Программное обеспечение «Track & Trace System»

РУКОВОДСТВО АДМИНИСТРАТОРА

версия 1.0.0

Листов 34

2024

АННОТАЦИЯ

В настоящем документе приведены сведения о назначении и возможностях программного обеспечения «Track & Trace System» (далее – ПО), сведения о технических и программных средствах, необходимых для его функционирования, сведения о структуре и составных частях ПО, о связях внутри ПО и связях с внешними системами.

СОДЕРЖАНИЕ

1. ОБЩИЕ СВЕДЕНИЯ

1.1. Назначение ПО

ПО предназначено для обеспечения бесперебойного процесса маркировки табачной продукции на производстве с учетом актуальных требований законодательства Российской Федерации и стран Евразийского экономического союза (далее – ЕАЭС) к порядку проведения операций нанесения и учета кодов маркировки продукции.

Компоненты ПО реализуют следующие задачи в рамках процессов маркировки табачной продукции на производстве:

1) эмиссия кодов маркировки согласно производственному заказу в станции управления заказами (далее – СУЗ);
2) формирование и отправка на печатные устройства объектов для печати информации о маркировке продукции;
3) сбор и обработка отчетов об агрегации и нанесении;
4) формирование и отправка отчетов о нанесении и отчетов об агрегации в СУЗ;
5) ручная паллетная агрегация;
6) контроль данных, обрабатываемых на ПО, согласно определенным правилам;
7) управление справочной информацией, используемой для функционирования ПО;
8) настройка и параметризация шаблонов для печати;
9) визуализация состояния исполняемых действий и процессов;
10) краткосрочное хранение истории по завершенным процессам;
11) архивное хранение данных об агрегации;
12) авторизация пользователей. Позволяет авторизовываться пользователям на основе ролевой модели, с соответствующими правами для данной роли. Создавать пользователей и назначать им роли;
13) добавление и редактирование производственных линий и их индивидуальных характеристик с заполнением атрибутивного состава полей;
14) добавление шаблонов печати. Добавление шаблонов печати заранее подготовленного файла. При добавлении шаблона поддерживается версионность, которая позволяет просмотреть версии файла, а также скачать или удалить любую из версий;
15) добавление продукции. Позволяет создать описание атрибутивного состава производимой продукции. Создание продукта может осуществляться как в пользовательском интерфейсе, так и путем загрузки готовых данных, заранее подготовленных файлов.

1.2. Описание ролей

В рамках работы ПО в части пользователей предполагается использование ролевой модели.

Функциональные роли пользователей ПО:

оператор – роль пользователя. Доступ осуществляется через внутреннюю сеть предприятия с рабочего места оператора.

администратор – роль пользователя. Доступ осуществляется через внутреннюю сеть предприятия с рабочего места администратора.

1.3. Сведения о функционировании ПО

ПО представляет собой распределенную информационную систему.

Схема развертывания ПО представлена на рисунке 1.

