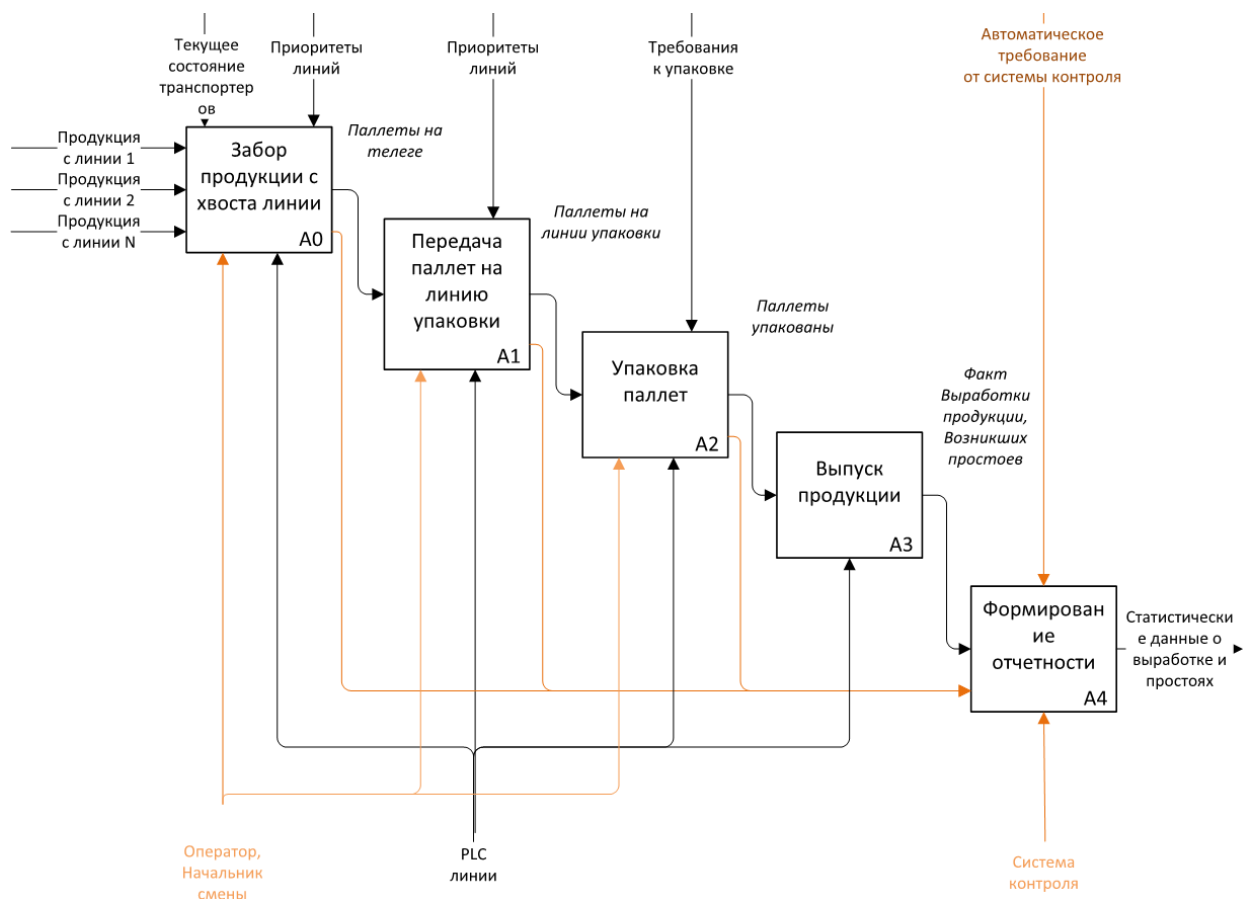


Рисунок 1.6 – Диаграмма «КАК ДОЛЖНО БЫТЬ» декомпозиции бизнес-процесса «работы упаковочно-транспортной линии»

Фактически для оператора упаковочно-транспортной линии ничего не меняется, за исключением способа предоставления отчетной информации.

### 1.3 Постановка задачи на разработку информационной системы

#### 1.3.1 Цель и назначение автоматизированного варианта решения задачи

Выявлены и согласованны с руководством как важные следующие исключительные ситуации и события упаковочно-транспортной линии:

1. Факт и продолжительность несвоевременного снятия продукции с хвостовых транспортеров линий переработки.
2. Фиксация по времени включения/отключения светофоров линий, разрешающих выход продукции с хвостовых транспортеров линий на транспортную телегу.

3. Фиксация по времени включения/отключения отдельных узлов упаковочно-транспортной линии.
4. Очередь паллет продукции перед обмотчиком, продолжительность.
5. Очередь паллет продукции перед упаковщиком, продолжительность.
6. Фиксация выработки, или факт выхода паллеты продукции с последнего наземного транспортера

Таким образом общая задача цифровизации упаковочно-транспортной линии декомпозирована до конкретного перечня проблем, на которых необходимо сосредоточиться. Назначение реализации данной работы будет:

- повышение производительности упаковочно-транспортной линии за счет снижения человеческого фактора и повышения управляемости процесса упаковки продукции;
- увеличение скорости реакции на исключительные ситуации начальника смены и ремонтного персонала;
- появление достоверной и полной статистической информации и возможность полноценного анализа;
- снижение нагрузки в виде отчетности с операторов и начальника смены.

### 1.3.2 Общая характеристика организации решения задачи на ЭВМ

Для формулировки целей и задач работы и выделим основные требования к реализации:

- интеграция систем не должна вносить изменения в конструктив линии и логику работы контроллера, интеграция строго односторонняя, влияние на систему управления извне и изменение ее тэгов и состояния недопустимо;
- система реального времени, или минимальные задержки обновления данных во внешней системе;
- предоставить только важную с точки зрения учета, анализа и принятия управленческих решений информацию;

- надежное хранение и резервирование статистической информации, автоматическое удаление устаревшей информации;
- предоставление статистической информации в виде составных отчетов, с учетом пользовательских фильтров и периодов статистических данных;
- интерфейс визуализации упаковочно-транспортной линии по типу цифрового двойника;
- возможность расширения проекта до границ всего цеха;
- реализация программного интерфейса дублирования поступающих данных в другие внешние системы.

### 1.3.3 Формализация расчетов подзадач

Учитывая сложность решаемых задач и объем требований, общая задача цифровизации довольно сложна и требует декомпозиции по отдельным исполнительным элементам системы. Возможные будущие потребности к расширению границ системы диктуют модульный подход к разработке, система должна быть легко интегрируемой в иные производственные системы в будущем. Общее решение будет состоять из следующих модулей / подзадач:

- сбор и передача информации;
- принятие, обработка и распределение информации;
- хранение информации;
- визуализация и предоставление отчетности.

Сбор и передача информации предполагает использование отдельного устройства или программного обеспечения. Основной задачей здесь является реализация интерфейса коммуникации с подконтрольной системой, в данном случае SCADA системой из программного комплекса Siemens WinCC. На данном этапе обработка данных минимальна, скорость опроса источника информации приближена к реальному времени.

Принятие, обработка и распределение информации реализует отдельный модуль серверной части программного обеспечения. Здесь происходит фильтрация и распределение информации согласно заданному алгоритму.

Хранение информации, как общее понятие, предполагает обеспечение конфиденциальности, целостности и доступности информации. Способ размещения информации – единая реляционная база данных.

Визуализация и предоставление отчетности как последний заключительный этап разработки проекта. Предполагается использовать карту цеха и итоговые показатели на ней для визуализации процесса работы упаковочно-транспортной линии. Доступ к статистическим данным будет реализован через составные отчеты Crystal Reports [25] в клиентской части информационной системы для удобства пользователя.

### **Выводы по параграфу 1.3**

Рассмотрев узкие места и недостатки текущей реализации предметной области, выявлены и согласованы основные проблемы и способы их решения. Общая задача цифровизации декомпозирована до конкретных достижимых подзадач. Плановый эффект от внедрения является значимым с точки зрения производительности всего цеха.

### **1.4 Анализ существующих разработок и обоснование выбора технологии проектирования**

Тема цифровизации производств и перехода к модели «Индустрия 4.0» актуальна и трудоемка, проанализируем наличие существующих разработок на рынке по подзадачам, декомпозированным в параграфе 1.3.

Задачи хранения информации в наше время не представляет особой сложности выбора или реализации. В случае ГП ПЦБК выбор СУБД продиктован общей политикой использования продуктов Microsoft. Следовательно, к использованию выбран продукт Microsoft SQL Server, для начала Express версии.

Задача визуализации и предоставления отчетности также не нуждается в анализе и выборе технологий. Разработка пользовательских приложений по технологии WPF привычна и эффективна. Потокное формирование отчетов средствами SAP Crystal Reports также привычно для предприятия и позволит быстро начать использовать функционал отчетности, не прибегая к изучению иных технологий.

Принятие, обработка и распределение информации – задачи крайне индивидуальные. Правила обработки и распределения информации продиктованы технологическими процессами. Здесь основная задача сделать систему гибкой, а значит определить простой и перспективный протокол обмена данными. Привычные для локальной автоматизации промышленные протоколы Modbus, Profibus и подобные требуют сложной проработки и поддержки. Пора уступать место протоколам верхнего уровня, лучшим из которых является протокол MQTT [13]. Это подтверждают и производители промышленного оборудования, выпуская шлюзы преобразования всевозможных протоколов в MQTT.

С точки зрения сбора и передачи информации. Заранее определимся, что SCADA система предоставляет для коммуникации протокол OPC [15], от этого и будем исходить в своих требованиях. С учетом будущих потребностей в расширении границ проекта и подключения новых линий и устройств, использующих протоколы коммуникации отличные от OPC, целесообразно использовать шлюзы преобразования OPC, ... в MQTT. Это даст нам гибкость и масштабируемость всей системы.

По преобразователям протоколов в MQTT рынке есть что предложить. Устройства начали массово выпускаться ведущими производителями оборудования промышленного назначения, такими как MOXA [16] или ICP DAS [11], несколько лет назад. Остановимся на анализе предложений рынка именно в этой сфере.

### 1.4.1 Определение критериев анализа

В выборе шлюза определимся с основными требованиями:

- надежность, эксплуатация в режиме 24/7 круглогодично, в условиях повышенной температуры;
- поддержка на выходе протокола MQTT;
- поддержка на входе протоколов OPC, Modbus, Profibus, Canbus, Profinet, ... самостоятельно, или средствами подключаемых модулей;
- возможность подключения дискретных и аналоговых сигналов самостоятельно, или средствами подключаемых модулей;
- масштабируемость и модульность;
- доступность и полнота информации в сети Интернет;
- цена и итоговая стоимость комплекта с учетом дополнительных модулей.

Стоит отдельно учесть отличия в спецификациях наиболее интересного нам протокола OPC. При анализе необходимо обращать внимание на поддержку устаревших стандартов OPC DA и AE, используемых системой упаковочно-транспортной линии. Новый универсальный стандарт OPC UA не сможет использоваться.

### 1.4.2 Сравнительная характеристика существующих разработок

В сравнительном анализе, приведенном в таблице 1.7, будут участвовать два инновационных решения от ведущих производителей промышленного оборудования. А также полностью персонализированное решение на базе платформы Raspberry PI [24] последней модели, обеспечивающее полную управляемость и возможности, ограниченные лишь опытом разработчика.

Таблица 1.7 – Сравнение характеристик

Характеристика	Шлюз		
	MOXA ThingsPro	ICP DAS	Raspberry PI
Надежность, температура эксплуатации	Надежный металлический корпус и работа в диапазоне температур от -40 до +70	Надежный металлический корпус и работа в диапазоне температур от -25 до +75	Возможен цельный алюминиевый корпус и работа в диапазоне температур от -40 до +85
Поддержка	Программная поддержка,	В режиме Modbus в	Программная поддержка

Таблица 1.7 – Сравнение характеристик

Характеристика	Шлюз		
	MOXA ThingsPro	ICP DAS	Raspberry PI
протокола MQTT на выходе	включая облачные технологии	MQTT	любых спецификаций
Поддержка протоколов на входе	Аппаратная или программная поддержка любых протоколов	Только Modbus, поддержка любых протоколов через модули преобразования в Modbus	Аппаратная или программная поддержка любых протоколов
Подключение дискретных и аналоговых сигналов	Через модули	Через модули преобразования в Modbus	Прямая поддержка
Масштабируемость	Через модули	Через модули	Через модули
Доступность и полнота документации	Минимальна, наличие дистрибьютера ООО «Ниеншанц-Автоматика»	Минимальна, наличие дистрибьютера ООО «АйПиСи2Ю»	В избытке в открытом доступе
Цена шлюза	31 500р.	35 000р.	3 600р.
Итоговая стоимость типового комплекта	46 700р.	55 500р.	8 500р.

В ходе изучения информации по продукции компании MOXA выяснилось, что компания позаботилась о легкости перехода от решений на базе Raspberry PI до решений MOXA ThingsPro, предоставив пользователю инструмент для миграции.

### Выводы по главе 1

В первой главе ВКР были исследованы узкие места наиболее критичной производственной упаковочно-транспортной линии. Был выполнен анализ предметной области и определена сущность задачи автоматизации.

Выявлены недостатки текущей реализации и определены основные требования к проектной реализации системы.

Анализ существующих на рынке решений позволил более точно выявить подход к реализации, с учетом будущего развития информационной системы.

Группа предприятий ПЦБК динамично развивалось на протяжении всей истории, внедрение подобных проектов позволит открыть возможности и вывести предприятие на совершенно новый уровень.

## Глава 2 РАЗРАБОТКА ПРОЕКТНЫХ РЕШЕНИЙ

### 2.1 Архитектура информационной системы сбора и визуализации данных

Настоящая выпускная квалификационная работа предполагает сбор и визуализацию данных на примере упаковочно-транспортной линии цеха гофротары ООО «Уралбумага». Но разрабатываемая информационная система должна быть единой и, при необходимости, иметь возможность расширяться и включать сбор данных с других производственных линий, их узлов и управляющих устройств.

Приведенная на рисунке 2.1 архитектура информационной системы подразумевает унификацию процесса сбора, транспорта, обслуживания и предоставления конечным пользователям данных.

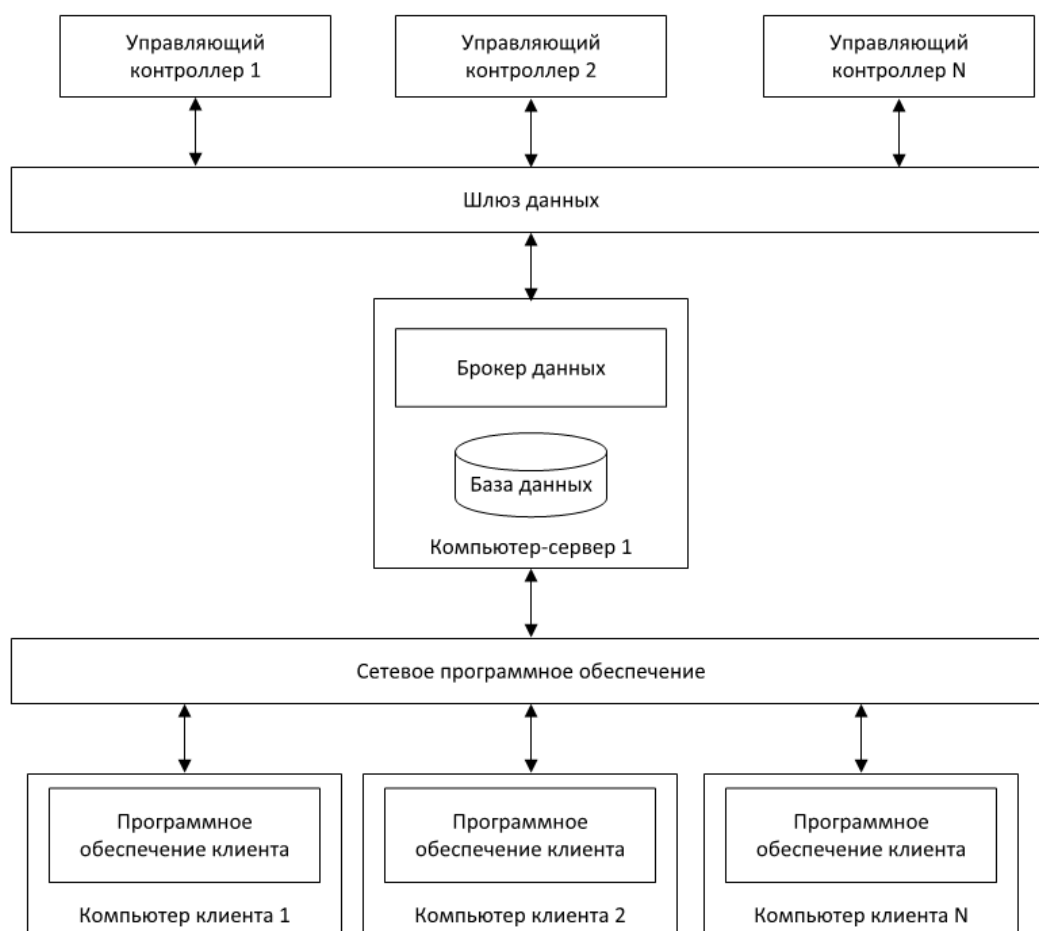


Рисунок 2.1 – Архитектура информационной системы



Для этого необходимо:

- ввести понятие шлюза данных, на нем ляжет основная нагрузка общения с любыми конечными устройствами и транспорта данных на верхний уровень;
- шлюзы данных разрабатывать универсальными и взаимозаменяемыми, реализация новых функций означает форсированное обновление возможностей всех шлюзов;
- логика нижнего уровня сведена к минимуму и заключается в простом преобразовании данных отдельного конечного устройства к формату понятному шлюзу данных;
- информационная система верхнего уровня едина и универсальна, обеспечивает хранение и предоставления данных со всех шлюзов данных, что обеспечивает удобства пользователя и обслуживающего персонала;
- распределение нагрузки вычислительных сетей, шлюзы данных должны общаться с устройствами нижнего уровня в реальном времени, когда для верхнего уровня данные можно транслировать значительно реже;
- быстрое внедрение типовых решений, научив шлюз общаться с одним конечным устройством, получаем возможность общения со всеми однотипными устройствами.

В дальнейшем данная архитектура позволит без серьезных финансовых и временных затрат подключать к разрабатываемой информационной системе другие производственные узлы и линии и их устройства. Можно выделить отдельные подзадачи реализации информационной системы, для удобства распределения по параграфам:

- получение и передача данных от SCADA системы упаковочно-транспортной линии, т.е. роль шлюза данных;
- получение данных и организация их хранения, т.е. роль сервера или брокера данных;

- визуализация данных по типу цифрового двойника и предоставление отчетности, как часть пользовательского приложения;
- структура базы данных и возможность интеграции с другими системами.

## **2.2 Логическое моделирование информационной системы сбора и визуализации данных**

### 2.2.1 Логическая модель и ее описание

Рассмотрим проектируемую информационную систему, точнее ее часть взаимодействия с конечными пользователями с точки зрения диаграммы вариантов использования. Диаграммы вариантов использования позволяют выделить функциональную структуру, не углубляясь в детали ее реализации.

С клиентской частью информационной системы, предоставляющей данные конечным пользователям, будут взаимодействовать следующие актеры:

- начальник смены;
- руководство;
- обслуживающий персонал;
- отдел учета.

Можно выделить следующие прецеденты, доступные всем актерам, описанные в таблице 2.2, которые должны быть реализованы в проектном решении.

