

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

Факультет
вычислительной техники

Кафедра
МОиПЭВМ

Направление подготовки 09.03.02 «Информационные системы и технологии»

Направленность Информационные системы и технологии

БАКАЛАВРСКАЯ РАБОТА

на тему

**Механизм синхронизации полиструктурированных
данных в режиме, близком к реальному времени на
основе триггерных событий**

Студент _____ Федулова Юлия Алексеевна
(подпись, дата) (ФИО полностью)

Руководитель _____ Попова Н.А.
(подпись, дата) (фамилия, инициалы)

Нормоконтролёр _____ Такташкин Д.В.
(подпись, дата) (фамилия, инициалы)

Работа допущена к защите (протокол заседания кафедры от 03.06.2026 г. № 12)

Заведующий кафедрой _____ Козлов А.Ю.
(подпись) (фамилия, инициалы)

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

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

Пенза, 2026

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

Факультет

Вычислительной техники

Кафедра

МО и ПЭВМ

«Утверждаю»
Заведующий кафедрой
_____ А.Ю. Козлов «26»
декабря 2025 г

**ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
БАКАЛАВРА**

1. Студент _____ Федулова Юлия Алексеевна _____ гр. 22ВИ1 _____ факультета ВТ
<Фамилия, имя, отчество полностью>

2. Тема работы _____ Механизм синхронизации полиструктурированных данных в режиме,
близком к реальному времени на основе триггерных событий

Тема утверждена приказом ПГУ №1583/О от "30" декабря 2025 г.

3. Руководитель работы к.т.н. доцент, доцент кафедры «МО и ПЭФВМ» Попова Н.А.

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

Назначение: _____ разработка _____ автоматизированного _____ механизма _____ синхронизации
полиструктурированных данных в режиме, близком к реальному времени на основе
триггерных событий

Среда выполнения: ОС Windows

Языковые средства: 1C, Java, JavaScript, UML

Инструментальные средства: «1C: Предприятие», NetBeans, MS SQL Server, Visual
Paradigm

Технологии разработки: фреймворк Spring Boot, событийная модель (триггеры), паттерн
Transactional Outbox, REST API, многослойная архитектура

Основные функции:

а) автоматический захват изменений объектов (создание, изменение, удаление);

б) асинхронное накопление заданий в очереди синхронизации;

в) формирование и трансляция полиструктурированных JSON-пакетов;

г) прием данных и их сохранение в реляционной СУБД;

д) динамическая визуализация и оперативный мониторинг бизнес-показателей.

5. Перечень подлежащих разработке вопросов

1) Анализ предметной области и постановка задачи

2) Проектирование;

3) Реализация;

4) Контроль качества системы;

5) Оформление пояснительной записки.

6. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1	Анализ предметной области и требований к системе	10%	08.04.2026 – 10.04.2026	
2	Проектирование программных средств	20%	11.04.2026 – 20.04.2026	
3	Реализация программных средств	30%	21.04.2026 – 06.05.2026	
4	Тестирование программных средств	20%	07.05.2026 – 09.05.2026	
5	Оформление пояснительной записки	20%	10.05.2026 – 28.06.2026	

Дата выдачи задания «26» декабря 2025г.

Руководитель бакалаврской работы _____ 26.12.2025 г.
(подпись, дата)

Задание к исполнению принял студент _____ 26.12.2025 г.
(подпись, дата)

Работу к защите допустить
Декан ФВТ _____ Л.Р. Фионова
(подпись, дата)

Реферат

Пояснительная записка содержит 124 листа, 47 рисунок, 34 таблицы, 21 использованный источник и 5 приложений.

1С:ПРЕДПРИЯТИЕ 8.3, JAVA, SPRING BOOT, JAVASCRIPT, СИНХРОНИЗАЦИЯ ДАННЫХ, ПОЛИСТРУКТУРИРОВАННЫЕ ДАННЫЕ, РЕЖИМ БЛИЗКИЙ К РЕАЛЬНОМУ ВРЕМЕНИ, REST API, JSON, ТРИГГЕРНЫЕ СОБЫТИЯ.

Объект работы: процесс информационного взаимодействия между корпоративными учётными системами и внешними сервисами мониторинга.

Предмет работы: разработка модуля синхронизации полиструктурированных данных в режиме, близком к реальному времени, для обеспечения актуальности показателей системы мониторинга.

Цель работы: проектирование и реализация программного средства автоматизированной синхронизации данных.

Средства разработки: «1С:Предприятие 8.3», Java, MS SQL Server, Spring Boot, JavaScript, Visual Paradigm.

Результаты работы: спроектирован и реализован механизм синхронизации полиструктурированных данных. Механизм работает в режиме, близком к реальному времени, и обеспечивает актуальность бизнес-показателей предприятия на веб-портале мониторинга.

Область применения: предприятие, использующее учётную систему 1С и нуждающееся в оперативном мониторинге данных через веб-портал.

					ПГУ 09.03.02–8BP221.18 ПЗ			
Изм.	Лист	№ докум.	Подпись	Дата				
Разраб.		Федулова Ю.А.			«Механизм синхронизации полиструктурированных данных в режиме, близком к реальному времени на основе триггерных событий» Пояснительная записка	Лит.	Лист	Листов
Провер.		Попова Н.А.					4	124
Н. Контр.		Такташкин Д.В.				Группа 22ВИ1		
Утверд.		Козлов А.Ю.						

Содержание

1	Постановка задачи и анализа требований к механизму синхронизации полиструктурированных данных в режиме близкому к реальному времени на основе триггерных событий	10
1.1	Анализ предметной области и постановка задачи	10
1.2	Сравнительный анализ существующих решений	11
1.3	Выбор технологии проектирования	13
1.4	Выбор инструментальных средств.....	14
1.5	Моделирование функциональных требований к механизму синхронизации	16
1.6	Моделирование нефункциональных требований к механизму синхронизации	18
1.7	Планирование разработки и оценка бюджета	19
2	Проектирование механизма синхронизации	22
2.1	Обоснование архитектурных решений и выбор стратегии	22
2.1.1	Сравнительный анализ и выбор архитектурной стратегии взаимодействия	22
2.1.2	Триггерный механизм регистрации изменений в системе-источнике.....	25
2.2	Проектирование структур данных и моделей хранения	25
2.2.1	Структура данных системы-источника (1С).	26
2.2.2	Структура данных системы-приемника	31
2.2.3	Спецификации сопоставления и правила трансформации данных	37
2.3	Моделирование динамики и алгоритмов функционирования.....	46
2.3.1	Проектирование динамики межсистемного взаимодействия.....	47
2.3.2	Алгоритмизация процедур обработки и синхронизации данных	48
2.3.3	Формализация жизненного цикла объектов обмена.	50

2.4	Проектирование архитектуры системы-приемника.....	51
2.4.1	Проектирование программной архитектуры и компонентной модели.....	51
2.4.2	Проектирование механизмов визуализации данных	55
2.5	Проектирование архитектуры развертывания и логической структуры системы	57
3	Программная реализация механизма синхронизации.....	60
3.1	Реализация системы-источника	60
3.1.1	Реализация компонентов системы-источника.....	60
3.1.2	Реализация событийного механизма захвата изменений	70
3.1.3	Программная реализация алгоритмов обмена в системе- источнике.....	72
3.1.4	Реализация режима, близкого к реальному времени	77
3.2	Программная реализация принимающей стороны.....	78
3.2.1	Реализация серверной части.....	78
3.2.2	Реализация механизмов обработки полиструктурированных данных	83
3.2.3	Реализация клиентской части.....	84
4	Контроль качества механизма синхронизации.....	86
4.1	Функциональное тестирование	86
4.1.1	Классы эквивалентности	86
4.1.2	Граничные значения	88
4.2	Сквозное тестирование	89
4.2	Нагрузочное тестирование	91
4.3	Метрики программного кода.....	95
4.3.1	Метрики SLOC.....	95
4.3.2	Метрики Холстеда	96
	Заключение	99
	Список использованных источников	101
	Приложение А (справочное) Глоссарий	104

Приложение Б (обязательное) Листинг системы-источника 1С	106
Приложение В (обязательное) Листинг серверной части веб-портала	111
Приложение Г (обязательное) Листинг клиентской части веб-портала	114
Приложение Д (рекомендуемое) Результаты сквозного тестирования.....	118

Введение

Эффективное управление современным предприятием невозможно без оперативного доступа к качественной и, что еще важнее, актуальной информации. Сегодня компании вынуждены использовать сразу несколько информационных систем для решения разных задач: одни отвечают за детальный внутренний учет ресурсов, другие обеспечивают удаленный контроль через веб-интерфейсы мониторинга.

Однако в такой архитектуре возникает острая проблема – задержка синхронизации данных. Традиционные методы обмена, основанные на периодической выгрузке больших массивов по редкому расписанию, уже не отвечают современным требованиям рынка. Бизнесу нужен принципиально иной подход. А именно, переход к передаче данных об изменениях в режиме, близком к реальному времени.

Особую сложность при этом представляет работа с информацией, имеющей гибкую или меняющуюся структуру. Проектировать жесткие схемы обмена для таких объектов нецелесообразно. Более того, это технически трудоемко. Поэтому возникает необходимость внедрения механизмов, способных эффективно обрабатывать полиструктурированные данные, то есть объекты с разным набором полей и реквизитов.

Объектом работы является процесс информационного взаимодействия между корпоративными учетными системами и внешними сервисами мониторинга.

Предметом работы является разработка модуля для синхронизации полиструктурированных данных в режиме, близком к реальному времени, с целью обеспечения актуальности показателей в системе мониторинга.

Актуальность темы очевидна. Бизнесу жизненно необходима свежая информация. Традиционные методы пакетной выгрузки этого не дают. Разработанный механизм на основе программных триггеров позволяет мгновенно транслировать изменения полиструктурированных объектов на

внешние панели мониторинга. В итоге повышается оперативность управления, а риск принятия решений на основе устаревших показателей практически исключается.

Целью работы является проектирование и реализация программного средства для автоматизированной синхронизации данных, обеспечивающего актуальность показателей в системе мониторинга. Для достижения поставленной цели необходимо решить следующие задачи:

- определить основные требования к работе будущего модуля;
- изучить предметную область и выбрать наиболее эффективный способ синхронизации данных;
- спроектировать логическую и физическую структуры баз данных (системы-источника и системы-приемника), а также составить графические схемы процессов обмена;
- разработать программные средства для автоматической передачи и приема информации;
- протестировать готовое решение и убедиться в корректности передачи данных.

Реализация такого модуля синхронизации дает ряд важных преимуществ. Процесс обновления сведений между различными узлами информационной среды полностью автоматизируется. Потребность в ручных операциях по переносу данных практически сводится к нулю. За счет того что изменения фиксируются непосредственно в момент их возникновения, обеспечивается высокая прозрачность всех ключевых процессов. Итог очевиден: наличие такого инструмента позволяет принимать управленческие решения, опираясь исключительно на актуальные показатели.

1 Постановка задачи и анализа требований к механизму синхронизации полиструктурированных данных в режиме близкому к реальному времени на основе триггерных событий

1.1 Анализ предметной области и постановка задачи

В деятельности современных коммерческих организаций именно информационные технологии выступают базой для сбора, обработки и хранения сведений по всем регистрируемым операциям. Предметная область данной работы охватывает процессы оперативной синхронизации и обмена данными между внутренней учётной системой предприятия и сторонними программными платформами.

В качестве практической базы взята компания, которая занимается оптово-розничной продажей строительных материалов. Она выбрана, потому что это напрямую связано со спецификой тех сведений, которые требуется передавать. В подобной учётной среде приходится иметь дело с документами, сильно отличающимися друг от друга по своему составу и списку реквизитов. Разные типы документов несут разную информацию, причём конкретный набор полей может дополняться либо изменяться в зависимости от стоящей задачи. Поскольку единого жёсткого шаблона для всех объектов не существует, подобные сведения принято относить к категории полиструктурированных данных [1]. Из этого следует важное ограничение: используемый модуль синхронизации обязан обладать достаточной гибкостью и корректно передавать объекты с совершенно разными наборами полей.

Практика показывает, что довольно часто предприятиям требуются дополнительные (вспомогательные) системы для контроля текущей деятельности. Основной массив данных продолжает обрабатываться в «тяжёлой» учётной системе. Однако для оперативного мониторинга и получения быстрой отчётности гораздо удобнее пользоваться лёгкими веб-интерфейсами. Такое решение даёт руководству возможность видеть ключевые показатели из любой точки, причём запускать специализированное ПО не нужно.

Но при одновременной эксплуатации нескольких независимых систем почти неизбежно возникает проблема рассинхронизации сведений. Информация о новом заказе или поступившей оплате должна отражаться на сайте мониторинга практически мгновенно. Иначе смысл оперативного контроля теряется. Если же применять стандартные методы периодической выгрузки по расписанию, данные на сайте быстро устаревают, а сам мониторинг перестаёт быть эффективным. Поэтому была сформулирована следующая задача: спроектировать и реализовать специализированное средство синхронизации полиструктурированных данных, работающее в режиме, близком к реальному времени. Базовая идея достаточно проста – необходимо создать подсистему, которая отслеживает действия пользователя внутри учётной программы и затем автоматически транслирует все изменения на внешний сервер.

Таким образом, разрабатываемое средство должно решить следующие задачи:

- устранить временной разрыв между совершением операции и её отображением на панели мониторинга;
- обеспечить передачу данных без потерь;
- гарантировать доставку даже при нестабильной работе сети.

Реализация такого модуля создаст надёжную технологическую базу. Это позволит поддерживать актуальность информации. А значит принимать оперативные управленческие решения.

1.2 Сравнительный анализ существующих решений

Для обмена данными между учётной системой и внешними сервисами существует несколько готовых подходов. С целью выяснить, применимы ли существующие решения в рамках данной задачи или требуется разработка собственного механизма, был проведён анализ доступных на рынке аналогов

Поскольку в качестве учётной системы в работе используется платформа «1С:Предприятие», имеет смысл рассмотреть готовые варианты обмена данными, которые предоставляет сама платформа или рынок решений для неё.

Планы обмена – это штатный механизм 1С для синхронизации между двумя базами 1С. Он надёжен и хорошо работает внутри экосистемы. Однако при попытке передавать данные во внешнюю систему (не на 1С) настройка становится сложной. Механизм оказывается тяжёлым, а добиться от него малых задержек при больших объёмах информации трудно.

Готовые коннекторы для связки 1С с популярными сервисами. На рынке существуют модули, которые реализуют обмен между 1С и конкретными внешними системами, например Битрикс, МойСклад или Salesforce. Они работают «из коробки», но каждый такой коннектор жёстко привязан к структуре данных того сервиса, для которого он сделан. В данной работе внешняя система – это самостоятельно разработанный веб-портал. Для него готового коннектора нет. А переделка существующего модуля под новую структуру по трудозатратам не отличается от написания собственного механизма.

Облачные сервисы-посредники (Zapier, Albato, Make и подобные) – Это целые платформы, куда можно подключить одну систему, потом другую, и настроить между ними передачу данных в визуальном редакторе. Это удобно и быстро. Но у такого подхода есть два серьёзных минуса. Первый – все данные проходят через чужие серверы. Для многих компаний это неприемлемо, особенно когда речь идёт о финансах или персональных данных. Второй – такие сервисы платные. Чем больше запросов, тем дороже. И ещё они не дают полного контроля над логикой, например над повторными попытками при сбоях.

На основе рассмотрения всех решений была составлена сводная таблица 1.

Таблица 1 – Сравнение готовых решений синхронизации

Критерий	Планы обмена	Готовые коннекторы	Облачные сервисы
Надёжность	Высокая	Средняя	Средняя
Возможность передачи на произвольный веб-портал	Сложно, много кода	Нет (привязаны к конкретному сервису)	Частично
Пригодность для полиструктурных данных	Низкая	Отсутствует	Ограниченная

Продолжение таблицы 1

Критерий	Планы обмена	Готовые коннекторы	Облачные сервисы
Безопасность (данные не уходят на сторону)	Да	Да	Нет
Стоимость	Бесплатно	Различается	Платная подписка

Готовые решения не удовлетворяют требованиям, предъявляемым к механизму синхронизации в данной работе. Планы обмена сложно настраивать под внешний портал. Коннекторы привязаны к конкретным системам, а здесь внешняя система разрабатывается индивидуально. Облачные сервисы небезопасны, дороги и не дают полного контроля над процессом. Поэтому необходим собственный механизм синхронизации.

1.3 Выбор технологии проектирования

Любой программный продукт начинают проектировать с визуальных моделей – они помогают структурировать логику будущей системы. Для таких задач используется графический язык UML. Это международный стандарт. С его помощью через набор универсальных диаграмм можно описать любые процессы и связи внутри системы, что делает проект понятным для всех участников разработки.

В процессе подготовки работы были изучены различные инструменты для построения UML-моделей. В частности, рассматривалось приложение ArgoUML. Оно является полностью бесплатным и кроссплатформенным благодаря работе на Java. Этот инструмент удобен для быстрого создания базовых схем, но его интерфейс довольно аскетичен. Также изучались возможности StarUML. Данное средство отличается высокой скоростью работы, однако в бесплатной версии имеет серьезные ограничения.

В качестве основного средства проектирования была выбрана среда Visual Paradigm (Community Editor). Этот инструмент представляет собой профессиональное CASE-средство, которое предоставляет разработчику

максимально полный набор функций для графического анализа и моделирования [2].

Удобство пользовательского интерфейса. В отличие от многих аналогов, здесь реализована очень продуманная система навигации. Рабочая область позволяет легко управлять несколькими диаграммами, объединяя их в логические группы. Инструменты рисования поддерживают функцию drag-and-drop, что заметно ускоряет процесс проектирования и дает возможность быстро менять расположение элементов на схеме.

Полноценная поддержка стандартов. Программа корректно работает со всеми типами диаграмм и обеспечивает строгое соблюдение правил нотации UML. Это гарантирует, что создаваемые модели будут технически грамотными и профессиональными.

Высокое качество документации. Одним из важных плюсов для написания пояснительной записки является возможность качественного экспорта изображений. Схемы, созданные в Visual Paradigm, сохраняют четкость при масштабировании, что критично при оформлении дипломной работы.

Гибкость и расширяемость. Среда позволяет адаптировать стандартные элементы под конкретные нужды и поддерживает интеграцию с различными плагинами. Это делает ее универсальным решением, которое можно настроить под требования любого проекта.

1.4 Выбор инструментальных средств

Для создания надёжного механизма синхронизации необходимо тщательно подобрать программные инструменты. Чтобы определиться с технологиями, которые лягут в основу всех компонентов системы, учитывались следующие факторы:

- высокая скорость обработки информационных потоков;
- стабильность работы при постоянных сетевых нагрузках;
- удобство сопровождения и возможность масштабирования в будущем.

С учётом этих критериев в качестве базы для системы-источника данных была выбрана платформа «1С:Предприятие 8.3». Такое решение наиболее рационально. Потому что на этой платформе уже реализована и успешно эксплуатируется конфигурация для учёта деятельности предприятия, занимающегося торговлей строительными материалами. Именно в этой базе хранится вся первичная информация о товарах и операциях, которую предстоит транслировать во внешнюю среду.

Платформа «1С:Предприятие 8.3» представляет собой универсальную программную среду, предназначенную для автоматизации самых разных бизнес-процессов [3]. Её архитектура строится на чётком разделении: технологическая платформа и прикладное решение существуют отдельно. Такое разделение позволяет разработчику сосредоточиться на бизнес-логике, не отвлекаясь на низкоуровневое программирование.

Одна из ключевых особенностей системы – встроенный режим «Конфигуратор». Это полноценная среда разработки. С её помощью (через визуальное проектирование и встроенный язык программирования) можно создавать уникальные структуры данных: справочники, документы, регистры.

Для реализации механизма синхронизации в «1С:Предприятие 8.3» предусмотрен мощный событийный инструментарий. Важнейший элемент здесь «Подписки на события». Они позволяют системе автоматически реагировать на действия пользователя и мгновенно запускать программный код для отправки данных. Благодаря этому отпадает необходимость постоянно опрашивать базу данных, а работа системы приближается к режиму реального времени. Кроме того, платформа обладает развитыми средствами для работы с сетевыми протоколами и форматами передачи данных. Это делает её отличным поставщиком информации для внешних веб-сервисов.

Для реализации принимающей стороны, а также интерфейса мониторинга, выбор остановился на языке Java и фреймворке Spring Boot. Такое сочетание продиктовано необходимостью обеспечить высокую производительность серверной части. Java считается одним из самых надёжных инструментов для

создания серверных приложений, способных работать в непрерывном режиме и обрабатывать большое количество запросов. Фреймворк Spring Boot, в свою очередь, позволяет быстро развернуть программный интерфейс, который будет выступать «входной точкой» для данных [4]. В качестве среды разработки использована NetBeans, обеспечивающая удобный функционал для написания и отладки серверного кода.

В качестве системы управления базами данных на стороне сайта используется реляционная СУБД Microsoft SQL. Она обеспечивает надёжное хранение информации и поддерживает работу с текстовыми полями неограниченной длины. Последнее критически важно для хранения полиструктурных данных.

На клиентской стороне для отображения информации предполагается использование JavaScript. Он позволит гибко работать с полиструктурированными данными, поступающими с сервера.

Сочетание перечисленных инструментов позволяет создать надёжный и производительный механизм синхронизации. Он способен работать в режиме, близком к реальному времени, и обрабатывать данные любой сложности.

1.5 Моделирование функциональных требований к механизму синхронизации

При анализе требований к разрабатываемому механизму синхронизации сначала нужно определить, где проходят границы системы и как её участники взаимодействуют друг с другом. Для этого была построена диаграмма вариантов использования (Use Case Diagram) в нотации UML. С помощью такого инструмента формируется концептуальная модель, причём без привязки к конкретной программной реализации [5].

Был определён список ключевых функций, необходимых для поддержания актуальности данных в распределённой среде предприятия:

- своевременная фиксация изменений (автоматическое обнаружение модификаций объектов в учётной системе);

- регистрация информации для обмена (сохранение нужных сведений об объекте перед отправкой) ;
- подготовка и трансляция данных (формирование информационных пакетов и их передача на принимающую сторону) ;
- актуализация состояния (расшифровка полученных данных и обновление показателей на панели мониторинга) ;
- обеспечение отказоустойчивости (обработка сбоев и планирование повторных попыток обмена).

Визуально эти требования отражены на диаграмме вариантов использования на рисунке 1.