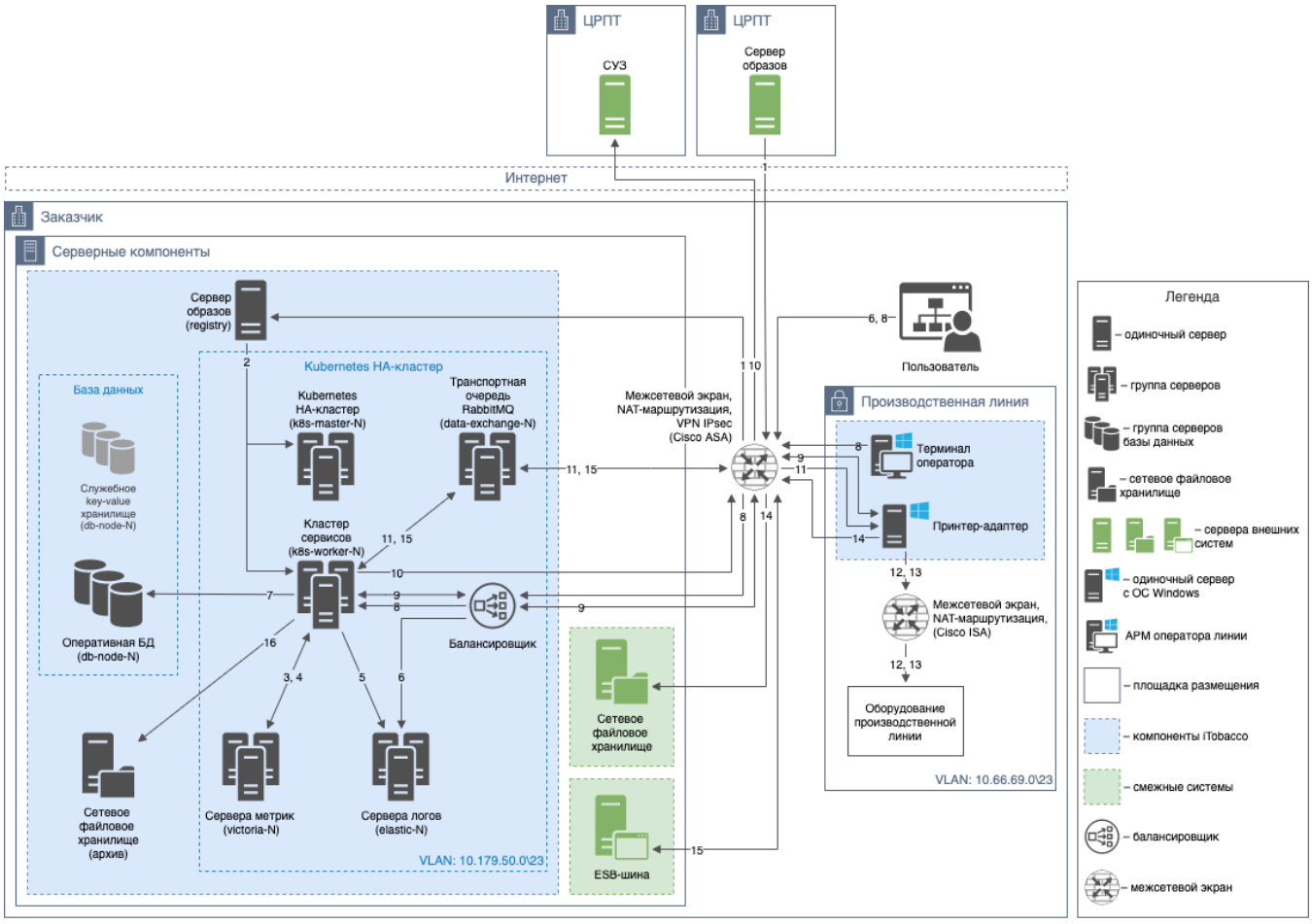

Рисунок 1. Схема развертывания ПО

Устанавливается отдельный кластер Kubernetes High Availability, в котором работают прикладные приложения ПО.

Операционная база данных, сервер логов ElasticSearch, транспортная очередь RabbitMQ и сервер метрик размещаются вне кластера Kubernetes High Availability.

Роли компонентов ПО приведены в таблице ниже.

Таблица 1. Роли компонентов ПО

Роль	Функции и комментарии
Kubernetes High Availability (k8s-master-N)	Master-узлы, обеспечивающие отказоустойчивость кластера Kubernetes сети Docker
Кластер сервисов (k8s-worker-N)	Рабочий узел кластера Kubernetes сети Docker, на котором разворачиваются прикладные сервисы ПО
Сервер образов (registry)	Осуществляет хранение образов ППО и необходимых компонентов обслуживающих сервисов, разворачиваемых в kubernetes
Транспортная очередь RabbitMQ (data-exchange-N)	Кластер RabbitMQ для межсервисного взаимодействия
Оперативная база данных (db-node-N)	СУБД на базе Postgres 13 для хранения пользователей, ролей, политик доступа, операционной информации
Сервер логов Elasticsearch (elastic-N)	Сервис хранения логов ElasticSearch
Сервер метрик (victoria-N)	Сервис хранения метрик системы VictoriaMetrics
Сетевое файловое хранилище (архив)	Файловое хранилище архивов
ESB-шина	Шина для обмена данными с 1С
Пользователи	Интернет-браузер на стороне операторов и администраторов
Балансировщик	Осуществляет прием и перенаправление запросов
Межсетевой экран, NAT-маршрутизация, VPN IPsec	Для внешнего подключения к инфраструктуре производства
Терминал оператора	АРМ оператора
Принтер адаптер	АРМ оператора
Межсетевой экран NAT-маршрутизация (Cisco ISA)	Для внешнего подключения к оборудованию производственной линии
Оборудование производственной линии	

1.4. Сведения о технических и иных средствах, обеспечивающих функционирование ПО

Серверная часть ПО устанавливается на серверах заказчика.

Требования к характеристикам технических средств, на которых разворачивается ПО, приведены в таблице ниже.

Таблица 2. Виртуальные машины для размещения серверных компонентов.

Название	IP-адрес	Роль	CPU	RAM	Примечание
docker-registry	10.179.51.5	docker-registry, rancher-host	2	4GB	
k8s-master-1	10.179.51.10	Kubernetes master	4	8GB	Входит в состав кластера k8s
k8s-master-2	10.179.51.11	Kubernetes master	4	8GB	Входит в состав кластера k8s
k8s-master-3	10.179.51.12	Kubernetes master	4	8GB	Входит в состав кластера k8s
elastic-1	10.179.51.21	Logging	4	8GB	Входит в состав кластера k8s
elastic-2	10.179.51.22	Logging	4	8GB	Входит в состав кластера k8s
elastic-3	10.179.51.23	Logging	4	8GB	Входит в состав кластера k8s
k8s-worker-1	10.179.51.31	Kubernetes worker	10	18GB	Входит в состав кластера k8s
k8s-worker-2	10.179.51.32	Kubernetes worker	10	18GB	Входит в состав кластера k8s
k8s-worker-3	10.179.51.33	Kubernetes worker	10	18GB	Входит в состав кластера k8s
victoria-1	10.179.51.41	Monitoring	4	6GB	Входит в состав кластера k8s
victoria-2	10.179.51.42	Monitoring	4	6GB	Входит в состав кластера k8s
victoria-3	10.179.51.43	Monitoring	4	6GB	Входит в состав кластера k8s
data-exchange-1	10.179.51.61	RabbitMQ	4	4GB	Входит в состав кластера k8s
data-exchange-2	10.179.51.62	RabbitMQ	4	4GB	Входит в состав кластера k8s
data-exchange-3	10.179.51.63	RabbitMQ	4	4GB	Входит в состав кластера k8s
etcd-node-01	10.179.51.71	etcd	2	2GB	Для координации работы Postgresql-кластера

Название	IP-адрес	Роль	CPU	RAM	Примечание
etcd-node-02	10.179.51.72	etcd	2	2GB	Для координации работы Postgresql-кластера
etcd-node-03	10.179.51.73	etcd	2	2GB	Для координации работы Postgresql-кластера
db-node-01	10.179.51.81	Postgresql	8	20GB	
db-node-02	10.179.51.82	Postgresql	8	20GB	
db-node-03	10.179.51.83	Postgresql	8	20GB	

Требования к составу дополнительно используемого программного обеспечения при функционировании ПО приведены в таблице 3.