Таблица 2.2 – Краткое описание прецедентов

<b>Прецедент</b>	<b>Краткое описание</b>
Получение текущего состояния оборудования	Получение текущего состояния оборудования в графическом виде на карте цеха
Получение архивных данных состояния оборудования	Получение архивных данных состояния оборудования в графическом виде на карте цеха по шкале времени
Формирование и просмотр отчетности	Формирование отчетности по накопленным данным и отображение на экране клиентской

Таблица 2.2 – Краткое описание прецедентов

Прецедент	Краткое описание
	части информационной системы
Сохранение сформированной отчетности	Сохранение сформированной отчетности в формате PDF

Разработанная диаграмма вариантов использования (use case diagram) в нотации UML для основных прецедентов проектируемой системы представлена на диаграмме рисунка 2.3.

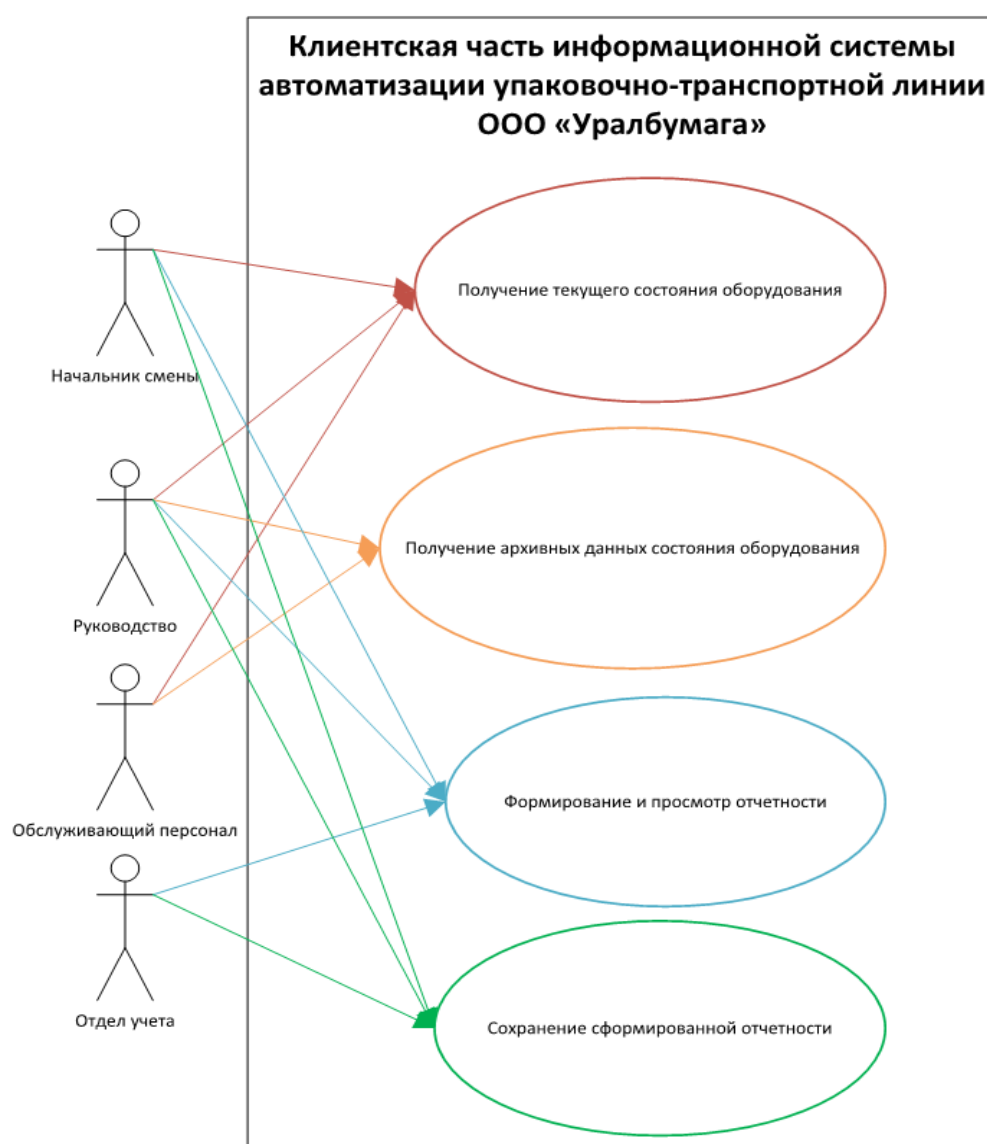


Рисунок 2.3 – Диаграмма вариантов использования

По диаграмме вариантов использования рисунка 2.3 видно, что конечные пользователи имеют возможности:

- видеть текущее состояние оборудования в графическом виде на карте цеха, по типу цифрового двойника;
- имеют возможность получить архивные данные состояния оборудования в графическом виде на карте цеха;
- формировать и просматривать отчеты по накопленным данным в интерфейсе информационной системы;
- сохранять сформированный отчет в формате PDF.

Конечных пользователей может быть множество, каждый имеет неограниченный доступ к информационной системе и ее возможностям. Автоматизированная ИС получает данные от базы данных, которая пополняется и обслуживается серверной частью информационной системы.

С точки зрения всей проектной информационной системы, состоящей из нескольких частей, графическое представление логической структуры представлено на рисунке 2.4 логической модели данных в нотации UML.

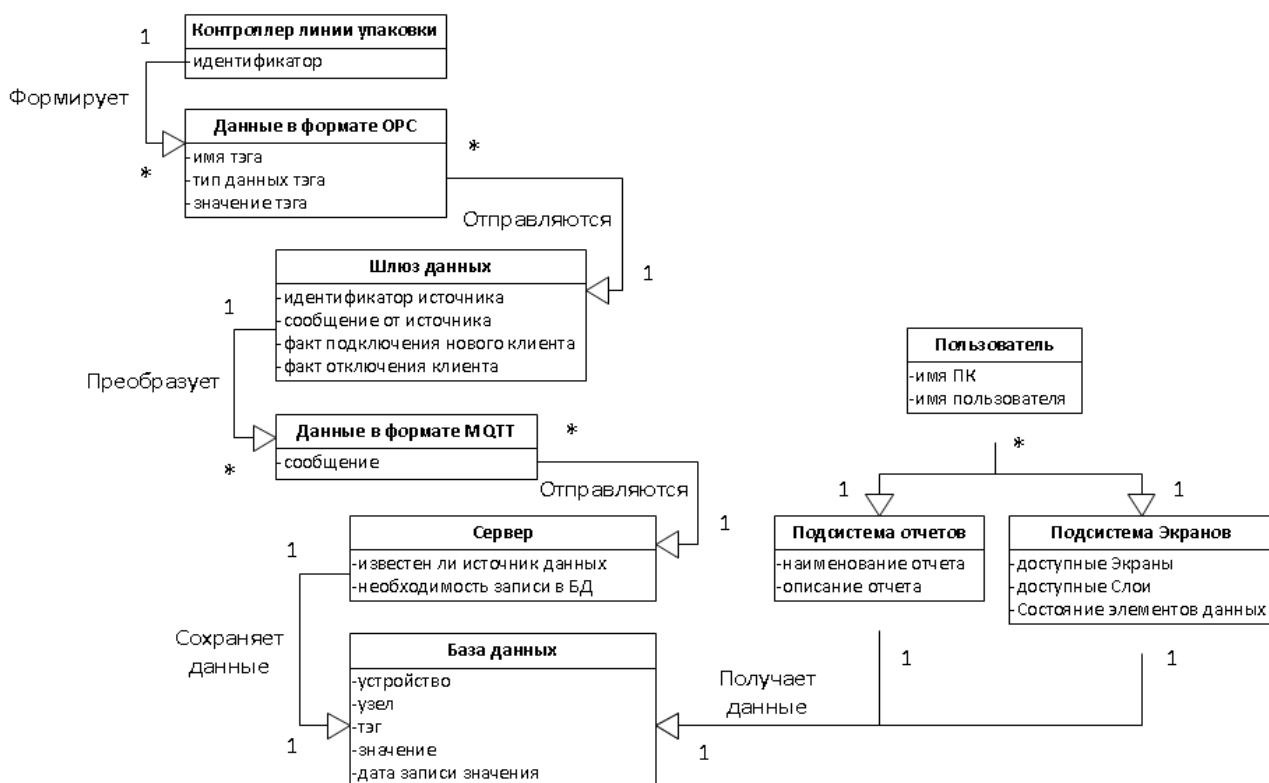


Рисунок 2.4 – Логическая модель данных

Общее решение разрабатываемой информационной системы в среде Microsoft Visual Studio состоит из нескольких отдельных проектов. Из них 3 полноценные программы и вспомогательные библиотеки.

Серверная часть, отвечающая за сбор данных от шлюзов данных и сохраняющая информацию в базе данных, называется «Server\_Control». Структура иерархии классов представлена на диаграмме классов в нотации UML рисунка 2.5.

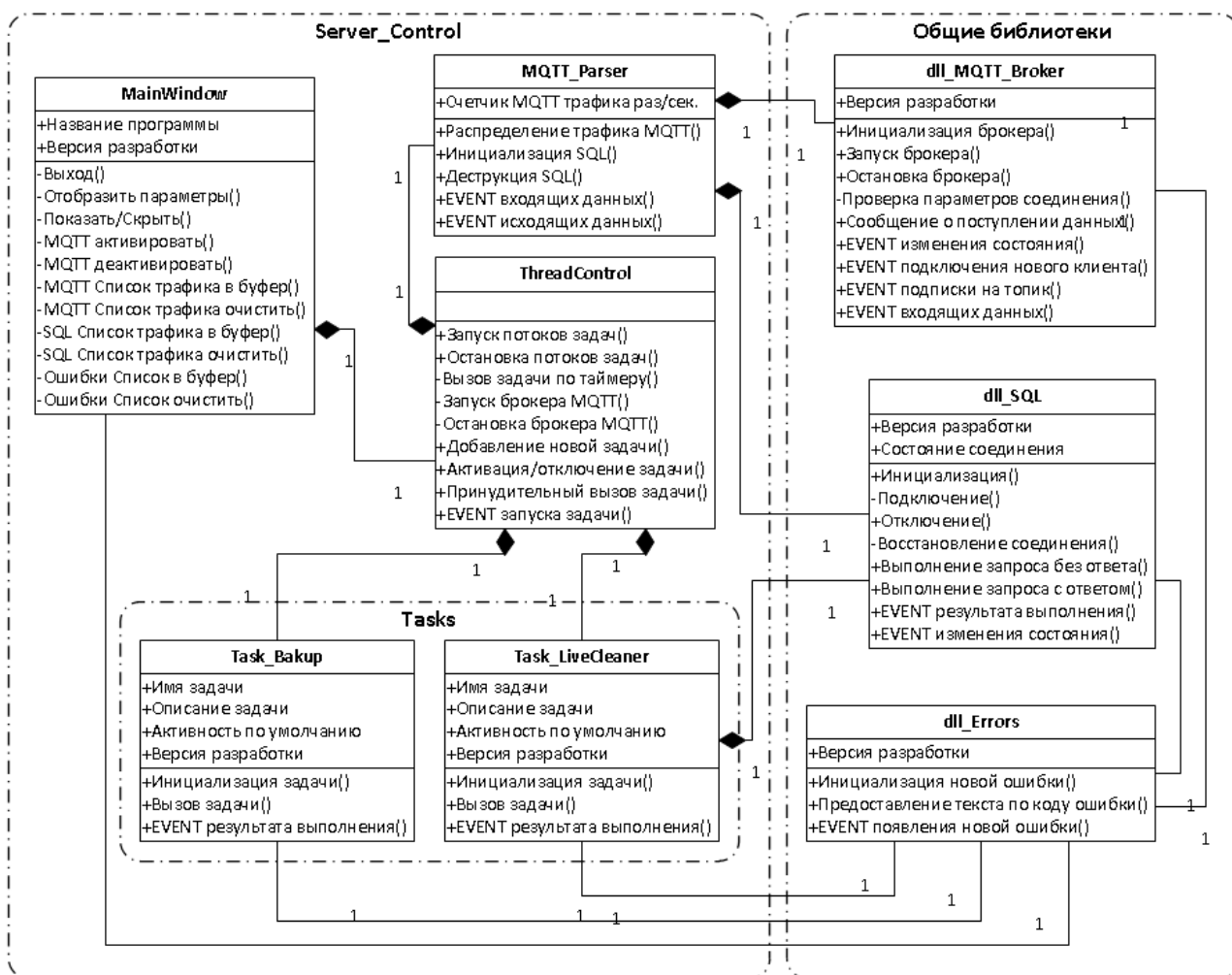


Рисунок 2.5 – Диаграмма классов Server\_Control

Серверная часть также управляет задачами обслуживания базы данных и периодической отчетностью администратору информационной системы для контроля над состоянием информационной системы.

Шлюз данных, как часть общего решения, отвечающий за сбор данных с конечных устройств и передающий данные серверу стандартизовано,

реализован программно и называется «GatewayP OPC Logitek». Структура иерархии классов представлена на диаграмме классов в нотации UML рисунка 2.6.

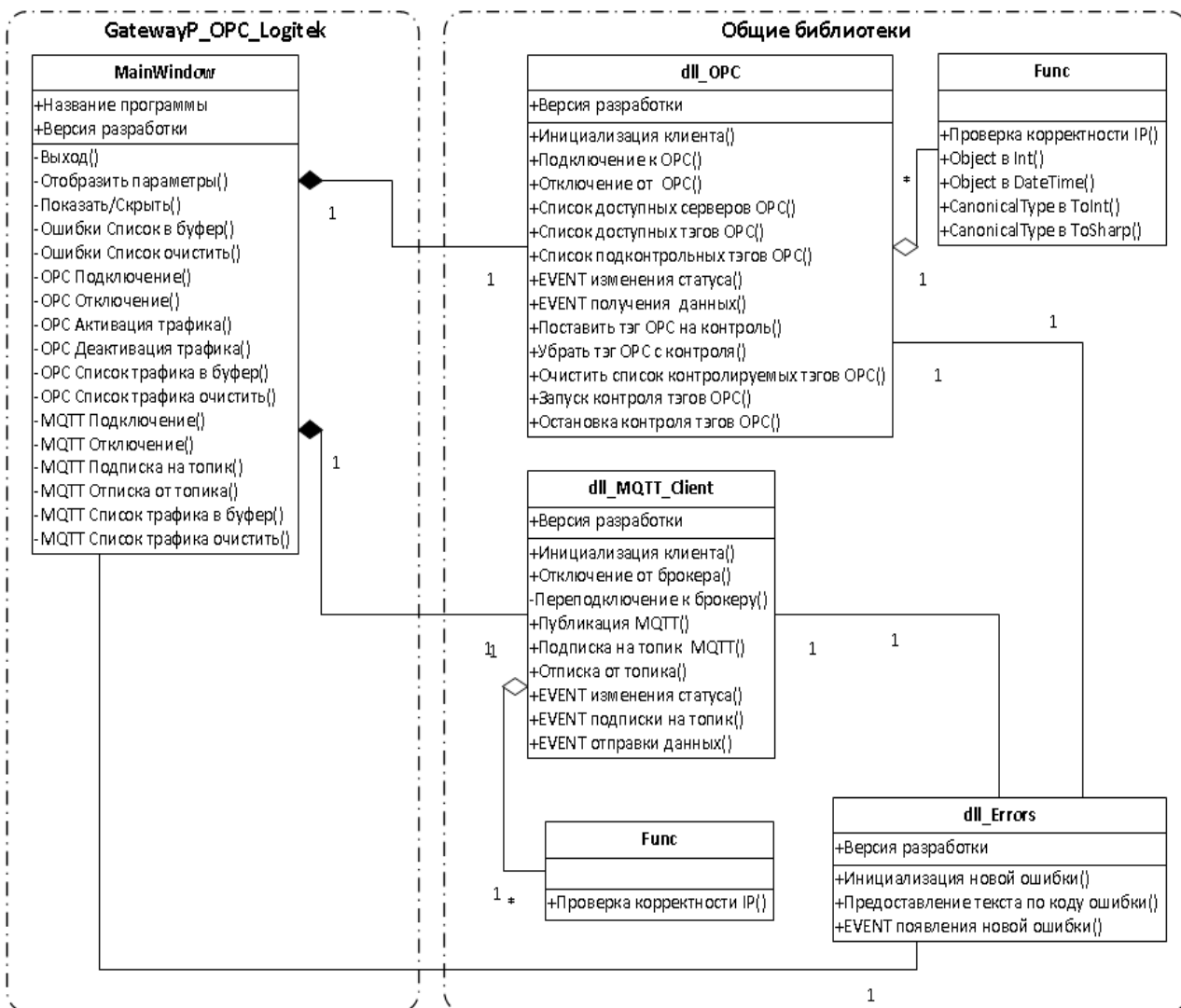


Рисунок 2.6 – Диаграмма классов GatewayP OPC Logitek

Клиентская часть общего решения, отвечающая за предоставление данных конечным пользователям, называется «Client\_Visual». Структура иерархии классов представлена на диаграмме классов в нотации UML рисунка 2.7.

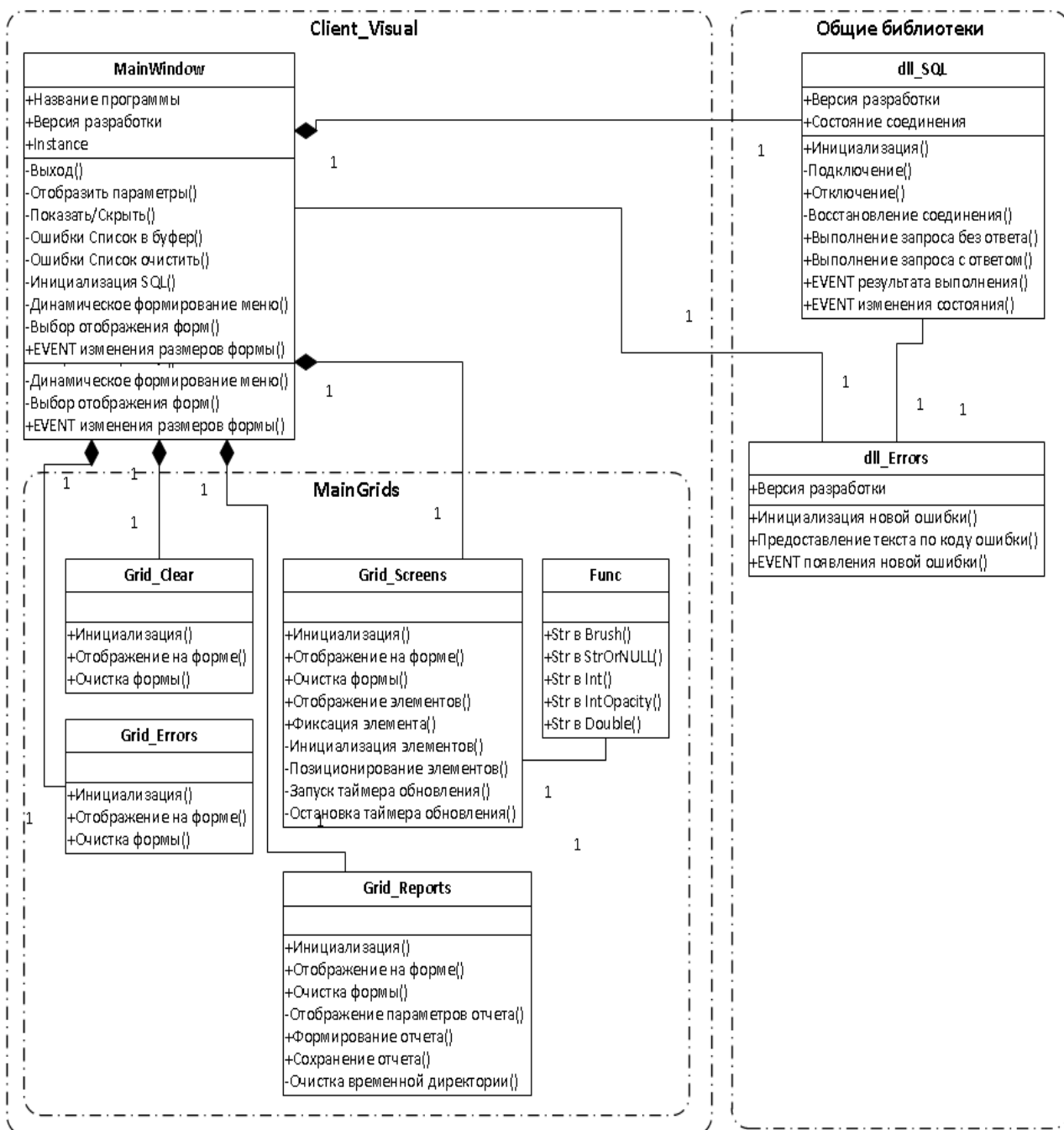


Рисунок 2.7 – Диаграмма классов Client\_Visual

### 2.2.2 Определение требований к информационной системе

Клиентскую часть информационной системы планируется запускать на современных пользовательских персональных компьютерах.