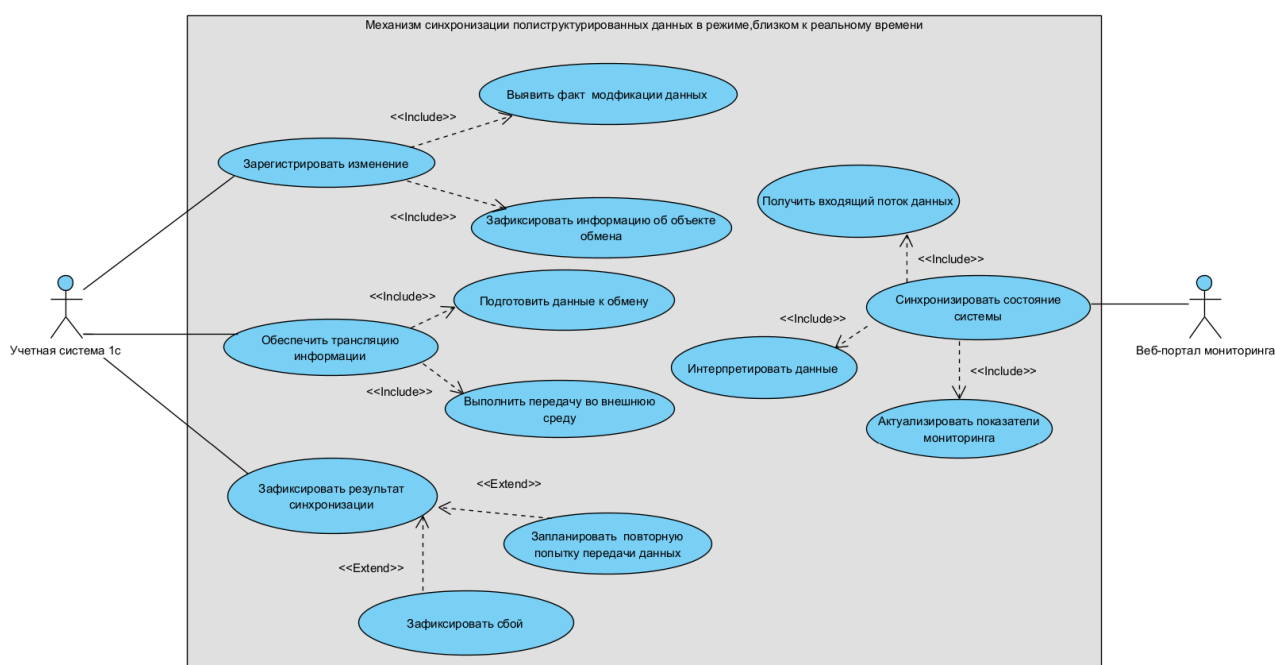


Рисунок 1 – Диаграмма вариантов использования механизма синхронизации

Согласно построенной модели, в системе выделены два ключевых актёра: Учётная система (1С) и Веб-портал мониторинга.

Учётная система (1С) выступает инициатором взаимодействия и источником данных. Её роль сводится к выполнению двух укрупнённых функций. Первая это «Зарегистрировать изменения в системе-источнике». Она включает в себя (include) процедуры обнаружения факта модификации и фиксации информации об объекте обмена. Вторая это «Обеспечить трансляцию

информации». Здесь подразумевается подготовка данных к передаче и непосредственно сама выгрузка во внешнюю среду. Третья это обеспечить обработку сбоев. Веб-портал мониторинга работает как система-приёмник. Его основная задача «Синхронизировать состояние системы». Этот прецедент распадается на подзадачи: приём входящего потока, его интерпретация и обновление визуальных индикаторов на панели мониторинга. Выполнение этих требований даёт конечным пользователям возможность получать актуальную информацию в режиме, близком к реальному времени.

1.6 Моделирование нефункциональных требований к механизму синхронизации

В ходе анализа были сформулированы следующие группы нефункциональных требований.

Требования к объёму и полноте синхронизации данных. Механизм должен обеспечивать корректную обработку и передачу информации обо всех состояниях объектов в учётной системе, а именно о создании новых записей, модификации существующих данных и удалении объектов. Система обязана гарантировать, что информация на принимающей стороне полностью идентична состоянию данных в системе-источнике на момент совершения последней операции.

Требования к производительности и оперативности. Временной интервал между фиксацией события в учётной системе и его отражением на панели мониторинга должен быть минимальным и в штатном режиме не превышать 60 секунд. Выполнение процедур подготовки и трансляции данных не должно приводить к снижению производительности пользовательского интерфейса учётной системы, поэтому весь процесс обмена должен протекать в фоновом режиме. Именно такое требование и подпадает под понятие «режим, близкий к реальному времени». Для данной работы это означает, что задержка не превышает 60 секунд, а сам процесс не мешает текущей работе сотрудников [6].

Требования к надёжности и отказоустойчивости. Ни одно событие в системе-источнике не должно быть проигнорировано.

1.7 Планирование разработки и оценка бюджета

Любую разработку начинают с планирования сроков и затрат. В управлении проектами чаще всего используют два подхода: гибкий и каскадный.

Для этой дипломной работы был выбран каскадный метод. Каждый следующий этап начинается только после того, как предыдущий полностью закончен и задокументирован. Для сокращения общего цикла разработки внутри этапов проектирования и реализации было предусмотрено параллельное выполнение независимых задач. Для отслеживания сроков и трудозатрат была использована программа ProjectLibre. Она бесплатная, работает на разных операционных системах и позволяет рассчитывать критический путь [7].

Весь механизм синхронизации был разбит на четыре крупных блока. В них вошло 30 конкретных задач. По плану начало работ 8.04.2026 года, окончание – 19.05.2026 года. Иерархия задач со сроками показана на рисунке 2.

Название	Продолжительность	Начало	Окончание
Начало проекта	0 дней?	08.04.2026, 08:00	08.04.2026, 08:00
☒ Постановка задачи и анализ требований	4 дней?	08.04.2026, 08:00	13.04.2026, 17:00
Анализ предметной области и постановка задачи	1 день?	08.04.2026, 08:00	08.04.2026, 17:00
Сравнительный анализ существующих решений	0,5 дней?	09.04.2026, 08:00	09.04.2026, 13:00
Выбор технологии проектирования	0,5 дней?	09.04.2026, 13:00	09.04.2026, 17:00
Выбор инструментальных средств	0,5 дней?	10.04.2026, 08:00	10.04.2026, 13:00
Моделирование функциональных требований	0,5 дней?	10.04.2026, 13:00	10.04.2026, 17:00
Моделирование нефункциональных требований	1 день?	13.04.2026, 08:00	13.04.2026, 17:00
☒ Проектирование механизма синхронизации	13,6 дней?	14.04.2026, 08:00	01.05.2026, 13:48
☒ Обоснование архитектурных решений	0,85 дней?	14.04.2026, 08:00	14.04.2026, 15:48
Сравнительный анализ стратегий взаимодействия	0,35 дней?	14.04.2026, 08:00	14.04.2026, 10:48
Триггерный механизм регистрации изменений	0,5 дней?	14.04.2026, 10:48	14.04.2026, 15:48
☒ Проектирование структур данных и моделей хранения данных	3,25 дней?	14.04.2026, 15:48	20.04.2026, 08:48
Структура данных системы-источника	1,25 дней?	14.04.2026, 15:48	16.04.2026, 08:48
Структура данных системы-приемника	0,5 дней?	14.04.2026, 15:48	15.04.2026, 10:48
Спецификации сопоставления и правила трансформации данных	2 дней?	16.04.2026, 08:48	20.04.2026, 08:48
☒ Моделирование динамики и алгоритмов функционирования	3 дней?	20.04.2026, 08:48	23.04.2026, 08:48
Проектирование динамики неконсистентного взаимодействия	1 день?	20.04.2026, 08:48	21.04.2026, 08:48
Алгоритмизация процедур обработки и синхронизации данных	1 день?	21.04.2026, 08:48	22.04.2026, 08:48
Формализация жизненного цикла объектов обмена	1 день?	22.04.2026, 08:48	23.04.2026, 08:48
☒ Проектирование архитектуры системы-приемника	4,5 дней?	23.04.2026, 08:48	29.04.2026, 13:48
Программное и компонентное проектирование	2,5 дней?	23.04.2026, 08:48	27.04.2026, 13:48
Проектирование механизмов визуализации данных	2 дней?	27.04.2026, 13:48	29.04.2026, 13:48
Проектирование архитектуры развертывания и логической структуры	2 дней?	29.04.2026, 13:48	01.05.2026, 13:48
☒ Программная реализация механизма	9,5 дней?	01.05.2026, 13:48	15.05.2026, 08:48
☒ Реализация системы-источника (ИС)	8,5 дней?	01.05.2026, 13:48	14.05.2026, 08:48
Реализация компонентов системы-источника	3 дней?	01.05.2026, 13:48	06.05.2026, 13:48
Реализация событийного механизма захвата изменений	1,5 дней?	06.05.2026, 13:48	08.05.2026, 08:48
Программная реализация алгоритмов обмена	3,5 дней?	08.05.2026, 08:48	13.05.2026, 13:48
Реализация режима, близкого к реальному времени	0,5 дней?	13.05.2026, 13:48	14.05.2026, 08:48
☒ Программная реализация принимающей стороны	9,5 дней?	01.05.2026, 13:48	15.05.2026, 08:48
Реализация серверной части	5 дней?	01.05.2026, 13:48	08.05.2026, 13:48
Реализация механизмов обработки полнструктурированных данных	1,5 дней?	08.05.2026, 13:48	12.05.2026, 08:48
Реализация клиентской части	3 дней?	12.05.2026, 08:48	15.05.2026, 08:48
☒ Контроль качества механизма синхронизации	3 дней?	15.05.2026, 08:48	20.05.2026, 08:48
☒ Функциональное тестирование	0,75 дней?	15.05.2026, 08:48	15.05.2026, 15:48
Классы эквивалентности	0,5 дней?	15.05.2026, 08:48	15.05.2026, 13:48
Граничные значения	0,25 дней?	15.05.2026, 13:48	15.05.2026, 15:48
Сквозное тестирование	0,5 дней?	15.05.2026, 15:48	16.05.2026, 10:48
Нагрузочное тестирование	1 день?	16.05.2026, 10:48	19.05.2026, 10:48
☒ Метрики программного кода	0,75 дней?	19.05.2026, 10:48	20.05.2026, 08:48
Метрики SLOC	0,25 дней?	19.05.2026, 10:48	19.05.2026, 13:48
Метрики Холстеда	0,5 дней?	19.05.2026, 13:48	20.05.2026, 08:48
Завершение проекта	0 дней?	20.05.2026, 08:48	20.05.2026, 08:48

Рисунок 2 – Спецификация этапов и задач процесса разработки

Для визуального анализа последовательности работ и выявления технологических зависимостей в системе была сформирована диаграмма Ганта представленная на рисунке 3. Данная модель наглядно демонстрирует «лесенку» выполнения задач и позволяет контролировать соблюдение сроков на каждом участке пути [8].



Рисунок 3 – Диаграмма Ганта

Важной частью проектного управления является определение необходимых ресурсов и их стоимостных характеристик. Под ресурсами в данном контексте понимаются трудозатраты специалистов, вовлеченных в процесс создания системы. Для каждой роли (аналитик, разработчик 1с, веб разработчик и тестировщик) была установлена часовая ставка, соответствующая текущим рыночным показателям. Перечень задействованных ресурсов и их финансовые параметры отражены на рисунке 4.

Название	Тип	Стандартная ставка
Системный аналитик	Работа	900/час
Разработчик 1с	Работа	1 000/час
Веб-разработчик	Работа	1 400/час
Тестировщик	Работа	800/час

Рисунок 4 – Трудовые ресурсы проекта и их тарифные ставки

Расчет итогового бюджета производился путем суммирования стоимости каждой отдельной задачи, которая вычислялась как произведение фактически затраченного времени на ставку исполнителя. Использование автоматизированных средств расчета в ProjectLibre позволило исключить ошибки в вычислениях и получить детализированную смету проекта. Итоговые данные по оценке бюджета представлены на рисунке 5.

Название	Стоимость
Начало проекта	0,00 Р
☒ Постановка задачи и анализ требований	29400,00 Р
Анализ предметной области и постановка задачи	7200,00 Р
Сравнительный анализ существующих решений	3600,00 Р
Выбор технологии проектирования	3600,00 Р
Выбор инструментальных средств	4200,00 Р
Моделирование функциональных требований	3600,00 Р
Моделирование нефункциональных требований	7200,00 Р
☒ Проектирование механизма синхронизации	101804,00 Р
☒ Обоснование архитектурных решений	2604,00 Р
Сравнительный анализ стратегий взаимодействия	2604,00 Р
Триггерный механизм регистрации изменений	0,00 Р
☒ Проектирование структур данных и моделей хранения данных	30800,00 Р
Структура данных системы-источника	9500,00 Р
Структура данных системы-приемника	6900,00 Р
Спецификации сопоставления и правила трансформации данных	14400,00 Р
☒ Моделирование динамики и алгоритмов функционирования	21600,00 Р
Проектирование динамики межсистемного взаимодействия	7200,00 Р
Алгоритмизация процедур обработки и синхронизации данных	7200,00 Р
Формализация жизненного цикла объектов обмена	7200,00 Р
☒ Проектирование архитектуры системы-приемника	32400,00 Р
Программное и компонентное проектирование	18000,00 Р
Проектирование механизмов визуализации данных	14400,00 Р
Проектирование архитектуры развертывания и логической структуры	14400,00 Р
☒ Программная реализация механизма	174400,00 Р
☒ Реализация системы-источника (1С)	68000,00 Р
Реализация компонентов системы-источника	24000,00 Р
Реализация событийного механизма захвата изменений	12000,00 Р
Программная реализация алгоритмов обмена	28000,00 Р
Реализация режима, близкого к реальному времени	4000,00 Р
☒ Программная реализация принимающей стороны	106400,00 Р
Реализация серверной части	56000,00 Р
Реализация механизмов обработки полиструктурированных данных	16800,00 Р
Реализация клиентской части	33600,00 Р
☒ Контроль качества механизма синхронизации	19200,00 Р
☒ Функциональное тестирование	4800,00 Р
Классы эквивалентности	3200,00 Р
Граничные значения	1600,00 Р
Сквозное тестирование	3200,00 Р
Нагрузочное тестирование	6400,00 Р
☒ Метрики программного кода	4800,00 Р
Метрики SLOC	1600,00 Р
Метрики Холстеда	3200,00 Р
Завершение проекта	0,00 Р
	Итого: 323 704,00 Р

Рисунок 5 – Оценка общего бюджета проекта

Таким образом, на основе проведенного анализа и календарного планирования было установлено, что реализация механизма синхронизации полиструктурированных данных требует 294 рабочих часов. Суммарная стоимость разработки, составила 323 704 рублей 00 копеек.

2 Проектирование механизма синхронизации

2.1 Обоснование архитектурных решений и выбор стратегии

В данном разделе рассматриваются возможные способы организации обмена данными между учётной системой и веб-порталом. На основе анализа выбирается наиболее подходящая стратегия, которая ляжет в основу проектируемого механизма синхронизации.

2.1.1 Сравнительный анализ и выбор архитектурной стратегии взаимодействия

Для обеспечения взаимодействия между ученой системой и внешней панелью мониторинга могут быть применены различные архитектурные стратегии. Выбор конкретного подхода определяет, каким образом изменения в базе данных будут фиксировать и передаваться принимающей стороне. В рамках данного раздела проводится анализ четырех основных стратегий интеграции для обоснования выбора наиболее эффективного решения.

Пакетный файловый обмен. Здесь накопленные изменения выгружаются в текстовые файлы (XML, JSON) по расписанию. Системы взаимодействуют опосредованно через файловую систему или FTP-сервер. Однако, как отмечено в тематических исследованиях [9], такая схема обладает заметной инерционностью. Скорость обновления данных целиком привязана к настройкам планировщика задач. Для оперативного контроля показателей этот вариант малопригоден из-за больших задержек. Кроме того, работа с полиструктурированными объектами через файлы заметно усложняет интерпретацию данных и контроль за целостностью транзакций.

Прямая синхронная API-интеграция. При таком подходе учётная система сама отправляет сетевой запрос к веб-службе непосредственно в момент проведения документа. С точки зрения скорости – это самый оперативный вариант. Но, согласно исследованиям по проектированию API [10], здесь возникает проблема «жёсткой связанности». Если сайт временно недоступен или интернет-соединение даёт сбой, интерфейс основной программы блокируется до

получения ответа от сервера. В реальной эксплуатации такие простои считаются недопустимыми, поскольку они напрямую снижают общую производительность предприятия.

Асинхронная интеграция с использованием внешнего брокера сообщений. Эта стратегия предполагает наличие промежуточного звена (например, Apache Kafka или RabbitMQ) [11]. Система-источник передаёт событие в шину данных, после чего мгновенно продолжает работу, не дожидаясь ответа от приёмника. Несмотря на эталонную надёжность и полную изоляцию систем, внедрение брокеров требует развёртывания сложной инфраструктуры и значительных затрат на администрирование. Для локальной задачи мониторинга одного предприятия использование таких инструментов часто оказывается избыточным.

Асинхронная интеграция с использованием внутренней очереди. Этот вариант представляет собой гибридную схему, построенную на паттерне Transactional Outbox (исходящая очередь транзакций) [12]. Данная стратегия считается одной из наиболее совершенных для организации обмена данными в современных распределённых системах. Её архитектурная ценность в обеспечении атомарности операций: процесс изменения данных в основной базе и процесс регистрации этого изменения для внешней системы происходят в рамках одной транзакции. Реализация такого подхода подразумевает разделение единого процесса синхронизации на две функционально независимые фазы.

Первая фаза – захват событий. При совершении любой операции система-источник не пытается немедленно связаться с внешним API. Вместо этого она лишь фиксирует факт изменения в специальной таблице-очереди, которая находится в той же базе данных.

Вторая фаза – асинхронная трансляция. Специализированный фоновый процесс в автономном режиме опрашивает очередь и выполняет фактическую передачу накопленных данных принимающей стороне.

Применение паттерна Transactional Outbox даёт несколько результатов, важных для проекта. Системы оказываются полностью изолированными друг от друга. Производительность и стабильность учётной системы 1С перестают

зависеть от того, работает ли внешний интернет-канал и насколько нагружен веб-сервер.

На основе проведенного анализа была составлена сравнительная характеристика рассмотренных подходов, представленная в таблице 2. Полученные выводы позволили выделить наиболее подходящий метод для реализации механизма синхронизации.

Таблица 2 – Сравнение архитектурных стратегий синхронизации данных

Критерий сравнения	Пакетный файловый обмен	Прямая синхронная API - интеграция	Асинхронная с внешним брокером	Асинхронная с внутренней очередью
Оперативность	Низкая	Максимальная	Высокая	Высока
Связанность система	Слабая	Жесткая	Слабая	Слабая
Риск блокировки источника	Нет	Высокий	Нет	Нет
Сложность инфраструктуры	Низкая	Средняя	Высокая	Средняя
Надежность доставки	Средняя	Низкая	Высокая	Высока

На основе данных таблицы был выбран вариант асинхронной синхронизации с использованием внутренней очереди. Внешние брокеры сообщений для этой задачи тоже показывают неплохую эффективность, однако они оказались избыточными. Причина – сложность настройки и последующего сопровождения. Внешние брокеры требуют установки дополнительного программного обеспечения и выделения отдельных серверов, что неоправданно усложняет всю систему. Таким образом, выбранный подход обеспечивает нужную оперативность обмена без лишних затрат на инфраструктуру, сохраняя при этом высокую отказоустойчивость всей архитектуры.

2.1.2 Триггерный механизм регистрации изменений в системе-источнике

В рамках проектирования механизма синхронизации определено, что фиксация изменений в системе-источнике должна выполняться событийным способом. Это означает, что система обязана реагировать на действия пользователя непосредственно в момент завершения операции с данными. В качестве технической основы для такого подхода выбраны подписки на события (триггерный механизм), поддерживаемые платформой учётной системы. Проектным решением предусмотрено, что отслеживанию подлежат не все объекты учётной системы, а только те, которые влияют на показатели панели мониторинга. В результате анализа предметной области выделены следующие функциональные контуры, изменения в которых должны регистрироваться:

- контур продаж и спроса (объекты, отражающие поступление заказов и факты отгрузки);
- логистический контур (объекты, связанные с пополнением товарных запасов);
- финансовый контур (объекты, фиксирующие поступление и расходование денежных средств).

Для каждого из выделенных контуров проектируемый триггерный механизм обеспечивает регистрацию факта изменения соответствующих объектов без вмешательства в типовую логику учётной системы.

2.2 Проектирование структур данных и моделей хранения

Проектирование структуры данных является фундаментом для построения отказоустойчивого механизма синхронизации. В качестве технологической базы системы-источника выступает прикладное решение на платформе «1С:Предприятие 8.3», автоматизирующее деятельность предприятия по торговле строительными материалами. В качестве базы-приёмника используется MS SQL Server. Спроектированная структура должна обеспечивать однозначное

соответствие между объектами учётной системы и записями в базе данных портала.

2.2.1 Структура данных системы-источника (1С).

Для формирования оперативных показателей на панели мониторинга были выделены пять приоритетных типов документов, отражающих основные бизнес-процессы организации:

- «ЗаказПокупателя», первичный документ, фиксирующий спрос и плановые показатели продаж;
- «ПоступлениеТоваров», документ, аккумулирующий данные о закупках и качественных характеристиках материалов;
- «РеализацияТоваров», документа, объединяющая сведения о фактических отгрузках и логистических условиях доставки;
- «ПоступлениеДенежныхСредств» и «РасходДенежныхСредств» документы для контроля входящих и исходящих финансовых потоков.

Логическая взаимосвязь этих объектов и их типизация представлены в виде диаграмме классов на рисунке 6. Использование данной нотации позволяет наглядно отобразить метаструктуру 1С, где каждый объект является классом с уникальным набором реквизитов и вложенных табличных частей.

На схеме каждый класс соответствует отдельному объекту конфигурации. Были предусмотрены как простые поля (номер, дата, сумма), так и связи между объектами.

Для сохранения данных и производительности использованы типизированные указатели на объекты метаданных (справочники, документы, перечисления)). Это обеспечивает точную идентификацию каждой связанной сущности при отправке во внешнюю среду. Отдельно было продумано устройство табличных частей («СоставЗаказа», «ТехническиеПараметры»). Для них применён принцип композиции. Документ и его строки рассматриваются как единый блок, что важно для формирования полиструктурных JSON-сообщений.

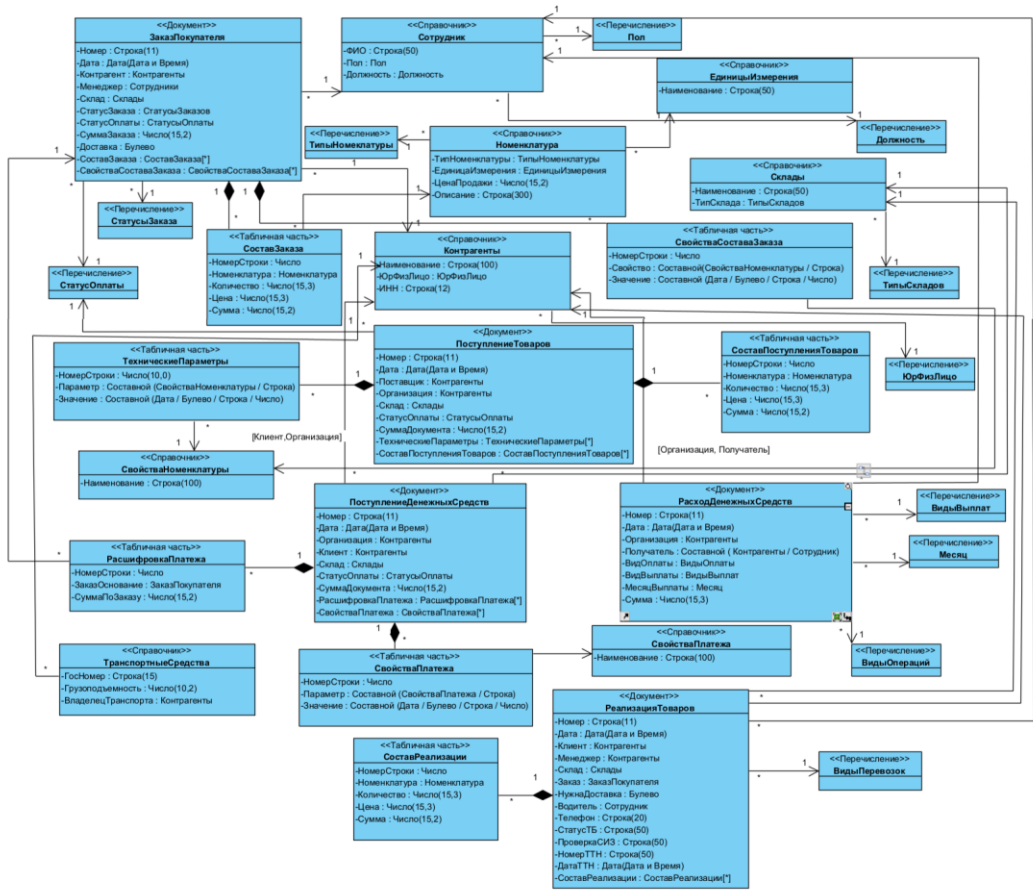


Рисунок 6 – Логическая модель метаданных контура бизнес-логики системы-источника

На диаграмме классов представлены основные объекты учётной системы, участвующие в синхронизации. На диаграмме используются логические имена классов. Технические типы данных платформы «1С:Предприятие», такие как СправочникСсылка.Контрагенты, указаны в таблицах ниже. Это позволяет сохранить наглядность диаграммы и одновременно дать точную информацию для реализации синхронизации.