Таблица 3. Список дополнительно используемого программного обеспечения

Название	Минимальная версия	Описание	Примечание
Kubernetes	1.21	Средство управления контейнерами	Сервисы системы развертываются и управляются Kubernetes.
RabbitMQ	13&9	Брокер сообщений, используется в качестве распределенной шины обмена данными	
Victoria Metrics	1.83.0	Сервис, отвечающий за хранение метрик ПО	
ElasticSearch	2.2.1	Программная поисковая система, служит для хранения и работы с логами ППО и общесистемных сервисов	
Kibana	2.2.1	Сервис для работы с логами общесистемных сервисов и ППО	
Grafana	8.5.11	Веб сервис для отображения метрик виртуальных машин, общесистемных сервисов и ППО	

Таблица 3. Список дополнительно используемого программного обеспечения

Название	**Минимальная версия**	**Описание**	**Примечание**
Alert Manager	0.25.0	Сервис для отправки настраиваемых уведомлений	
PostgreSQL	13.1	СУБД	

Клиентская часть ПО представляет собой веб-интерфейс, который пользователь открывает через свой браузер. Поддерживаемые браузеры указаны ниже:

- Google Chrome;
- Edge;
- Mozilla Firefox;
- Yandex-браузер;
- Opera.

2. СТРУКТУРА ПО

2.1. Сведения о структуре ПО

ПО строится по принципам микросервисной архитектуры, которая предполагает создание ПО из множества модулей меньшего размера (микросервисов), поддерживающих независимое развертывание. Как правило, каждый сервис ограничен только своей выполняемой функцией и взаимодействует с другими сервисами посредством программных интерфейсов.

При разработке ПО использовался принцип микросервисной архитектуры, когда каждый микросервис из состава ПО выполняет определенный набор функций. Микросервисы поставляются как docker-контейнеры, которые содержат в себе зависимости и базовые настройки (за исключением специфичных для окружения настроек и конфиденциальных данных, а именно паролей, сертификатов и т.п.), необходимые для запуска. Управление и запуск контейнеров осуществляется в кластере Kubernetes.

ПО состоит из следующих программных модулей – см. таблицу 4.

Таблица 4. Программные модули ПО

Сервис	Название	Описание
po-manager	Модуль управления заказами	Раздел 2.2.1
suz-adapter	Модуль управления станцией управления заказом	Раздел 2.2.5
operational-service	Модуль обработки событий технологического процесса	Раздел 2.2.2
archive-loader	Модуль архивного хранения	Раздел 2.2.4
factory-layer-adapter	Модуль взаимодействия с Factory Layer	Раздел 2.2.7
esb-adapter	Модуль взаимодействия с ESB-шиной	Раздел 2.2.6
ui-app	Ui/frontend	

Таблица 5. Программные компоненты производственной линии

Сервис	Название	Описание
printer-adapter	Модуль управления печатными устройствами	Раздел 2.2.3
printer-driver-sewtec-telnet	Драйвер линии Sewtec	Раздел 2.2.3.1
printer-driver-videojet-ultimate	Драйвер принтера Videojet (Ultimate protocol)	Раздел 2.2.3.2

2.2. Сведения о составных частях ПО

2.2.1. Описание модуля управления заказами

Модуль управления заказами предназначен для централизованного управления и контроля за исполняемыми процессами, а также управления настройками ПО и ведения единой справочной информации.

Модуль управления заказами предусматривает пользовательский интерфейс со следующими функциональными возможностями:

1) создание производственных заказов;
2) визуализация состояния производственных заказов, связанных объектов и процессов;
3) загрузка и параметризация шаблонов для печати;
4) настройка топологии печатных устройств;
5) ведение справочной информации;
6) управление пользователями и ролями;
7) управление общими настройками ПО.

Модуль управления заказами реализует следующие автоматизированные функции:

1) формирование (эмиссия) буфера кодов маркировки согласно производственному заказу;
2) генерация кодов агрегатов согласно определенному алгоритму;
3) отправка объектов для печати в модуль управления печатными устройствами и обработка результатов нанесения;
4) ведение статусной модели производственных заказов и связанных объектов;
5) формирование и отправка отчетов о нанесении и агрегации;
6) сбор информации о состоянии компонентов ПО;
7) формирование и отправка сведений о произведенной продукции.

Подробное описание работы с пользовательским интерфейсом приведено в документе «Руководство пользователя».

2.2.2. Описание модуля обработки событий технологического процесса

Модуль обработки событий технологического процесса предназначен для фиксации данных оперативной обработки и проверки обрабатываемой в ПО информации.

Модуль обработки событий технологического процесса реализует следующие автоматизированные функции:

1) прием и обработка информации о производственных заказах, информации о нанесении, отчетов об агрегации;

2) выполнение проверок на соответствие требованиям к консистентности и непротиворечивости данных в процессе производства продукции;
3) включение и отключение проверок;
4) формирование протокола по результатам проверок.

Модуль обработки событий технологического процесса не предусматривает пользовательский интерфейс. Отображение результатов выполненных проверок реализуется посредством единого пользовательского интерфейса модуля управления заказами.

2.2.3. Описание модуля управления печатными устройствами

Модуль управления печатными устройствами предназначен для обеспечения бесперебойного процесса нанесения кодов маркировки и агрегации путем пополнения буферов печатных устройств объектами для печати.

Модуль управления печатными устройствами реализует следующие автоматизированные функции:

1) прием и обработка информации о кодах маркировки и агрегации, планируемых к нанесению;
2) преобразование информации о кодах маркировки и агрегации в объекты для печати с использованием установленных шаблонов и драйверов печатных устройств;
3) пополнение буфера печатных устройств объектами для печати с учетом логики маршрутизации кодов согласно типам печатных устройств и количеству печатных устройств каждого типа;
4) отправка результатов нанесения в модуль управления заказами;
5) ведение статусной модели процесса нанесения кодов.