Сформируем требования к разрабатываемой системе, таблица 2.8 описывает функциональные требования, а таблица 2.9 – нефункциональные требования.

Таблица 2.8 – Функциональные требования к системе

<b>№</b>	<b>Требование</b>	<b>Важность</b>
1	Информационная система должна иметь простой и удобный графический интерфейс.	абсолютно необходимая
2	Меню информационной системы должно динамически выстраиваться в зависимости от того, какие данные существуют в базе данных.	абсолютно необходимая
3	Информационная система должна сворачиваться в область уведомлений при событии сворачивания окна информационной системы.	желательная
4	Двойное нажатие левой кнопки мыши по значку информационной системы в области уведомлений должно инициировать показ/скрытие окна информационной системы.	желательная
5	При запуске информационной системы, если соединение с базой данных сервера не установлено, блокировать меню и действия пользователя.	абсолютно необходимая
6	Возникающие в процессе работы исключительные ситуации должны обрабатываться и выводиться в список ошибок.	абсолютно необходимая
7	Обновление данных по циклу на карте цеха должно выполняться в отдельном потоке для исключения заморозки всего приложения.	абсолютно необходимая
8	Обновление элементов данных по циклу на карте должно происходить с интервалом в 3 секунды.	абсолютно необходимая
9	При изменении размера окна информационной системы, графические элементы данных карты цеха должны менять свое позиционирование и размер элемента в соответствии с новыми размерами окна.	абсолютно необходимая
10	Генерация и отображение отчета должна начинаться по	абсолютно



Таблица 2.8 – Функциональные требования к системе

№	Требование	Важность
	факту выбора из списка отчетов в меню.	необходимая
11	Генерация отчета, как длительная операция, должна происходить в отдельном потоке.	абсолютно необходимая
12	Сохранение отчета должно поддерживать формат PDF.	абсолютно необходимая
13	Возникающие в процессе генерации отчетов исключения должны обрабатываться по аналогии с общими исключениями информационной системы.	

Таблица 2.9 – Нефункциональные требования к системе

№	Требование	Важность
<b>Требования к производительности</b>		
1	Время реакции системы на события должно быть не более трех секунд	абсолютно необходимая
<b>Требования к операционной среде и пропускной способности каналов связи с серверами</b>		
1	Операционная система Windows 7 и новее.	абсолютно необходимая
2	Установленный .NET Framework версии 4.6 и новее.	абсолютно необходимая
3	Установленная среда исполнения SAP CrystalReports Runtime версии 13.0.24 [17].	абсолютно необходимая
4	Клиентские пользовательские компьютеры должны быть подключены к общей подсети с сервером и базой данных информационной системы	абсолютно необходимая
5	Минимальное разрешение экрана 800 x 600.	абсолютно необходимая

Таблица 2.9 – Нефункциональные требования к системе

№	Требование	Важность
6	Минимальная скорость соединения с сервером и базой данных – 100mb/s	
<b>Требования к точности вычислений</b>		
1	Ошибки при конвертации типов данных должны приводить к исключениям.	абсолютно необходимая
2	При подозрении на плохое качество данных, данные исключаются из результата.	абсолютно необходимая
<b>Требования к технической безопасности и надежности системы</b>		
1	Конечные пользователи должны обладать привелегиями доступа уровня чтения базы данных информационной системы.	абсолютно необходимая
2	Доступ к данным 24 часа в сутки, допускается плановое обслуживание не более 24-х часов в месяц.	абсолютно необходимая
3	Среднее время восстановления работоспособного состояния не более 1 суток.	абсолютно необходимая
<b>Требования к технической и сопроводительной документации, обучению пользователей</b>		
1	Документация распространяется вместе с клиентской частью информационной системы	желательная
2	Документация является частью проекта среды разработки и актуализируется с выходом нового релиза	желательная

Разрабатываемая информационная система не влияет прямо на производственные процессы предприятия, поэтому допустимы продолжительное плановое обслуживание и частые обновления.

## 2.3 Реализация шлюза данных

В начальном периоде жизненного цикла разрабатываемой информационной системы, в качестве шлюза данных будет выступать программное решение, это позволит ускорить процесс разработки на начальном этапе и более качественно отловить все исключения в процессе разработки. При этом программное решение будет разрабатываться отдельными библиотеками, что позволит в дальнейшем с минимальными трудозатратами портировать его на физическое устройство шлюза данных.

### 2.3.1 OPC протокол и его возможности

Технологии OPC были созданы для того, чтобы можно было легко и безопасно обмениваться информацией между различными платформами от разных поставщиков и обеспечивать бесшовную интеграцию этих платформ без дорогостоящей и длительной разработки программного обеспечения.

Как видно из рисунка 2.10, OPC протокол состоит из серверной и клиентской части [19].

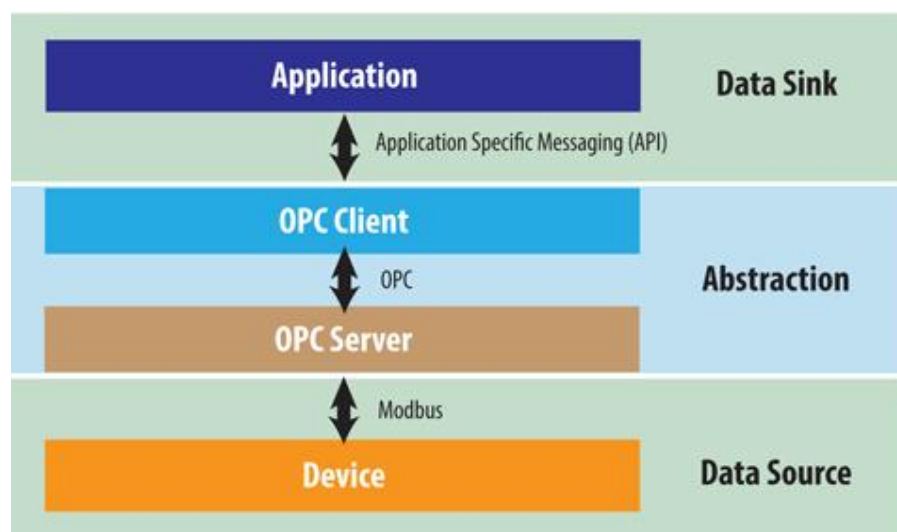


Рисунок 2.10 – Архитектура реализации OPC

В данном случае роль серверной части, предоставляющей данные состояний, является сама SCADA система на базе WinCC. В роли клиентской части OPC будет выступать разрабатываемая информационная система. Также

разрабатываемая информационная система будет реализовывать протокол верхнего уровня для транспорта данных во вне.

Фонд OPC является отраслевым консорциумом, который создает и поддерживает стандарты протокола OPC для открытой связи устройств и систем промышленной автоматизации. Также OPC Foundation выпускает, поддерживает и документирует стандарт и библиотеки, реализующие протокол OPC, в том числе для использования в C#.

OPC – набор спецификаций стандартов, каждый стандарт описывает набор функций определенного назначения. В нашем случае SCADA система упаковочно-транспортной линии является устаревшей и поддерживает устаревший стандарт OPC DA (Data Access). OPC DA описывает набор функций обмена данными в реальном времени с ПЛК, РСУ, ЧМИ, ЧПУ и другими устройствами.

Фактически OPC протокол позволяет получить доступ к данным конечного устройства, а именно набору тэгов и их текущим значениям. В данном случае в SCADA системе упаковочно-транспортной линии прописаны тэги данных, содержащие адресацию и типизацию данных управляющего линией контроллера, а протокол OPC позволяет обращаться к их значениям на верхнем уровне. Данный подход исключает вмешательство в логику работы управляющего контроллера.

OPC протокол также удобен тем, что самостоятельно инициирует сообщения об изменении данных, что позволяет получать данные в режиме реального времени.

### 2.3.2 Структура программного решения шлюза данных

Как было сказано ранее, программное обеспечение должно быть модульным, для обеспечения легкой миграции на аппаратную реализацию шлюза в будущем. Такой подход диктует разбить общее программное решение шлюза данных на отдельные составные части, как видно из рисунка 2.11:

- интерфейс для управления и отображения текущего состояния подключений и исключительных ситуаций;
- библиотека общения по протоколу OPC, реализующая спецификацию в необходимой степени и предоставляющая методы использования;
- библиотека транспорта данных во внешнюю систему.

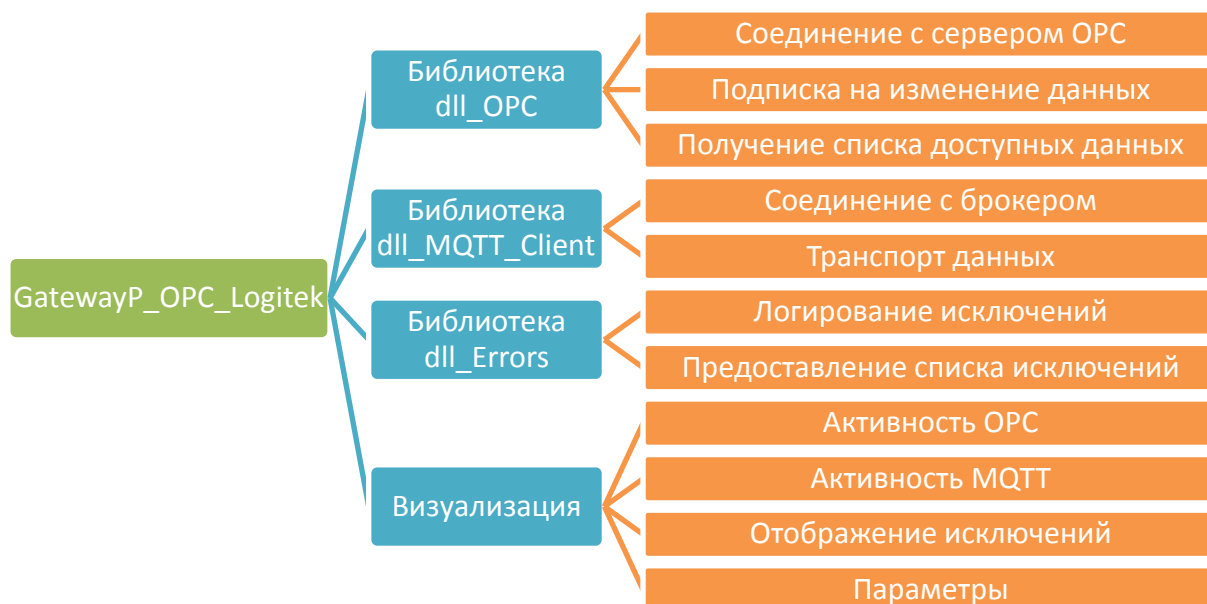


Рисунок 2.11 – Структура программного решения шлюза данных

Как видно из рисунка 2.11, программное решение шлюза данных использует подключаемые внешние библиотеки, а описание интерфейса и связи библиотек незначительны и собраны в одном месте. Для примера, подключаемая библиотека «dll\_Errors» будет использована другими частями информационной системы.

При миграции на аппаратную реализацию шлюза, будут использованы подключаемые библиотеки «как есть», изменения коснутся лишь визуализации.

### 2.3.3 Реализация библиотеки сбора данных по протоколу OPC

Исследования на реальной системе управления упаковочно-транспортной линией осложняются ее безостановочной работой. Фактически компьютер управления линией отключается для обслуживания только раз в год. Упаковочно-транспортная линия одна из самых критичных в цехе, нельзя

недооценивать риски, связанные с проведением исследований на рабочей системе. Принято решение клонировать систему управления в виртуальную среду, для начальных опытов этого будет достаточно.

Библиотека «dll\_OPC», как отдельный проект в среде Microsoft Visual Studio, состоит из двух классов и одного подключаемого класса отдельной библиотеки, как видно на диаграмме классов рисунка 2.12.

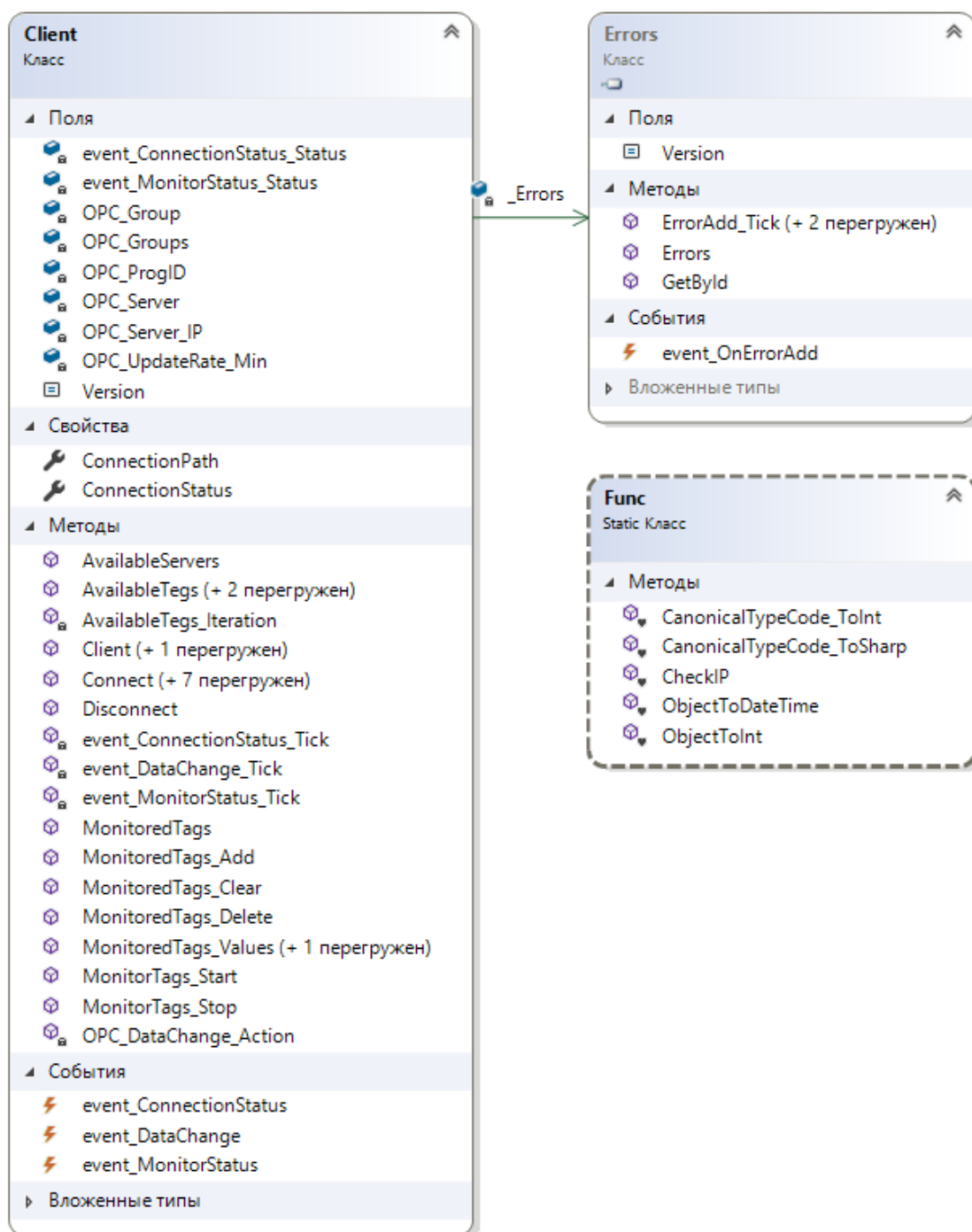


Рисунок 2.12 – Диаграмма классов библиотеки dll\_OPC

Основной публичный класс Client содержит основные методы общения по протоколу OPC. Внутренний статический класс Func предоставляет вспомогательные методы преобразования типов данных.

Основной класс Client реализует спецификацию OPC Foundation в необходимой мере, а именно:

- предоставление коллекции доступных серверов OPC;
- подключение к серверу OPC и отключение от него;
- предоставление коллекции доступных групп и тэгов OPC сервера;
- добавление, удаление и очистка коллекции наблюдаемых в реальном времени тэгов сервера OPC, своего рода подписка на события изменения;
- запуск и остановка потока наблюдения изменения состояния тэгов;
- инициализация события при изменении состоянии тэгов, позволяющее реагировать всем подписанным на событие классам.

Тэги OPC сервера имеют типизацию отличную от C#. Для преобразования типов и удобства дальнейшей обработки, вспомогательный класс Func содержит метод CanonicalTypeCode\_ToSharp(), частично приведенный во фрагменте кода 2.13.

```
internal static Type CanonicalTypeCode_ToSharp(Object Canonical)
{
    int res = CanonicalTypeCode_ToInt(Canonical);
    switch (res)
    {
        case 11:
        case 8203:
            return typeof(bool);
        case 17:
        case 8209:
        case 2:
        case 8194:
        case 8210:
        case 18:
        case 8195:
        case 3:
            return typeof(int);
        case 19:
        case 8211:
            return typeof(UInt32);
        case 21:
        case 8213:
            return typeof(UInt64);
        case 5:
```

Листинг 2.13 – Пример преобразования типов тэгов

Сама библиотека `dll_OPC` базируется на сторонней публичной библиотеке `OPCDAAuto` [23], подключаемой как ресурс. Для работы необходима регистрация библиотеки `OPCDAAuto` на целевой системе.

Основной класс `Client` библиотеки также реализует статические параметры, заложенные в коде как пределы допустимых значений, например минимальная частота опроса [10]. Для связи с внешними классами и основным кодом программы используются события, к примеру изменения статуса контроля тэгов, входящих данных, соединения с сервером.

Структура передаваемых в событиях данных типизирована отдельными классами-структурами, описывающими набор данных и их типы, а также методы пополнения коллекции. Жесткая типизация значительно снижает возможность совершения ошибки при программировании.

Листинг структуры классов и методов библиотеки представлен в приложении А, полный код не приведен ввиду его объемов.

#### 2.3.4 Реализация интерфейса шлюза данных

Основной функционал программы вынесен в отдельные библиотеки, это дает возможность пренебречь интерфейсом, так как при будущей миграции проекта на физическое исполнение шлюза данных визуализация будет не нужна или сведена к минимуму. Фактически программа призвана работать автономно, оператор не заметит и не увидит ее работы. Главная роль основного кода программы – инициализация, передача параметров и контроль подключаемых библиотек.

На этапе разработки важной ролью является отображение процессов и протекающих данных. Логирование потока данных, а также возникающих в ходе работы программы исключительных ситуаций реализовано в виде таблицы, обновляемой в режиме реального времени.

Интерфейс собран с использованием языка разметки XAML, что делает его разработку более гибкой, позволяя наделять элементы нетипичными для



них свойствами. Интерфейс представлен на рисунке 2.14. На рисунке 2.15 представлен рисунок, иллюстрирующий контекстное меню управления подключениями транспорта данных.