Для детального рассмотрения структуры данных выбран документ «ЗаказПокупателя».

Реквизиты шапки документа описаны в таблице 3.

Таблица 3 – Описание атрибутов «ЗаказПокупателя»

Атрибут	Тип данных	Назначение
Ссылка	УникальныйИдентификатор	Глобальный идентификатор объекта

Продолжение таблицы 3

Атрибут	Тип данных	Назначение
Номер	Строка(11)	Учетный номер документа в системе 1С
Дата	Дата (Дата и время)	Момент совершения операции
Контрагент	СправочникСсылка.Контрагенты	Ссылка на покупателя (заказчика)
Менеджер	СправочникСсылка.Сотрудники	Ответственное лицо со стороны организации
Склад	СправочникСсылка.Склады	Место хранения или отгрузки товаров
СтатусЗаказа	ПеречислениеСсылка.СтатусыЗаказов	Состояние жизненного цикла документа
СуммаЗаказа	Число(15, 2)	Итоговый финансовый показатель сделки
Доставка	Булево	Флаг необходимости логистических услуг

Количественные данные и изменяемые характеристики были вынесены в подчинённые таблицы. Табличная часть с товарами заказа описана в таблице 4.

Таблица 4 – Описание атрибутов «СоставЗаказа»

Атрибут	Тип данных	Назначение
НомерСтроки	Число	Порядковый индекс позиции в документе
Номенклатура	СправочникСсылка.Номенклатура	Ссылка на товарную позицию или услугу
Количество	Число(15, 3)	Объем заказываемой продукции
Цена	Число(15, 3)	Стоимость единицы товара
Сумма	Число(15, 2)	Итоговая стоимость позиции с учетом объема
КлючСвязи	Число(10)	Технический идентификатор для связи со свойствами строки

Для обеспечения максимальной гибкости системы и поддержки передачи неограниченного набора параметров товара используется механизм дополнительных характеристик. Такой подход позволяет передавать на портал любые свойства товара без изменения структуры базы данных.

Структура этой подчиненной сущности «СвойстваСоставаЗаказа» представлена в таблице 5.

Таблица 5 – Состав атрибутов табличной части «СвойстваСоставаЗаказа»

Атрибут	Тип данных	Назначение
НомерСтроки	Число	Технический индекс записи
КлючСвязи	Число(10)	Ссылка на строку товара из таблицы «СоставЗаказа»
Свойство	СоставнойТип(Строка, СправочникСсылка)	Наименование или категория характеристики
Значение	СоставнойТип(Дата, Булево, Строка, Число)	Конкретное значение выбранного свойства

Для самого механизма обмена данными была спроектирована отдельная группа объектов. Она показана на рисунке 7.

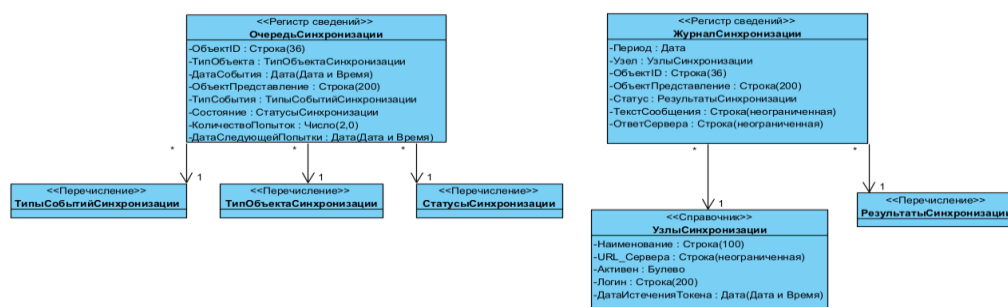


Рисунок 7 – Логическая модель метаданных технического контура системы-источника

В таблицах ниже указаны технические типы данных платформы «1С:Предприятие», которые будут использоваться при формировании JSON-пакетов и в маппинге Регистр «ОчередьСинхронизации» хранит записи о событиях, требующих синхронизации, детальное описание приведено в таблице 6.

Таблица 6 – Состав атрибутов регистра сведений «ОчередьСинхронизации»

Атрибут	Тип данных	Назначение
ОбъектID	Строка(36)	Уникальный идентификатор объекта в 1С (GUID)
ТипОбъекта	ПеречислениеСсылка.ТипОбъекта Синхронизации	Тип объекта (документ, справочник)
ДатаСобытия	Дата (Дата и время)	Дата и время регистрации события
ОбъектПредставление	Строка(200)	Текстовое представление объекта

Продолжение таблицы 6

Атрибут	Тип данных	Назначение
ТипСобытия	ПеречислениеСсылка.ТипыСобытийСинхронизации	Вид операции (создание, обновление, удаление)
Состояние	ПеречислениеСсылка.СтатусыСинхронизации	Текущий статус обработки записи
КоличествоПопыток	Число(2,0)	Счётчик неудачных попыток отправки
ДатаСледующейПопытки	Дата (Дата и время)	Время следующей попытки при ошибке

Регистр «ЖурналСинхронизации» фиксирует историю попыток обмена данными, детальное описание атрибутов приведено в таблице 7.

Таблица 7 – Состав атрибутов регистра сведения «ЖурналСинхронизации»

Атрибут	Тип данных	Назначение
Период	Дата	Дата и время выполнения попытки
Узел	СправочникСсылка.УзлыСинхронизации	Ссылка на узел, через который выполнялась отправка
ОбъектID	Строка(36)	Идентификатор синхронизируемого объекта
ОбъектПредставление	Строка(200)	Текстовое представление объекта
Статус	ПеречислениеСсылка.РезультатыИнтеграции	Итог попытки (успех или ошибка)
ТекстСообщения	Строка(неограниченная)	Отправленный пакет данных
ОтветСервера	Строка(неограниченная)	Ответ от веб-портала

Справочник «УзлыСинхронизации» содержит настройки подключений к внешним системам, детальное описание атрибутов в таблице 8.

Таблица 8 – Состав атрибутов справочника «УзлыСинхронизации»

Атрибут	Тип данных	Назначение
Ссылка	УникальныйИдентификатор	Идентификатор узла (GUID)
Наименование	Строка(100)	Название узла (например, «Основной портал»)
URL_Сервера	Строка(неограниченная)	Адрес веб-портала

Продолжение таблицы 8

Атрибут	Тип данных	Назначение
Активен	Булево	Признак использования узла
Логин	Строка(200)	Логин для портала
ДатаИстеченияТокена	Дата (Дата и время)	Срок действия токена

Перечисления определяют фиксированный набор значений для ключевых параметров механизма синхронизации, детально описание в таблице 9.

Таблица 9 – Перечисления служебной системы

Атрибуты	Ключевое значение
ТипыСобытийСинхронизации	Вид операции над объектом
СтатусыСинхронизации	Состояние записи в очереди
РезультатыИнтеграции	Итог попытки обмена
ТипОбъектаСинхронизации	Тип синхронизируемого объекта

Все служебные объекты работают независимо от бизнес-логики.

2.2.2 Структура данных системы-приемника

При создании механизма синхронизации нужно спроектировать структуру хранения данных на стороне принимающей системы. База данных веб-портала мониторинга работает как оперативное хранилище. В основу проектирования лег принцип разделения данных на заголовки документов и их детальный состав. Полная логическая модель, , представлена на рисунке 8.

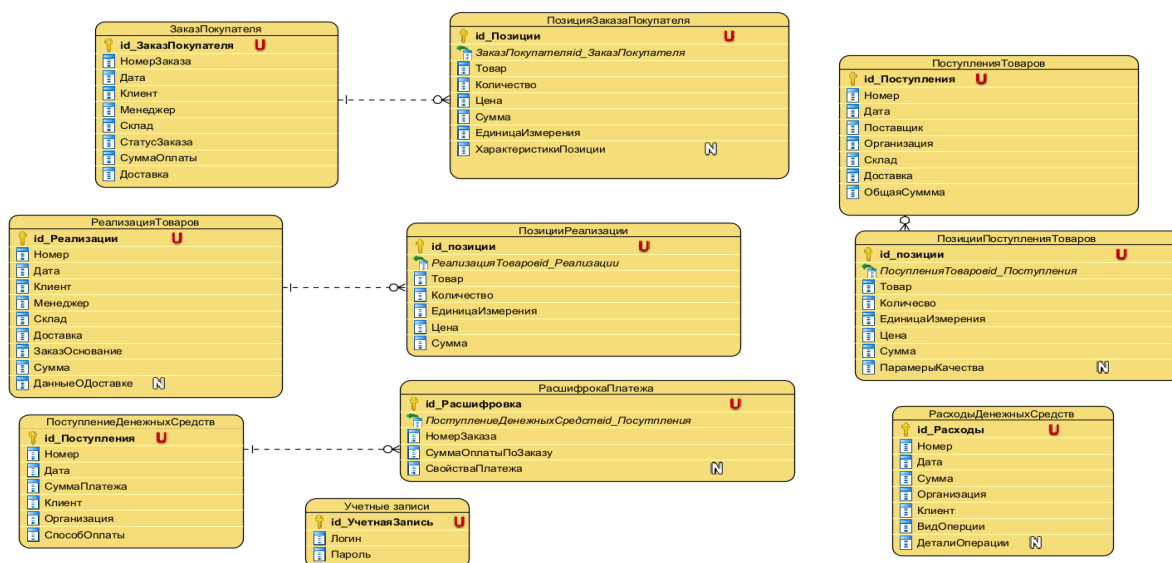


Рисунок 8 – Логическая схема базы данных системы мониторинга

Ниже приведено детальное описание ключевых сущностей, которые были выделены для обеспечения полноценной синхронизации торговых и финансовых операций.

Сущность «Заказы покупателей» является отправной точкой в цепочке продаж. Она аккумулирует общие сведения о намерениях клиентов и плановых суммах. Подробный состав атрибутов приведен в таблице 10 .

Таблица 10 – Описание полей таблицы «ЗаказПокупателя»

Атрибуты	Нулевое значение	Ключевое значение
Id_ЗаказаПокупателя	Не допустимо	РК
Номер_Заказа	Не допустимо	
Дата	Не допустимо	
Клиент	Не допустимо	
Менеджер	Не допустимо	
Склад	Не допустимо	
СтатусОплаты	Не допустимо	
СтатусЗаказа	Не допустимо	
Доставка	Не допустимо	

Для хранения информации о товарных позициях внутри заказа используется подчинённая сущность. В ней присутствует специальное полиструктурное поле «ХарактеристикиПозиции». Через это поле и обеспечивается гибкость модели: сюда попадают все изменяемые свойства товара , причём для каждой строки заказа набор таких свойств может быть своим. Описание её полей представлено в таблице 11.

Таблица 11 – Описание полей таблицы «ПозицииЗаказаПокупателя»

Атрибуты	Нулевое значение	Ключевое значение
Id_Позиции	Не допустимо	РК
ЗаказПокупателяId_ЗаказаПокупателя	Не допустимо	FK
Товар	Не допустимо	
Количество	Не допустимо	

Продолжение таблицы 11

Атрибуты	Нулевое значение	Ключевое значение
ЕдиницаИзмерения	Не допустимо	
Цена	Не допустимо	
Сумма	Не допустимо	
ХарактеристикиПозиции	Не допустимо	

Аналогичным образом была спроектирована структура для блока закупок. Сущность «ПоступленияТоваров» фиксирует все взаимодействия с поставщиками. К ней привязана сущность «ПозицииПоступленияТоваров», которая содержит детальную расшифровку принятой продукции. В таблице состава поступлений предусмотрено полиструктурное поле «ПараметрыКачества». Оно нужно для записи технических параметров качества материалов – эти параметры могут различаться от партии к партии. Такое решение позволяет хранить разнородные сведения о товаре без жёсткой привязки к фиксированному набору реквизитов.

Для документа «Поступлений денежных средств» разработана многокомпонентная модель. Она связывает сущность «ПоступлениеДенежныхСредств» с расшифровкой по конкретным заказам. Ещё одна особенность: система позволяет фиксировать индивидуальные свойства для каждой строки оплаты. Например, форму подтверждения платежа или признак аванса. Для этого спроектировано полиструктурное поле «СвойстваПлатежа» в таблице «РасшифровкаПлатежа»

Блок отгрузок построен вокруг сущности «РеализацииТоваров». В логическую модель добавлены не только стандартные реквизиты, но и специальное поле для хранения сведений о доставке «ДанныеОДоставке». Благодаря этому информация о водителе, транспортном средстве и логистических требованиях объединяется в рамках одного объекта. Общая схема базы данных не усложняется, но при этом сохраняется возможность работать с полиструктурными данными – каждая доставка может иметь свой уникальный набор характеристик,

Для Расходов денежных средств спроектирована сущность «РасходДенежныхСредств» В ней добавлено поле «ДеталиОперации» для фиксации специфические детали транзакции

Помимо объектов предметной области, в логическую структуру базы данных включен системный контур, представленный сущностью «УчетныеЗаписи» Описание представлено в таблице 12.

Таблица 12 – Описание полей таблицы «УчетныеЗаписи»

Атрибуты	Нулевое значение	Ключевое значение
Id_УчетнаяЗапись	Не допустимо	PK
Логин	Не допустимо	
Пароль	Не допустимо	

Данная сущность необходима для реализации механизмов санкционированного доступа к данным.

На основе логической диаграммы была спроектирована физическая модель базы данных принимающей стороны. Были определены, как именно будут храниться данные, и какие типы полей использовать с учётом того, что СУБД выбрана Microsoft SQL Server. Полная физическая схема базы данных с указанием всех типов данных и связей представлена на рисунке 9.

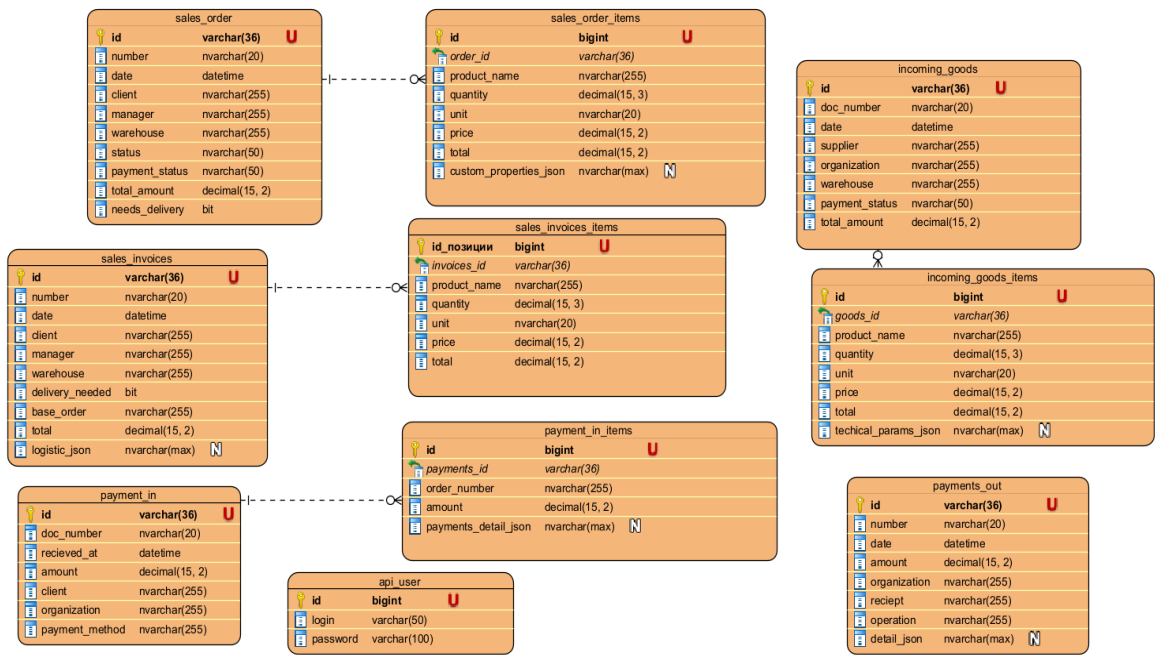


Рисунок 9 – Физическая модель базы данных принимающей стороны

В качестве примера, ниже в таблицах 13 и 14 приведены детальные спецификации полей таблиц sales_orders, sales_orders_items.

Таблица 13 – Описание полей таблицы sales_orders

Атрибуты	Тип данных	Нулевое значение	Ключевое значение
Id	Varchar(36)	Не допустимо	РК
number	Nvarchar(20)	Не допустимо	
date	DateTime	Не допустимо	
client	Nvarchar (255)	Не допустимо	
manager	Nvarchar (255)	Не допустимо	
warehouse	Nvarchar (255)	Не допустимо	
status	Nvarchar (255)	Не допустимо	
payment_status	Nvarchar (50)	Не допустимо	
total_amount	DECIMAL (15, 2)	Не допустимо	
needs_delivery	Bit	Не допустимо	

Для обеспечения возможности работы с полиструктурными данными, такими как меняющиеся характеристики стройматериалов или детальные реквизиты платежей, был выбран тип данных Nvarchar (max). Это техническое решение позволяет сохранять любые наборы атрибутов в формате JSON внутри одной колонки. Такой подход делает физическую структуру базы данных «гибкой».

Таблица 14 – Описание полей таблицы sales_orders_items

Атрибуты	Тип данных	Нулевое значение	Ключевое значение
id	Bigint	Не допустимо	РК
orders_id	Nvarchar (36)	Не допустимо	FK
product_name	nvarchar(255)	Не допустимо	
quantity	Decimal(15, 3)	Не допустимо	
price	Decimal(15, 2)	Не допустимо	

Продолжение таблицы 14

Атрибуты	Тип данных	Нулевое значение	Ключевое значение
total	Decimal(15, 2)	Не допустимо	
unit	Nvarchar(20)	Не допустимо	
custom_properties_json	Nvarchar(max)	Допустимо	

Остальные таблицы базы данных построены по тому же принципу. У сущностей, которые работают с закупками (incoming_goods, incoming_goods_items), отгрузками (sales_invoices, sales_invoice_items) и поступлениями платежей (payments_in, payments_items), структура повторяет логику «заголовок – состав». В них тоже используется тип данных Nvarchar (max) для передачи параметров качества продукции и логистических анкет. Таблица расходов денежных средств (payments_out) реализована по аналогичной схеме.

В системном блоке базы данных есть сущность apiusers. Она нужна, чтобы контролировать санкционированный доступ к информации. Описание её спецификации приведено в таблице 15.

Таблица 15 – Описание полей таблицы api_user

Атрибуты	Тип данных	Нулевое значение	Ключевое значение
id	Bigint	Не допустимо	PK
login	Nvarchar(50)	Не допустимо	
password	Nvarchar(200)	Не допустимо	

Данная таблица предназначена для идентификации внешних систем-источников, участвующих в процессе синхронизации. В ней хранятся уникальные параметры доступа. Благодаря этому сервер приложений может проверять входящие запросы. Так гарантируется, что в базу данных мониторинга попадают сведения только от доверенного учётного контура предприятия. Исключается ситуация, когда недостоверные данные приходят из

неавторизованных источников. Таким образом была спроектирована база данных принимающей стороны.

2.2.3 Спецификации сопоставления и правила трансформации данных

Важным этапом проектирования механизма синхронизации является разработка схем сопоставления данных между системой-источником и системой-приемником. На этом этапе абстрактные сущности логической модели переводятся в объекты метаданных платформы «1С:Предприятие 8.3». Операционные сущности сопоставляются с документами, а справочные – со справочниками и перечислениями. Для обеспечения однозначной идентификации объектов на стороне веб-портала, каждый пакет данных в формате JSON содержит служебный блок мета-данных `meta_info`, включающий два обязательных поля: `type`, техническое имя объекта в конфигурации 1С и `id`, уникальный идентификатор объекта (GUID), выступающий в качестве первичного ключа в СУБД MS SQL. Таблицы сопоставления задают правила, по которым данные из объектной модели 1С будут преобразованы в JSON и отправлены на веб-портал. Правила преобразования данных для документа «Заказ покупателя» из представлены в таблице 16.

Таблица 16 – Сопоставление данных для документа «Заказ покупателя»

Поле/ Табличная часть 1С	Тип данных в 1С	Ключ в JSON	Поле в БД SQL	Тип данных в SQL	Логика перевода
			Таблица sales_order		
Ссылка	Уникальный Идентификатор	meta_info. id	id	varchar (36)	Перевод id в строку (UUID).
Основные реквизиты					
Номер	Строка(11)	number	doc_number	Nvarchar (20)	Прямой перенос строкового значения.
Дата	Дата (ДатаВремя)	date	date	datetime	Перевод даты в строк (ISO 8601).

Продолжение таблицы 16

Поле/ Табличная часть 1С	Тип данных в 1С	Ключ в JSON	Поле в БД SQL	Тип данных в SQL	Логика перевода
Контрагент	СправочникСсылка Контрагенты	client	client	Nvarchar (255)	Получение наименования по ссылке (строка).
Менеджер	СправочникСсылка Сотрудник	manager	manager	Nvarchar (255)	Получение наименования по ссылке (строка).
Склад	СправочникСсылка Склады	warehouse	warehouse	Nvarchar (255)	Получение наименования по ссылке (строка).
Статус заказа	Перечисление Ссылка СтатусыЗаказов	status	status	Nvarchar (50)	Получение наименования по ссылке (строка).
Статус оплаты	Перечисление Ссылка СтатусыОплаты	payment	payment_ status	Nvarchar (50)	Получение наименования по ссылке (строка).
Сумма заказа	Число(15, 2)	total	total_ amount	decimal (15, 2)	Перенос числового значения.
Доставка	Булево	delivery	needs_ delivery	bit	Приведение логического типа.
Табличная часть «Состав Заказа»		items[]	Таблица sales_ order_ items		
Номенклатура	СправочникСсылка Номенклатура	items[]._ name	product_ name	Nvarchar (255)	Получение наименования по ссылке (строка).
Количество	Число(15, 3)	items[]._qty	quantity	decimal (15, 3)	Прямой перенос числового значения.
Ед. измерения	СправочникСсылка Единицы Измерения	items[]._ unit	unit	Nvarchar (20)	Получение наименования по ссылке (строка).
Цена	Число(15, 2)	items[]._ price	price	decimal (15, 2)	Прямой перенос числового значения.