Модуль управления печатными устройствами не предусматривает пользовательский интерфейс. Настройка и управление печатными устройствами реализуется посредством единого пользовательского интерфейса модуля управления заказами.

2.2.3.1. Драйвер линии Sewtec

Драйвер линии Sewtec предназначен для формирования инструкций линии Sewtec, загрузки шаблонов печати и изображений, а также обеспечения бесперебойного процесса нанесения кодов агрегации путем пополнения буферов печатных устройств линии Sewtec объектами для печати.

Драйвер линии Sewtec реализует следующие автоматизированные функции:

1) получение шаблонов печати и их публикация в сетевую директорию, доступную линии Sewtec;
2) получение изображений, используемых при печати, и их публикация в сетевую директорию, доступную линии Sewtec;

3) формирование файлов-инструкций для линии Sewtec в соответствии с характеристиками и настройками производственного заказа;
4) публикация файлов-инструкций в сетевую директорию, доступную линии Sewtec;
5) получение кодов агрегации, их преобразование в объекты для печати, пополнение буфера линии Sewtec объектами для печати.

2.2.3.2. Драйвер принтера Videojet (Ultimate protocol)

Драйвер принтера Videojet предназначен для загрузки в принтер Videojet шаблонов печати, а также обеспечения бесперебойного процесса нанесения кодов маркировки путем пополнения буферов принтеров Videojet объектами для печати.

Драйвер принтера Videojet реализует следующие автоматизированные функции:

1) получение шаблонов печати и их загрузка в принтеры Videojet;
2) получение кодов маркировки, их преобразование в объекты для печати, пополнение буферов принтеров Videojet объектами для печати.

2.2.4. Описание модуля архивного хранения

Модуль архивного хранения предназначен для фиксации и долгосрочного хранения сведений о нанесении кодов маркировки, агрегации продукции по типам упаковки (потребительская в групповую и групповая в транспортную), паллетной агрегации.

Модуль архивного хранения реализует следующие автоматизированные функции:

1) получение и сохранение отчетов о нанесении;
2) получение и сохранение отчетов об агрегации;
3) получение и сохранение сведений о паллетной агрегации.

Модуль архивного хранения не предусматривает пользовательский интерфейс.

2.2.5. Описание модуля взаимодействия со станцией управления заказа (далее – СУЗ)

Модуль взаимодействия с СУЗ предназначен для обеспечения возможности обмена компонентов ПО с СУЗ (в локальной и облачной реализации) согласно определенным форматам и спецификациям взаимодействия.

Модуль взаимодействия с СУЗ реализует следующие автоматизированные функции:

1) динамическое получение токена доступа (для облачной СУЗ);
2) эмиссия кодов маркировки;
3) регистрация отчета о нанесения кодов маркировки;
4) регистрация отчета об агрегации.

Модуль взаимодействия с СУЗ не предусматривает пользовательский интерфейс. Настройка и управление подключениями к СУЗ реализуется посредством единого пользовательского интерфейса модуля управления заказами.

2.2.6. Модуль взаимодействия с ESB-шиной

Модуль взаимодействия с ESB-шиной предназначен для получения производственных заказов и направлении сведений о готовой продукции.

Модуль взаимодействия с ESB-шиной реализует следующие автоматизированные функции:

1) получение производственных заказов из ESB-шины;
2) передача производственных заказов в модуль управления заказами;
3) получение сведений о готовой продукции из модуля управления заказами;
4) передача сведений о готовой продукции в ESB-шину.

Модуль взаимодействия с ESB-шиной не предусматривает пользовательский интерфейс. Настройка подключения к ESB-шине реализуется посредством единого пользовательского интерфейса модуля управления заказами.

2.2.7. Модуль взаимодействия с Factory Layer

Модуль взаимодействия с Factory Layer предназначен для взаимодействия с имеющимся программным обеспечением (Factory Layer) в части получения и передачи информации о кодах маркировки и кодах агрегации.

Модуль взаимодействия с Factory Layer реализует следующие автоматизированные функции:

1) получение запросов статуса короба с произведенной продукцией;
2) передача запросов статуса в Factory Layer;
3) передача запросов статуса в модуль обработки событий технологического процесса;
4) получение информации об агрегации от производственной линии;
5) передача информации об агрегации в Factory Layer;
6) передача информации об агрегации в шину данных.

2.2.8. Модуль управления заказами на производственной линии (Терминал)

Модуль управления заказами на производственной линии (Терминал) предназначен для управления заказами с терминала оператора производственной линии.

Модуль управления заказами на производственной линии (Терминал) реализует следующие автоматизированные функции:

1) отображение производственных заказов;
2) запуск, остановка и завершение производственных заказов;
3) формирование и печать паллетного паспорта;
4) повторная печать паллетного паспорта;
5) создание заказов на переупаковку;
6) запуск, остановка и завершение заказов на переупаковку.

2.2.9. Описание сервиса миграций

Сервис миграций отвечает в приложении за применение изменений схемы базы данных на PostgreSQL. Сервис работает в автоматическом режиме: запускается, выполняет миграции и выключается.

2.2.10. Система мониторинга

Система мониторинга выполнена на базе программного обеспечения Grafana, Victoria Metrics, ElasticSearch. Мониторинг выполняется с целью наблюдения за работоспособностью и за уровнем нагруженности ПО.

Взаимодействие с системой мониторинга осуществляется в рамках решения, основанного на базе системы Prometheus. Сервер Prometheus с заданными администратором интервалами при помощи языка PromQL собирает значения по требуемым метрикам от программных компонентов ПО, затем сохраняет эти данные в СУБД временных рядов Victoria Metrics. Для визуализации и контроля работы ПО сервис «Система мониторинга» использует Grafana.