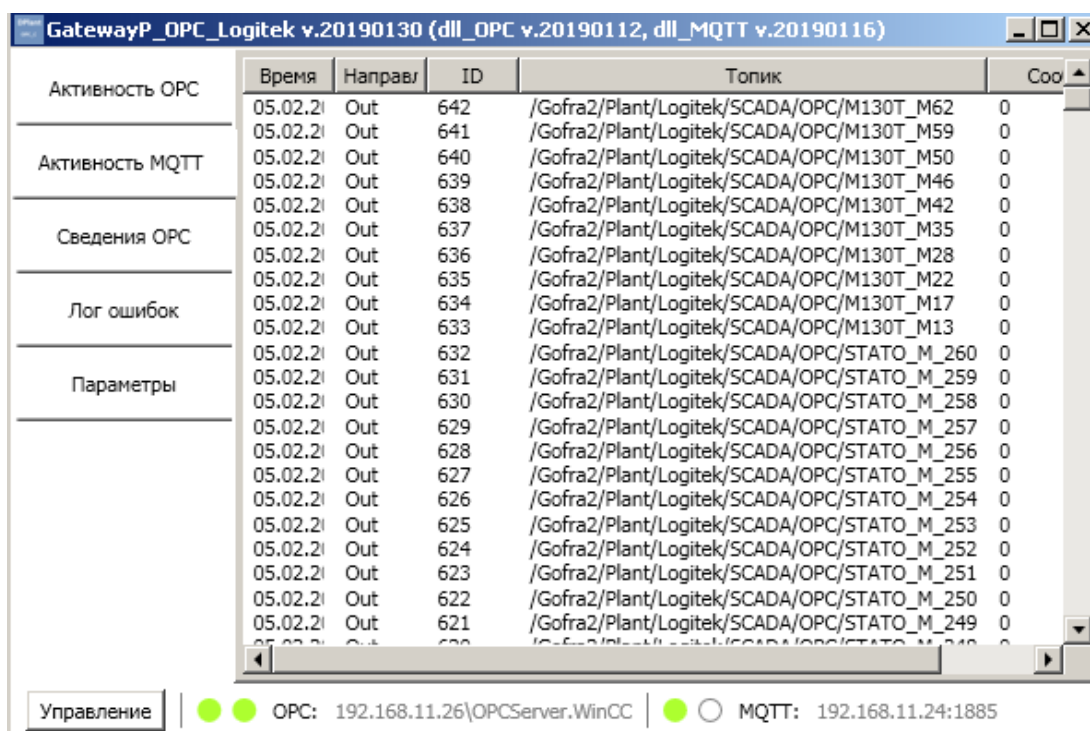


Рисунок 2.14 – Интерфейс программы – общий вид

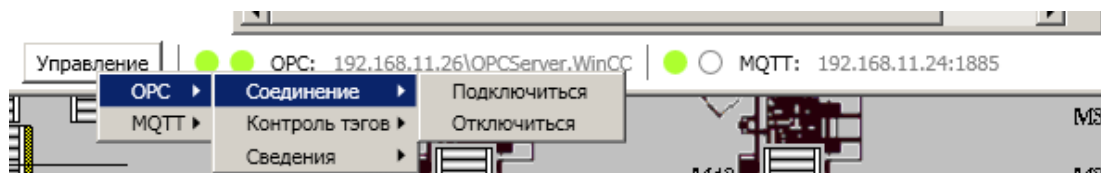


Рисунок 2.15 – Интерфейс программы – управление

Основной код программы довольно прост и имеет всего один класс MainWindow, инициализирующий возможности подключаемых библиотек. Управление возможно с помощью одноименной кнопки со всплывающим контекстным меню.

## 2.4 Реализация сервера или брокера данных

Назначение серверной части ИС – получение и обработка данных от шлюза данных, разработанного в предыдущем параграфе, и сохранение их в централизованной базе данных. С ростом количества шлюзов данных и объема

передаваемого трафика, разрабатываемая ИС будет испытывать высокие нагрузки обработки трафика, это требование ляжет на выбранный протокол транспорта данных.

### 2.4.1 Протокол транспорта данных

Протокол обмена данными на верхнем уровне должен удовлетворять следующим требованиям:

- современным;
- популярным с точки зрения поддержки разработчиков производственного оборудования;
- простым в реализации;
- быстрым и безопасным с точки зрения передачи информации;
- обеспечивать гибкость и масштабируемость всего решения.

Все эти качества относятся к протоколу MQTT [21]. Он относительно нов и стремительно набирает обороты. MQTT (Message Queue Telemetry Transport) – упрощённый сетевой протокол, работающий поверх TCP/IP, ориентированный для обмена сообщениями между устройствами по принципу издатель-подписчик, что отражено на рисунке 2.16.



Рисунок 2.16 – Протокол MQTT

Клиент как издатель может отправлять данные брокеру в определенную тему. Клиент как подписчик получает от брокера данные в соответствии со своей подпиской на определенный топик. Брокер руководит подписками к топикам и публикует новые данные. Адреса топика просты и наглядны:

/ компания / площадка / линия / узел / датчик / конкретное значение датчика

Протокол MQTT общепризнан как в бытовой автоматизации, так и производителями промышленного оборудования. Уже начали появляться готовые решения – шлюзы преобразования промышленных протоколов от компаний-гигантов MOXA и ICP DAS [14]. Это означает гарантированную поддержку и наличие устройств промышленного назначения, преобразующих любые протоколы нижних уровней в MQTT, унифицируя поток данных и упрощая структуру транспорта данных в целом.

Важным моментом, говоря о протоколе MQTT, является поддержка трех уровней качества обслуживания QoS:

- QoS 0 задействует стратегию «максимум однократная доставка сообщений». Приёмник сообщения не подтверждает их получение, отправитель, соответственно, передаёт сообщение лишь раз, не предпринимая попыток по их повторной передаче. Это – метод «отправил и забыл».
- QoS 1. Здесь применяется подход «минимум однократная доставка сообщений». Гарантируется, что приёмник получит сообщение хотя бы один раз. При этом подписчик может получить одно и то же сообщение несколько раз. А отправитель будет предпринимать повторные попытки отправки до тех пор, пока не получит подтверждение в успешной доставке сообщения.
- QoS 2. Этому уровню качества обслуживания соответствует самая медленная процедура доставки сообщений, но при этом он – самый надёжный. Его основная особенность – реализация стратегии

«однократная доставка сообщений». При его использовании применяется четырёхступенчатая процедура подтверждения доставки сообщений.

Подходящим подходом наших задач является уровень 0 качества обслуживания. Цифровизация в текущем контексте не оказывает влияния на производственные процессы, значит частичная потеря данных лишь снизит качество аналитических данных и не окажет пагубного влияния на производство, следовательно, можно пренебречь качеством данных ради скорости обслуживания.

#### 2.4.2 Структура серверного решения

Серверная часть с точки зрения общего решения является брокером MQTT, с последующим хранением поступающих данных. На этом уровне абстракции от конечных устройств и шлюзов данных фильтрация потока данных отсутствует. Вместо фильтрации данных, структура базы данных позволяет задать для каждого шлюза данных и каждого его тэга подход к хранению данных:

- перезапись последнего входящего состояния тэга, так называемая таблица текущих состояний;
- запись и фиксация по времени каждого входящего значения, так называемая живая таблица данных.

При реализации таблицы живых данных, возникает вопрос об объемах поступающей информации – необходимо контролировать объемы информации в соответствии с требованиями пользователя. К примеру, данные о перемещении паллет продукции по наземным транспортерам могут быть востребованы для анализа в течении 5 дней, тогда как температурные показатели конечных устройств стоит хранить дольше. Решение нашлось довольно простое и полностью соответствует концепции гибкости разрабатываемого решения – для каждого шлюза данных и отдельного тэга указывается срок хранения данных в днях. Сервер раз в сутки стартует процесс

нормализации базы данных и чистит устаревшие данные. Структура базы данных будет более подробно освещена в последующем параграфе.

Серверная часть по своей структуре и функционалу декомпозирована на составляющие, представленные на рисунке 2.17.

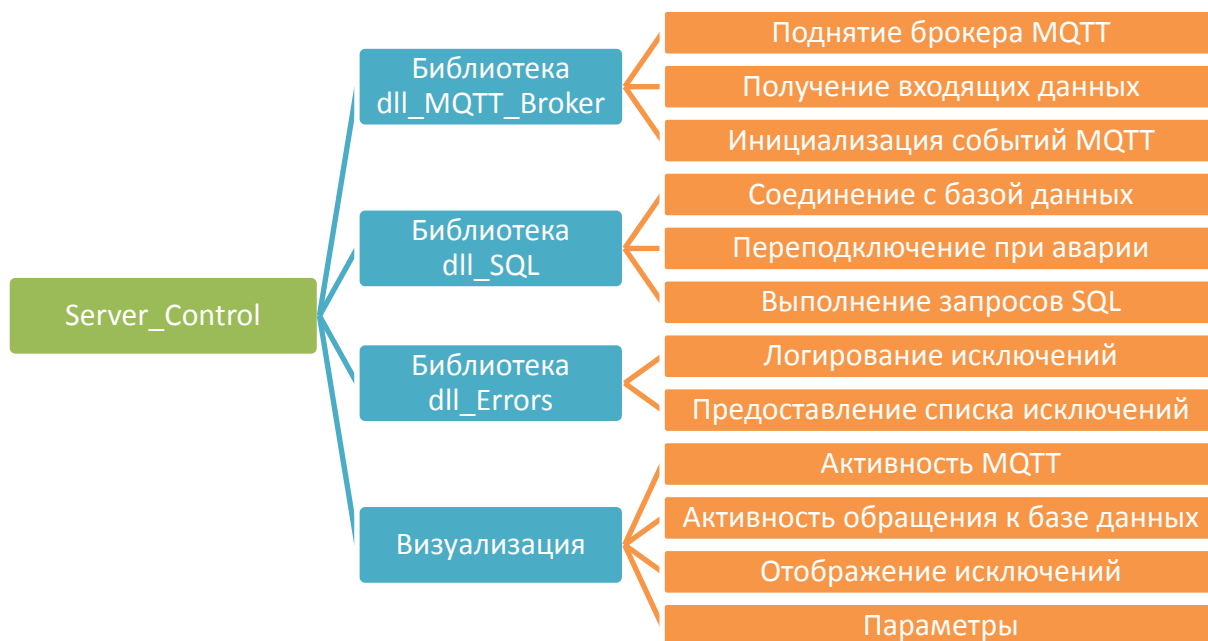


Рисунок 2.17 – Структура программного решения

Стоит отметить, что библиотека `dll_Errors` одинакова как для данного решения, так и для решения шлюза данных, описанного в предыдущем параграфе. Модульный подход способствует сокращению объема кода и большей управляемости.

### 2.4.3 Реализация библиотек MQTT транспорта данных

Библиотека `dll_MQTT_Broker` как отдельный проект реализуется единым публичным классом `Broker` и в достаточной мере реализует стороннюю библиотеку `GrantMQ.Net` [22], а именно:

- поднятие брокера с определенными параметрами;
- запуск и остановка брокера по требованию;
- предоставление событий активации или остановки брокера;
- предоставление событий подключения или отключения клиента MQTT;

- предоставление события входящих данных MQTT.

Листинг структуры классов и методов библиотеки представлен в приложении Б.

Клиентская библиотека `dll_MQTT_Client`, использованная в предыдущем параграфе, обратна брокеру и базируется на другой сторонней библиотеке `M2Mqtt.Net` [23]. Ее роль обратна – подключение и отправка сообщений брокеру.

Здесь пришлось столкнуться с одной особенностью сторонней библиотеки – она использует `.Net Framework` версии 4.5, что не поддерживается устаревшей операционной системой `Windows XP`. Возможность обновления операционной системы на компьютере управления транспортной линией `Logitek` отсутствует, поэтому принято решение детальнее изучить и внести изменения в исходный код сторонней библиотеки `M2Mqtt.Net`.

Оказалось, требование `.Net Framework` версии 4.5 обусловлено поддержкой возможности шифрования `TLS` версий 1.1 и 1.2. при подключении клиента к брокеру, что видно из фрагмента кода 2.18.

```
/// <summary> MQTT SSL utility class
public static class MqttSslUtility
{
    public static SslProtocols ToSslPlatformEnum(MqttSslProtocols mqttSslProtocol)
    {
        switch (mqttSslProtocol)
        {
            case MqttSslProtocols.None:
                return SslProtocols.None;
            case MqttSslProtocols.SSLv3:
                return SslProtocols.Ssl3;
            case MqttSslProtocols.TLSv1_0:
                return SslProtocols.Tls;
            case MqttSslProtocols.TLSv1_1:
                return SslProtocols.Tls11;
            case MqttSslProtocols.TLSv1_2:
                return SslProtocols.Tls12;
            default:
                throw new ArgumentException("SSL/TLS protocol version not supported");
        }
    }
}
```

Листинг 2.18 – Метод перечисления шифрования сторонней библиотеки

Отключив поддержку данных версий шифрования, появляется возможность перекомпилировать стороннюю библиотеку под .Net Framework версии 4.0, последней версии поддерживаемой Windows XP.

Листинг структуры классов и методов библиотеки представлен в приложении В.

#### 2.4.4 Реализация библиотеки обращения к базе данных

Данная библиотека разрабатывалась для удобства использования и унификации обращения к базе данных из серверного и пользовательского приложений.

Библиотека реализует следующие возможности:

- подключение и отключение от базы данных, с переданными при инициализации библиотеки параметрами;
- перегруженный метод выполнения SQL запроса без ответа;
- перегруженный метод выполнения SQL запроса с ответом типа DataTable.

Особенностью библиотеки является возможность автоматического восстановления соединения с базой данных после сбоя. Фактически, метод подключения приватен и вызывается библиотекой самостоятельно при первом обращении.

Листинг структуры классов и методов библиотеки представлен в приложении Г.

#### 2.4.5 Реализация интерфейса серверной части

Серверная часть программы задействует библиотеки общения по MQTT и обращения к базе данных. Активность также отражается на одноименных вкладках, при условии, что программа не является свернутой в системную область уведомлений.

Внешний вид серверной части представлен на рисунке 2.19.

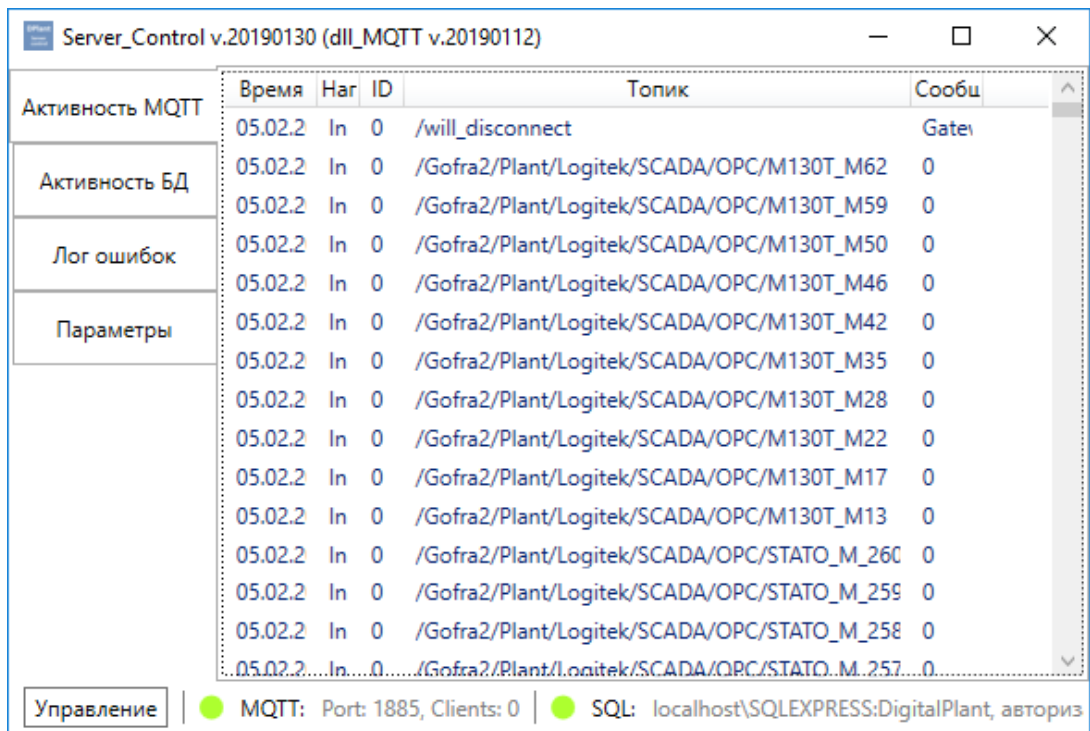


Рисунок 2.19 – Внешний вид серверной части

Все управление осуществляется через одноименную кнопку и содержит меню, представленное на рисунке 2.20.

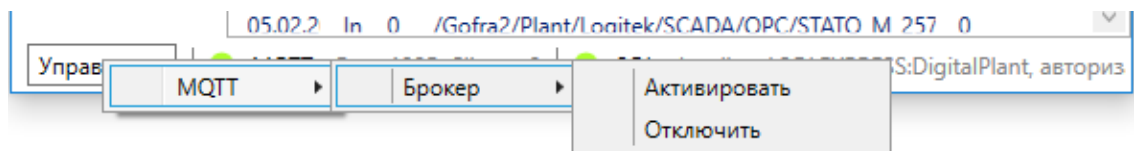


Рисунок 2.20 – Меню управления серверной части

Управление SQL соединением не требуется, так как оно поднимается автоматически по факту необходимости, а отключение от SQL сервера происходит в момент деструкции формы.

В дальнейшем планируется уйти от пользовательского интерфейса программы и мигрировать код в тип системной службы. Это позволит серверной части работать без необходимости входа в систему, что довольно удобно и важно с точки зрения безопасности при использовании серверов. Но, на этапе отладки, решение в виде отдельной программы позволит более



тщательно контролировать информационные потоки и возникающие исключения.

#### **Вывод по параграфу 2.4**

Результатом параграфа является разработка библиотек реализации протокола MQTT и библиотеки упрощенного обращения к серверу баз данных. Тестирование серверной части в виртуальной среде показало хорошие показатели по производительности и минимальные по потребляемым ресурсам.

На данном этапе данные с нижнего уровня уже попадают в базу данных и представляют собой ценность для аналитики. Остается лишь предоставить конечным пользователям интерфейс обращения к данным и нормализовать структуру базы данных.

### **2.5 Предоставление данных конечным пользователям**

Политика развития предприятия способствует обновлению парка компьютеров и операционных систем. Это означает возможность использования последних версий .Net Framework, что снимает множество ограничений. Становится возможным реализовать инновационные методы предоставления информации конечным пользователям.

Для начала определим два направления, два вида представления, по которым пользователь сможет получить данные:

- предоставление отчетности в форматах PDF и MS Excel;
- визуализация по типу цифрового двойника.

Также определим основные требования к интерфейсу – это гибкость и удобство использования.

#### **2.5.1 Предоставление отчетности**

Текущая задача сводится к извлечению информации из базы данных и ее представления в удобном читабельном виде. При этом конечный пользователь

должен иметь возможность сохранить данные как отдельный документ на локальном компьютере.

По удобству восприятия информации среди прочих лидирует формат документа PDF. Документы такого формата читабельны, помимо текста могут содержать изображения и графики, формат поддерживается большинством устройств.

Основной недостаток PDF – на данные документа нельзя ссылаться из другого документа. В файлах формата Excel становится возможным использование связей ссылочного типа сторонними документами. Также стоит выделить третий формат данных – CSV, использующий табличные данные по разделителю. Формат CSV отбрасывает форматирование Excel формата и содержит только сами данные. Это полезно в задачах импорта данных в сторонних системах.

Подход, когда формирование отчетности предопределено в программном коде, удобен для разработчика, но не отвечает условиям гибкости программного решения. Лучшее что можно сделать – предоставить конечному пользователю, точнее администратору программного решения, возможность добавлять новые отчеты самостоятельно.