Окончание таблицы 16

Поле/ Табличная часть 1С	Тип данных в 1С	Ключ в JSON	Поле в БД SQL	Тип данных в SQL	Логика перевода
Сумма	Число(15, 2)	items[]. total	total	decimal (15, 2)	Прямой перенос числового значения.
Табличная часть «Свойства Состава Заказа»		items[]. props	custom_ properties_ json	Nvarchar (max)	-
Свойство	Составной тип (СправоникСсылка СвойстваНомер/ Строка)	items[].pro ps. {ключ}			Получение наименования по ссылке или прямой перенос строки (становится ключом в JSON).
Значение	Составной тип (дата/ строка/ булево)				Становится значением ключа; числа передаются как есть, остальное преобразуется в строку

Правила трансформации данных для документа «Поступление товаров» при синхронизации с веб-порталом приведены в таблице 17.

Таблица 17 – Сопоставление данных для документа «Поступление товаров»

Поле/ Табличная часть 1С	Тип данных в 1С	Ключ в JSON	Поле в БД SQL	Тип данных в SQL	Логика перевода
			Таблица incoming_ goods		
Ссылка	Уникальный Идентификатор	meta_info. id	id	varchar (36)	Перевод id в строку (UUID).
Основные реквизиты					
Номер	Строка(11)	number	doc_number	Nvarchar (20)	Прямой перенос строкового значения.

Продолжение таблицы 17

Поле/ Табличная часть 1С	Тип данных в 1С	Ключ в JSON	Поле в БД SQL	Тип данных в SQL	Логика перевода
Дата	Дата (ДатаВремя)	date	date	datetime	Перевод даты в строку (ISO 8601).
Поставщик	СправочникСсылка Контрагенты	supplier	supplier	nvarchar (255)	Получение наименования по ссылке (строка).
Организация	СправочникСсылка Контрагенты	organization	organization	Nvarchar (255)	Получение наименования по ссылке (строка).
Склад	СправочникСсылка Склады	warehouse	warehouse	Nvarchar (255)	Получение наименования по ссылке (строка)
Статус оплаты	Перечисление СсылкаСтатусы Оплаты	payment_ status	payment_ status	Nvarchar (50)	Перевод значения в строку.
Сумма документа	Число(15, 2)	total	total_ amount	decimal (15, 2)	Прямой перенос числового значения.
Табличная часть «Состав Поступления Товара»		items[]	Таблица incoming_ goods_ items		
Номенклатура	СправочникСсылка Номенклатура	items[]._ name	product_ name	nvarchar (255)	Получение наименования по ссылке (строка).
Количество	Число(15, 3)	items[]._ qty	quantity	decimal (15, 3)	Прямой перенос числового значения.
Ед. измерения	СправочникСсылка Единицы Измерения	items[]._ unit	unit	nvarchar (20)	Получение наименования по ссылке (строка).

Окончание таблицы 17

Поле/ Табличная часть 1С	Тип данных в 1С	Ключ в JSON	Поле в БД SQL	Тип данных в SQL	Логика перевода
Цена	Число(15, 2)	items[]. price	price	decimal (15, 2)	Прямое перенос значения
Сумма	Число(15, 2)	items[]. total	total	decimal (15, 2)	Прямой перенос числового значения.
Табличная часть «Технические Параметры»		items[]. tech_ params	technical_ params_ json	nvarchar (max)	
Параметр	Составной тип (СправочникСсылка Свойства Номеклатуры/ Строка)	items[]. tech_ params. . {ключ}			Получение наименования по ссылке или перенос строки (становится ключом в JSON)..
Значение	Составной тип (дата/ число/ срока/ булево)				Становится значением ключа; числа передаются как есть, остальное преобразуется в строку

Сопоставление полей документа «Расход денежных средств» для обмена с веб-порталом описано в таблице 18.

Таблица 18 – Сопоставление данных для документа «Расход денежных средств»

Поле/ Табличная часть 1С	Тип данных в 1С	Ключ в JSON	Поле в БД SQL	Тип данных в SQL	Логика перевода
			Таблица payment_ out		
Ссылка	Уникальный Идентификатор	meta_info. id	id	varchar (36)	Перевод id в строку (UUID).
Основные реквизиты					

Продолжение таблицы 18

Поле/ Табличная часть 1С	Тип данных в 1С	Ключ в JSON	Поле в БД SQL	Тип данных в SQL	Логика перевода
Номер	Строка(11)	number	number	nvarchar (20)	Прямой перенос строкового значения.
Дата	Дата (Дата и Время)	date	date	datetime	Перевод даты в строку (ISO 8601).
Организация	СправочникСсылка Контрагенты	organization	organization	nvarchar (255)	Получение наименования по ссылке (строка).
Получатель	Составной тип (СправочникСсылка Контрагенты, СправочникСсылка Сотрудник)	recipient	recipient	nvarchar (255)	Получение наименования по ссылке (строка).
Вид операции	Перечисление СсылкаВиды Операций	operation	operation	nvarchar (100)	Получение строкового значения по ссылке
Сумма	Число(15, 2)	amount	amount	decimal (15, 2)	Прямой перенос числового значения.
Вариативные данные		details	details_ json	nvarchar (max)	
Сценарий: Зарплата					В JSON включаются кадровые реквизиты.
Месяц выплаты	Справочник СсылкаМесяц	details. month			Перевод значения в строку.
Вид выплаты	Справочник СсылкаВидыВыплат	details. pay_kind			Получение наименования по ссылке
Сценарий: Оплата					В JSON включаются финансовые реквизиты.
Документ закупки	ДокументСсылка. ПоступлениеТоваров	details. invoice_ref			Получение строки по ссылке.

Правила конвертации данных документа «Поступление денежных средств» из 1С в JSON-формат представлены в таблице 19.

Таблица 19 – Сопоставление данных для документа «Поступление Денежных Средств»

Поле/ Табличная часть 1С	Тип данных в 1С	Ключ в JSON	Поле в БД SQL	Тип данных в SQL	Логика перевода
			Таблица pay_in		
Ссылка	Уникальный Идентификатор	meta_info. id	id	varchar (36)	Перевод id в строку (UUID).
Основные реквизиты					
Номер	Строка(11)	number	doc_number	nvarchar (20)	Прямой перенос строкового значения.
Дата	Дата (ДатаВремя)	date	received_at	datetime	Перевод даты в строку (ISO 8601).
Организация	Справочник СсылкаКонтрагент	organiza- tion	organization	nvarchar (255)	Получение наименования по ссылке (строка).
Клиент	Справочник СсылкаКонтрагент	client	client	nvarchar (255)	Получение наименования по ссылке (строка).
Сумма документа	Число(15, 2)	amount	amount	Decimal (15, 2)	Прямой перенос числового значения.
Способ оплаты	Перечисление СсылкаСпособ Оплаты	method	payment_ method	nvarchar (50)	Перевод значения в строку.
Табличная часть «Расшифровка Платежей»		invoices[]	Таблица pay_in_ items		
Заказ (Основание)	ДокументСсылка ЗаказПокупателя	invoices[]. order_id	order_ number	Nvarchar (255)	Получение наименования по ссылке (строка).
Сумма по заказу	Число(15, 2)	invoices[]. sum	amount	Decimal (15, 2)	Прямой перенос
Табличная часть «Свойства Платежей»		invoices[]. attributes	payment_ details_json	nvarchar (max)	

Продолжение таблицы 19

Поле/ Табличная часть 1С	Тип данных в 1С	Ключ в JSON	Поле в БД SQL	Тип данных в SQL	Логика перевода
Свойство	Составной тип (СправочникСсылка СвойстваПлатежа/Ст рока)	invoices[attributes. {ключ}			Получение наименования по ссылке или перенос строки (становится ключом в JSON)..
Значение	Составной тип (дата/ число/ срока/ булево)				Становится значением ключа; числа передаются как есть, остальное преобразуется в строку

Схема преобразования данных для документа «Реализация товаров» при интеграции с веб-порталом приведена в таблице 20.

Таблица 20 – Сопоставление данных для документа «Реализация Товаров»

Поле / Табличная часть 1С	Тип данных в 1С	Ключ в JSON	Поле в БД SQL	Тип в SQL	Логика перевода
			Таблица sales_ invoices		
Ссылка	Уникальный Идентификатор	meta_info. id	id	varchar (36)	Перевод id в строку (UUID).
Основные реквизиты					
Номер	Строка(11)	number	number	nvarchar (20)	Прямой перенос .
Дата	Дата (ДатаВремя)	date	date	datetime	Перевод даты в строку (ISO 8601).
Клиент	СправочникСсылка Контрагент	client	client	nvarchar (255)	Получение наименования по ссылке (строка).
Менеджер	СправочникСсылка Контрагент	manager	manager	nvarchar (255)	Получение наименования по ссылке

Продолжение таблицы 20

Поле / Табличная часть 1С	Тип данных в 1С	Ключ в JSON	Поле в БД SQL	Тип в SQL	Логика перевода
Склад	СправочникСсылка Склад	warehouse	warehouse	nvarchar (255)	Получение наименования по ссылке (строка).
Заказ основание	ДокументСсылка ЗаказПокупателя	base_order	base_ order	nvarchar (255)	Получение номера документа по ссылке (строка).
Сумма документа	Число(15, 2)	total	total_ amount	decimal (15, 2)	Прямой перенос числового значения.
Нужна доставка	Булево	delivery_ needed	delivery_ needed	bit	Перевод логического типа.
Логистика		delivery_ info	Logistics json	nvarchar (max)	
ФИО_водителя	СсылкаСправочник Сотрудник	delivery_ info.driver .fio			Получение наименования по ссылке или прямой перенос строки.
Телефон	Строка(20)	delivery_ info.driver .tel			Прямой перенос строкового значения.
Статус ТБ	Строка(50)	delivery_ info. driver.stat us			Перевод значения в строку.
ПроверкаСИЗ	Строка(50)	delivery_ info. driver. siz			Прямой перенос строкового значения.
Номер ТТН	Строка(50)	delivery_ info.. transport. numttn			Перенос строкового значения.
Дата ТТН	Дата (ДатаВремя)	delivery_ info. transport. data			Перевод даты в строку (ISO 8601).

Окончание таблицы 20

Поле / Табличная часть 1С	Тип данных в 1С	Ключ в JSON	Поле в БД SQL	Тип в SQL	Логика перевода
Табличная часть «Состав Реализации»		items[]	Таблица sales_invoice_items		
Номенклатура	СправочникСсылка Номенклатура	items[]. name	product_ name	nvarchar (255)	Получение наименования по ссылке (строка).
Количество	Число(15, 3)	items[]. qty	quantity	decimal (15, 3)	Прямой перенос числового значения.
Ед. измерения	СправочникСсылка ЕдиницаИзмерений	items[]. unit	unit	nvarchar (20)	Получение наименования по ссылке (строка).
Цена	Число(15, 2)	items[]. price	price	decimal (15, 2)	Прямой перенос числового значения.
Сумма	Число(15, 2)	items[]. sum	total	decimal (15, 2)	Прямой перенос числового значения.

Приведённые в таблицах сопоставления обеспечивают однозначную интерпретацию данных при их передаче из учётной системы в базу данных веб-портала. Таблицы сопоставления помогают точно понять, какие данные откуда берутся. При написании кода синхронизации эти таблицы служат готовым справочником для разработчика.

2.3 Моделирование динамики и алгоритмов функционирования

На этапе проектирования важно не только определить состав объектов системы, но и описать, как они взаимодействуют во времени. Для этого используются динамические диаграммы UML. Они помогают зафиксировать последовательность действий между компонентами, условия переходов и жизненные циклы данных.

2.3.1 Проектирование динамики межсистемного взаимодействия

При проектировании программного решения для обмена данными важно описать, как система работает во времени. Для детальной фиксации хронологии взаимодействия между распределёнными узлами была спроектирована диаграмма последовательности (Sequence Diagram). С помощью этого инструмента графического моделирования можно описать жизненный цикл отдельных транзакций и определить, в каком порядке компоненты обмениваются управляющими сигналами и информационными пакетами.

Спроектированная модель взаимодействия представлена на рисунке 10.

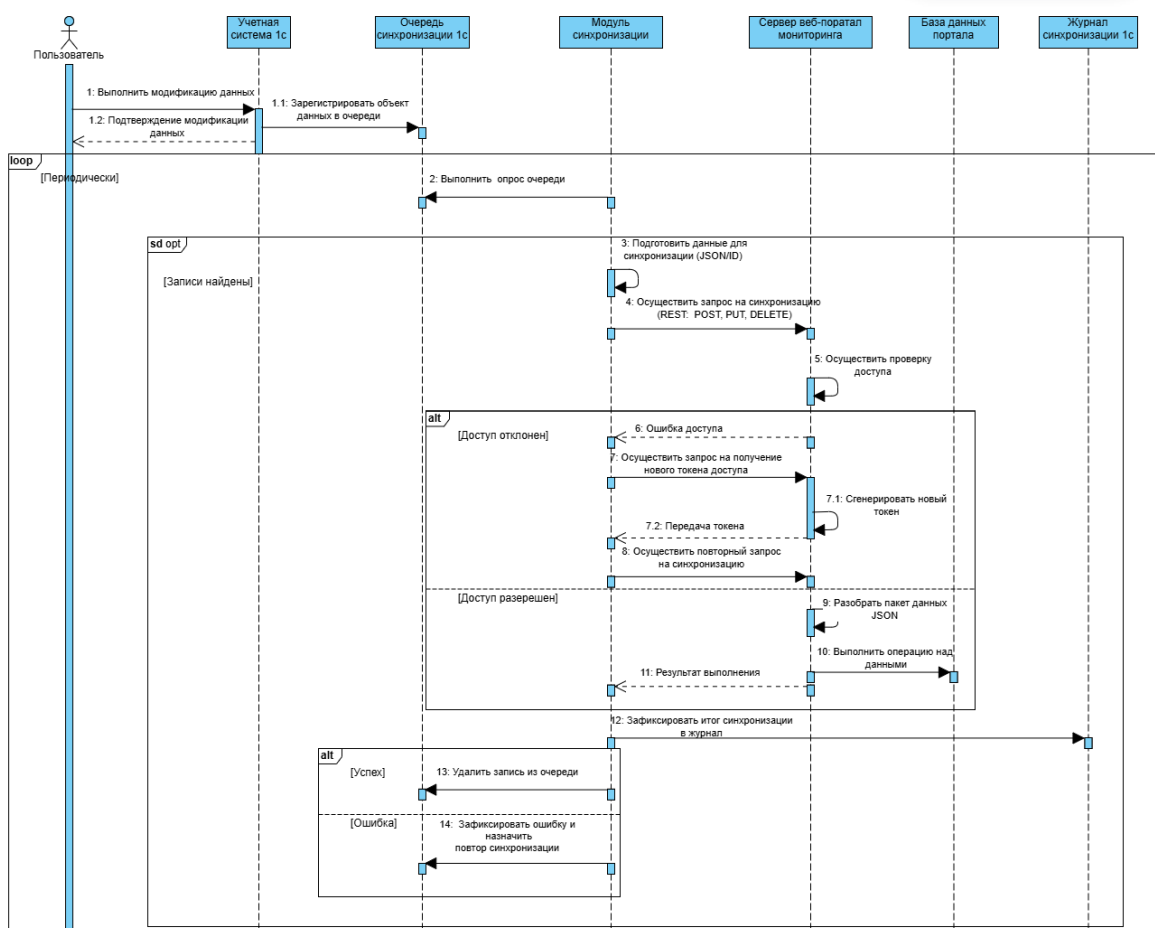


Рисунок 10 – Диаграмма последовательности процесса синхронизации данных

1С и веб-портал обмениваются данными асинхронно. Это сделано, чтобы синхронизация не тормозила работу пользователя. Сначала пользователь меняет данные в учётной системе. 1С сразу записывает это событие в очередь

синхронизации. На этом участие пользователя заканчивается, и система подтверждает, что данные сохранены.

Дальше в дело вступает модуль синхронизации. Он работает в фоне. Периодически модуль опрашивает очередь. Если есть записи, он готовит из них JSON-пакет и отправляет на сервер веб-портала через REST-интерфейс. Для безопасности используется токен доступа.

Если сервер отклонил запрос из-за устаревшего токена, модуль сам запрашивает новый ключ, сохраняет его и повторяет отправку.

Когда токен правильный, сервер разбирает пакет и обновляет информацию в базе данных портала. После этого он возвращает ответ.

Завершается обмен записью результата в журнал синхронизации. Если всё прошло успешно, запись удаляется из очереди. Если произошла ошибка, система назначает время для повторной попытки.

2.3.2 Алгоритмизация процедур обработки и синхронизации данных

Для описания внутренней логики работы механизма спроектирована диаграмма деятельности. Она детально раскрывает последовательность технологических операций, правила принятия решений и алгоритмы преобразования данных на каждом этапе [13]. В отличие от модели межсистемного взаимодействия, здесь основной упор сделан на внутреннее устройство программных модулей и условия их переходов из одного состояния в другое.

Основная цель проектирования алгоритма в нотации UML заключается в декомпозиции сложного процесса синхронизации на простые, логически завершённые действия. Использование дорожек (swimlanes) на диаграмме позволяет четко разграничить зоны ответственности между действиями человека, внутренней логикой учетной системы, автономным модулем синхронизации и принимающим веб-порталом. Это обеспечивает прозрачность проектирования и позволяет на раннем этапе предусмотреть обработку

всевозможных нештатных ситуаций. Спроектированный алгоритм функционирования механизма представлен на рисунке 11.

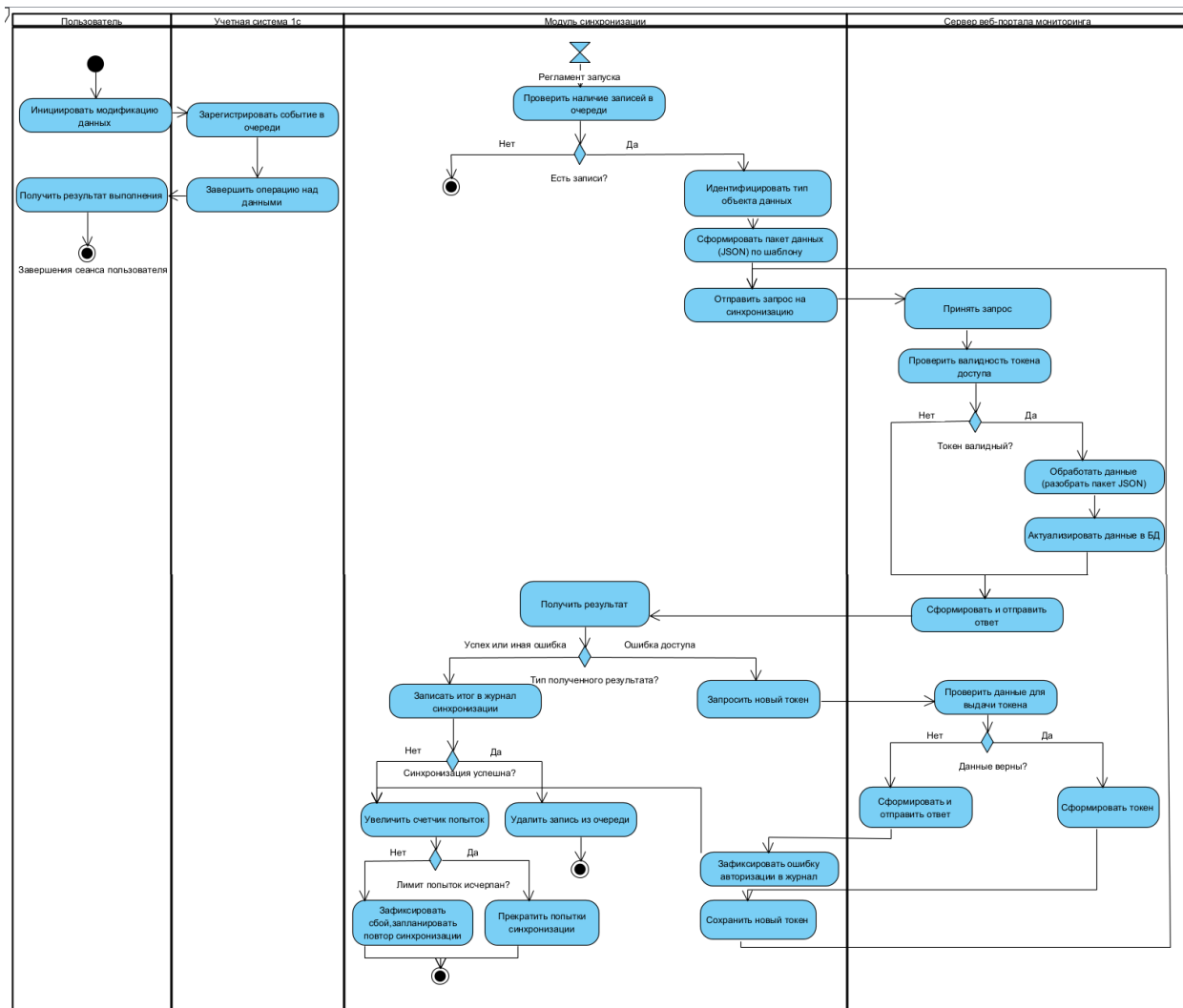


Рисунок 11 – Диаграмма деятельности механизма синхронизации

В основе механизма синхронизации лежат два независимых процесса.

Сначала система фиксирует изменения в тот момент, когда пользователь работает с данными в интерфейсе 1С. Тут срабатывает программный триггер. Он регистрирует событие в очереди обмена. Благодаря этому работа сотрудников не тормозится. Основной этап обмена запускается автоматически по расписанию (на схеме он отмечен песочными часами). Алгоритм проверяет, есть ли записи в очереди. Если есть – формирует JSON-пакет и отправляет его на веб-портал.

При отправке сервер проверяет защитный токен доступа. Если токен оказался недействительным, система сама запрашивает новый ключ, сохраняет

его в настройках и в этом же цикле повторяет отправку. Все результаты взаимодействия с сервером записываются в журнал синхронизации. Это нужно, чтобы администратор мог всё проверить. Если передача прошла успешно, запись из очереди удаляется полностью. Если случился сбой, система увеличивает счётчик попыток и назначает время следующего запуска. Когда он достигается лимит попыток, автоматическая обработка записи прекращается. Это защищает систему от лишней нагрузки, если произошла серьёзная ошибка связи.

В итоге такой подход даёт надёжную интеграцию.

2.3.3 Формализация жизненного цикла объектов обмена.

Для обеспечения гарантированной доставки информации и соблюдения режима, близкого к реальному времени необходимо детально проработать все фазы существования данных внутри интеграционного контура. Жизненный цикл каждой записи в очереди обмена представляет собой последовательность переходов между строго определенными статусами. Для визуализации этой логики была разработана диаграмма состояний, представленная на рисунке 12.