2.2.11. Описание брокера RabbitMQ

RabbitMQ – это брокер сообщений с открытым исходным кодом. Он маршрутизирует сообщения по всем базовым принципам протокола AMQP, описанным в спецификации. Отправитель передает сообщение брокеру, а тот доставляет его получателю. RabbitMQ реализует и дополняет протокол AMQP.

Основная идея модели обмена сообщениями в RabbitMQ заключается в том, что producer (издатель) не отправляет сообщения непосредственно в очередь.

Подробные сведения можно найти на сайте https://www.rabbitmq.com.

2.2.12. Описание баз данных ПО

Информационное обеспечение ПО реализует возможность ввода, обработки, накопления и хранения информации. Вся информация, содержащаяся в ПО, хранится централизованно и размещается в базах данных.

В качестве операционной базы данных ПО применяется СУБД PostgreSQL. Подробную информацию об этой СУБД можно найти на сайте https://www.postgresql.org/.

Также предусмотрена встроенная база данных H2 - локальное хранилище модуля управления печатными устройствами.

Операционная база данных, хранящая все данные, с которыми работает ПО в ходе своего функционирования, представляет собой набор следующих таблиц – см. таблицу ниже.

Таблица 6. Состав таблиц базы данных ПО.

№	Наименование таблицы	Описание
1.	Схема esb_adapter	

№	Наименование таблицы	Описание
1.1.	databasechangelog	Сервисная таблица для обеспечения миграций
1.2.	databasechangeloglock	Сервисная таблица для обеспечения миграций
1.3.	message_log	Таблица для хранения истории входящих и исходящих конвертов
1.4.	shedlock	Таблица для синхронизации шедулеров при поднятии более одного инстанса одного сервиса
2.	Схема factory_layer_adapter	
2.1.	aggregated_code	Таблица для хранения агрегации по кодам маркировки
2.2.	databasechangelog	Сервисная таблица для обеспечения миграций
2.3.	databasechangeloglock	Сервисная таблица для обеспечения миграций
2.4.	pallet	Таблица для хранения информации о произведенных паллетах
2.5.	shedlock	Таблица для синхронизации шедулеров при поднятии более одного инстанса одного сервиса
3.	Схема fm_manager	
3.1.	databasechangelog	Сервисная таблица для обеспечения миграций
3.2.	databasechangeloglock	Сервисная таблица для обеспечения миграций
3.3.	line	Таблица для хранения атрибутов линий фильтр делательного производства
3.4.	outbox	Таблица для обеспечения сохранения в БД и отправки сообщения в MQ в рамках одной транзакции
3.5.	pallet	Таблица для хранения произведенных паллет фильтроделательного производства

№	Наименование таблицы	Описание
3.6.	pallet_sequence	Таблица для хранения количества произведенных паллет в рамках фильтроделательного производства в разрезе линии
3.7.	product	Справочник продуктов фильтроделательного производства
3.8.	product_order	Реестр производственных заказов фильтроделательного производства
3.9.	shedlock	Таблица для синхронизации шедулеров при поднятии более одного инстанса одного сервиса
3.10.	terminal	Таблица для хранения атрибутов терминалов фильтроделательного производства
4.	Схема oms_order_creator	
4.1.	databasechangelog	Сервисная таблица для обеспечения миграций
4.2.	databasechangeloglock	Сервисная таблица для обеспечения миграций
4.3.	shedlock	Таблица для синхронизации шедулеров при поднятии более одного инстанса одного сервиса
5.	Схема oms_order_loader	
5.1.	databasechangelog	Сервисная таблица для обеспечения миграций
5.2.	databasechangeloglock	Сервисная таблица для обеспечения миграций
6.	Схема oms_report_sender	
6.1.	databasechangelog	Сервисная таблица для обеспечения миграций
6.2.	databasechangeloglock	Сервисная таблица для обеспечения миграций

№	Наименование таблицы	Описание
6.3.	shedlock	Таблица для синхронизации шедулеров при поднятии более одного инстанса одного сервиса
7.	Схема operational_service	
7.1	application_report_batch	Таблица для хранения отчетов о нанесении и агрегации по произведенным коробам
7.2	archived_order	Таблица для хранения списка архивированных заказов
7.3	box_to_aggregate	Таблица для хранения списка коробов, готовых к паллетированию
7.4	codes	Таблица для хранения кодов маркировки
7.5	completed_order	Таблица для хранения списка завершенных производственных заказов
7.6	databasechangelog	Сервисная таблица для обеспечения миграций
7.7	databasechangeloglock	Сервисная таблица для обеспечения миграций
7.8	disaggregated_unit	Таблица для хранения истории переагрегации кодов групповых и транспортных упаковок
7.9	emission_batch	Таблица для хранения батчей эмитированных кодов маркировки
7.10	line_pallet_count	Таблица для хранения количества произведенных паллет в разрезе линии
7.11	order_print_audit	Лог производственных заказов
7.12	outbox	Таблица для обеспечения сохранения в БД и отправки сообщения в MQ в рамках одной транзакции
7.13	pallet	Таблица для хранения произведенных паллет
7.14	product_info	Справочник продуктов с атрибутами
7.15	shedlock	Таблица для синхронизации шедулеров при поднятии более одного инстанса одного сервиса