На рынке существует готовое решение данной задачи – потоковое создание отчетов по требованию на основании программируемых документов-шаблонов. Решение Crystal Reports от компании SAP. Сами отчеты Crystal Reports создаются на одноименной платформе, либо в среде Visual Studio с установленным дополнением разработчика от SAP.

Создадим первый отчет Crystal Reports в среде Visual Studio. Роль отчета – вывести список простоев линии Logitek по причине не съёма продукции с хвостовых транспортеров линий переработки. На предыдущих этапах разработки, на стороне сервера, фильтр входящих по MQTT данных формировал коллекцию фактического наличия продукции на наземных транспортерах по линиям переработки. К примеру, если после линии переработки и перед транспортной телегой линии упаковки доступно 5

транспортеров и в один момент все 5 транспортеров заняты продукцией – это есть начало простоя, т.к. линия переработки парализуется по отсутствию места схода продукции. При высвобождении хотя бы одного транспортера – линия вновь может выпускать продукцию – окончание простоя. Эти данные попадали в отдельную таблицу базы данных, эти данные и необходимо будет использовать для формирования отчета.

Для начала необходимо создать новый отчет Crystal Reports и определить источник данных, простым указанием таблицы базы данных, как показано на рисунке 2.21.

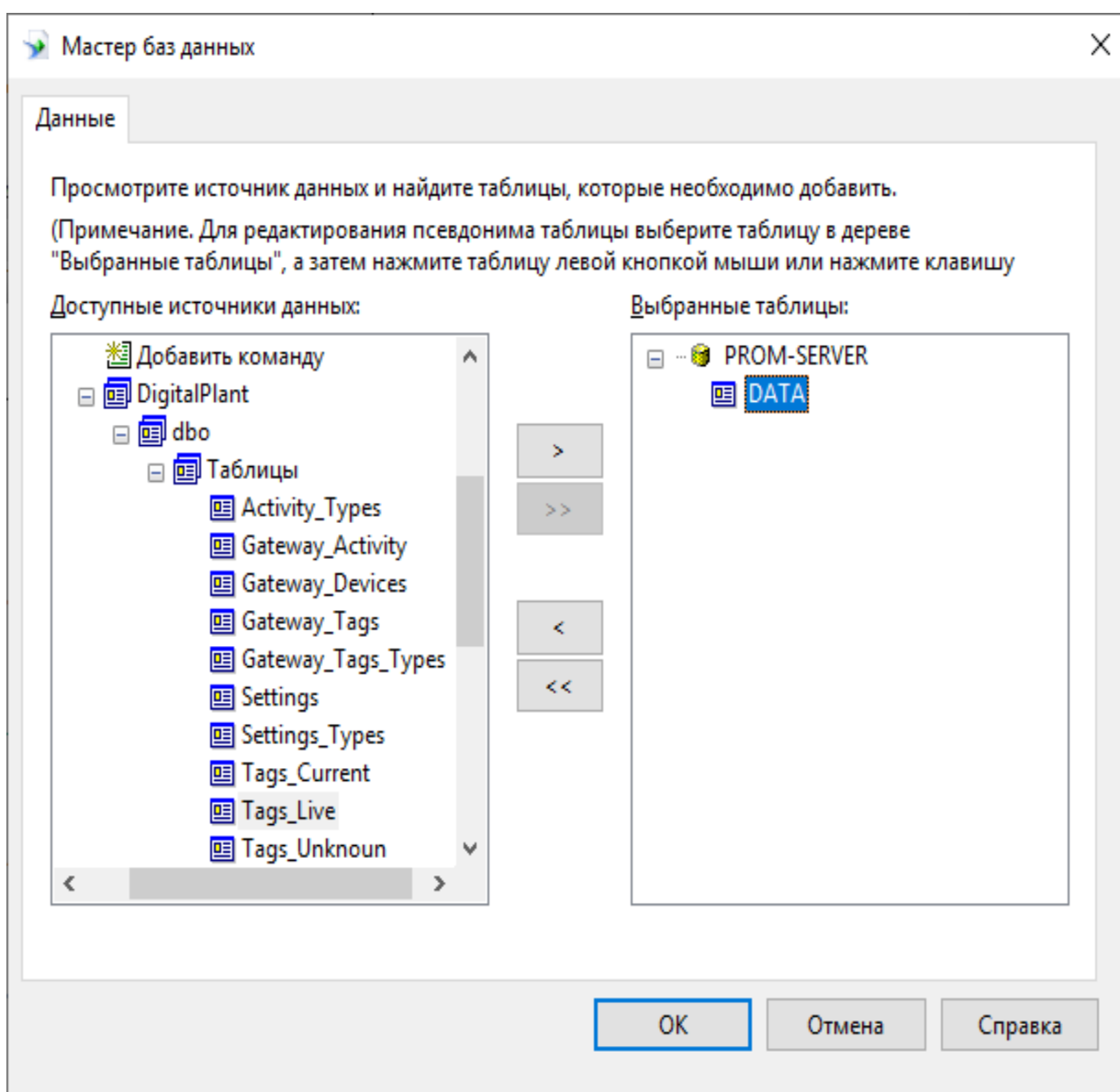


Рисунок 2.21 – Источник данных шаблона отчета

Теперь в отчете доступны поля таблицы базы данных с соблюдением их типов. В самом простом варианте доступные поля данных достаточно просто переместить в определенную зону отчета, добавить заголовок и немного отформатировать внешний вид отчета. Доступные поля данных и форматирование шаблона отчета представлен на рисунке 2.22.

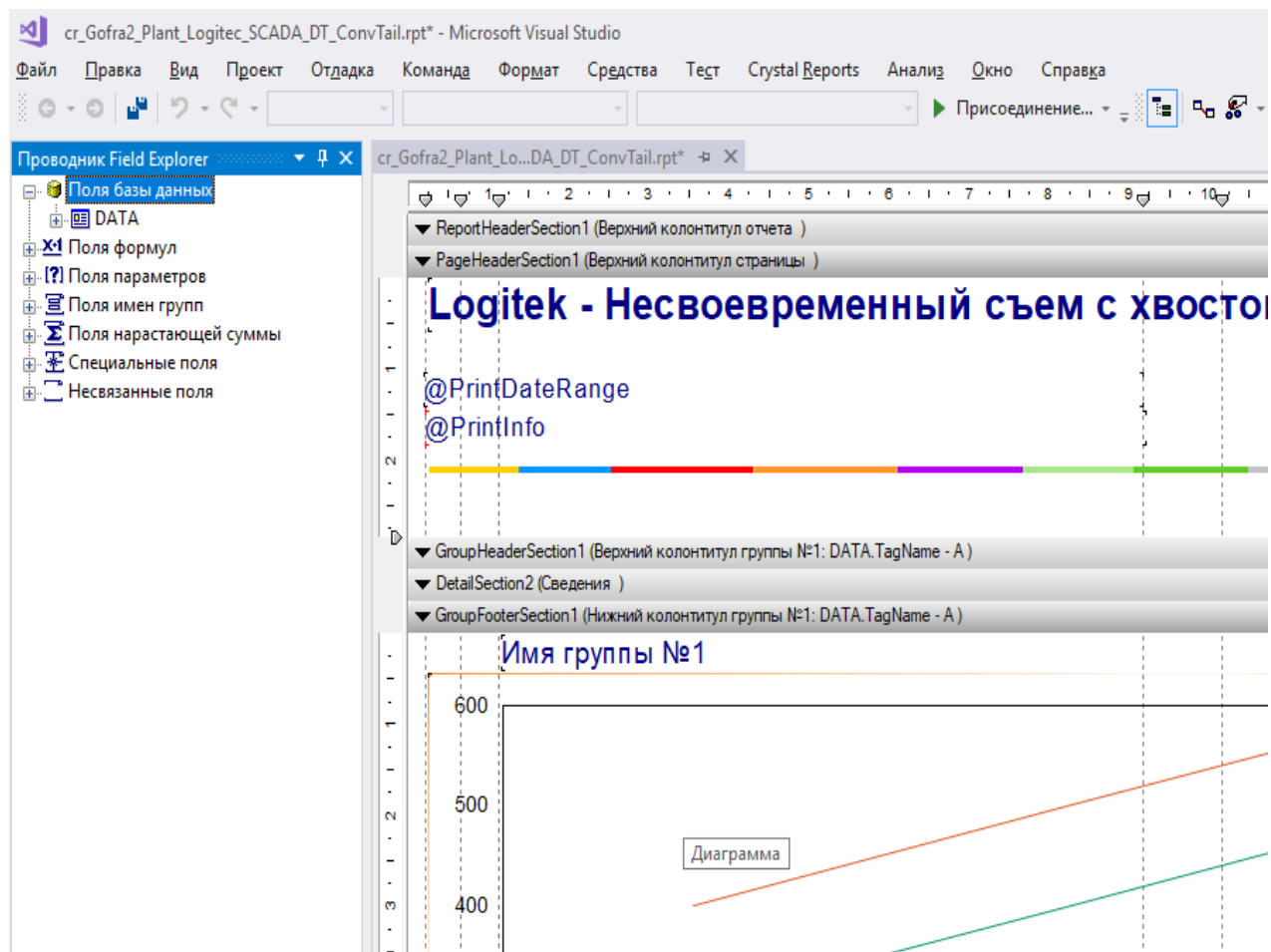


Рисунок 2.22 – Пример создания шаблона отчета

Созданный отчет сохраняется в заранее определенном месте и с определенным именем. Остается только указать в базе данных наименование, комментарий и наименование файла отчета для его отображения на форме. Меню формы строится динамически на основании базы данных, отчеты распределяются по группам, для удобства пользователя. Конечный вид отчета, сформированный нашей программой, отражен на рисунке 2.23.

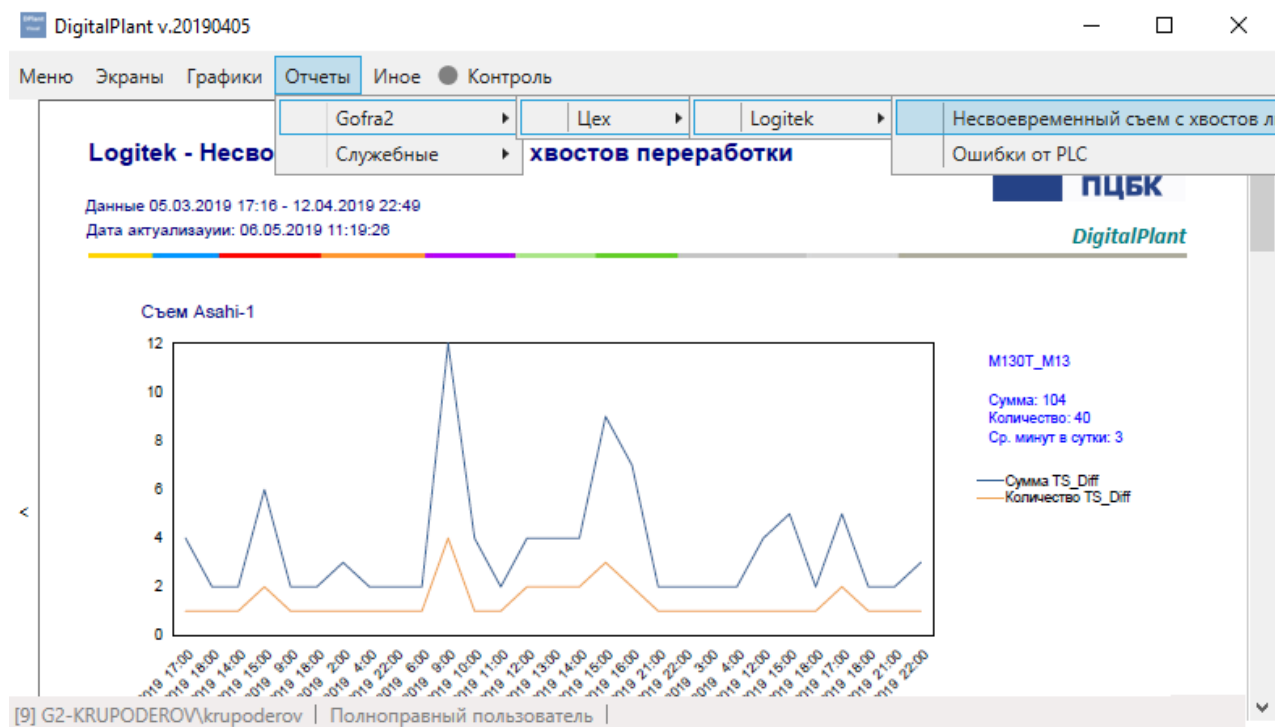


Рисунок 2.23 – Пример сформированного отчета

Объект формирования и отображения шаблонов отчетов делает большинство работы за нас – уже доступны функции печати и сохранения сформированного отчета. При этом перечисленные выше необходимые форматы документа при сохранении поддерживаются по умолчанию.

Единственным недостатком такого подхода является формирование отчета на стороне конечного пользователя, что означает необходимость установки на локальной машине пользователя довольно объемного пакета SAP Crystal Reports Runtime Engine. В дальнейшем запланирован перенос формирования отчета на сторону серверной части.

Пример последней страницы отчета представлен в приложении Д. На нем наглядно отражена частота появления простоя по причине несвоевременного снятия продукции с хвостовых транспортеров, а также приведены круговые диаграммы частоты и длительности простоев в разрезе перерабатывающих линий.

## 2.5.2 Визуализация данных по типу Цифрового двойника

Цифровой двойник – виртуальное представление физической системы. Представление данных по типу цифрового двойника обладает исключительной наглядностью и восприятием данных.

Упаковочно-транспортная линия цеха предоставляет следующий объем данных:

- факт наличия паллеты продукции на определенном наземном транспортере;
- состояние пропускающих светофоров, блокирующих движение паллет продукции;
- факт отключения оборудования и наземных транспортеров;
- состояние готовности к работе наземных транспортеров;
- счетчики приоритетов хвостовых транспортеров линий переработки, отвечающих за очередность забора паллет продукции на транспортную телегу;
- возникновение ошибок, определенных управляющим контроллером линии.

При этом начальнику смены потребуется контролировать перемещения продукции и выявлять заторы, а ремонтному персоналу наоборот потребуется информация о состоянии устройств. Необходимо иметь возможность отобразить пользователю только те данные, которые необходимы в данный момент и для решения данной задачи. Это позволит пользователю сфокусировать внимание на конкретной проблеме, а также значительно снизит нагрузку систем и объем потока данных.

Для реализации вывода информации в пользовательский интерфейс определены новые термины:

- Экран – элемент меню, содержащий ссылку на определенное изображение цеха, узла линии или иного пространства;

- Слои – наборы Элементов с указанием их смещений по осям X и Y, размерами, типами данных и прочей информацией необходимой для прорисовки Элементов на Экране;
- Элемент – единица источника данных, один тэг одного устройства.

Т.е. Экраны содержат Слои, которые в свою очередь содержат Элементы данных. Подобный подход стал возможен благодаря продуманной структуре базы данных и ее связей, приведенной в следующем параграфе.

К примеру, для отображения продукции цеха на упаковочно-транспортной линии используется картинка карты цеха, внешний вид и доступные Слои (справа вверху) представлены на рисунке 2.24.

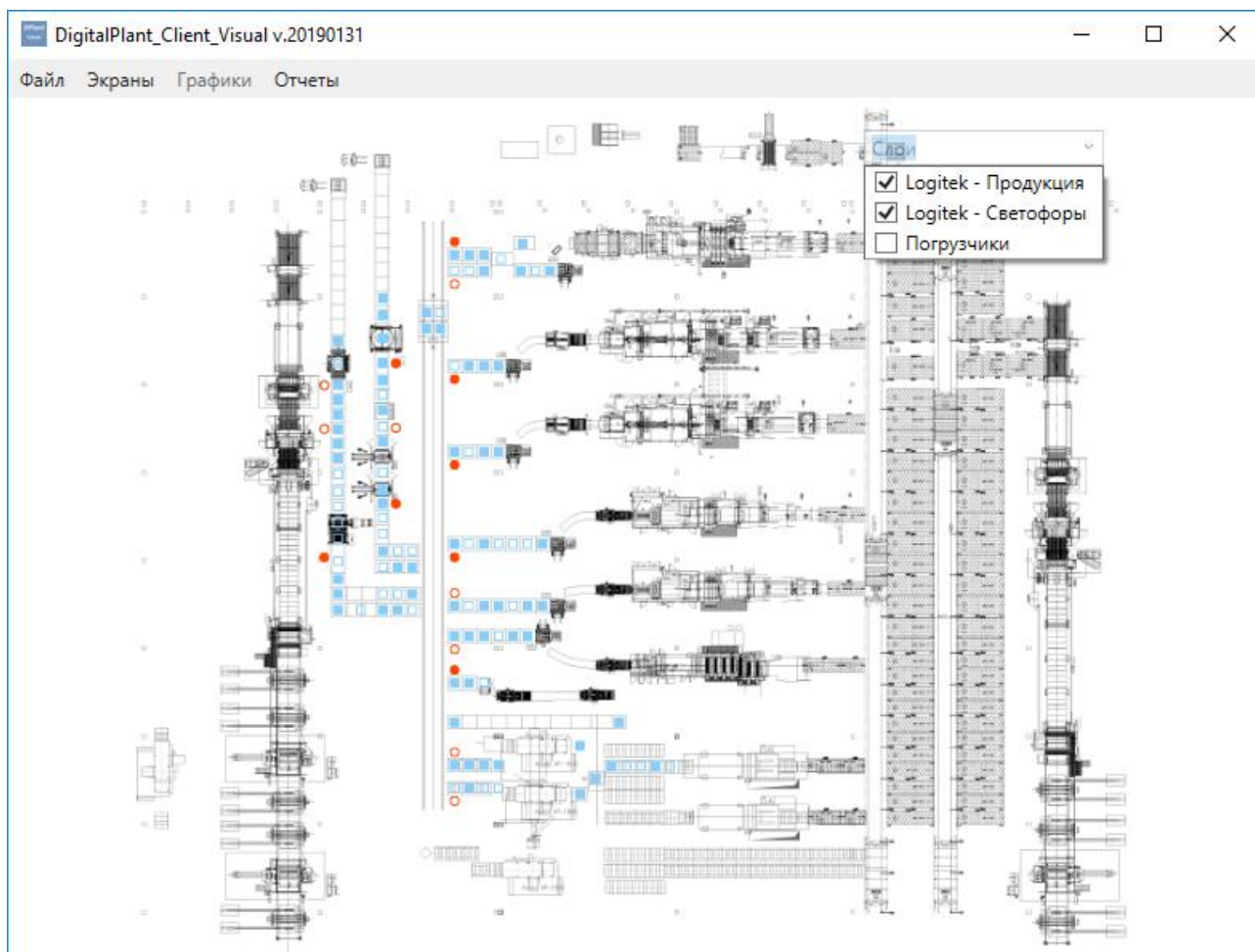


Рисунок 2.24 – Пример Экрана цеха

Для отображения Элементов используются примитивы, определенные для каждого Элемента в таблице определенного Слоя. В данном случае

закрашенный синий квадрат означает наличие паллеты продукции на этом наземном транспортере, пустой синий квадрат – отсутствие паллеты продукции, а красные эллипсы – состояние светофоров, блокирующих выпуск продукции на определенном участке. При выборе или отключении отдельных Слоев, набор Элементов данных появится или исчезнет с Экрана.

Элементы также могут быть представлены в виде цифровых значений. Для примера выберем другой тестовый Экран, на котором будут отражены температурные показатели пультов управления линий производства и активность сетевого оборудования, на рисунке 2.25.