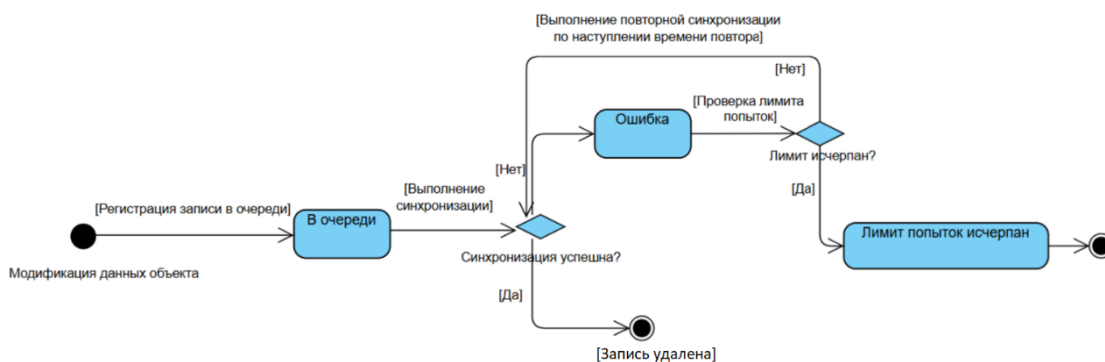


Рисунок 12 – Диаграмма состояний записи в очереди

Жизненный цикл данных в процессе синхронизации начинается с этапа модификации объекта. В этот момент срабатывает триггер, который регистрирует запись в очереди. Первичное стабильное состояние для объекта статус «В очереди». В этой фазе данные ждут запуска фонового процесса.

Дальнейшее изменение статуса зависит от того, как прошла синхронизация. Если от принимающей системы получено подтверждение,

запись вообще удаляется из очереди. Это значит, что цикл обмена завершился успешно и данные в обеих базах синхронизированы. Если же произошёл сбой (ошибка сети или сервера), объект переходит в состояние «Ошибка».

В спроектированном механизме есть специальный алгоритм управления повторами. Когда запись находится в состоянии «Ошибка», система проверяет, сколько попыток уже было сделано. Если лимит не исчерпан, механизм планирует повторный сеанс связи. При этом статус «В очереди» повторно не назначается – так снижается транзакционная нагрузка на базу данных. Система запускает повторную синхронизацию прямо из текущего состояния, когда наступает запланированное время.

Если же установленный лимит попыток полностью исчерпан, статус переходит в состояние «Лимит попыток исчерпан». Дальнейшие переходы прекращаются. Это состояние является финальным для ошибочных записей. В такой ситуации требуется вмешательство администратора, чтобы разобраться в причинах несовместимости данных. Спроектированная модель даёт высокую отказоустойчивость и гарантирует, что ни одна операция не будет потеряна при временных сбоях .

2.4 Проектирование архитектуры системы-приемника

Веб-портал состоит из серверной и клиентской частей. Серверная часть принимает данные от 1С, хранит их и обрабатывает. Клиентская часть показывает эти данные пользователю. Ниже спроектированы обе части.

2.4.1 Проектирование программной архитектуры и компонентной модели

При разработке серверной части механизма синхронизации нужно системно подойти к организации программного кода. Система должна работать в режиме, близком к реальному времени, и обрабатывать данные со сложной, меняющейся структурой. Поэтому выбрана многослойная архитектура [14]. Это необходимо, чтобы разделить ответственность. Каждый программный модуль

отвечает за свою конкретную задачу. В этом проекте архитектура веб-портала мониторинга разбита на следующие логические уровни.

Уровень представления (Presentation Layer). Это внешний фасад системы. Он принимает входящие сетевые соединения от учётной системы и отвечает за визуализацию данных для пользователя. Здесь проектируются API-интерфейсы. Они слушают внешние запросы, проверяют подлинность и передают данные дальше на обработку.

Слой бизнес-логики (Service Layer). Центральный компонент системы. Здесь проектируются основные алгоритмы обработки информации. Слой сервисов занимается расшифровкой входящих пакетов, управлением транзакциями и логикой работы с полиструктурными данными.

Слой доступа к данным (Data Access Layer). Этот слой работает как мост между Java-кодом и реляционными таблицами SQL. Он берёт на себя сложность формирования SQL-запросов. Разработчик может оперировать объектами предметной области, не вникая в детали работы с базой.

Слой базы данных (Database). Уровень физического хранения. Здесь обеспечивается сохранность и целостность информации. Взаимодействие этих уровней и общая компоновка системы представлены на диаграмме компонентов на рисунке 13.

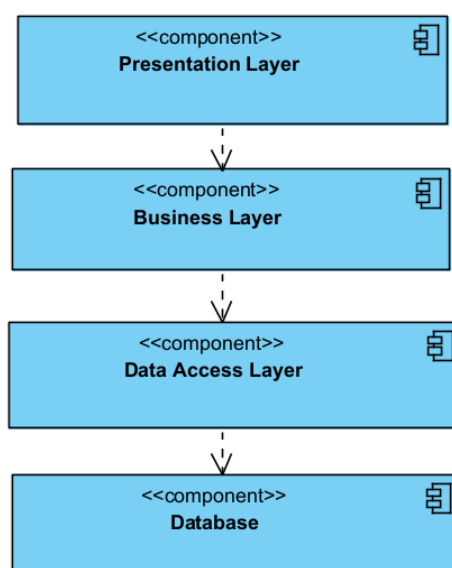


Рисунок 13 – Диаграмма компонентов веб-портала мониторинга



Основные уровни архитектуры – это слой бизнес-логики (Business Layer), слой взаимодействия с данными (Data Access Layer) и физическая база данных (Databases).

Контроллеры работают как программные интерфейсы. Они принимают внешние вызовы и распределяют потоки информации. В системе спроектированы такие компоненты:

- AuthController, принимает запросы на аутентификацию и возвращает JWT-токен при успешной авторизации;
- SyncController, принимает входящие REST-запросы от системы 1С и направляет их в сервис синхронизации для обработки данных;
- DataViewController, обрабатывает GET-запросы, возвращая HTML-страницы с данными для интерфейса мониторинга.

Сервисы – это логическое ядро приложения. Здесь выполняются основные алгоритмы обработки данных. В проекте выделены:

- AuthService, проверяет права доступа и подтверждает подлинность входящих запросов, не допуская несанкционированного доступа к базе данных;
- SyncService, решает ключевую задачу проекта: разбирает и преобразует сложные данные из 1С, и готовит информацию к сохранению;
- DataViewService, отвечает за подготовку данных для отображения на страницах веб-портала;

Объектные модели – это программное отражение документов и справочников предприятия. Их структура позволяет хранить не только основные реквизиты, но и расширенные наборы характеристик. В рамках системы используются:

- SalesOrder и SalesOrderItem (модели для работы с заказами покупателей и перечнем товаров.);
- SalesInvoice и SalesInvoiceItem (блоки для регистрации отгрузок и сопутствующих логистических данных);

- IncomingGoods и IncomingGoodsItem (объекты для учёта закупок и параметров качества продукции);
- PaymentIn и PaymentInItem (модели для фиксации входящих платежей с детализацией по конкретным заказам);
- PaymentOut (блок для регистрации исходящих финансовых операций);
- ApiUser (модель, содержащая параметры доступа для доверенных систем).

Репозитории – это связующее звено между программными объектами и таблицами базы данных.

Слой базы данных (Databases) – физическое хранилище на СУБД Microsoft SQL Server. Архитектура базы включает полный набор таблиц (api_users, sales_orders, sales_order_items, sales_invoices, sales_invoice_items, incoming_goods, incoming_goods_items, payments_in, payments_in_items, payments_out).

2.4.2 Проектирование механизмов визуализации данных

Визуальная оболочка веб-портала была спроектирована таким образом, чтобы получить функциональную среду для оперативного анализа синхронизированной информации. В данной работе пользовательский интерфейс рассматривается как конечный узел обработки полиструктурных данных. Здесь данные должны превращаться из технического формата базы данных в удобный для чтения вид. Ключевой особенностью интерфейса стала его адаптивность: структура отображаемых реквизитов автоматически подстраивается под изменения в составе документа.

Интерфейс был спроектирован с расчётом на то, что состав реквизитов документа может меняться, и он должен подстраиваться под это автоматически. Концепция интерфейса оперативной панели мониторинга показана на рисунке 15.

Панель мониторинга предприятия

Заказы Закупки Отгрузки Приход денег Расход денег

№ Заказа	Дата	Клиент	Менеджер	Склад	Доставка	Статус	Оплата	Сумма
000000025	07.06.2026 17:45	ИП Волков С.П.	Кузнецова Татьяна Борисов	Магазин-склад "ТехноСтрой"	Самовывоз	Новый	Не оплачен	4370,00 Р

Состав заказа:

Товар	Кол-во	Ед.	Цена	Сумма	Характеристики
Кирпич облицовочный красный	60,000	шт	24,50	1470,00	<div>Особые условия транспортировки</div> <div>Запрещена перевалка (не перегружать)</div> <div>Класс опасности груза</div> <div>Класс Б (средняя опасность)</div> <div>Зона выгрузки на объекте</div> <div>Сектор А (Фундамент)</div> <div>Дополнительная маркировка</div> <div>Наклейка "Хрупкое"</div>
Цемент М-500 ПЦ (50 кг)	5,000	упак	580,00	2900,00	<div>Тип упаковки</div> <div>На поддонах</div>

000000024	04.06.2026 15:42	Учредитель (Инвестиции)	Кузнецова Татьяна Борисов	Магазин-склад "ТехноСтрой"	Самовывоз	Новый	Не оплачен	122,50 Р
000000023	03.06.2026 17:39	Наша компания (ТехноСтрой-М)	Сидоров Андрей Николаевич	Транзитная площадка (Юг)	Самовывоз	Новый	Не оплачен	2900,00 Р
000000022	31.05.2026 13:21	Учредитель (Инвестиции)	Морозов Артем Александров	Транзитная площадка (Юг)	Самовывоз	Новый	Оплачен	2900,00 Р
000000021	31.05.2026 11:39	Учредитель (Инвестиции)	Иванова Елена Дмитриевна	Магазин-склад "ТехноСтрой"	Самовывоз	Новый	Оплачен	1203,00 Р
000000016	31.05.2026 09:17	ИП Волков С.П.	Сидоров Андрей Николаевич	Транзитная площадка (Юг)	Самовывоз	Новый	Не оплачен	98,00 Р

Рисунок 15 – Интерфейс веб-портала мониторинга

Архитектура панели построена по модульному принципу. Была использована система вкладок. Каждая вкладка соответствует своему бизнес-контуру:

- заказы;
- закупки;
- отгрузки;
- приход денег;
- расход денег.

При проектировании интерфейса отдельно была продумана визуализация полиструктурных характеристик. На схематичном рисунке интерфейса видно: при детализации заказа система выводит вложенную таблицу с составом и свойствами товаров. Логика отображения таких атрибутов построена на динамическом формировании элементов. Эти характеристики хранятся в СУБД в формате JSON, поэтому в проект была заложена клиентская логика на JavaScript. Алгоритм визуализации работает так. В момент загрузки страницы содержимое полиструктурированных полей автоматически разбирается. Получается гибкий интерфейс: состав информационных блоков на экране собирается программно из метаданных того пакета, который пришёл. Система становится визуально устойчивой. Если в учётной системе 1С появятся новые

типы данных, они отобразятся на портале сами – переделывать HTML-разметку не потребуется. В итоге спроектированные механизмы визуализации замыкают цикл обработки данных и дают инструмент для оперативного контроля предприятия в режиме , близком к реальному времени

2.5 Проектирование архитектуры развертывания и логической структуры системы

Для того чтобы показать, как физически будут размещены компоненты механизма синхронизации, была построена диаграмма развёртывания , которая представлена на рисунке 16. На этапе проектирования такая схема помогает заранее определить, какие программные среды и артефакты понадобятся для устойчивой работы системы.

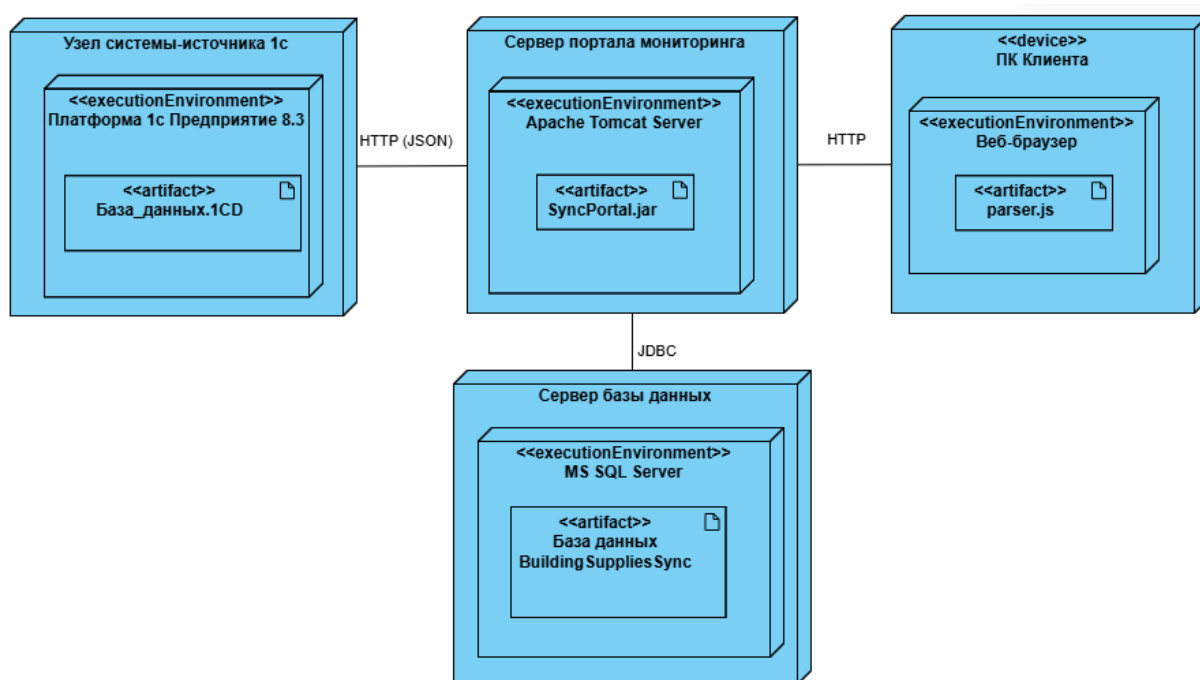


Рисунок 16 – Диаграмма развертывания

Узел системы-источника (1С). В этом узле развёрнута платформа «1С:Предприятие 8.3». Внутри неё находится информационная база (артефакт База данных.1CD). Именно здесь хранятся все данные предприятия: документы, справочники, регистры и другие объекты учёта. Из этого узла инициируется отправка данных на веб-портал.

Сервер портала мониторинга. Здесь в среде Apache Tomcat будет запущено Java-приложение (артефакт SyncPortal.jar). Данный узел спроектирован для приёма REST-запросов как от системы 1С, так и от браузеров пользователей. Он будет обрабатывать эти запросы и взаимодействовать с базой данных.

Клиентский узел (ПК пользователя). На нём предполагается работа веб-браузера. Через HTTP браузер будет отправлять запросы к серверу портала и получать в ответ HTML-страницы, CSS и JavaScript. На стороне клиента предусмотрено выполнение парсинга полиструктурных данных и их отображение пользователю.

Сервер базы данных. Это отдельный узел с СУБД Microsoft SQL Server. На нём будет развёрнута база данных Building Supplies Sync. В ней предполагается хранение всех данных, синхронизированных с учётной системой.

В дополнение к физической схеме развёртывания была разработана диаграмма компонентов, представленная на рисунке 17. Она показывает логическую структуру системы: из каких модулей она состоит и как между ними распределены функции.

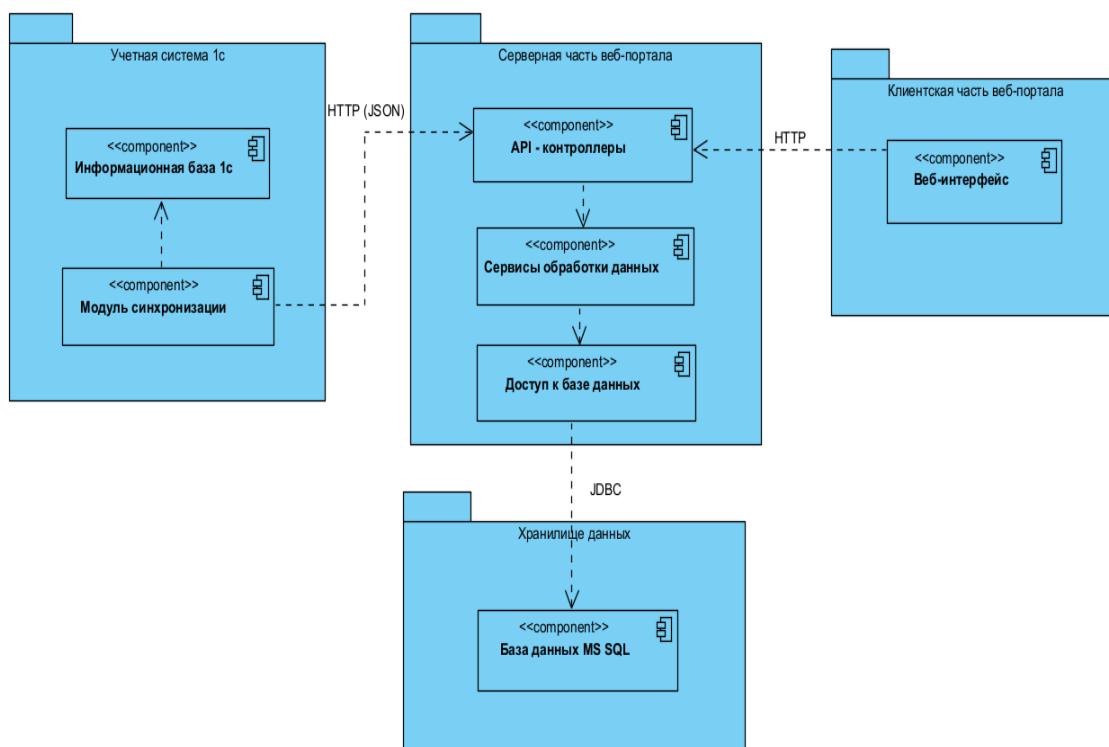


Рисунок 17 – Диаграмма компонентов

Учётная система 1С выступает источником данных. В ней спроектированы два основных компонента: информационная база (хранилище документов, справочников и регистров) и модуль синхронизации, который отвечает за регистрацию изменений и отправку данных на портал.

Серверная часть веб-портала включает три компонента. API-контроллеры принимают REST-запросы от 1С и от браузера. Сервисы обработки данных реализуют бизнес-логику: проверку токенов, парсинг JSON, вызов репозитория. Компонент доступа к базе данных обеспечивает сохранение и извлечение информации через JPA и JDBC.

Клиентская часть веб-портала содержит компонент «Веб-интерфейс», который работает в браузере пользователя. Он отвечает за отображение данных, выполнение JavaScript-кода и парсинг полиструктурных JSON-полей.

Хранилище данных представлено компонентом «База данных MS SQL». Он обеспечивает физическое хранение всех синхронизированных данных.

Взаимодействие между компонентами организовано через протоколы HTTP для связи клиента с сервером и 1С с сервером и JDBC для связи сервера с базой данных. Данные передаются в формате JSON, что обеспечивает гибкость при работе с полиструктурной информацией.

3 Программная реализация механизма синхронизации

3.1 Реализация системы-источника

3.1.1 Реализация компонентов системы-источника

Программная реализация механизма синхронизации была выполнена на основе существующей конфигурации, предназначенной для автоматизации торговой деятельности магазина строительных материалов. Данная система уже содержала базовый набор объектов для ведения учёта (справочники номенклатуры, контрагентов, складов и т. д.).

В среде «1С:Предприятие 8.3» основные работы были сосредоточены на двух направлениях. Было выполнено проектирование и внедрение новых объектов метаданных, которые составили техническое ядро синхронизации. Также была проведена модернизация существующих документов для обеспечения передачи полиструктурированных данных. Типовые объекты учёта, не задействованные в процессе обмена, в рамках данной работы не описываются – это позволило сконцентрироваться на разработанном механизме. Диаграмма компонентов системы-источника представлена на рисунке 18.

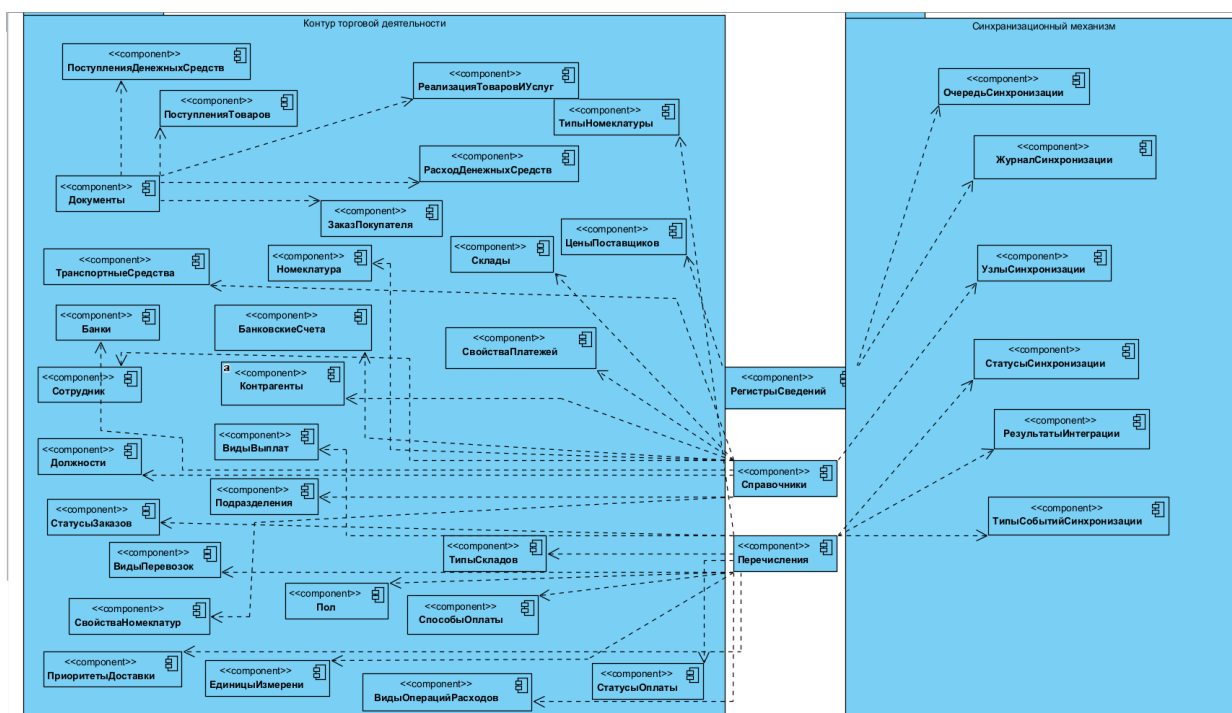


Рисунок 18 – Диаграмма компонентов системы-источника

Построенная модель наглядно разделяет информационную систему на два независимых и автономных контура. Первый контур отвечает за торговую деятельность предприятия. В него входят прикладные объекты: справочники, документы, перечисления и так далее. Они выступают в роли первичных источников информации. В структуру этих объектов были встроены дополнительные механизмы для фиксации и захвата изменений в данных. Второй контур – это сам механизм синхронизации. Он состоит из служебных компонентов, которые были разработаны с нуля. Эти компоненты обеспечивают работу очереди на отправку, автоматическое ведение журналов транзакций и безопасное хранение настроек подключения к внешним ресурсам.

Дальше дано детальное описание реализованных компонентов.

Перечисление – это объект метаданных, который хранит фиксированный список значений. В отличие от справочников, набор вариантов в перечислении задаётся только на этапе разработки. Такие объекты нужны для типизации данных и управления логикой алгоритмов [14].