№	Наименование таблицы	Описание
7.16	temp_repacking_codes	Таблица для хранения восстановленной агрегации по коробу при переупаковке
7.17	use_codes_agg	Таблица для хранения промежуточных агрегатов для отчета по использованию кодов
8.	Схема po_manager	
8.1	audit	Таблица аудита изменений схемы
8.2	databasechangelog	Сервисная таблица для обеспечения миграций
8.3	databasechangeloglock	Сервисная таблица для обеспечения миграций
8.4	driver	Справочник типов драйверов устройств
8.5	operator_working_shift	Лог производственных смен
8.6	operator_working_shift_order	Справочник производственных заказов в разрезе смен
8.7	outbox	Таблица для обеспечения сохранения в БД и отправки сообщения в MQ в рамках одной транзакции
8.8	printer_adapter_status	Таблица для хранения статусов принтер-адаптеров
8.9	printer_adapter_status_history	Таблица для хранения истории статусов принтер-адаптеров
8.10	printer_properties	Таблица для хранения параметров принтеров
8.11	printer_property_definitions	Таблица для хранения перечня возможных настроек принтеров
8.12	printer_settings	Таблица для хранения настроек принтеров
8.13	property_info	Реестр со всеми возможными атрибутами для продукции.
8.14	property_type_binding	Реестр с маппингами между атрибутом и условиями его применимости для конкретного принтера, драйвера

№	Наименование таблицы	Описание
8.15	shedlock	Таблица для синхронизации шедулеров при поднятии более одного инстанса одного сервиса
8.16	user_credentials	Таблица для хранения данных пользователей
9.	Схема public	
9.1	bitmap	Справочник изображений для шаблонов печати
9.2	bitmap_template_binding	Таблица для хранения отношений продукта к шаблонам и изображениям шаблонов
9.3	country	Справочник стран
9.4	databasechangelog	Сервисная таблица для обеспечения миграций
9.5	databasechangeloglock	Сервисная таблица для обеспечения миграций
9.6	error_journal	Лог ошибок сервисов
9.7	factory_info	Таблица для хранения сведений о фабрике
9.8	line	Справочник линий
9.9	line_queue	Справочник очередей RMQ
9.10	material_specifications	Справочник ресурсных спецификаций
9.11	oms_connection_status	Таблица для хранения информации о статусе подключения к СУЗ(ам)
9.12	oms_orders	Таблица для хранения заказов в СУЗ на эмиссию кодов маркировки
9.13	oms_registered_connections	Справочник СУЗов
9.14	po_batch	Таблица хранения батчей кодов маркировки для отправки на принтер-адаптер
9.15	product	Справочник продуктов
9.16	product_aggregation_template_binding	Таблица для хранения маппинга уровней упаковки и шаблонов печати

№	Наименование таблицы	Описание
9.17	product_attribute	Справочник атрибутов продуктов для шаблонов печати
9.18	product_order	Реестр производственных заказов
9.19	product_order_counter	Таблица для хранения счетчиков различных статусов кодов в разрезе производственного заказа
9.20	production_mode	Справочник режимов производства
9.21	report_batch	Таблица для хранения отчетов о нанесении и агрегации
9.22	system_setting	Системные настройки
9.23	template	Справочник шаблонов печати

2.3. Связи с другими информационными системами

В процессе функционирования ПО взаимодействует со следующими смежными системами:

- с системой проверки электронной подписи «КриптоПро SVS»;
- с Государственной информационной системой мониторинга за оборотом товаров, подлежащих обязательной маркировке средствами идентификации;
- 1С.

3. НАСТРОЙКА ПО

3.1. Общие сведения по настройке ПО

Для обеспечения требуемого уровня сопровождаемости разработка ПО велась в соответствии с практиками «непрерывная интеграция/непрерывное развертывание» (CI/CD, Continuous Integration/Continuous Delivery).

Разработка велась с использованием инструмента GitLab, который имеет встроенную систему контроля версий и позволяет выполнять совместную разработку силами нескольких команд, применять обновления кода, выполнять сборку и при необходимости откатывать изменения. Сервер GitLab развернут на оборудовании, используемым Аймарк на основании договора.

Сборка релизов происходит на в среде непрерывной интеграции и доставки GitLab CI/CD в соответствии со скриптом (сценарием) сборки приложения. Также на технических средствах развернута тестовая среда (так называемый, контур разработки и тестирования), в которой происходит развертывание разработанных компонентов программного обеспечения их автономное и совместное тестирование.

Для доставки ПО в промышленную среду заказчика создаются docker-образы сервисов приложения и передаются в контур заказчика для развертывания и настройки. В качестве опытной среды используется контур тестирования и разработки. Предполагается развертывание ПО в составе общей сборки с компонентами прикладной инфраструктуры и технологическими компонентами.

Развертывание ПО выполняется в следующей последовательности:

1) общая подготовка серверов;
2) установка docker;
3) установка docker-registry;
4) установка rancher;
5) установка кластера Kubernetes;
6) инициализация кластера Kubernetes;
7) получение kubeconfig;
8) развертывание системных сервисов в кластер;
9) развертывание компонентов системы мониторинга;
10) установка компонентов системы логирования;
11) развертывание и настройка кластера СУБД PostgreSQL.

3.2. Общая подготовка серверов

Установка производится на предустановленную операционную систему Ubuntu Server 22.04.

Для настройки используется утилита ansible версии не ниже 2.13.