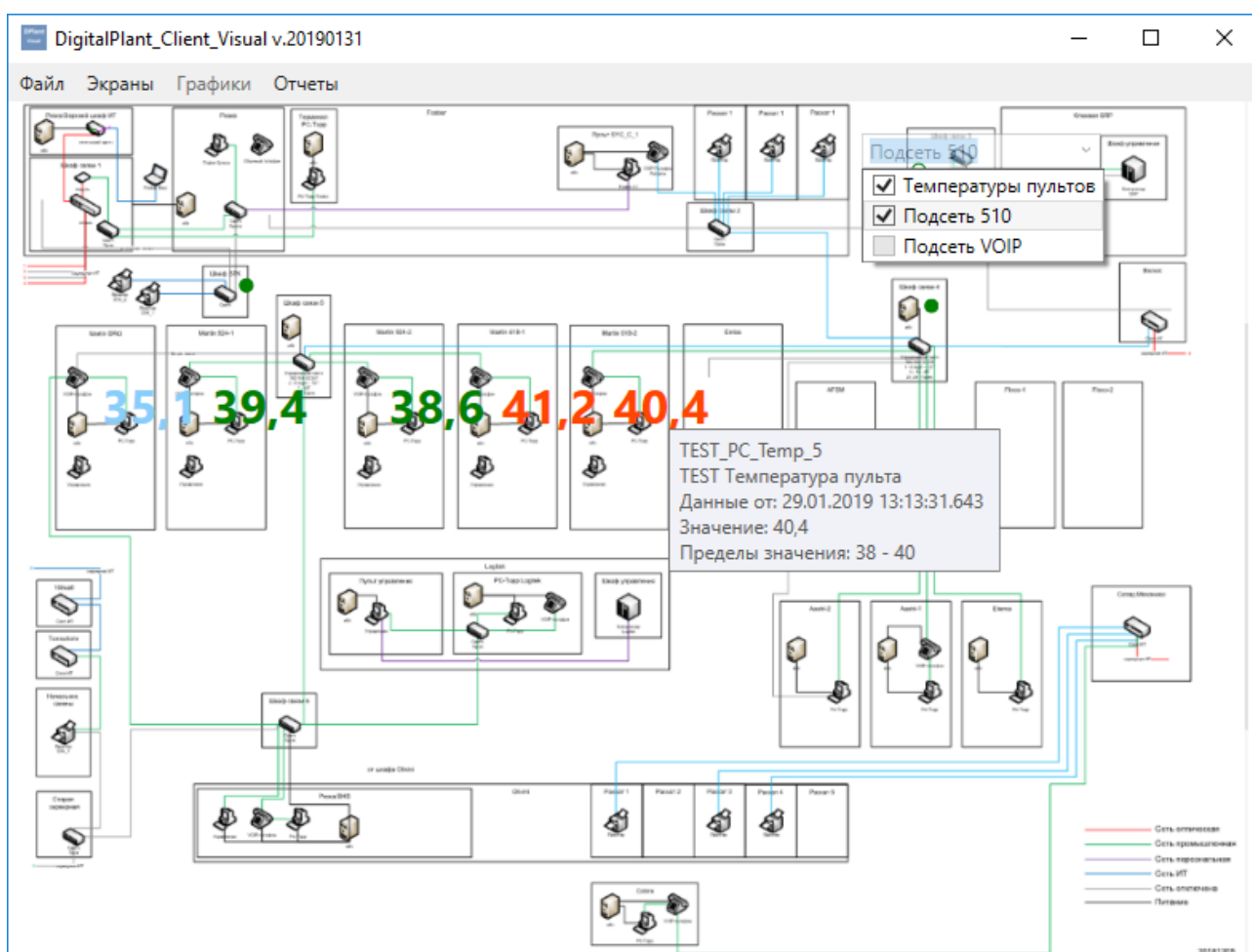


Рисунок 2.25 – Пример Экрана температурных показателей

Элементы данных позиционируются на Экране по осям X и Y, predetermined при добавлении Элемента в информационную систему. При этом позиция Элемента данных зависит не только от исходных статических



значений, но и от коэффициента изменения размера изображения Экрана и его положения на форме программы. Позиция Элементов также пересчитывается автоматически при изменении размеров формы приложения. Благодаря этому становится возможным использовать файлы изображения Экранов больших размеров и пользовательские мониторы разного формата без искажений прорисовки Элементов.

Стоит заметить, что для цифровых показателей уже доступны пределы значений и их цветное изменение на Экране относительно текущего показания. С дальнейшим развитием системы будет добавлена анимация для большей концентрации внимания. Также, если навести мышкой на Элемент, можно увидеть дополнительную информацию по Элементу, включая его предельные значения, время последнего обновления данных, его подробное описание и назначение.

Количество Слоев не ограничено и не привязано к определенному шлюзу данных. Каждый Слой каждого Экрана персонализирован по типу представления информации и может содержать Элементы нескольких шлюзов данных. К примеру, один Слой может отображать перемещение продукции всего цеха, собирая информацию с разных шлюзов данных.

Визуализация данных по типу цифрового двойника позволяет наглядно оценить технологические процессы производственного оборудования, выявить его неэффективное использование или нарушение регламента, своевременно реагировать и принимать организационные решения. Для примера, становится возможным выявить затор продукции на упаковочно-транспортной линии и скорректировать распределение мощностей производственных линий. Своевременное обнаружение технических неисправностей или отключения оборудования позволяет сократить простои оборудования, наглядно оценить масштаб в режиме реального времени.

### 2.5.3 Доступ к архивным данным

Фактически это система реального времени, задержка обновления данных Экранов задана искусственно как 3 секунды, ввиду объема данных. Уже возможно наблюдать реальное положение в цехе, находясь за рабочим местом.

Система разрабатывалась в виртуальной среде и, думая как на презентации показать изменения данных, появилась инновационная идея – возможность отобразить на Экране исторические данные. Инновационной идеей является благодаря подходу позиционирования всех Элементов Экрана относительно одного Элемента.

Данный функционал позволяет наглядно оценить, к примеру, положение автопогрузчиков цеха в момент затора транспортной линии, или состояние хвостовых транспортеров линий в момент блокировки светофоров, определить причину затора продукции.

Фактически каждый Элемент встречается в таблице живых данных множество раз, по факту его изменения. При каждой записи в базу данных фиксируется его текущее время вхождения, причем время берется со стороны сервера. В режиме реального времени из базы данных выбираются последние вхождения всех Элементов в базе живых данных. В процессе возврата в прошлое необходимо остановить потоки реального времени, выяснить временную метку выбранного Элемента и выбрать из живой базы состояния всех Элементов до этой временной метки.

На рисунке 2.26 выбран Элемент и все остальные Элементы представлены относительно него. Выбранный Элемент подсвечивается красной рамкой, что означает переход отображения в режим просмотра архивных данных. Внизу формы появляется временная шкала, от первого вхождения выбранного Элемента в базу до последнего. Стрелками возможно управлять выбором текущей предельной временной метки выборки из базы данных.

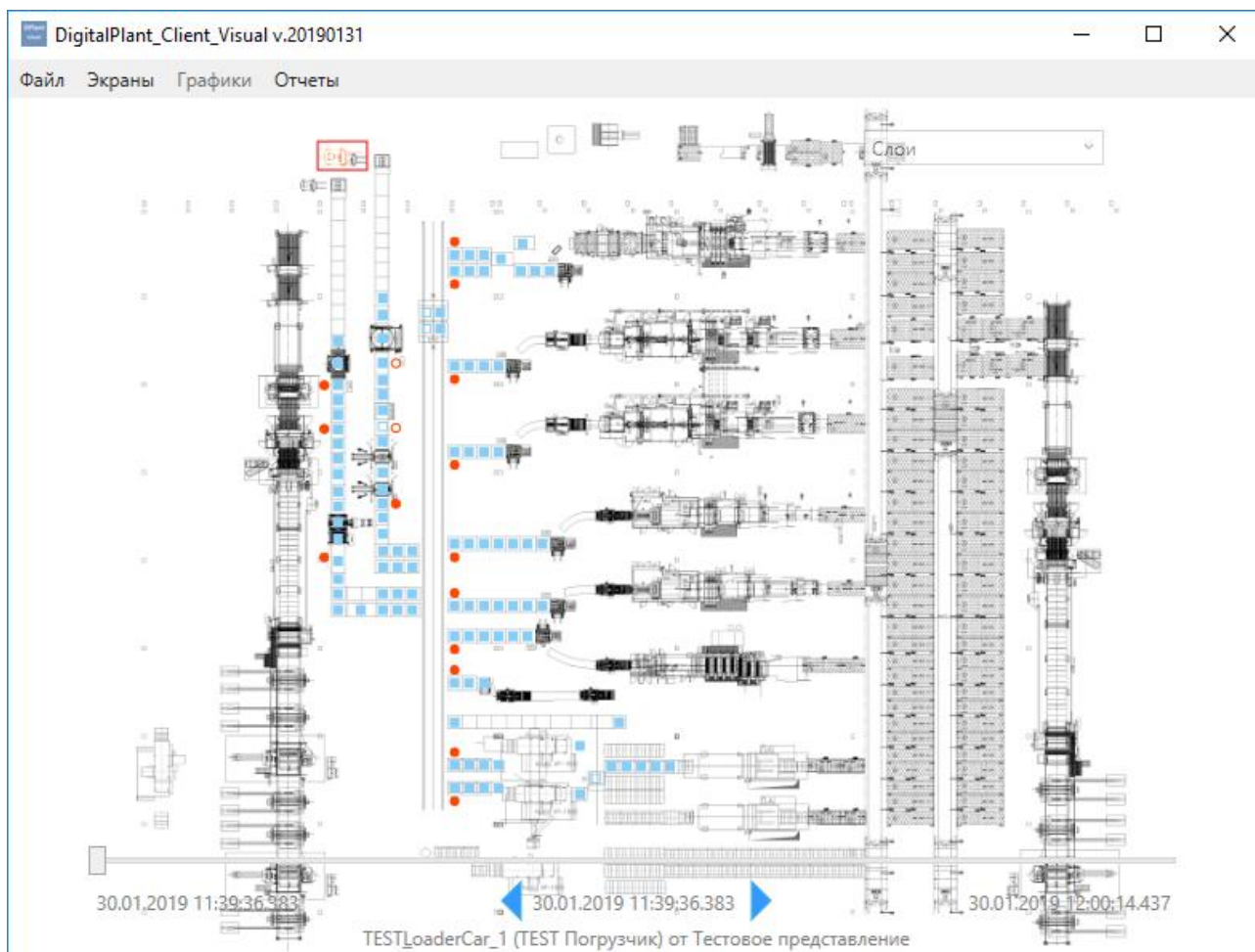


Рисунок 2.26 – Пример возврата во времени

Как отражено, все Элементы сменили свое состояние относительно выбранной временной метки. Таким образом возможно позиционироваться относительно любого Элемента Экрана. Сброс и возврат к отображению в реальном времени осуществляется двумя щелчками по любому Элементу или пустому пространству Экрана.

Довольно много идей, еще больше потенциальных возможностей. Пример нереализованного функционала может быть:

- постановка Элемента на контроль пользователем и возможность получения оповещений при переходе его пиковых значений;
- увеличение и уменьшение картинка Экрана роликом мышки, т.к. карты могут быть довольно большими;

- отображение графика изменения Элемента в режиме позиционирования и просмотра исторических данных на Экране;
- создание задачи обслуживания для системы TOIR прямо из представления Экрана выбрав нужный Элемент.

### **Вывод по параграфу 2.5**

На данном этапе разработки пользовательская часть системы уже обладает инновационным подходом к отображению данных. Задача реализации по типу цифрового двойника выполнена.

Конечному пользователю предоставлен доступ к отчетности, а также возможность видеть текущее состояние показателей цеха в реальном времени. При этом возможность каждому пользователю выделить из всего массива только нужную информацию, позволит сконцентрироваться на решении конкретных задач.

Динамическое построение формы, от меню до набора данных, позволит администратору системы добавлять новые шлюзы данных, Экраны и отчеты довольно простым способом. А алгоритм позиционирования элементов на картинке Экрана позволяет масштабировать представление без видимых дефектов и смещений на картинке.

Возможность возврата к прошлым показателям относительно любого Элемента Экрана дает возможность выявить причину возникновения проблемы также, как будто наблюдается состояние Элементов в реальном времени.

### **2.6 Возможности интеграции с другими системами**

В целом система открывает возможности сбора технических и технологических данных в реальном времени. Эти данные имеют ценность не только как отображение для конечного пользователя, но и для других информационных систем предприятия.

### 2.6.1 Связь информационной системы с внешними системами

Возможности интеграции ограничены лишь воображением и целесообразностью.

К примеру, если на предприятии есть действующие системы учета неисправностей, становится возможным автоматически инициировать новую задачу автоматически, основываясь на предельных значениях поступающих данных. В отличие от обнаружения неисправностей ручными методами автоматизированный подход имеет ряд преимуществ:

- неисправность фиксируется на ранней стадии, до момента, когда неисправность становится видна человеком;
- данные основываются на датчиках и более достоверны, исключается человеческий фактор;
- при инициализации неисправности становится возможным передать внешней системе дополнительную информацию, например, динамику показаний.

Интеграция с системами оповещения, таких как Zabbix, позволит информировать конечных пользователей и ремонтные службы средствами E-mail и SMS уведомлений. Это значительно повысит время реакции на проблему, еще в ее начальной стадии.

В будущем собранные данные станут хорошим источником для питания систем имитационного моделирования, дополненной реальности и анализа Больших Данных.

### 2.6.2 Структура базы данных информационной системы

База данных как основной источник данных для интеграции с другими системами имеет простую и понятную структуру. Структура отображена на рисунке 2.27.

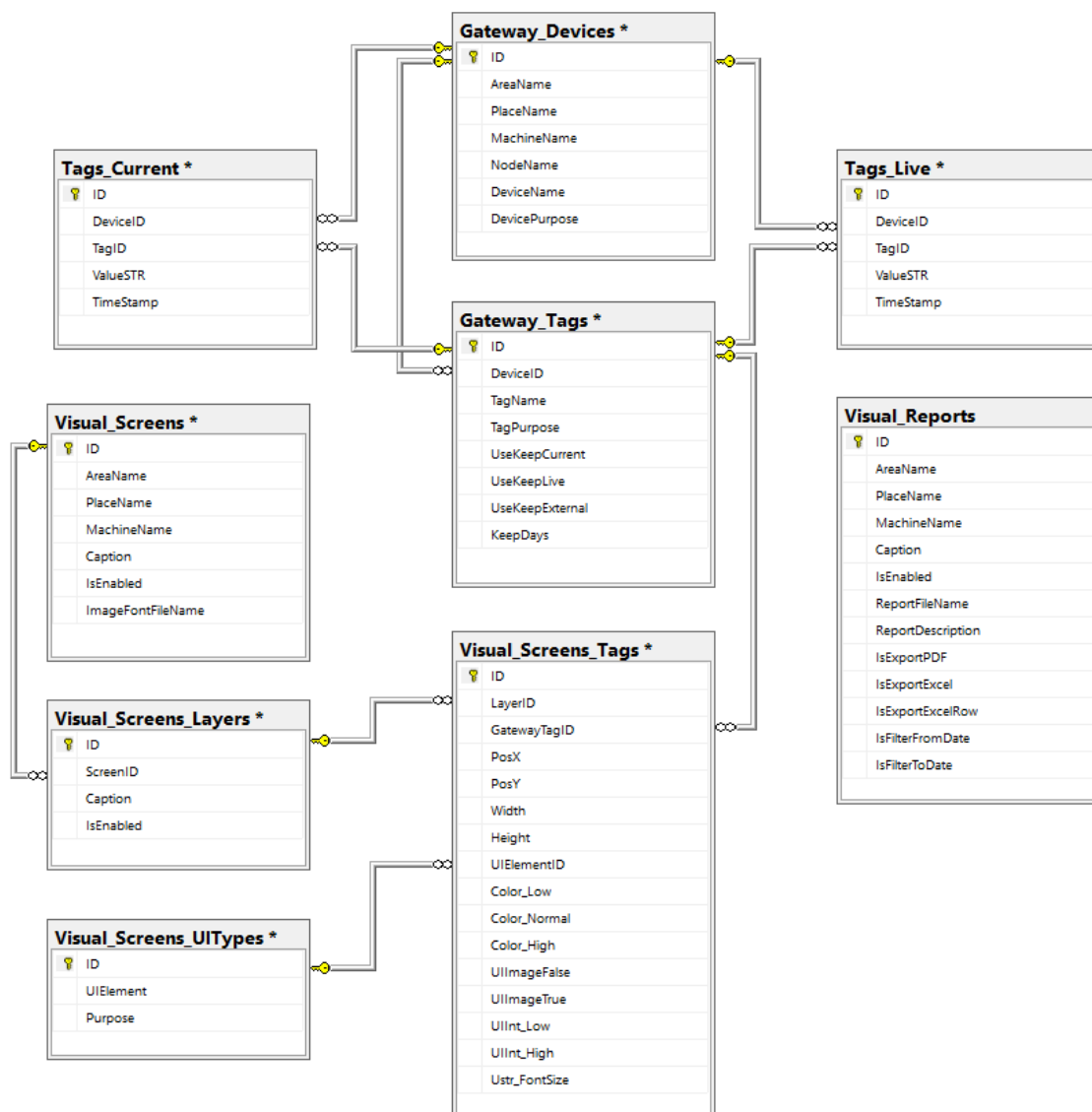


Рисунок 2.27 – Структура базы данных

Для добавления нового устройства шлюза данных используется таблица Gateway\_Devices с указанием расположения устройства (производственная площадка -> цех или подразделение -> производственная машина -> узел машины). Также указываются все тэги нового устройства в таблице Gateway\_Tags. Для каждого тэга указываются способ и срок хранения данных.

Таблицы Tags\_Current и Tags\_Live служат для хранения последних значений тэга и всех входящих значений тэга соответственно.

Для отображения данных в пользовательском приложении используются таблицы Visual\_Screens и Visual\_Reports. На основании этих таблиц формируется меню экранов и отчетов соответственно.

В случае с таблицей отчетов, таблица имеет ссылку на документ шаблона отчета Crystal Reports. Дальнейшие данные для отчета и их извлечение указано внутри самого файла шаблона отчета.

Таблица Visual\_Screens\_Layers содержит слои экрана по идентификатору последнего. Таблица Visual\_Screens\_Tags содержит ссылки на идентификаторы экрана и тэга шлюза данных, а также содержит всю необходимую информацию для прорисовки элемента на экране. Вспомогательная таблица Visual\_Screens\_UITypes служит унификацией типов представления элементов на экране, т.е. содержит перечисление будет ли элемент отображаться квадратом, либо числом, либо изображением.

## **Выводы по главе 2**

Во второй главе ВКР рассмотрены требования к информационной системе, были спроектированы логическая и физическая модели, выбраны программные средства разработки, а именно: язык программирования C#, система управления базами данных Microsoft SQL и среда программирования Microsoft Visual Studio.

Для реализации информационной системы были изучены готовые библиотеки сторонних разработчиков, а также протоколы транспорта данных OPC и MQTT. Была разработана рабочая версия информационной системы, позволяющая решить цели и задачи автоматизации упаковочно-транспортной линии цеха производства гофротары. Основным достижением является разработка общей архитектуры информационной системы, позволяющей масштабировать решение с минимальными затратами времени и усилий.

Для предоставления информации конечным пользователям были представлены подходы составной отчетности и создания цифрового двойника. Цифровой двойник является инновационным инструментом визуализации данных в настоящее время, позволяющий оценивать технологические и технические процессы в реальном времени и реагировать на аварийные ситуации заблаговременно.

## Глава 3 ОЦЕНКА И ОБОСНОВАНИЕ ЭКОНОМИЧЕСКОЙ ЭФФЕКТИВНОСТИ ПРОЕКТА

### 3.1 Выбор и обоснование методики расчета экономической эффективности

В качестве базовой методики для обоснования экономической эффективности работы выбран расчет прямой эффективности от внедрения информационной системы по сравнению с базовым вариантом существующей организации обработки информации. В основе описания экономической эффективности лежит сопоставление существующего и внедряемого технологических процессов, анализ затрат, необходимых для выполнения всех операций технологического процесса.