Для разработанного механизма созданы следующие перечисления, отвечающие за техническую сторону синхронизации:

- «СтатусыСинхронизации»;
- «ТипыСобытийСинхронизации»;
- «РезультатыСинхронизации».

Перечисление «Статусы синхронизации» нужно для управления жизненным циклом записи в очереди обмена. Оно включает три значения: «В очереди», «Ошибка», «ЛимитПопытокИсчерпан». Перечисления показаны на рисунке 19.

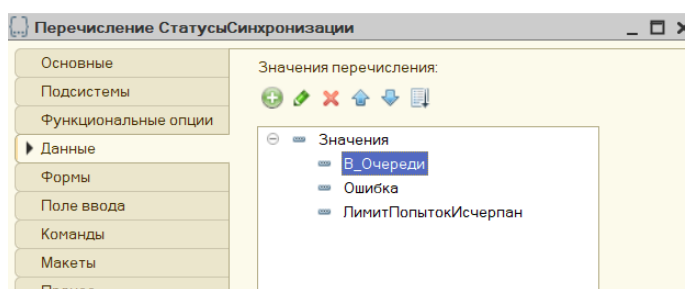


Рисунок 19 – Перечисление «СтатусыСинхронизации»

Перечисление «ТипыСобытийСинхронизации» используется для определения действий пользователя над документами. В его состав входят значения: «Создание», «Обновление» и «Удаление». Перечисления с типами показаны на рисунке 20.

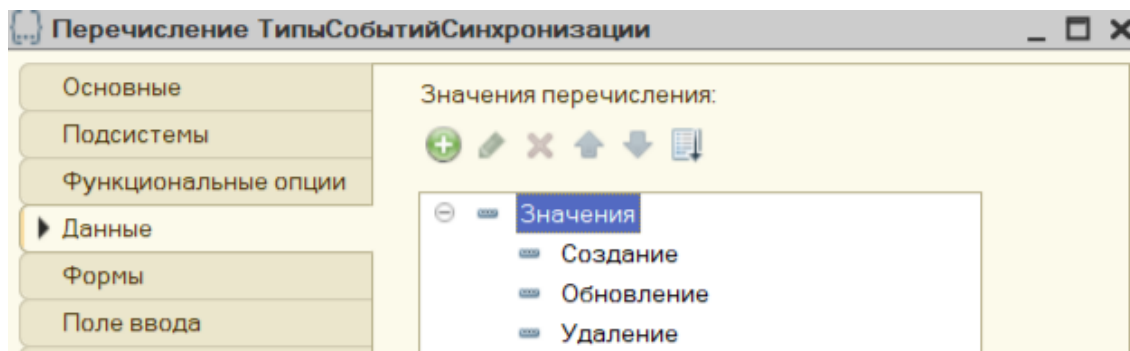


Рисунок 20 – Перечисление «ТипыСобытийСинхронизации»

Перечисление «РезультатыСинхронизации» сделано специально для электронного журнала истории. Оно содержит два варианта: «Успех» и «Ошибка». Благодаря этому перечислению система автоматически классифицирует каждую попытку обмена. Перечисления с результатами показаны на рисунке 21.

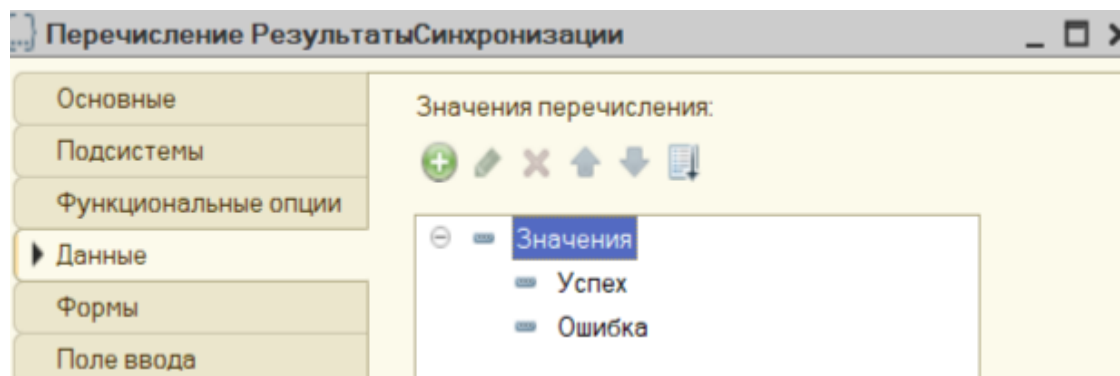


Рисунок 21 – Перечисление «РезультатыСинхронизации»

Справочник – это объект системы, где хранятся списки данных с одинаковой структурой. У каждого элемента справочника есть уникальный код и наименование, это помогает легко идентифицировать записи [13].

Для работы механизма синхронизации в системе были созданы и доработаны и несколько справочников.

Справочник «УзлыСинхронизации». В нём хранятся все настройки для связи с веб-порталом. Доступ к этому справочнику имеют только администраторы системы. Обычные пользователи не могут просматривать или изменять реквизиты этого справочника, В справочнике есть реквизиты для URL-адреса сервера и признака активности канала обмена. Также здесь хранятся логин и дата истечения токена. Объект доступен в подсистеме «Управление синхронизацией данных». Справочник показан на рисунке 22.

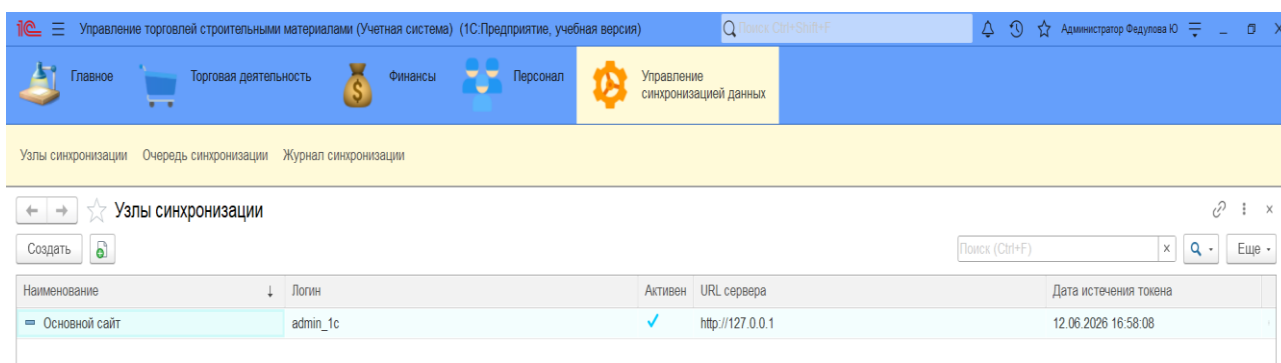


Рисунок 22 – Справочник «УзлыСинхронизации»

В системе 1С документ – это главный объект для регистрации событий хозяйственной жизни предприятия [15]. В отличие от справочников, документы всегда привязаны к конкретному моменту времени (дата и время). Документы, перечисленные ниже, выступают источниками данных для обмена.

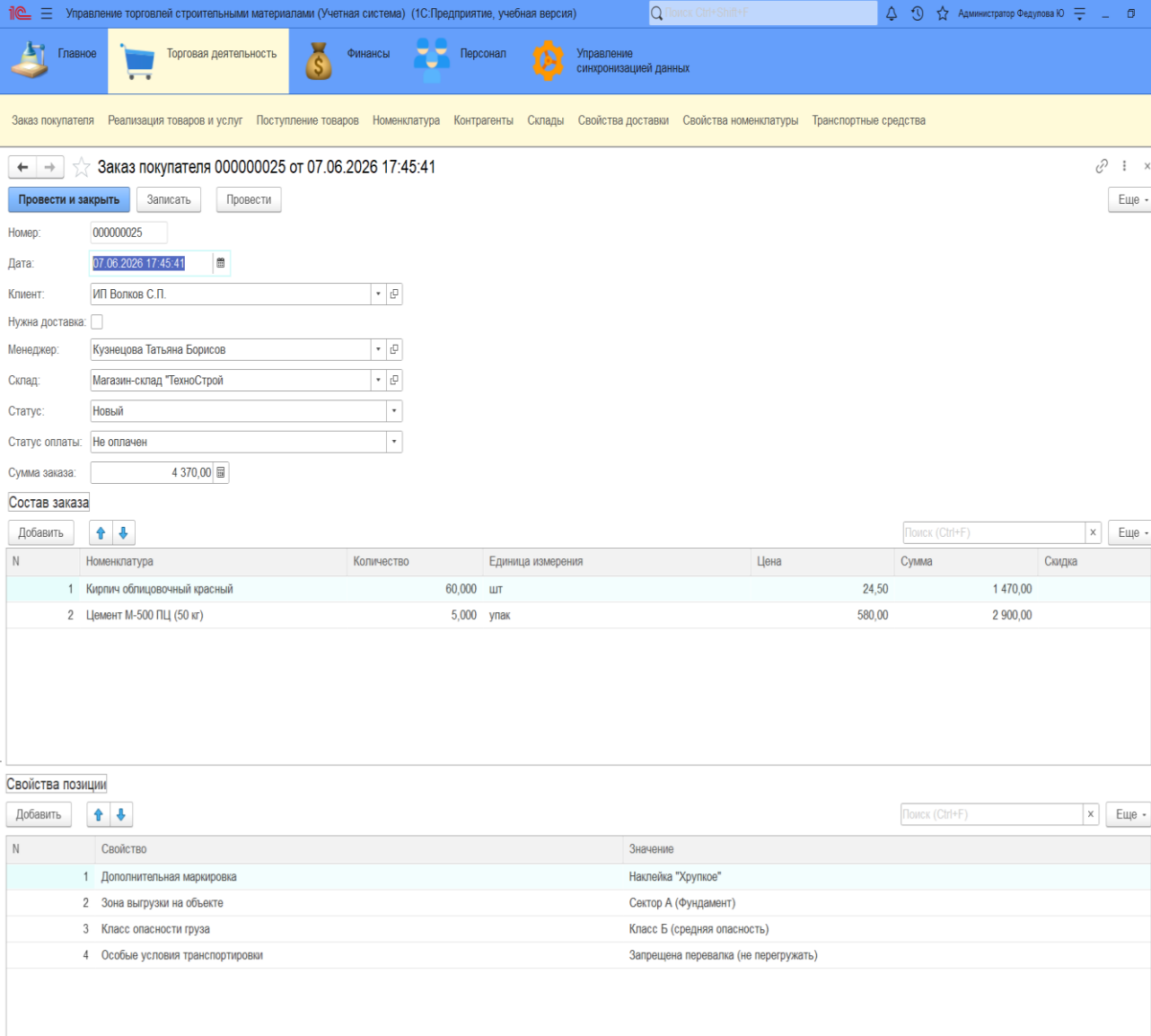
Для работы механизма синхронизации были доработаны и задействованы пять документов:

- «ЗаказПокупателя»;
- «ПоступлениеТоваров»;
- «РеализацияТоваров»;
- «ПоступлениеДенежныхСредств»;
- «РасходДенежныхСредств».

Документ «ЗаказПокупателя» фиксирует предварительную договорённость с клиентом о поставке материалов.

Чтобы поддерживать гибкий состав данных, в документ добавлена табличная часть «Свойства позиции» для хранения свойств товаров. Связь

между товаром и его свойствами обеспечивается скрытым полем «Ключ связи». Документ доступен в подсистеме «Торговая деятельность» и показан на рисунке 23.



Управление торговлей строительными материалами (Учетная система) (1С:Предприятие, учебная версия)

Главное Торговая деятельность Финансы Персонал Управление синхронизацией данных

Заказ покупателя Реализация товаров и услуг Поступление товаров Номенклатура Контрагенты Склады Свойства доставки Свойства номенклатуры Транспортные средства

Заказ покупателя 000000025 от 07.06.2026 17:45:41

Провести и закрыть Записать Провести

Номер: 000000025

Дата: 07.06.2026 17:45:41

Клиент: ИП Волков С.П.

Нужна доставка: ☐

Менеджер: Кузнецова Татьяна Борисов

Склад: Магазин-склад "ТехноСтрой"

Статус: Новый

Статус оплаты: Не оплачен

Сумма заказа: 4 370,00

Состав заказа

Добавить

N	Номенклатура	Количество	Единица измерения	Цена	Сумма	Скидка
1	Кирпич облицовочный красный	60,000	шт	24,50	1 470,00	
2	Цемент М-500 ПЦ (50 кг)	5,000	упак	580,00	2 900,00	

Свойства позиций

Добавить

N	Свойство	Значение
1	Дополнительная маркировка	Наклейка "Хрупкое"
2	Зона выгрузки на объекте	Сектор А (Фундамент)
3	Класс опасности груза	Класс Б (средняя опасность)
4	Особые условия транспортировки	Запрещена перевалка (не перегружать)

Рисунок 23 – Документ «ЗаказПокупателя»

Документ «ПоступлениеТоваров» используется при закупке стройматериалов у поставщиков. В структуру документа возможность регистрировать параметры качества для каждой принятой позиции для этого создана табличная часть «Технические параметры».

Объект доступен в подсистеме «Торговая деятельность» и показан на рисунке 24.

Управление торговых строительных материалами (Учетная система) (1С:Предприятие, учебная версия) Поиск (Ctrl+F) Администратор Федюлова Ю

Главное Торговая деятельность Финансы Персонал Управление синхронизацией данных

Заказ покупателя Реализация товаров Поступление товаров Номенклатура Контрагенты Склады Свойства доставки Свойства номенклатуры Транспортные средства

Поступление товаров 000000004 от 04.06.2026 15:46:33 *

Провести и закрыть Записать Провести Еще

Номер: 000000004

Дата: 04.06.2026 15:46:33

Поставщик: ОАО "ЦементМастер"

Склад: Транзитная площадка (Юг)

Организация: Учредитель (Инвестиции)

Сумма документа: 52,00

Статус оплаты: Не оплачен

Состав поступления

Добавить Поиск (Ctrl+F) Еще

N	Номенклатура	Количество	Единицы измерения	Цена	Сумма
1	Кирпич облицовочный красный	4,000	шт	13,00	52,00

Технические параметры

Добавить Поиск (Ctrl+F) Еще

N	Параметр	Значение
1	Особые условия транспортировки	Запрещена перевалка (не перегружать)
2	Дата прохождения контроля	01.06.2026

Рисунок 24 – Документ «ПоступлениеТоваров»

Документ «РеализацияТоваров» отражает факт отгрузки материалов со склада. Этот документ ключевой для логистического мониторинга. Его форма расширена специальными анкетами, где фиксируются сведения о водителе, конкретном автомобиле и приоритете доставки. Этот документ отличается от других тем, что его полиструктурные данные относятся не к товарам, а к процессу доставки. Документ тесно связан с другими объектами системы – заказом покупателя и складскими остатками. При проведении реализации автоматически проверяется наличие товара на складе и фиксируется его списание. Документ доступен в подсистеме «Торговая деятельность» Основная форма его представлена на рисунке 25.

Управление торговлей строительными материалами (Учетная система) (1С:Предприятие, учебная версия)

Главное Торговая деятельность Финансы Персонал Управление синхронизацией данных

Заказ покупателя Реализация товаров Поступление товаров Номенклатура Контрагенты Склады Свойства доставки Свойства номенклатуры Транспортные средства

← → ☆ Реализация товаров 000000005 от 09.06.2026 12:07:21

Провести и закрыть Записать Провести

Еще -

Основные данные Данные о доставке

Номер: 000000005

Дата: 09.06.2026 12:07:21

Менеджер: Кузнецова Татьяна Борисовна

Заказ основание: Заказ покупателя 0000000025 от 07.06.2026 17:45:41

Нужна доставка: ☒

Клиент: ИП Волков С.П.

Сумма документа: 4 370,00

Склад: Магазин-склад "ТехноСтрой"

Добавить

N	Номенклатура	Количество	Единицы измерения	Цена	Сумма
1	Кирпич облицовочный красный	60,000	шт	24,50	1 470,00
2	Цемент М-500 ПЦ (50 кг)	5,000	упак	580,00	2 900,00

Поиск (Ctrl+F)

Еще -

Рисунок 25 – Документ «РеализацияТоваров»

При активации поля «Нужна доставка» динамически появляется вкладка с данными о доставке. По ней открывается специализированная форма представленная на рисунке 26.

Управление торговлей строительными материалами (Учетная система) (1С:Предприятие, учебная версия)

Главное Торговая деятельность Финансы Персонал Управление синхронизацией данных

Заказ покупателя Реализация товаров Поступление товаров Номенклатура Контрагенты Склады Свойства доставки Свойства номенклатуры Транспортные средства

← → ☆ Реализация товаров 000000005 от 09.06.2026 12:07:21 *

Провести и закрыть Записать Провести

Еще -

Основные данные Данные о доставке

Данные о перевозчике: Водитель: Морозов Артем Александров, Тел: +7 (900) 666-77-8

Транспортные данные: Номер: 34353531135, Дата: 09.06.2026, Компания: ИП Грузовоз

Приоритет доставки: Низкий

Рисунок 26 – Вкладка «Данные о доставке» в документе реализации

При нажатии на поле «Данные о перевозчике» появляется форма с реквизитами ответственного за доставку. В этой форме отображаются все ключевые реквизиты ответственного за доставку. Эти дополнительные поля не являются обязательными для документа «Реализация товаров» и появляются только при определённых условиях, что делает документ полиструктурным. Форма представлена на рисунке 27.

The screenshot shows the '1C: Enterprise' interface with the 'Transportation' module selected. The form is titled 'Реквизиты ответственного за доставку' (Requisites of the person responsible for delivery). It contains the following fields:

- ФИО: Морозов Артем Александров
- Телефон: +7 (900) 666-77-88
- Статус инструктажа по ТБ: Спец-инструктаж по работе
- Проверка СИЗ: Комплект полный (маска, жилет, обувь)

There is a 'Завершить ввод' (Finish input) button at the bottom left and an 'Еще' (More) button at the top right.

Рисунок 27 – Форма «Реквизиты ответственного за доставку» в документе

При нажатии на поле «Транспортные данные» появляется форма со спецификацией транспортной логистики. Она представлена на рисунке 28.

The screenshot shows the '1C: Enterprise' interface with the 'Transportation' module selected. The form is titled 'Спецификация транспортной логистики' (Specification of transport logistics). It contains the following fields:

- Номер ТТН: 34353531135
- Дата ТТН: 09.06.2026
- Вид перевозки: Грузовой тяжелый
- Перевозчик: ИП Грузовоз (Иванов)
- Транспорт: Volvo FH16 (Длинномер)

There is a 'Завершить ввод' (Finish input) button at the bottom left and an 'Еще' (More) button at the top right.

Рисунок 28 – Форма «Спецификация транспортной логистики» в документе

Документ «Поступление Денежных Средств» фиксирует оплату от клиентов. Документ содержит таблицу «Свойства платежа». Это позволяет для каждого оплачиваемого заказа указать индивидуальный набор параметров. Например, для одного платежа могут быть заполнены только сумма и форма оплаты, а для другого – дополнительно номер чека, терминал, признак аванса, банк плательщика и комментарий кассира. Состав свойств не фиксирован жёстко и определяется в момент проведения оплаты в зависимости от ситуации. Такая модель делает финансовый поток полиструктурным – состав свойств может быть уникальным для каждого входящего платежа. Объект доступен в подсистеме «Финансы». Документ «Поступление Денежных Средств» представлен на рисунке 29.

Управление торговлей строительными материалами (Учетная система) (1С:Предприятие, учебная версия)

ГлавноеТорговая деятельностьФинансыПерсоналУправление синхронизацией данных

Поступление денежных средствРасход денежных средствБанкиБанковские счетаЕдиницы измеренияСвойства платежа

Поступление денежных средств 000000016 от 01.06.2026 13:50:09 *

Провести и закрытьЗаписатьПровести

Номер: 000000016
Дата: 01.06.2026 13:50:09
Организация: Наша компания (ТехноСтрой-М)
Клиент: ОАО "ЦементМастер"
Способ оплаты: Наличные
Сумма документа: 2 900,00

Расшифровка платежа

Добавить

N	Заказ	Сумма
1	Заказ покупателя 000000022 от 31.05.2026 13:21:43	2 900,00

Свойства платежа

Добавить

N	Свойство	Значение
1	Процент банковской комисс	3%
2	Промокод	A563521

Рисунок 29 – Документ «ПоступлениеДенежныхСредств»

Документ «РасходДенежныхСредств» отвечает за выплаты поставщикам и сотрудникам При его проектировании было учтено, что состав полей на форме может динамически меняться в зависимости от выбранного вида операции. В этом и заключается полиструктурированность для этого документа Объект включён в подсистему «Финансы» и представлен н рисунке 30.

Управление торговлей строительными материалами (Учетная система) (1С:Предприятие, учебная версия)

ГлавноеТорговая деятельностьФинансыПерсоналУправление синхронизацией данных

Поступление денежных средствРасход денежных средствБанкиБанковские счетаЕдиницы измеренияСвойства платежа

Расход денежных средств 000000007 от 13.05.2026 23:39:02

Провести и закрытьЗаписатьПровести

Номер: 000000007
Дата: 13.05.2026 23:39:02
Организация: Наша компания (ТехноСтрой-М)
Вид операции: Оплата поставщику
Получатель: ОАО "ЦементМастер"
Документ закупки: Поступление товаров 000000003 от 08.05.2026 17:19:33
Сумма: 2 465,00

Рисунок 30 – Документ «РасходДенежныхСредств»

Регистры отличаются от справочников и документов. Они хранят данные в разрезе нескольких измерений. Благодаря этому система быстро находит нужную информацию и эффективно управляет очередностью задач [16].

Для работы механизма синхронизации создали два регистра сведений: «ОчередьСинхронизации» и «ЖурналСинхронизации».

Регистр «ОчередьСинхронизации». Его главная задача сделать хранить данные для синхронизации. В измерениях объекта фиксируются идентификатор документа, тип события, и точное время. В ресурсах хранятся текущее состояние (например, «В очереди» или «Ошибка»), объектное представление и данные для повторных попыток отправки. Объект доступен в подсистеме «Управление синхронизацией данных» и представлен на рисунке 31.

Объект ID	Тип события	Дата события	Тип объекта	Объект представление	Состояние
09a92f1b-663c-11f1-bac3-20c19b10792a	Обновление	12.06.2026 13:48:55	ЗаказПопкупателя	Заказ покупателя 000000026 от 12.06.2026 13...	В очереди
17bb671d-5da7-11f1-bac2-c4170e60682e	Обновление	12.06.2026 13:49:08	ПоступлениеДенежныйСредств	Поступление денежных средств 000000016 о...	В очереди
1dc12b96-6009-11f1-bac3-20c19b10792a	Обновление	12.06.2026 13:49:01	ПоступлениеТоваров	Поступление товаров 000000004 от 04.06.20...	В очереди
36d803bb-5cd0-11f1-bac2-c4170e60682e	Обновление	12.06.2026 13:49:08	ЗаказПопкупателя	Заказ покупателя 000000022 от 31.05.2026 13...	В очереди
40cdc8ec-5ccc-11f1-bac2-c4170e60682e	Обновление	12.06.2026 13:48:45	ЗаказПопкупателя	Заказ покупателя 000000021 от 31.05.2026 11...	В очереди
6c2b6518-5f1e-11f1-bac3-20c19b10792a	Обновление	12.06.2026 13:48:48	ЗаказПопкупателя	Заказ покупателя 000000023 от 03.06.2026 17...	В очереди
836f3e1a-664c-11f1-bac3-20c19b10792a	Создание	12.06.2026 13:50:24	ЗаказПопкупателя	Заказ покупателя 000000027 от 12.06.2026 13...	В очереди
8de886e8-627f-11f1-bac6-20c19b10792a	Обновление	12.06.2026 13:48:50	ЗаказПопкупателя	Заказ покупателя 000000025 от 07.06.2026 17...	В очереди

Рисунок 31 – Регистр сведений «Очередьсинхронизации»

Регистр «ЖурналСинхронизации» нужен для аудита и контроля качества обмена. В очереди записи постоянно обновляются. А в журнале сохраняется полная история каждой попытки синхронизации. Здесь фиксируется статус идентификатор объекта, его представление, статус синхронизации, само содержимое отправленного пакета в формате JSON, а также ответ от веб-портала. Такой подход делает механизм синхронизации полностью прозрачным и упрощает его отладку.