3.3. Установка кластера Kubernetes

Создать inventory-файл, с указанием ip-адресов виртуальных машин, следующего содержания:

```
all:
  vars:
    domain: "itobacco.i-tob.net"
    dnsServers:
      - 10.179.3.71
      - 10.179.3.72
    ntp-server:
      - 10.179.3.71
      - 10.179.3.72
  hosts:
    docker-registry:
      ansible_host: 10.179.51.5
    k8s-master-1:
      ansible_host: 10.179.51.10
    k8s-master-2:
      ansible_host: 10.179.51.11
    k8s-master-3:
      ansible_host: 10.179.51.12
    elastic-1:
      ansible_host: 10.179.51.21
    elastic-2:
      ansible_host: 10.179.51.22
    elastic-3:
      ansible_host: 10.179.51.23
    k8s-worker-1:
      ansible_host: 10.179.51.31
    k8s-worker-2:
      ansible_host: 10.179.51.32
    k8s-worker-3:
      ansible_host: 10.179.51.33
    victoria-1:
      ansible_host: 10.179.51.41
```

```
  victoria-2:
    ansible_host: 10.179.51.42
  victoria-3:
    ansible_host: 10.179.51.43
  data-exchange-1:
    ansible_host: 10.179.51.61
  data-exchange-2:
    ansible_host: 10.179.51.62
  data-exchange-3:
    ansible_host: 10.179.51.63
children:
  k8s-cluster:
    vars:
      k8s_labels: []
      k8s_taints: []
    children:
      kube-node:
        hosts:
          k8s-worker-1:
          k8s-worker-2:
          k8s-worker-3:
      kube-master:
        hosts:
          k8s-master-1:
          k8s-master-2:
          k8s-master-3:
      logging-nodes:
        hosts:
          elastic-1:
          elastic-2:
          elastic-3:
      mq-nodes:
        hosts:
          data-exchange-1:
          data-exchange-2:
          data-exchange-3:
```

```
      monitoring-nodes:
        hosts:
          victoria-1:
          victoria-2:
          victoria-3:
docker-nodes:
  hosts:
    docker-registry:
    k8s-master-1:
    k8s-master-2:
    k8s-master-3:
    elastic-1:
    elastic-2:
    elastic-3:
    k8s-worker-1:
    k8s-worker-2:
    k8s-worker-3:
    victoria-1:
    victoria-2:
    victoria-3:
    data-exchange-1:
    data-exchange-2:
    data-exchange-3:
docker-registries:
  hosts:
    docker-registry:
rancher:
  hosts:
docker-registry:
```

3.4. Установка docker

В рабочей папке ansible запустить ansible-playbook с следующими параметрами:

```
ansible-playbook -i environment/prod/inventory.yaml \
playbooks/service/docker.yaml \
playbooks/service/timesyncd.yaml \
-u %USER_NAME%
```

Значение %USER_NAME% - имя пользователя, с привилегиями администратора, от которого производятся операции в системе виртуальной машины.

3.5. Установка docker-registry

В рабочей папке ansible запустить ansible-playbook с следующими параметрами:

```
ansible-playbook -i environment/prod/inventory.yaml \
playbooks/service/docker-registry.yaml -u %USER_NAME%
```

Значение %USER_NAME% - имя пользователя, с привилегиями администратора, от которого производятся операции в системе виртуальной машины.

3.6. Установка Rancher

В файле environment/prod/host_vars/rancher/rancher.yaml изменить значение переменной rancher_datadir на путь до папки хранения данных Rancher.

В рабочей папке ansible запустить ansible-playbook с следующими параметрами:

```
ansible-playbook -i environment/prod/inventory.yaml \
playbooks/service/rancher.yaml -u %USER_NAME%
```

Значение %USER_NAME% - имя пользователя, с привилегиями администратора, от которого производятся операции в системе виртуальной машины.

3.7. Инициализация кластера Kubernetes

1) После установки Rancher открыть веб интерфейс https://%RANCHER_HOST%:8443/g/clusters;
2) Нажать «Add cluster»;
3) Выбрать «Existing nodes»;
4) Установить следующие значения параметров:

```
Cluster Name:  itobacco
    Kubernetes Version: v1.21
    Network Provider: Calico
    Advanced options:
        Nginx Ingress: Disabled
        Node Port Range: 25000-35000
        Metrics Server Monitoring: Disabled
```

5) Далее нажать «Next»;
6) Скопировать значения server, token, ca-checksum и вписать их в файл environment/prod/group_vars/k8s-cluster/rancher.yaml:

```
---
rancher_host: "%server%"
```

```
rancher_token: "%token%"
rancher_ca_checksum: "%ca-checksum%"
```

7) В рабочей папке ansible запустить ansible-playbook с следующими параметрами:

```
ansible-playbook -i environment/prod/inventory.yaml \
        playbooks/rancher-node.yaml -u %USER_NAME%
```

Значение %USER_NAME% - имя пользователя, с привилегиями администратора, от которого производятся операции в системе виртуальной машины.

8) Дождаться инициализации кластера Kubernetes.

3.8. Получение kubeconfig

1) Открыть веб интерфейс https://%RANCHER_HOST%:8443/g/clusters;
2) Выбрать кластер;
3) Нажать «Kubeconfig File» и сохранить содержимое в домашнюю директорию в файл .kube/config.

3.9. Установка системных сервисов в кластер

Выполнить следующие команды:

```
# CertManager
helm -n cert-manager --create-namespace install cert-manager \
 ./k8s/helm/charts/cert-manager -f\
 ./k8s/helm/values/prod/cert-manager/values.yaml
# Ingress controller
helm -n ingress-nginx --create-namespace install nginx \
 ./k8s/helm/charts/ingress-nginx \
 -f ./k8s/helm/values/prod/ingress-nginx/values.yaml
# Load Balancer
helm -n metallb-system --create-namespace install metallb \
 ./k8s/helm/charts/metallb \
 ./k8s/helm/values/prod/metallb/values.yaml
helm -n metallb-system install extra-manifests \
 ./k8s/helm/charts/metallb \
 ./k8s/helm/values/prod/extra-manifests/metallb-system.yaml
# Storage controller
helm -n kube-system install csi-driver-smb \
 ./k8s/helm/charts/csi-driver-smb \
 ./k8s/helm/values/prod/csi-driver-smb/values.yaml
helm -n kube-system install local-path-provisioner \
 ./k8s/helm/charts/local-path-provisioner \
```

```
 ./k8s/helm/values/prod/local-path-provisioner/values.yaml
# RabbitMQ Operator
helm -n kube-system install rabbitmq-operator \
 ./k8s/helm/charts/rabbitmq-operator \
 ./k8s/helm/values/prod/rabbitmq-operator/values.yaml
```