Для расчета прямого эффекта от внедрения разработанного программного продукта необходимо рассмотреть показатели трудовых и стоимостных затрат. Вычисления в основном будут построены на временных затратах обслуживающего персонала при аварийных ситуациях и оперативном получении отчетной информации уполномоченным лицом.

К трудовым показателям относятся следующие:

1. Абсолютное снижение трудовых затрат, рассчитываемое по формуле:

$$T = T_0 - T_1, \quad (3.1)$$

где  $T_0$  – время, затрачиваемое на выполнение автоматизируемых операций в базовом варианте,

$T_1$  – время, затрачиваемое на выполнение автоматизируемых операций в проектном варианте.

2. Коэффициент относительного снижения трудовых затрат  $K_T$  (в процентах), для расчета которого используется следующая формула:

$$K_T = (\Delta T / T_0) * 100 \% \quad (3.2)$$

3. Индекс снижения трудовых затрат, рассчитываемый следующим образом:

$$Y_T = T_0 / T_1 \quad (3.3)$$

К стоимостным показателям относятся следующие:

1. Абсолютное снижение стоимостных затрат.



$$\Delta C = C_0 - C_1 \quad (3.4)$$

где  $C_0$  – стоимостные затраты на обработку информации по базовому варианту,  $C_1$  – стоимостные затраты на обработку информации по предлагаемому варианту.

2. Коэффициент относительного снижения стоимостных затрат  $K_C$  (в процентах), определяемый по следующей формуле:

$$K_C = (\Delta C / C_0) * 100 \% \quad (3.5)$$

3. Индекс снижения стоимостных затрат, рассчитываемый по формуле:

$$Y_C = C_0 / C_1 \quad (3.6)$$

Коэффициенты  $K_C$  и  $Y_C$  характеризуют рост производительности труда за счет внедрения более экономичного варианта проектного решения.

Капитальные затраты на создание и внедрение проекта фактически отсутствуют и не рассчитываются. Аппаратное обеспечение доступно и не приобреталось, программное обеспечение распространяется на бесплатной основе, разработка и внедрение проектного решения велась силами сотрудников предприятия и не требовала дополнительных затрат. Расчет срока окупаемости затрат на внедрение проекта равен нулю, а расчетный коэффициент эффективности стремится к бесконечности.

На основании описанной методики выполним расчет показателей экономической эффективности и сделаем вывод об экономической обоснованности внедрения информационной системы.

### **3.2 Расчет показателей экономической эффективности проекта**

Трудовые затраты до внедрения информационной системы составляют в среднем 9 минут на реакцию обслуживающего персонала при аварийной ситуации (в среднем 7 раз в смену – 12 часов), ввиду необходимости оповещения узких специалистов, и 1 минуту, после внедрения информационной системы, таким образом, по формуле (3.1)

$$T_0 = 9 * 7 * 60 = 63 \text{ (человеко-часов в месяц);}$$

$$T_1 = 1 * 7 * 60 = 7 \text{ (человеко-часов в месяц);}$$

$T = 63 - 7 = 56$  (человеко-часов в месяц), 672 (человеко-часов в год);

и 4 минуты каждые два часа рабочего времени машиниста в течение смены на фиксацию и передачу необходимой информации начальнику смены и 2 минуты каждые два часа рабочего времени начальника смены на обработку полученной информации, против 0 минут, поскольку начальник смены будет исключен из процесса транспорта информации

$T_0 = 6 * (12 / 2) * 60 = 36$  (человеко-часов в месяц); из них 24 часа машиниста и 12 часа начальника смены;

$T_1 = 0$  (человеко-часов в месяц);

$T = 36$  (человеко-часов в месяц), 432 (человеко-часов в год).

Коэффициент относительного снижения трудовых затрат по формуле (3.2) в первом случае составит:

$K_T = (56 / 63) * 100 \% = 89\%$  (снижение трудовых затрат)

и во втором соответствующе:

$K_T = (36 / 36) * 100 \% = 100\%$  (снижение трудовых затрат)

При этом индекс снижения трудовых затрат, рассчитываемый по формуле (3.3) в первом случае составит:

$Y_T = 63 / 7 = 9$  (т.е. трудовые затраты снижаются в 9 раз)

а во втором не является расчетным, поскольку трудовые затраты не снижаются, они исключаются из процесса.

В стоимостном отношении внедрение информационной системы выглядит следующим образом:

Возьмем для расчетов заработную плату машиниста – 30 000 руб./мес. (необходимо двое работающих посменно), узкого специалиста 35 000 руб./мес. (необходимо двое работающих посменно), начальника смены – 35 000 руб./мес. (необходимо двое работающих посменно).

В первом случае:

$C_0 = 63 * 30000 * 2 / 60 / 12 = 5\ 250$  руб. (затраты в месяц до внедрения информационной системы)

$C_1 = 7 * 35000 * 2 / 60 / 12 = 681$  руб. (затраты в месяц после внедрения информационной системы)

$\Delta C = C_0 - C_1 = 4\ 569$  руб. (снижение стоимостных затрат в месяц), 54 828 руб. (в год).

Во втором случае:

$C_0 = 24 * 30000 * 2 / 60 / 12 + 12 * 35000 * 2 / 60 / 12 = 3\ 166$  руб. (затраты в месяц до внедрения информационной системы)

$C_1 = 0 * 35000 * 2 / 60 / 12 = 0$  руб. (затраты в месяц после внедрения информационной системы)

$\Delta C = C_0 - C_1 = 3\ 166$  руб. (снижение стоимостных затрат), 37 992 руб. (в год).

Коэффициент относительного снижения стоимостных затрат, определяемый по следующей формуле (3.5):

В первом случае:

$$K_C = (\Delta C / C_0) * 100 \% = 4\ 569 / 5\ 250 * 100 \% = 87\%$$

Во втором случае:

$$K_C = (\Delta C / C_0) * 100 \% = 3166 / 3166 * 100 \% = 100\%$$

Индекс снижения стоимостных затрат, рассчитываемый по формуле (3.6):

В первом случае:

$$Y_C = 5250 / 681 = 7,7 \text{ (т.е. стоимостные затраты снижаются в 7,7 раз)}$$

Во втором случае не является расчетным, поскольку стоимостные затраты не снижаются, они исключаются из расчета.

Аккумулируем полученные результаты в таблицу 3.1.

Таблица 3.1 – Показатели эффективности внедрения информационной системы

	Затраты		Абсолютное изменение затрат	Коэффициент изменения затрат	Индекс изменения затрат
	Базовый вариант	Проектный вариант			
Трудоемкость	$T_0$ (час)	$T_1$ (час)	$\Delta T = T_0 - T_1$	$K_T = \Delta T / T_0 * 100 \%$	$Y_T = T_0 / T_1$

Таблица 3.1 – Показатели эффективности внедрения информационной системы

	Затраты		Абсолютное изменение затрат	Коэффициент изменения затрат	Индекс изменения затрат
	Базовый вариант	Проектный вариант			
1-й вариант	756	84	672	89%	9
2-й вариант	432	0	432	100%	-
Стоимость	$C_0$ (руб)	$C_1$ (руб)	$\Delta C = C_0 - C_1$	$K_C = \Delta C / C_0$ * 100 %	$Y_C = C_0 / C_1$
1-й вариант	63000	8172	54 828	87%	7,7
2-й вариант	37992	0	37992	100%	-

Показатели  $Y_T$  и  $Y_C$  неравны в связи с разной стоимостной оценкой труда рассматриваемых лиц.

Кроме того, из показателей  $K_T$  и  $K_C$  видно, что трудовые показатели эффективнее стоимостных, что позволяет сделать вывод о последующем росте эффективности в связи с индексацией заработной платы.

Диаграммы сравнения трудовых и стоимостных затрат представлены на рисунках 3.2 и 3.3.

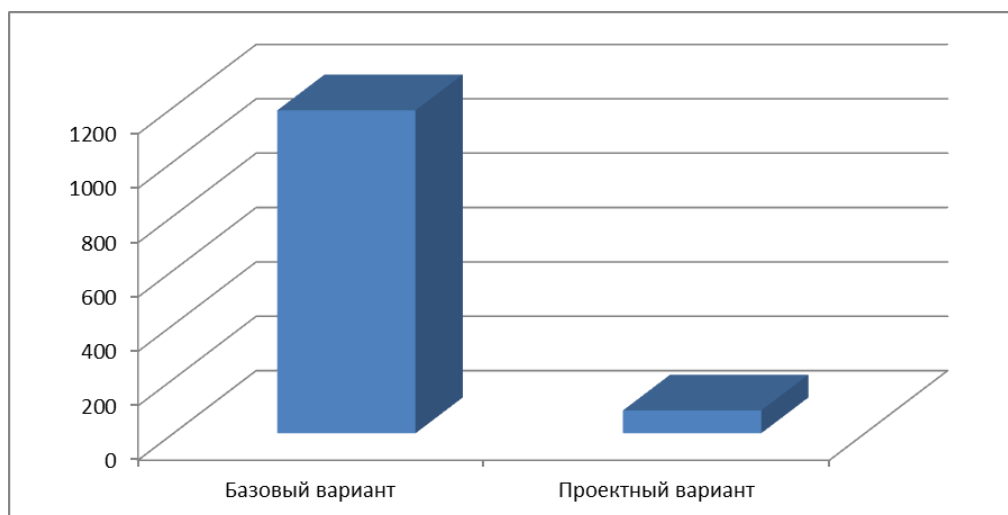


Рисунок 3.2 – Сравнение трудовых затрат

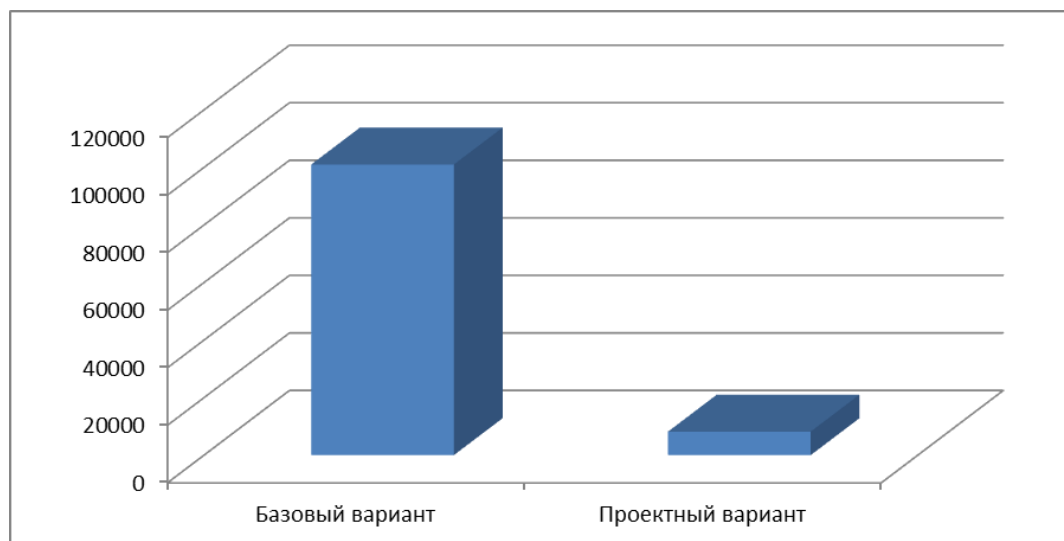


Рисунок 3.3 – Сравнение стоимостных затрат

Таким образом годовой экономический эффект составляет 92 820 рублей, снижая затраты на оповещение ремонтного персонала на 87% и исключая полностью затраты на передачу отчетной информации.

### Выводы по главе 3

В третьей главе был описан и обоснован выбор методики расчета экономической эффективности, а также проведен анализ внедрения программного продукта, доказывающий эффективность проделанной работы.

Кроме вышеперечисленного, внедрение информационной системы позволяет избежать возникновения ошибок в учетных данных, экономит время, затрачиваемое на обработку данных, снижает продолжительность снятия продукции автопогрузчиком с хвостовых транспортеров упаковочно-транспортной линии, повышает достоверность передаваемой информации.

## ЗАКЛЮЧЕНИЕ

В ходе работы над выпускной квалификационной работой был разработан программный продукт. Были автоматизированы следующие процессы: передача информации о выработке и простоях, извещение обслуживающего персонала. Процессы и события упаковочно-транспортной линии стали подконтрольны начальнику смены, ремонтному персоналу и руководству.

В первой главе выпускной квалификационной работы была построена организационная структура предприятия и дана характеристика упаковочно-транспортной системы цеха, определена сущность автоматизации. Выполнено концептуальное моделирование в следствии которых были обнаружены недостатки и методы их устранения. Определена цель и назначение автоматизированного варианта, дана общая характеристика организации решения на ЭВМ и произведена формализация расчетов подзадач. Произведен анализ существующих разработок путем определения критериев сравнения и разбор по ним выбранных программных продуктов.

Во второй главе построены диаграммы использования, отражающие возможность взаимодействия конечных пользователей с программным продуктом. Изучены технологии и протоколы транспорта данных. Фактически разработан программный продукт, состоящий из нескольких решений – сервера, шлюза данных и клиентской части.

Разработанный программный продукт обладает понятным графическим интерфейсом, что не приведет к длительному обучению оператора и руководителя. Так же позволяет достигнуть значительной экономии времени при вводе, обработке, просмотре, редактировании данных.

В третьей главе была выбрана и обоснована методика расчета экономической эффективности, а также произведен расчет, который показал, что трудовые и стоимостные затраты сокращены в проектном варианте.

На основании вышесказанного можно сделать вывод о том, что разработанный программный продукт является выполненной в полной мере и соответствует всем заявленным требованиям.

## СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

### *Нормативно-правовые акты*

1. Федеральный закон РФ от 9 февраля 2009 года N 8-ФЗ (ред. от 09.02.2009) «Об обеспечении доступа к информации о деятельности государственных органов и органов местного самоуправления» // Справочная правовая система Консультант плюс.

2. Постановление Правительства РФ от 10.07.2013 N 583 (ред. от 20.11.2018) «Об обеспечении доступа к общедоступной информации о деятельности государственных органов и органов местного самоуправления в информационно-телекоммуникационной сети Интернет в форме открытых данных» // Справочная правовая система Консультант плюс.

3. Распоряжение Правительства РФ от 10.07.2013 N 1187-р (ред. от 24.03.2018) «О Перечнях информации о деятельности государственных органов, органов местного самоуправления, размещаемой в сети Интернет в форме открытых данных» // Справочная правовая система Консультант плюс.

### *Учебники и учебные пособия*

4. Гутгарц, Р. Д. Проектирование автоматизированных систем обработки информации и управления : учебное пособие для академического бакалавриата / Р. Д. Гутгарц. — М.: ИД Юрайт, 2019. — 304 с.

5. Маркин, А. В. Программирование на sql в 2 ч. Часть 1 : учебник и практикум для бакалавриата и магистратуры / А. В. Маркин. — М.: ИД Юрайт, 2019. — 362 с.

6. Чистов, Д. В. Проектирование информационных систем : учебник и практикум для академического бакалавриата / Д. В. Чистов, П. П. Мельников, А. В. Золотарюк, Н. Б. Ничепорук ; под общей редакцией Д. В. Чистова. — М.: ИД Юрайт, 2019. — 258 с.

### *Научная и методическая литература*

7. Ковязин, Р. Р. Применение технологии ОРС : научно-технический вестник СПбГИТМО (ТУ). Выпуск 10. Информация и управление в

технических системах / Р. Р. Ковязин, А. Е., Платунов. – СПб.: ИД СПбГИТМО, 2003. – с.71–76.

8. Росляков, А. В. Интернет вещей / С. В. Ваняшин, А. Ю. Гребешков, М. Ю. Самсонов. – Самара: ИД ПГУТИ, 2014. – 340 с.

9. Федоренко, Д. Ю. Программирование клиентов OPC на C++ и C#. Часть 1. OPC DA. – М.: ИД Мариуполь, 2012. – 72 с.

*Электронные ресурсы*

10. КодСталь: Подключение к OPC DA серверу в WPF C# [Электронный ресурс] URL: <http://alex-elkin.blogspot.com/2016/11/opc-da-wpf-c.html> (дата обращения 05.03.2019).

11. Новый шлюз UA-5231M для интеграции Modbus устройств в облачные сервисы IoT [Электронный ресурс] URL: <https://icp-das.ru/news/new/shlyuz-ua-5231m/> (дата обращения 05.03.2019).

12. Производство гофрокартона и гофроупаковки в Перми [Электронный ресурс] URL: <https://pcbк.ru/projects/> (дата обращения 05.03.2019).

13. MQTT — Википедия [Электронный ресурс] URL: <https://ru.wikipedia.org/wiki/MQTT> (дата обращения 05.03.2019).

14. MQTT и Modbus: сравнение протоколов, используемых в шлюзах для IoT / Блог компании Intel / Хабр [Электронный ресурс] URL: <https://habr.com/company/intel/blog/304228/> (дата обращения 05.03.2019).

15. OPC — Википедия [Электронный ресурс] URL: <https://ru.wikipedia.org/wiki/OPC> (дата обращения 05.03.2019).

16. Boost Your IIOT Development [Электронный ресурс] URL: <https://www.moxa.com/Event/industrial-computers/thingspro-data-acquisition-device-management-platform/index.htm> (дата обращения 05.03.2019).

17. Crystal Reports, Developer for Visual Studio Downloads - Business Intelligence (BusinessObjects) - SCN Wiki [Электронный ресурс] URL: <https://wiki.scn.sap.com/wiki/display/BOBJ/Crystal+Reports%2C+Developer+for+Visual+Studio+Downloads> (дата обращения 05.03.2019).



18. Emmepi Groip. Handling division system [Электронный ресурс]. URL: <http://www.emmepigroup.com/handling-division-system/> (дата обращения 05.03.2019).

19. Home Page - OPC Foundation [Электронный ресурс] URL: <https://opcfoundation.org/> (дата обращения 05.03.2019).

20. Linea uscita prodotto finito 4 [Электронный ресурс] URL: [https://www.youtube.com/watch?v=7ZMFNDE\\_174](https://www.youtube.com/watch?v=7ZMFNDE_174) (дата обращения 05.03.2019).

21. MQTT [Электронный ресурс] URL: <http://mqtt.org/> (дата обращения 05.03.2019).

22. M2Mqtt & GnatMQ | MQTT Client Library & Broker for .Net platform [Электронный ресурс] URL: <https://m2mqtt.wordpress.com/> (дата обращения 05.03.2019).

23. OPC Foundation - Classic [Электронный ресурс] URL: <https://opcfoundation.org/developer-tools/specifications-classic/data-access/> (дата обращения 05.03.2019).

24. Raspberry Pi 3 Model B+ - Raspberry Pi [Электронный ресурс] URL: <https://www.raspberrypi.org/products/raspberry-pi-3-model-b-plus/> (дата обращения 05.03.2019).