Объект доступен в подсистеме «Управление синхронизацией данных» и представлен на рисунке 32.

Управление торговой строительными материалами (Учетная система) (1С:Предприятие, учебная версия)

ГлавноеТорговая деятельностьФинансыПерсоналУправление синхронизацией данных

Узлы синхронизацииОчередь синхронизацииЖурнал синхронизации

Журнал синхронизации

Создать

Поиск (Ctrl+F)

Еще

Период	Объект ID	Объект представления	Статус	Текст сообщения	Ответ сервера	Узел
04.06.2026 12:25:28	9bfca488-5fe6-11f1-bac3-20c19b10792a	Заказ покупателя 0000000024 от 04.06.2026 12:21:39	Успех	{...}	UPDATED	Основной сайт
04.06.2026 12:25:53	9bfca488-5fe6-11f1-bac3-20c19b10792a	Заказ покупателя 0000000024 от 04.06.2026 12:25:51	Успех	{...}	UPDATED	Основной сайт
04.06.2026 12:36:37	9bfca488-5fe6-11f1-bac3-20c19b10792a	Заказ покупателя 0000000024 от 04.06.2026 12:36:33	Успех	{...}	UPDATED	Основной сайт
04.06.2026 12:55:50	9bfca488-5fe6-11f1-bac3-20c19b10792a	Заказ покупателя 0000000024 от 04.06.2026 12:55:50	Успех	{...}	UPDATED	Основной сайт
04.06.2026 14:32:52	1dc12b96-6009-11f1-bac3-20c19b10792a	Поступление товаров 0000000004 от 04.06.2026 14:3...	Успех	{...}	CREATED	Основной сайт
04.06.2026 14:33:52	1dc12b96-6009-11f1-bac3-20c19b10792a	Поступление товаров 0000000004 от 04.06.2026 14:3...	Успех	{...}	UPDATED	Основной сайт
04.06.2026 14:34:34	1dc12b96-6009-11f1-bac3-20c19b10792a	Поступление товаров 0000000004 от 04.06.2026 14:3...	Успех	{...}	UPDATED	Основной сайт
04.06.2026 14:36:50	1dc12b96-6009-11f1-bac3-20c19b10792a	Поступление товаров 0000000004 от 04.06.2026 14:3...	Успех	{...}	UPDATED	Основной сайт
04.06.2026 15:05:31	17bb671d-5da7-11f1-bac2-c4170e60682e	Поступление денежных средств 0000000016 от 01.06...	Успех	{...}	UPDATED	Основной сайт
04.06.2026 15:05:31	36d803bb-5cd0-11f1-bac2-c4170e60682e	Заказ покупателя 0000000022 от 31.05.2026 13:21:43	Успех	{...}	UPDATED	Основной сайт
04.06.2026 15:22:10	9bfca488-5fe6-11f1-bac3-20c19b10792a	Заказ покупателя 0000000024 от 04.06.2026 15:22:08	Успех	{...}	UPDATED	Основной сайт
04.06.2026 15:42:21	9bfca488-5fe6-11f1-bac3-20c19b10792a	Заказ покупателя 0000000024 от 04.06.2026 15:39:59	Успех	{...}	UPDATED	Основной сайт
04.06.2026 15:42:26	9bfca488-5fe6-11f1-bac3-20c19b10792a	Заказ покупателя 0000000024 от 04.06.2026 15:42:25	Успех	{...}	UPDATED	Основной сайт
04.06.2026 15:43:02	1dc12b96-6009-11f1-bac3-20c19b10792a	Поступление товаров 0000000004 от 04.06.2026 15:4...	Успех	{...}	UPDATED	Основной сайт
04.06.2026 15:46:34	1dc12b96-6009-11f1-bac3-20c19b10792a	Поступление товаров 0000000004 от 04.06.2026 15:4...	Успех	{...}	UPDATED	Основной сайт
04.06.2026 15:49:50	17bb671d-5da7-11f1-bac2-c4170e60682e	Поступление денежных средств 0000000016 от 01.06...	Успех	{...}	UPDATED	Основной сайт
04.06.2026 15:49:50	36d803bb-5cd0-11f1-bac2-c4170e60682e	Заказ покупателя 0000000022 от 31.05.2026 13:21:43	Успех	{...}	UPDATED	Основной сайт
04.06.2026 15:50:41	17bb671d-5da7-11f1-bac2-c4170e60682e	Поступление денежных средств 0000000016 от 01.06...	Успех	{...}	UPDATED	Основной сайт

Рисунок 32 – Регистр сведений «ЖурналСинхронизации»

Перечисленные выше объекты метаданных составили техническую основу системы-источника и были полностью реализованы в среде «1С:Предприятие 8.3».

3.1.2 Реализация событийного механизма захвата изменений

Чтобы механизм синхронизации работал в автоматическом режиме, в системе реализовали событийную модель. В её основе лежат подписки на события – специальный инструмент платформы «1С:Предприятие 8.3». Он позволяет мгновенно запускать программный код, когда с данными совершаются какие-либо действия.

В этом проекте программные триггеры настроили для всех пяти ключевых документов, участвующих в обмене. Благодаря подпискам удалось организовать централизованный захват изменений. Как только пользователь нажимает кнопку «Провести» или «Удалить» в любом из этих документов, система автоматически обращается к процедурам в общем модуле. Те, в свою очередь, фиксируют

задачу на обмен в специальном техническом хранилище – «Очереди синхронизации».

Основной триггер привязан к системному событию «ПриЗаписи». Он срабатывает каждый раз, когда документ сохраняется в базе данных. За это отвечает процедура-обработчик под названием «ЗарегистрироватьИзменение».

У этой процедуры есть несколько последовательных действий. Сначала алгоритм определяет, впервые ли документ попал в контур синхронизации. Если в истории обмена нет записей об успешной передаче этого объекта, событию присваивается признак «Создание». Если записи есть классификация идёт как «Обновление». Затем обработчик выполняет физическую запись собранных сведений в технический регистр «Очередь синхронизации». Такая схема позволяет мгновенно зафиксировать факт изменения. Программная реализация этого алгоритма показана в листинге 1.

Листинг 1 – Процедура «ЗарегистрироватьИзменение»

```
Процедура ЗарегистрироватьИзменение (Источник, ТипСобытия) Экспорт
    Менеджер =
    РегистрыСведений.ОчередьСинхронизации.СоздатьМенеджерЗаписи ();
    Менеджер.ОбъектID =
    Строка (Источник.Ссылка.УникальныйИдентификатор ());
    Менеджер.ТипОбъекта = Источник.Метаданные ().Имя;
    Менеджер.ДатаСобытия = ТекущаяДата ();
    Если Источник.ПометкаУдаления Тогда
        Менеджер.ТипСобытия =
        Перечисления.ТипыСобытийСинхронизации.Удаление;
    Иначе
        // Проверка на существование успешной отправки в очереди
        Запрос = Новый Запрос;
        Запрос.Текст = "ВЫБРАТЬ ПЕРВЫЕ 1 ОбъектID ИЗ
        РегистрСведений.ОчередьСинхронизации
        | ГДЕ ОбъектID = &ID И Состояние =
        ЗНАЧЕНИЕ (Перечисление.СтатусыСинхронизации.Отправлено) ";
        Запрос.УстановитьПараметр ("ID", Менеджер.ОбъектID);
        Если Запрос.Выполнить ().Пустой () Тогда
            Менеджер.ТипСобытия =
            Перечисления.ТипыСобытийСинхронизации.Создание;
        Иначе
            Менеджер.ТипСобытия =
            Перечисления.ТипыСобытийСинхронизации.Обновление;
        КонецЕсли;
    КонецЕсли;
```

```

Менеджер.Состояние =
Перечисления.СтатусыСинхронизации.В_Очереди;
Менеджер.Записать ();
КонецПроцедуры

```

Вторая подписка настроена на событие «ПередУдалением». Она необходима для обработки ситуаций, когда документ полностью удаляется из информационной базы. В качестве обработчика для этого события была разработана процедура «ЗарегистрироватьУдаление».

Данный алгоритм позволяет зафиксировать уникальный идентификатор объекта за мгновение до его физического удаления. Процедура создает в очереди синхронизации запись с типом события «Удаление». Это гарантирует, что впоследствии, и данные будут удалены не только в 1С, но и во внешней базе данных SQL. Код обработчика приведен в листинге 2.

Листинг 2 – Процедура «ЗарегистрироватьУдаление»

```

Процедура ЗарегистрироватьУдаление(Источник, Отказ) Экспорт
Менеджер =
РегистрыСведений.ОчередьСинхронизации.СоздатьМенеджерЗаписи ();
Менеджер.ОбъектID =
Строка (Источник.Ссылка.УникальныйИдентификатор ());
Менеджер.ОбъектПредставление = Строка (Источник);
Менеджер.ТипОбъекта = Источник.Метаданные ().Имя;
Менеджер.ДатаСобытия = ТекущаяДата ();
Менеджер.ТипСобытия =
Перечисления.ТипыСобытийСинхронизации.Удаление;
Менеджер.Состояние =
Перечисления.СтатусыСинхронизации.В_Очереди;
Менеджер.Записать ();
КонецПроцедуры

```

Разработанный событийный механизм позволяет полностью исключить ручной контроль за актуальностью данных. Подписки на события работают автономно и гарантируют, что любое изменение в документах магазина строительных материалов мгновенно попадет в очередь на отправку.

3.1.3 Программная реализация алгоритмов обмена в системе-источнике

Для технической реализации системы в конфигурации создали отдельный общий модуль – «МенеджерСинхронизации». Решение вынести логику в

отдельный модуль объясняется просто: всю интеграцию нужно было централизовать. Благодаря этому технические процедуры формирования и отправки данных оказались изолированы от типовых модулей документов. Такой подход заметно упрощает поддержку системы и исключает риск нарушить целостность основной базы данных во время обмена.

В этом модуле собран полный цикл обработки информации на стороне системы-источника. Всё начинается с интерпретации данных, которые зафиксировали триггеры. Заканчивается управлением сетевыми сессиями с внешним веб-порталом.

Все разработанные процедуры и функции общего модуля, обеспечивающие полный цикл синхронизации, систематизированы в таблице 32. С полным кодом реализации можно ознакомиться на GitHub [16].

Таблица 32 – Основные процедуры и функции Модуля синхронизации

Название процедуры/функции	Описание
ЗарегистрироватьИзменение (Источник, ТипСобытия)	Программный триггер. Ловит запись документа и ставит его в очередь на синхронизацию.
ЗарегистрироватьУдаление (Источник, Отказ)	Программный триггер. Фиксирует удаление объекта и ставит задачу на удаление в очередь.
ВыполнитьОбменДанными()	Главный управляющий процесс. Проверяет очередь и запускает цикл отправки данных.
СформироватьПолиструктурныйПакет (ID, Тип, Действие)	Функция-диспетчер. Определяет тип документа и вызывает нужный шаблон для сборки пакета данных
СобратьПакетЗаказПокупателя (ДокОбъект, Данные)	Формирует пакет для документа «Заказ покупателя».
СобратьПакетРеализация (ДокОбъект, Данные)	Формирует пакет для документа «Реализация товаров»
СобратьПакетПоступлениеТоваров (ДокОбъект, Данные)	Формирует пакет для документ «Поступление товаров»
СобратьПакетПоступлениеДенег (ДокОбъект, Данные)	Формирует пакет для документа «Поступление денег»
СобратьПакетРасходДенег (ДокОбъект, Данные)	Формирует пакет для документа «Расход денег»
ФорматироватьЗначениеПолиструктуры (Значение)	Обработчик: очищает даты от лишнего времени и сохраняет типы данных для сайта.

Продолжение таблицы 32

Название процедуры/функции	Описание
ОтправитьПакетНаСервер (УзелСсылка, Пакет, Ответ, МетодHTTP, ДопПуть)	Модуль связи: отправляет данные и сам обновляет ключ (JWT), если связь оборвалась или ключ устарел.
ВыполнитьСетевойЗапрос (URL, Токен, Пакет, МетодHTTP, ДопПуть)	Техническая функция для прямой передачи данных на веб-портал по протоколу HTTP.
ОбновитьJWTТокен (Узел)	Автоматически заходит на сайт под логином/паролем 1С для получения нового ключа доступа.
ПарситьСтрокуВОбъект (Текст)	Вспомогательный алгоритм для преобразования текстовых записей 1С в структурированные данные.

При программной реализации модуля отдельное внимание было уделено обработке полиструктурированной информации. Состав реквизитов у документов продаж, закупок и финансовых транзакций сильно различается. Поэтому внутри модуля разработали гибкие алгоритмы «сборки» данных. Центральный узел этой системы универсальная функция «СформироватьПолиструктурныйПакет». Она работает как архитектурный диспетчер.

Для этого механизма были задействованы несколько специализированных типов объектов платформы «1С:Предприятие 8.3»:

- тип «Структура» используется как базовый контейнер для хранения реквизитов с фиксированным набором полей;
- тип «Массив» применяется для передачи иерархических списков (табличных частей документов) и обеспечивает правильную вложенность данных;
- тип «Соответствие» является ключевым инструментом для реализации полиструктурированности, позволяет использовать в качестве ключей произвольные строки это даёт возможность формировать набор свойств динамически, без жёсткой привязки к именам полей.

Для финальной сериализации подготовленных данных в текстовую строку формата JSON используется системный модуль «ЗаписьJSON».

Сначала формируется унифицированный блок `meta_info`. Независимо от типа документа, система в первую очередь генерирует технический заголовок. В нём содержатся GUID объекта, время события и тип операции. Это стандартизирует входящие потоки данных. Принимающая сторона может мгновенно классифицировать пакет ещё до того, как начнётся его глубокая обработка.

После того как метаданные заполнены, функция-диспетчер анализирует тип объекта и вызывает одну из специализированных процедур-шаблонов:

- «СобратьПакетЗаказПокупателя» объединяет данные заказа с вариативными характеристиками товаров;
- «СобратьПакетРеализация» упаковывает сложную иерархию логистического контура;
- «СобратьПакетПоступлениеТоваров» формирует спецификации закупок с параметрами качества;
- «СобратьПакетПоступлениеДенег» детализирует аналитику каждой строки платежа;
- «СобратьПакетРасходДенег» – реализует условную логику полей в зависимости от вида финансовой операции.

Чтобы подробнее разобраться, как работает механизм, был разобрана процедура «СобратьПакетЗаказПокупателя». Алгоритм выполняет вложенный цикл. При обходе каждой строки товара система обращается к связанной таблице характеристик. С помощью объекта «Соответствие» программа на лету сопоставляет технические ключи связи и собирает разрозненные свойства в один вложенный JSON-объект внутри каждой позиции товара. Ниже на листинге 3 и в Приложении Б приведён фрагмент программного кода, который реализует описанную логику диспетчеризации и детальный шаблон сборки данных для заказа покупателя:

Листинг 3 – Реализация шаблона сборки данных «Заказа покупателя» в 1С

```
Процедура СобратьПакетЗаказПокупателя (ДокОбъект, Данные)
    Данные.Вставить ("number", ДокОбъект.Номер);
```

```

Данные.Вставить ("date", Формат (ДокОбъект.Дата, "ДФ=yyyy-MM-
ddTHH:mm:ss")) ;
Данные.Вставить ("client", Строка (ДокОбъект.Клиент)) ;
Данные.Вставить ("manager", Строка (ДокОбъект.Менеджер)) ;
Данные.Вставить ("warehouse", Строка (ДокОбъект.Склад)) ;
Данные.Вставить ("status", Строка (ДокОбъект.Статус)) ;
Данные.Вставить ("payment", Строка (ДокОбъект.СтатусОплаты)) ;
Данные.Вставить ("total", ДокОбъект.СуммаЗаказа);
Данные.Вставить ("needs_delivery", ДокОбъект.НужнаДоставка);
МассивТоваров = Новый Массив;
Для Каждого СтрТ Из ДокОбъект.Товары Цикл
    Товар = Новый Структура ("name, qty, unit, price, total",
Строка (СтрТ.Номенклатура), СтрТ.Количество,
Строка (СтрТ.ЕдиницаИзмерения), СтрТ.Цена, СтрТ.Сумма);
    Свойства = Новый Соответствие;
    Для Каждого СтрСвойство Из ДокОбъект.СвойстваСтрок Цикл
        Если СтрСвойство.КлючСвязи = СтрТ.КлючСвязи Тогда
            Свойства.Вставить (Строка (СтрСвойство.Свойство),
ФорматироватьЗначениеПолиструктуры (СтрСвойство.Значение)) ;
        КонецЕсли;
    КонецЦикла;
    Товар.Вставить ("props", Свойства);
    МассивТоваров.Добавить (Товар);
КонецЦикла;
Данные.Вставить ("items", МассивТоваров);
КонецПроцедуры

```

В результате работы данного алгоритма формируется унифицированный информационный пакет. Ниже на рисунке 33 представлен пример итогового полиструктурированного JSON-объекта, который передается на веб-портал.

```

{
  "meta_info": {
    "type": "ЗаказПокупателя",
    "id": "1f334e1f-5cc7-11f1-bac2-c4170e60682e",
  },
  "number": "000000022",
  "date": "2026-05-31T13:21:43",
  "client": "ИП Волков С.П.",
  "manager": "Иванова Елена Дмитриевна",
  "warehouse": "Основной склад",
  "status": "К отгрузке",
  "payment": "Не оплачен",
  "total": 56250.00,
  "needs_delivery": true,
  "items": [
    {
      "name": "Кирпич облицовочный",
      "qty": 1500.00,
      "unit": "шт",
      "price": 25.00,
      "total": 37500.00,
      "attributes": {
        "Марка прочности": "M150",
        "Зона выгрузки объекта": "Сектор Б (Склад)"
      }
    },
    {
      "name": "Цемент М-500 (50 кг)",
      "qty": 50.00,
      "unit": "мешок",
      "price": 375.00,
      "total": 18750.00,
      "attributes": {
        "Тип упаковки": "На поддонах",
        "Класс опасности груза": "Класс А (очень опасный груз)"
      }
    }
  ]
}

```

Рисунок 33 – JSON-объекта документа «Заказ покупателя»

Данный рисунок является ключевой иллюстрацией практической реализации полиструктурированности данных в разработанном механизме синхронизации.

3.1.4 Реализация режима, близкого к реальному времени

Чтобы механизм работал полностью автономно и соблюдал режим, близкий к реальному времени в системе 1С реализован автоматический запуск процедур обмена. Для этого использовали объект метаданных «Регламентное задание».. Настройки этого компонента показаны на рисунке 34.

Регламентное задание автоматически вызывает экспортную процедуру МенеджерСинхронизации.ВыполнитьОбменДанными(). Так исключается риск пропуска событий, и данные регулярно передаются на веб-портал мониторинга.

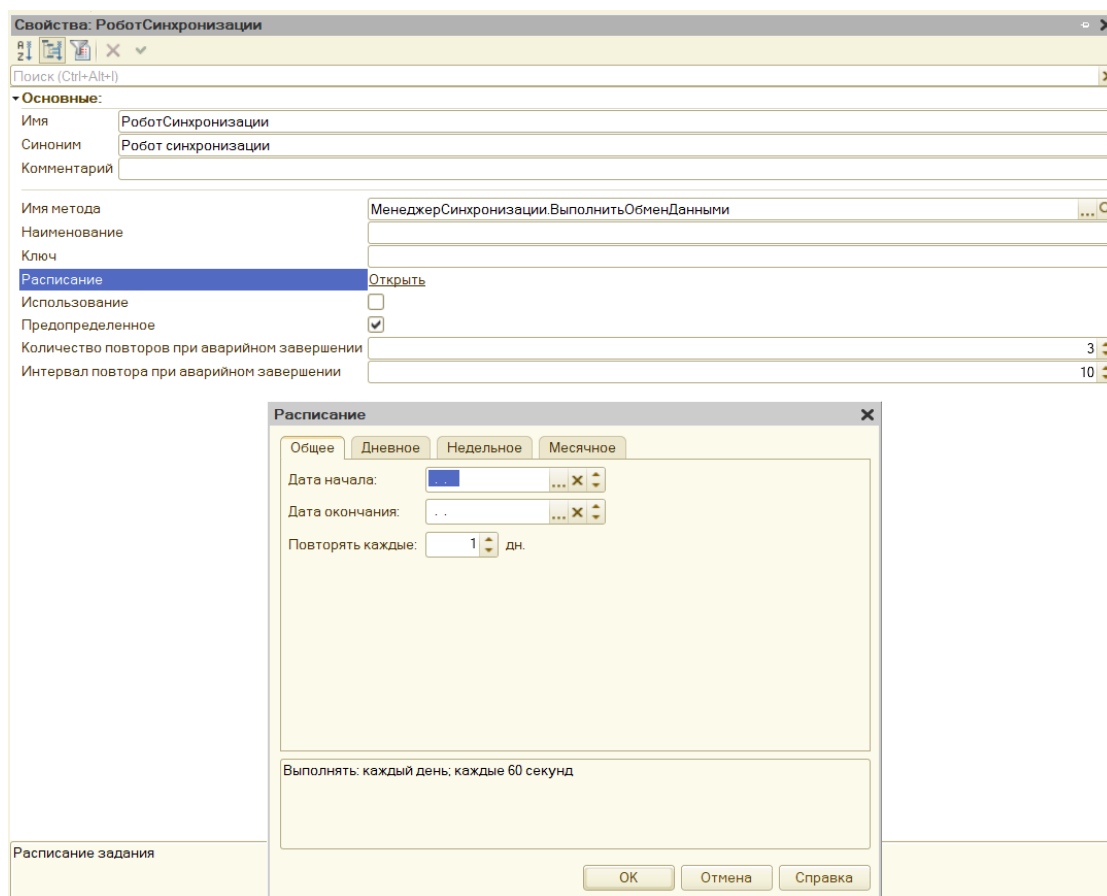


Рисунок 34 – Настройка расписания и параметров регламентного задания

Для регламентного задания «РоботСинхронизации» был установлен интервал запуска 60 секунд. В проекте интервал повтора установили в 60 секунд.

Такой шаг выбрали не случайно – он жёстко ограничивает задержку при передаче данных. Получается, что любое событие из очереди обработают не позже чем через минуту после его появления. Это и есть верхняя граница ожидания. Показатели на портале остаются свежими, а нагрузка на файловую базу 1С не выходит за разумные пределы.

3.2 Программная реализация принимающей стороны

3.2.1 Реализация серверной части

Серверная часть веб-портала мониторинга реализована на языке Java с использованием фреймворка Spring Boot. Архитектура построена по принципу многослойности: выделены уровни:

- контроллеров;
- сервисов;
- репозиторий;
- моделей данных.

Такой подход позволяет разделить ответственность между компонентами, упрощает тестирование и сопровождение кода. С полным кодом можно ознакомиться на GitHub [16].