3.10. Установка компонентов системы мониторинга

Выполнить следующие команды:

```
# VictoriaMetricsOperator &  VictoriaMetrics
helm -n monitoring --create-namespace install victoria-metrics \
 ./k8s/helm/charts/victoria-metrics \
 -f ./k8s/helm/values/prod/victoria-metrics/values.yaml
# GrafanaOprtator & Grafana
helm -n monitoring install grafana-operator \
 ./k8s/helm/charts/grafana-operator \
 -f ./k8s/helm/values/prod/grafana-operator/values.yaml
# Extra manifests
helm -n monitoring install extra-manifests \
 ./k8s/helm/charts/extra-manifests \
 -f ./k8s/helm/values/prod/extra-manifests/values-monitoring.yaml
# Dashboards & Rules & AlertRules
helm -n monitoring install monitoring-app \
 ./k8s/helm/charts/monitoring-app \
 -f ./k8s/helm/values/prod/monitoring/common.yaml \
 -f ./k8s/helm/values/prod/monitoring/monitoring-app.yaml
helm -n monitoring install monitoring-db \
 ./k8s/helm/charts/monitoring-db \
 -f ./k8s/helm/values/prod/monitoring/common.yaml \
 -f ./k8s/helm/values/prod/monitoring/monitoring-db.yaml
helm -n monitoring install monitoring-infra \
 ./k8s/helm/charts/monitoring-infra \
 -f ./k8s/helm/values/prod/monitoring/common.yaml \
 -f ./k8s/helm/values/prod/monitoring/monitoring-infra.yaml
helm -n monitoring install monitoring-k8s \
 ./k8s/helm/charts/monitoring-k8s \
 -f ./k8s/helm/values/prod/monitoring/common.yaml \
 -f ./k8s/helm/values/prod/monitoring/monitoring-k8s.yaml
```

```
helm -n monitoring install monitoring-mq \
 ./k8s/helm/charts/monitoring-mq \
 -f ./k8s/helm/values/prod/monitoring/common.yaml \
 -f ./k8s/helm/values/prod/monitoring/monitoring-mq.yaml
```

3.11. Установка компонентов системы логирования

Выполнить следующие команды:

```
# FluentBit
helm -n logging-system --create-namespace install fluent-bit \
 ./k8s/helm/charts/fluent-bit \
 -f ./k8s/helm/values/prod/fluent-bit/values.yaml
# OpenSearch
helm -n logging-system install opensearch-master \
 ./k8s/helm/charts/opensearch \
 -f ./k8s/helm/values/prod/opensearch/common.yaml \
 -f ./k8s/helm/values/prod/opensearch/opensearch-master.yaml
helm -n logging-system install opensearch-data \
 ./k8s/helm/charts/opensearch \
 -f ./k8s/helm/values/prod/opensearch/common.yaml \
 -f ./k8s/helm/values/prod/opensearch/opensearch-data.yaml
helm -n logging-system install opensearch-ingest \
 ./k8s/helm/charts/opensearch \
 -f ./k8s/helm/values/prod/opensearch/common.yaml \
 -f ./k8s/helm/values/prod/opensearch/opensearch-ingest.yaml
```

3.12. Развертывание кластера СУБД PostgreSQL

Создать inventory-файл, с указанием ip-адресов виртуальных машин, следующего содержания:

```
all:
  vars:
    ansible_python_interpreter: '/usr/bin/python3'
    subnet_cidr: "10.179.50.0/23"
    nameservers:
      - 10.179.3.71
      - 10.179.3.72
    ntpservers:
      - 10.179.3.71
```

```
    - 10.179.3.72
  hosts:
    etcd-node-1:
      ansible_host: 10.179.51.71
    etcd-node-2:
      ansible_host: 10.179.51.72
    etcd-node-3:
      ansible_host: 10.179.51.73
    db-node-1:
      ansible_host: 10.179.51.81
    db-node-2:
      ansible_host: 10.179.51.82
    db-node-3:
      ansible_host: 10.179.51.83

etcdcluster:
  hosts:
    etcd-node-1:
    etcd-node-2:
    etcd-node-3:
balancers:
  hosts:
    db-node-1:
    db-node-2:
    db-node-3:
master:
  hosts:
    db-node-1:
replica:
  hosts:
    db-node-2:
    db-node-3:
pgcluster:
  children:
    master:
    replica:
```

В рабочей папке ansible запустить ansible-playbook с следующими параметрами:

```
ansible-playbook -i environment/prod/inventory.yaml \
        playbooks/deploy_pgcluster.yml\
        -u %USER_NAME%
    ansible-playbook -i environment/prod/inventory.yaml \
        playbooks/configure_pg_cluster.yaml\
        -u %USER_NAME%
    ansible-playbook -i environment/prod/inventory.yaml \
        playbooks/monitoring.yaml \
        -u %USER_NAME%
```

Значение %USER_NAME% - имя пользователя, с привилегиями администратора, от которого производятся операции в системе виртуальной машины.

4. АДМИНИСТРИРОВАНИЕ ПО

Администрирование ПО включает в себя следующие основные задачи, выполняемые эксплуатирующим персоналом:

- сбор статистики и мониторинг за корректным функционированием ПО;
- определение и устранение возникающих проблем в работе ПО;
- обновление ПО;
- перезапуск серверов.

5. ПРОВЕРКА ПО

В интерфейсе сервера Rancher отображается список микросервисов ПО и их текущий статус.

Основными критериями успешности установки очередной версии являются:

1) запущены все компоненты ПО;
2) отсутствуют сообщения уровня ERROR в событиях сервисов;
3) все события сервисов сохраняются в систему агрегации логов ElasticSearch и доступны для просмотра и поиска в Kibana;
4) необходимо применить фильтр по атрибуту «level» и значением ERROR;
5) компоненты доступны и работоспособны;
6) доступен веб-интерфейс.