25. SAP Crystal Reports [Электронный ресурс] URL: <https://www.crystalreports.com/> (дата обращения 05.03.2019).

*Литература на иностранном языке*

26. Gaston С.Н. MQTT Essentials - A Lightweight IoT Protocol. Packt Publishing, 2017. – 280 с.

27. Joch K. Mosquitto - MQTT BROKER FOR IoT (Internet of Things). CTS GMBH, 2017. – 49 с.

28. Mahnke W., Leitner S., Damm M. OPC Unified Architecture. Springer. 2009. – 339 с.

## ПРИЛОЖЕНИЕ А

### Листинг структуры библиотеки DLL\_OPC

```
1 using ...
2
3
4
5
6 namespace dll_OPC
7 {
8     /// <summary> Клиент OPC
9     public class Client
10    {
11        /// <summary> Версия модуля
12        public const int Version = 20190112;
13        /// <summary> Класс коллекции ServersList
14        public class class_AvailableServers...
15        /// <summary> Класс коллекции контролируемых Тэгов
16        public class class_MonitoredTags...
17        /// <summary> Класс коллекции Тэгов и их значений при событии входящих данных
18        public class class_MonitoredTagsValues...
19        /// <summary> Класс коллекции доступных групп и тэгов сервера OPC
20        public class class_AvailableTags...
21        /// <summary> Делегат события изменения состояния подключения
22        public delegate void delegate_BoolStatus(bool Status);
23        /// <summary> Событие изменения состояния подключения
24        public event delegate_BoolStatus event_ConnectionStatus;
25        /// <summary> Событие изменения состояния монитора тэгов
26        public event delegate_BoolStatus event_MonitorStatus;
27        /// <summary> Делегат события входящих данных контроля тэгов
28        public delegate void delegate_DataChange(DateTime TimeStamp, List<class_MonitoredTagsValues> TagsList);
29        /// <summary> Событие входящих данных контроля тэгов
30        public event delegate_DataChange event_DataChange;
31        /// <summary> Строка соединения с сервером OPC
32        public string ConnectionPath...
33        /// <summary> Состояние подключения к серверу OPC
34        public bool ConnectionStatus...
35
36        private dll_Errors.Errors _Errors = new dll_Errors.Errors();
37        private OPCServer OPC_Server;
38        private OPCGroups OPC_Groups;
39        private OPCGroup OPC_Group;
40        private string OPC_Server_IP;
41        private string OPC_ProgID;
42        private bool event_ConnectionStatus_Status = false;
43        private void event_ConnectionStatus_Tick() { if (event_ConnectionStatus != null) event_ConnectionStatus(
44        private bool event_MonitorStatus_Status = false;
45        private void event_MonitorStatus_Tick() { if (event_MonitorStatus != null) event_MonitorStatus(event_Mon
46        private int OPC_UpdateRate_Min = 500;
47        private void event_DataChange_Tick(List<class_MonitoredTagsValues> ExportTagsList) { if (event_DataChang
48
49        /// <summary> Клиент OPC
50        public Client(string ServerIP)...
51
52        /// <summary> Клиент OPC
53        public Client()...
54
55        /// <summary> Подключение к серверу OPC
56        public bool Connect(string ProgID, string GroupName, int UpdateRate, List<class_MonitoredTags> Monitored
57
58        /// <summary> Подключение к серверу OPC
59        public bool Connect(string ProgID, string GroupName, List<class_MonitoredTags> MonitoredTags)...
60
61        /// <summary> Подключение к серверу OPC
62        public bool Connect(string ProgID, int UpdateRate, List<class_MonitoredTags> MonitoredTags)...
63
64        /// <summary> Подключение к серверу OPC
65        public bool Connect(string ProgID, List<class_MonitoredTags> MonitoredTags)...
66
67        /// <summary> Подключение к серверу OPC
68        public bool Connect(string ProgID, string GroupName, int UpdateRate)...
```

```

354
355 //  Подключение к серверу OPC
361 public bool Connect(string ProgID, string GroupName)...
365
366 //  Подключение к серверу OPC
372 public bool Connect(string ProgID, int UpdateRate)...
376
377 //  Подключение к серверу OPC
382 public bool Connect(string ProgID)...
386
387 //  Отключение от сервера OPC
391 public bool Disconnect()...
415
416 //  Список доступных серверов OPC
420 public List<class_AvailableServers> AvailableServers()...
452
453 //  Список доступных групп и тэгов сервера OPC в виде коллекции
457 public List<class_AvailableTags> AvailableTags()...
479
480 private List<class_AvailableTags> AvailableTags_Iteration(OPCBrowser cur_OPcBrowser)...
528
529 //  Список доступных групп и тэгов сервера OPC в виде текста
534 public string AvailableTags(bool IsShowIDs)...
538
539 private string AvailableTags(bool IsShowIDs, string sub_t, List<class_AvailableTags> AvailableGroupsTags
558
559 //  Список контролируемых тэгов
563 public List<class_MonitoredTags> MonitoredTags()...
587
588 //  Добавить тэг в список контролируемых
592 public bool MonitoredTags_Add(List<class_MonitoredTags> TagsNames)...
631
632 //  Удалить тэг из списка контролируемых
636 public bool MonitoredTags_Delete(List<class_MonitoredTags> TagsNames)...
673
674 //  Очистить список контролируемых тэгов
678 public bool MonitoredTags_Clear()...
704
705 //  Значение контролируемого тэга по имени
710 public List<class_MonitoredTagsValues> MonitoredTags_Values(string TagName)...
747
748 //  Значения всех контролируемых тэгов
752 public List<class_MonitoredTagsValues> MonitoredTags_Values()...
756
757 //  Запуск контроля тэгов
761 public bool MonitorTags_Start()...
787
788 //  Остановка контроля тэгов
792 public bool MonitorTags_Stop()...
811
812 private void OPC_DataChange_Action(int TransactionID, int NumItems, ref Array ClientHandles, ref Array I
840 }
841
842 internal static class Func
843 {
844     internal static bool CheckIP(string str)...
851
852     internal static int ObjectToInt(object obj)...
862
863     internal static DateTime ObjectToDateTime(object obj)...
873
874     internal static int CanonicalTypeCode_ToInt(Object Canonical)...
884
885     internal static Type CanonicalTypeCode_ToSharp(Object Canonical)...
924 }
925

```

## ПРИЛОЖЕНИЕ Б

### Листинг структуры библиотеки dll\_MQTT\_Broker

```
10 namespace dll_MQTT_Broker
11 {
12     /// <summary> Брокер MQTT
13     public class Broker
14     {
15         /* Links ...
16
17         /// <summary> Версия модуля
18         public const int Version = 20190112;
19
20         /// <summary> Делегат события изменения состояния подключения
21         public delegate void delegate_BooleanStatus(bool Status);
22         /// <summary> Событие изменения состояния подключения
23         public event delegate_BooleanStatus event_ConnectionStatus;
24         /// <summary> Делегат события подключения-отключения клиента MQTT
25         public delegate void delegate_Clients_ConnectDisconnect(string ClientId);
26         /// <summary> Событие подключения-отключения клиента MQTT
27         public event delegate_Clients_ConnectDisconnect event_Clients_ConnectDisconnect;
28         /// <summary> Делегат события подписки-отписки клиента MQTT
29         public delegate void delegate_Clients_SubscribeUnsubscribe(bool IsSubscribe, ushort MessageId, string[] Topics);
30         /// <summary> Событие подписки-отписки клиента MQTT
31         public event delegate_Clients_SubscribeUnsubscribe event_Clients_SubscribeUnsubscribe;
32         /// <summary> Делегат события сообщения от клиента MQTT
33         public delegate void delegate_Clients_Message(DateTime TimeStamp, string Topic, string Message);
34         /// <summary> Событие сообщения от клиента MQTT
35         public event delegate_Clients_Message event_Clients_Message;
36         /// <summary> Строка соединения с брокером MQTT
37         public string ConnectionPath...
38
39         private dll_Errors.Errors _Errors = new dll_Errors.Errors();
40         private bool event_ConnectionStatus_Status = false;
41         private void event_ConnectionStatus_Tick() { if (event_ConnectionStatus != null) event_ConnectionStatus(event_ConnectionStatus_Status); }
42         private void event_Clients_ConnectDisconnect_Tick(string ClientId) { if (event_Clients_ConnectDisconnect != null) event_Clients_ConnectDisconnect(ClientId); }
43         private void event_Clients_SubscribeUnsubscribe_Tick(bool IsSubscribe, ushort MessageId, string[] Topics) { if (event_Clients_SubscribeUnsubscribe != null) event_Clients_SubscribeUnsubscribe(IsSubscribe, MessageId, Topics); }
44         private void event_Clients_Message_Tick(string Topic, string Message) { if (event_Clients_Message != null) event_Clients_Message(DateTime.Now, Topic, Message); }
45         private MqttTcpCommunicationLayer MQTT_CommunicationLayer;
46         private MqttBroker MQTT_Broker;
47         private int MQTT_Broker_Port;
48         private bool MQTT_Broker_Secure;
49         private X509Certificate MQTT_Broker_serverCert;
50         private MqttSslProtocols MQTT_Broker_sslProtocol;
51         private int MQTT_Broker_ConnectTimeout;
52         private bool MQTT_Broker_IsReTransfer = false;
53         private MqttClientCollection MQTT_Clients = new MqttClientCollection();
54
55         /// <summary> Брокер MQTT
56         public Broker(int Port, bool Secure, X509Certificate serverCert, MqttSslProtocols sslProtocol, int ConnectTimeout, bool IsReTransfer)...
57
58         /// <summary> Брокер MQTT
59         public Broker(int Port, bool IsReTransfer)...
60
61         /// <summary> Брокер MQTT
62         public Broker(bool IsReTransfer)...
63
64         /// <summary> Брокер MQTT
65         public Broker()...
66
67         /// <summary> Запуск брокера MQTT
68         public bool Start(string User, string Pass)...
69
70         /// <summary> Запуск брокера MQTT
71         public bool Start()...
72
73         /// <summary> Остановка брокера MQTT
74         public bool Stop()...
75
76         private bool Auth_CheckParams(string User, string Pass)...
77
78         private bool Connect_RemoteCertificateValidation(object sender, X509Certificate certificate, X509Chain chain, SslPolicyErrors policyErrors)...
79
80         private X509Certificate Connect_LocalCertificateSelection(object sender, string targetHost, X509CertificateCollection localCertificates, X509CertificateCollection remoteCertificates)...
81
82         private void Event_ClientConnected(object sender, MqttClientConnectedEventArgs e)...
83
84         private void Event_ClientConnected_Connected(object sender, MqttMsgConnectEventArgs e)...
85
86         private void Event_ClientConnected_Disconnected(object sender, EventArgs e)...
87
88         private void Event_ClientConnected_ConnectionClosed(object sender, EventArgs e)...
89
90         private void Event_ClientConnected_Subscribe(object sender, MqttMsgSubscribeEventArgs e)...
91
92         private void Event_ClientConnected_Unsubscribe(object sender, MqttMsgUnsubscribeEventArgs e)...
93
94         private void Event_ClientConnected_Publish(object sender, MqttMsgPublishEventArgs e)...
95
96     }
97 }
```

## ПРИЛОЖЕНИЕ В

### Листинг структуры библиотеки dll\_MQTT\_Client

```
9 namespace dll_MQTT_Client
10 {
11     /// <summary> Клиент MQTT
14     public class Client
15     {
16         /* Links ...
22
23         /// <summary> Версия модуля
26         public const int Version = 20190116;
27
28         /// <summary> Тип направления данных
31         public enum enum_DataTransferWay { Outgoing = 0, Incoming = 1 }
32         /// <summary> Делегат события изменения состояния подключения
35         public delegate void delegate_BoolStatus(bool Status);
36         /// <summary> Событие изменения состояния подключения
39         public event delegate_BoolStatus event_ConnectionStatus;
40         /// <summary> Событие изменения состояния подписки
43         public event delegate_BoolStatus event_SubscribeStatus;
44         /// <summary> Делегат события транспорта данных
47         public delegate void delegate_DataChange(DateTime TimeStamp, enum_DataTransferWay DataTransferWay, ushort MesID);
48         /// <summary> Событие транспорта данных
51         public event delegate_DataChange event_DataChange;
52         /// <summary> Строка соединения с брокером MQTT
55         public string ConnectionPath...
63
64         private dll_Errors.Errors _Errors = new dll_Errors.Errors();
65         private bool event_ConnectionStatus_Status = false;
66         private void event_ConnectionStatus_Tick() { if (event_ConnectionStatus != null) event_ConnectionStatus(event_ConnectionStatus_Status); }
67         private bool event_SubscribeStatus_Status = false;
68         private void event_SubscribeStatus_Tick() { if (event_SubscribeStatus != null) event_SubscribeStatus(event_SubscribeStatus_Status); }
69         private string MQTT_Server_IP;
70         private int MQTT_Server_Port;
71         private bool MQTT_Server_Secure;
72         private X509Certificate MQTT_Server_caCert;
73         private X509Certificate MQTT_Server_clientCert;
74         private MqttSslProtocols MQTT_Server_sslProtocol;
75         private void event_DataChange_Tick(enum_DataTransferWay DataTransferWay, ushort MesID, string Topic, string Message) { }
76         private MqttClient MQTT_Client;
77         private string[] MQTT_SubscribedTopics = new string[0];
78         private DateTime MQTT_RestoreConnection_LastDate = default(DateTime);
79
80         /// <summary> Клиент MQTT
90         public Client(string ServerIP, int ServerPort, bool Secure, X509Certificate caCert, X509Certificate clientCert)
102
103         /// <summary> Клиент MQTT
108         public Client(string ServerIP, int ServerPort)...
120
121         /// <summary> Клиент MQTT
124         public Client()...
133
134         /// <summary> Подключение к брокеру MQTT https://nifi.apache.org/docs/nifi-docs/ ...
149         public bool Connect(string clientId, string username, string password, bool willRetain, byte willQosLevel, byte willDelayInterval)
212
213         /// <summary> Подключение к брокеру MQTT
220         public bool Connect(string clientId, string username, string password)...
235
```



# ПРИЛОЖЕНИЕ Г

## Листинг структуры библиотеки dll\_SQL

```
5 namespace dll_SQL
6 {
7     /// <summary> Обмен данными с MS SQL
10 public class SQL
11 {
12     /// <summary> Версия модуля
15     public const int Version = 20190122;
16     /// <summary> Делегат события изменения состояния подключения
19     public delegate void delegate_BooleanStatus(bool Status, string ConnectionPath);
20     /// <summary> Событие изменения состояния подключения
23     public event delegate_BooleanStatus event_ConnectionStatus;
24     /// <summary> Делегат события выполнения команды SQL
32     public delegate void delegate_WriteToDatabase(DateTime TimeStamp, string Server, string DataBase, string Command,
33     /// <summary> Событие выполнения команды SQL
36     public static event delegate_WriteToDatabase event_WriteToDatabase;
37     public string ConnectionInfo...
45
46     private dll_Errors.Errors _Errors = new dll_Errors.Errors();
47     private bool event_ConnectionStatus_Status = false;
48     private void event_ConnectionStatus_Tick() { if (event_ConnectionStatus != null) event_ConnectionStatus(event_Con
49     private void event_WriteToDatabase_Tick(string Command, bool Result) { if (event_WriteToDatabase != null) event_w
50     private string SQL_Server = @"localhost\SQLEXPRESS";
51     private string SQL_DataBase = "test";
52     private bool SQL_IntegratedSecurity = true;
53     private string SQL_User = "";
54     private string SQL_Pass = "";
55     private int SQL_ConnectTimeoutSec = 30;
56     private bool SQL_RestoreUse = true;
57     private int SQL_RestoreTimeoutSec = 20;
58     private bool SQL_MultipleActiveResultSets = true;
59     private DateTime Restore_LastDate = default(DateTime);
60     private SqlConnection p_Connection = null;
61
62     /// <summary> Обмен данными с MS SQL
73     public SQL(string Server, string DataBase, bool IntegratedSecurity, string User, string Password, int ConnectTime
86
87     /// <summary> Обмен данными с MS SQL
92     public SQL(string Server, string DataBase)...
104
105     ~SQL()...
127
128     private bool Connect()...
176
177     /// <summary> Отключение соединения
180     public bool Disconnect()...
203
204     /// <summary> Активно ли соединение
207     public bool IsActive()...
219
220     private bool RestoreConnection()...
243
244     /// <summary> Выполнить SQL команду без ответа
248     public bool Query_NonAnswer(SqlCommand Command, bool DisconnectAfterSend)...
283
284     /// <summary> Выполнить SQL команду без ответа
289     public bool Query_NonAnswer(string CommandStr, bool DisconnectAfterSend)...
298
299     /// <summary> Выполнить SQL команду без ответа
303     public bool Query_NonAnswer(string CommandStr)...
307
308     /// <summary> Выполнить SQL команду и получить ответ
314     public DataTable Query_WithAnswer(SqlCommand Command, bool DisconnectAfterSend)...
354
355     /// <summary> Выполнить SQL команду и получить ответ
361     public DataTable Query_WithAnswer(string CommandStr, bool DisconnectAfterSend)...
370
371     /// <summary> Выполнить SQL команду и получить ответ
376     public DataTable Query_WithAnswer(string CommandStr)...
380
381 }
```

# ПРИЛОЖЕНИЕ Д

## Пример отчета съема продукции

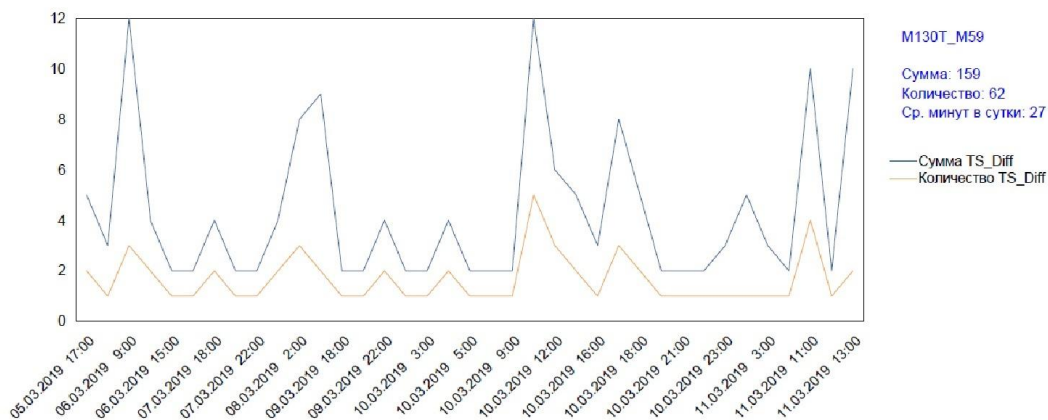
Logitek - Несвоевременный съем с хвостов переработки



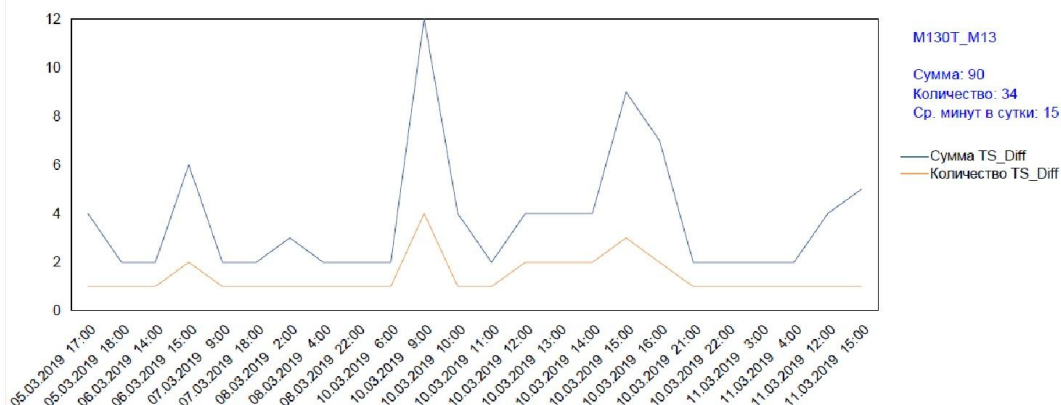
Данные 05.03.2019 16:23 - 11.03.2019 19:00  
Дата актуализауии: 12.03.2019 16:49:09

DigitalPlant

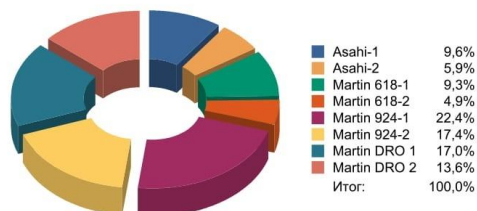
Съем Martin DRO 1



Съем Asahi-1



Суммарная длительность по линиям



Суммарное количество инцидентов по линиям