Контроллеры являются точкой входа для всех внешних запросов. В системе реализованы два контроллера. AuthController отвечает за аутентификацию системы-источника. SyncController принимает данные от 1С. Спецификация программных интерфейсов и маршрутов обработки данных приведена в таблице 21.

Таблица 21 – Спецификация программных интерфейсов и маршрутов обработки данных

Контроллер	Метод	Маршрут (URI)	Функциональное назначение
AuthController	POST	/api/auth/login	Аутентификация системы-источника

Продолжение таблицы 21

Контроллер	Метод	Маршрут (URI)	Функциональное назначение
SyncController	POST	/api/sync/{type}	Прием и первичная регистрация новых полиструктурных объектов данных, поступающих из системы 1С.
SyncController	PUT	/api/sync/{type}/{id}	Обновление существующих записей в базе данных портала на основе присланного идентификатора
DataViewController	DELETE	/api/sync/{type}/{id}	Инициация процедуры удаления объекта из системы мониторинга при его аннулировании в учетном контуре предприятия.
DataViewController	GET	/	Визуализация оперативной панели мониторинга «Заказы покупателей» с детализацией состава.
DataViewController	GET	/incoming-goods	Формирование интерфейса мониторинга поступления товаров и параметров их качества.
DataViewController	GET	/invoices	Отображение данных по отгрузкам (реализациям) и вложенной логистической информации.
DataViewController	GET	/payments-in	Контроль и визуализация входящих финансовых потоков по заказам покупателей.

Окончание таблицы 21

Контроллер	Метод	Маршрут (URI)	Функциональное назначение
SyncController	GET	/payments-out	Мониторинг исходящих денежных средств

Сервисы содержат основную бизнес-логику веб-портала. В системе реализованы три сервиса:

- AuthService отвечает за аутентификацию и генерацию токенов.
- SyncService выполняет центральную функцию приём, разбор и сохранение данных из 1С, а также их удаление.
- DataViewService предоставляет данные для отображения на страницах.

Спецификация сервисов представлена в таблице 22.

Таблица 22 – Спецификация сервисов

Название класса	Метод	Назначение
AuthService	authenticateAndGenerateToken(username, password)	Проверка логина и пароля в таблице и генерация JWT-токена
	validateToken(authHeader)	Проверка подлинности JWT-токена из заголовка Authorization
SyncService	processIncomingJson(rawJson)	Главный диспетчер. Определяет тип документа и вызывает соответствующий метод сохранения. Реализация приведена в Приложении В
	executeDelete(type, guid)	Удаление документа и всех его строк по типу и уникальному идентификатору. Реализация приведена в Приложении В
	saveOrder(JsonNode node)	Сохранение заказа покупателя. Реализация приведена в Приложении В

Продолжение таблицы 22

Название класса	Метод	Назначение
	saveInvoice(JsonNode node)	Сохранение отгрузки
	savePaymentIn(JsonNode node)	Сохранение входящего платежа
	savePaymentOut(JsonNode node)	Сохранение исходящего платежа
DataViewService	getAllOrders()	Получение всех заказов, сортировка по дате
	getAllIncomingGoods()	Получение всех поступлений, сортировка по дате
	getAllInvoices()	Получение всех накладных (отгрузок), сортировка по дате
	getAllPaymentsIn()	Получение всех входящих платежей, сортировка по дате

Репозитории обеспечивают доступ к базе данных. Все они наследуют интерфейс `JpaRepository`, поэтому каждый автоматически получает стандартные методы: `save()`, `findById()`, `findAll()`, `deleteById()`, `existsById()`. Эти методы не требуют дополнительной реализации.

Реализованные репозитории со специфическими методами приведены в таблице 23.

Таблица 23 – Спецификация интерфейсов репозиторий

Интерфейс репозитория	Работает с моделью	Специфический метод	Интерфейс репозитория
<code>ApiUserRepository</code>	<code>ApiUser</code>	<code>findByUsernameAndPassword(username, password)</code>	Поиск учётной записи по логину и паролю для аутентификации
<code>OrderItemRepository</code>	<code>SalesOrderItem</code>	<code>deleteByOrderId(String id)</code>	Удаление всех позиций заказа
<code>SalesInvoiceItem Repository</code>	<code>SalesInvoice Item</code>	<code>deleteByInvoiceId(String id)</code>	Удаление всех позиций реализации

Продолжение таблицы 23

Интерфейс репозитория	Работает с моделью	Специфический метод	Интерфейс репозитория
IncomingGoodsItem Repository	IncomingGoodsItem	deleteByParentDocId(String id)	Удаление всех строк поступления товаров
PaymentInItem Repository	PaymentInItem	deleteByPaymentId(String id)	Удаление всех строк входящего платежа

Модели представляют собой JPA-сущности. Каждая модель соответствует одной таблице в базе данных Microsoft SQL Server и одному типу документа из 1С. Спецификации реализованных моделей приведены в таблице 24.

Таблица 24 – Спецификация моделей

Название класса	Назначение	Полиструктурные поля
ApiUser	Учётные записи внешних систем для аутентификации (логин, пароль)	
SalesOrder	Шапка документа «Заказ покупателя»	
SalesOrderItem	Товарные позиции заказа	custom_properties_json характеристики товара
SalesInvoice	Шапка документа «Реализация товаров»	logistics_json – данные о водителе, транспорте и приоритете доставки
SalesInvoiceItem	Позиции реализации	
IncomingGoods	Шапка документа «Поступление товаров»	
IncomingGoodsItem	Позиции поступления товаров	technical_params_json технические параметры качества продукции
PaymentIn	Шапка документа «Поступление денежных средств»	
PaymentInItem	Детализация входящего платежа по заказам	payment_attributes_json атрибуты строки платежа (

Продолжение таблицы 24

Название класса	Назначение	Полиструктурные поля
PaymentOut	Документ «Расход денежных средств»	details_json детали операции в зависимости от вида расхода

Таким образом реализована серверная часть веб-портала.

3.2.2 Реализация механизмов обработки полиструктурированных данных

Для обработки входящих JSON-пакетов с переменным набором атрибутов в серверной части веб-портала использована библиотека Jackson в режиме Tree Model [17]. Такой подход позволяет разбирать данные без жёстко заданных классов любое изменение состава полей в 1С не потребует перекомпиляции приложения.

Два основных класса библиотеки обеспечивают динамический парсинг:

- ObjectMapper, преобразует входную JSON-строку в древовидную структуру;
- JsonNode , универсальный контейнер для любого узла, с его помощью алгоритм обходит пакет, проверяет наличие полей и извлекает значения на лету

Алгоритм на примере документа «Заказ покупателя».

Из дерева JsonNode сначала извлекается мета-информация (тип документа и идентификатор). Затем данные делятся на статические и динамические. Статические реквизиты записываются в поля объектной модели типизированными методами. Динамические (характеристики товаров) не разбираются, а сохраняются в исходном виде в поле БД типа nvarchar (max).Фрагмент реализации приведён в листинге 5.

Листинг 5 – Динамический разбор для документа «Заказ покупателя»

```
@Service
public class SyncService {
    private final ObjectMapper mapper = new ObjectMapper();
```

```

@Transactional
public void processIncomingJson(String rawJson) throws
Exception {
    JsonNode root = mapper.readTree(rawJson);
    String type =
root.path("meta_info").path("type").asText();
    if ("ЗаказПоккупателя".equals(type)) saveOrder(root);
}

private void saveOrder(JsonNode node) {
    String guid = node.path("meta_info").path("id").asText();
    orderItemRepo.deleteByOrderId(guid);
    SalesOrder order = new SalesOrder();
    order.setId(guid);
    order.setNumber(node.path("number").asText());
    order.setClient(node.path("client").asText());
    order.setTotalAmount(node.path("total").asDouble());
    orderRepo.save(order);
    for (JsonNode i : node.path("items")) {
        SalesOrderItem item = new SalesOrderItem();
        item.setOrder(order);
        item.setProductName(i.path("name").asText());

item.setCustomPropertiesJson(i.path("props").toString());
        orderItemRepo.save(item);
    }
}
}

```

Таким образом, использование `JsonNode` делает сервер независимым от изменений структуры данных в 1С. Новые реквизиты обрабатываются автоматически без правок кода, что обеспечивает масштабируемость и устойчивую работу синхронизации в режиме, близком к реальному времени.

3.2.3 Реализация клиентской части

Клиентская часть сделана в виде оперативной панели мониторинга. Её главная задача показывать результаты синхронизации наглядно и понятно.

При стандартной работе пользователь видит в браузере сырые JSON-строки с фигурными скобками, кавычками и служебными символами. Это нечитаемо.

Для отображения полиструктурных данных (JSON-строк из SQL) была разработана логика динамической отрисовки данных. На клиенте был реализован модуль на JavaScript, выполняющий разбор. После преобразования

строки в объект JavaScript алгоритм обходил полученный объект, и каждая пара «ключ – значение» превращалась в элемент маркированного списка. Так интерфейс был сделан универсальным. Фрагмент кода алгоритма динамической визуализации приведён в листинге 6. Полный код приведен в Приложении Г.

Листинг 6 – Реализация разбора и визуализации данных на стороне клиента

```
document.addEventListener("DOMContentLoaded", function() {
    document.querySelectorAll('.js-json-formatter').forEach(function(element) {
        let rawData = element.textContent.trim();
        try {
            let jsonData = JSON.parse(rawData);
            let resultHtml = '<ul class="dynamic-list">`;
            for (let key in jsonData) {
                resultHtml += `<li><b>${key}</b>`;
            }
            element.innerHTML = resultHtml + '</ul>';
        } catch (error) {
            element.innerHTML = '<span class="status-empty">данные отсутствуют</span>';
        }
    });
});
```

В итоговой программной реализации интерфейса были достигнуты высокая информативность и масштабируемость. Портал был подготовлен к корректному отображению любых новых типов аналитик, которые в будущем могут быть добавлены в учётную систему 1С.

4 Контроль качества механизма синхронизации

4.1 Функциональное тестирование

4.1.1 Классы эквивалентности

Метод деления на классы эквивалентности относится к техникам тест-дизайна [18]. Он часто применяется при функциональном тестировании. Смысл подхода в следующем: всё множество возможных входных данных разбивается на группы (классы). Внутри каждой группы поведение системы предполагается одинаковым.

В рамках верификации разработанного механизма синхронизации основным объектом тестирования выступает поле «URL сервера», расположенное в справочнике «Узлы синхронизации» системы 1С. Данный реквизит является критически важным элементом настройки, так как именно на его основе программный модуль формирует параметры низкоуровневого сетевого соединения и определяет адрес принимающего веб-портала. Ошибка в заполнении данного поля (например, использование неверного протокола или наличие невидимых символов) приведет к невозможности установления связи и срыву регламента синхронизации в режиме близком к реальному времени.

К значению поля «URL сервера» в рамках проекта предъявляются следующие функциональные требования:

- наличие протокола: строка должна начинаться строго с префикса `http://` или `https://` ;
- символьный состав: допустимо использование латинских букв, цифр, точек (.), двоеточий (:) и слэшей (/);
- ограничения: в адресе категорически запрещены пробелы и использование кириллических символов;
- формат порта: если в адресе указан порт, он должен быть отделен двоеточием и состоять только из цифр.

Для минимизации количества тестовых сценариев без потери качества проверки были выделены допустимые (правильные) и недопустимые (неправильные) классы эквивалентности, представленные в таблице 25.

Таблица 25 – Классы эквивалентности для поля «URL сервера»

Характеристика	Правильные КЭ	Неправильные КЭ
Сетевой протокол	Наличие http:// или https://(1)	Протокол отсутствует или указан неверно (ftp://и др.) (2)
Символьный состав	Латиница, цифры, точки, двоеточия (3)	Использование кириллицы или спецсимволов (!, @, #) (4)
Чистота строки	Строка не содержит пробелов (5)	Внутри адреса присутствуют пробелы (6)
Формат порта	Числовое значение после двоеточия (7)	Буквенное значение или спецсимволы после двоеточия (8)

На основе сформированных классов были разработаны тест-кейсы, результаты выполнения которых представлены в таблице 26.

Таблица 26 – Тест-кейсы для проверки поля «URL сервера»

Номер теста	Классы Эквивалентности	Входные данные	Выходные данные
1	1, 3, 5	http://127.0.0.1	Успешно
2	1, 3, 5, 7	http://localhost:8081	Успешно
3	2, 3, 5	127.0.0.1:80	Ошибка: Протокол обязателен
4	2, 3, 5	ftp://mysite.ru	Ошибка: Неверный протокол
5	1, 4, 5	http://мой-портал.рф	Ошибка: Использование кириллицы
6	1, 3, 6	http://127.0. 0.1	Ошибка: Наличие пробела
7	1, 4, 5	http://site#api.ru	Ошибка: Недопустимый символ «#»
8	1, 3, 8	http://localhost:port	Ошибка: Номер порта должен быть числом
9	1, 3, 5	https://cloud-server.com/api	Успешно

Тесты закрывают все варианты проверки адреса. Это значит, что связь настроится правильно, а неподходящие данные не пройдут. Разработанные тест-кейсы позволяют в полном объеме проверить корректность работы алгоритма

распознавания адреса внешней системы. Это гарантирует стабильное установление связи и исключает возможность передачи некорректных данных.

4.1.2 Граничные значения

Метод граничных значений является одной из фундаментальных техник верификации программной логики. Он направлен на проверку поведения системы в точках перехода между допустимыми состояниями [19]. В рамках проектирования отказоустойчивых механизмов синхронизации особую роль играет алгоритм обработки исключительных ситуаций. Поскольку сетевое взаимодействие подвержено влиянию внешних факторов, система должна обладать интеллектуальной стратегией повторных попыток (Retry Policy).

Ключевым параметром данного алгоритма является лимит количества попыток отправки пакета. Граничное тестирование этой величины необходимо для подтверждения того, что система не войдет в бесконечный цикл при возникновении критической ошибки и корректно прекратит автоматические действия при достижении установленного предела. Это гарантирует сохранность вычислительных ресурсов сервера.. Согласно заложенной в программный модуль 1С логике, максимальное количество попыток равно 5.

Для выделения границ был проведен анализ состояний счетчика попыток. Граничные условия тестирования представлены в таблицах 27 .

Таблица 27 – Граничные условия счетчика попыток

Границы	Условия	Границы
Логические	Совершение первой попытки после возникновения сбоя (Л1)	Логические
Технологические	Достижение предельного значения счетчика попыток (5 попыток) (Т1)	Технологические
Физические	Попытка инициации обмена при нулевом значении счетчика (Ф1)	Физические
Произвольные	Поведение системы при превышении лимита (6-я попытка и далее) (П1)	Произвольные

Тест-кейсы граничных условий представлены в таблице 28.

Таблица 28 – Тест-кейсы граничных условий

№	Граница	Входное состояние	Ожидаемый результат
1	Ф1	Количество попыток = 0	Система выполняет первичную отправку пакета в штатном режиме.
2	Л1	Количество попыток = 1	После первого сбоя система рассчитывает время паузы (60 сек) и сохраняет статус «Ошибка».
3	Т1	Количество попыток = 4	Предпоследняя попытка. Система планирует финальный автоматический сеанс связи.
4	Т1	Количество попыток = 5	Лимит достигнут. После неудачного сеанса статус меняется на «Лимит исчерпан», дата повтора обнуляется.
5	П1	Количество попыток = 6	Система полностью игнорирует запись в очереди и не инициирует сеансы связи.

Такой подход защищает систему от избыточной нагрузки при критических сбоях.

4.2 Сквозное тестирование

Сквозное тестирование проверяет полный цикл работы системы [20]. В данном случае, от создания документа в 1С до его отображения на веб-портале. Такой подход позволяет убедиться, что все компоненты механизма синхронизации работают вместе без сбоев. Цель тестирования – проверить, что полиструктурированные данные из учётной системы 1С корректно передаются на веб-портал и отображаются в читаемом виде. В ходе тестирования для каждого документа были выполнены операции создания, обновления. Особое внимание уделялось полиструктурированным полям: характеристикам товаров, техническим параметрам, логистическим данным, свойствам платежей и деталям расхода. Результаты сквозного тестирования представлены в таблице 29 и в приложении Д.

Таблица 29 – Результаты сквозного тестирования

№ теста	Документ	Описание теста	Ожидаемый результат	Фактический результат
1	Заказ покупателя	Создание документа с заполнением для каждого товара своих свойств	На портале в деталях документа отображается список свойств для каждой позиции	Успешно (рисунок Д.1)
2	Заказ покупателя	Обновление документа с изменением состава свойств у позиции	Список свойств на портале обновляется в соответствии с изменениями	Успешно (рисунок Д.2)
3	Поступление товаров	Создание документа с заполнением технических параметров качества для каждой позиции	В разделе «Закупки» отображаются технические параметры в читаемом виде	Успешно (рисунок Д.3)
4	Поступление товаров	Обновление документа с изменением технических параметров	На портале отображаются новые параметры, старые заменены	Успешно (рисунок Д.4)
5	Реализация товаров	Создание документа без заполнения данных о доставке	На портале отображается отгрузка, данные о доставке отсутствуют	Успешно (рисунок Д.5)
6	Реализация товаров	Обновление документа с добавлением данных о доставке (реквизиты ответственного за доставку, логистика, приоритет)приоритета	На портале отображаются карточки водителя, транспорта и приоритет	Успешно (рисунок Д.6)
7	Поступление денежных средств	Создание документа с заполнением свойств платежа (форма оплаты, признак аванса, номер чека)	В разделе «Приход денег» отображаются свойства платежа в виде информационных блоков	Успешно (рисунок Д.7)

Продолжение таблицы 29

№ теста	Документ	Описание теста	Ожидаемый результат	Фактический результат
8	Поступление денежных средств	Обновление документа с изменением свойств	Новые свойства отображаются на портале	Успешно (рисунок Д.8)
9	Расход денежных средств	Создание документа с заполнением деталей операции (месяц выплаты, вид выплаты, документ-основание)	В разделе «Расход денег» отображаются детали операции в читаемом виде	Успешно (рисунок Д.9)
10	Расход денежных средств	Обновление документа с изменением деталей	На портале отображаются актуальные детали	Успешно (рисунок Д.10)

После выполнения всех сценариев была проверена операция удаления документов. Для каждого типа документа удаление прошло успешно – записи исчезали из интерфейса портала в течение установленного интервала синхронизации. Скриншоты удаления не приводятся ввиду очевидности результата. Все проверенные сценарии завершились успешно. Система корректно обрабатывает создание, обновление и удаление документов. Полиструктурированные данные передаются на портал и отображаются в читаемом виде. Ошибок при выполнении тестов не зафиксировано. Разработанный механизм синхронизации работает в штатном режиме и готов к эксплуатации.

4.2 Нагрузочное тестирование

Помимо проверки правильности выполнения функций, критически важным этапом контроля качества является нефункциональное тестирование. Для механизмов синхронизации, работающих в режиме времени, близком к

реальному, ключевым показателем является производительность. Система должна, но и справляться с интенсивными потоками информации, характерными для периодов высокой операционной активности предприятия (например, при массовой отгрузке заказов со склада).

Целью данного этапа является определение пропускной способности веб-портала и выявление времени отклика серверной части при различной интенсивности входящих запросов. В ходе испытаний проверяется стабильность работы алгоритмов десериализации JSON и скорость физической записи данных в СУБД MS SQL Server.

Для имитации нагрузки был использован специализированный инструмент Apache JMeter. Сценарий тестирования заключался в циклической отправке HTTP-запросов методом POST на программный интерфейс (API) портала. В качестве полезной нагрузки использовался эталонный полиструктурированный пакет данных «Заказ покупателя», содержащий иерархический список товаров и динамический набор характеристик. Результаты нагрузочного тестирования представлены в таблице 30.

Таблица 30 – Результаты нагрузочного тестирования

Параметр теста	Тест №1 (100 запр.)	Тест №2 (1000 запр.)	Тест №3 (5000 запр.)
Среднее время (Average), мс	3 260	2 027	3 176
Медианное время, мс	3 532	6	2
90% Line (Перцентиль), мс	4 449	6 671	13 658
Минимальное время, мс	1 601	1	2
Максимальное время, мс	4 613	8 645	18 003
Доля ошибок (Error %)	0.00%	0.00%	0.00%

Ниже приведен детальный анализ каждого этапа тестирования на основе графических отчетов системы JMeter.

Результаты тестирования на 100 запросов представлены на рисунке 35.

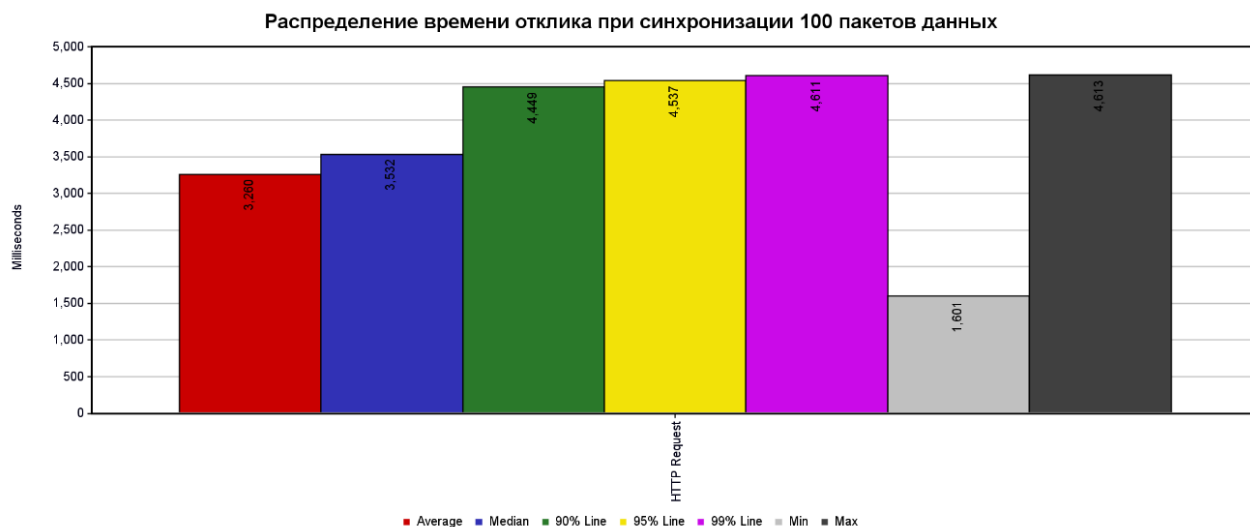


Рисунок 35 – Распределение времени отклика при синхронизации 100 пакетов данных

На графике показаны результаты теста при обычной нагрузке в 100 запросов. Это имитация спокойного рабочего дня предприятия. Среднее время обработки одного пакета данных составляет 3260 мс. Видно, что сервер работает стабильно, разрыв между быстрым и медленным ответом небольшой. Это значит, что система легко справляется с таким объемом данных и быстро записывает их в базу.

Результаты тестирования на 1000 запросов представлены на рисунке 36.



Рисунок 36 – Результаты теста на 1000 запросов

На рисунке 36 показано, как система ведет себя при более высокой нагрузке в 1000 запросов. Среднее время обработки одного заказа даже немного снизилось до 2027 мс, так как сервер «прогрелся» и стал работать быстрее. При этом время обработки самых долгих запросов выросло до 6,6 секунд. Это говорит о том, что сервер начал выстраивать очередь из задач, но всё равно успешно справился со всеми запросами без единой ошибки.

Результаты тестирования на 5000 запросов представлены на рисунке 37.

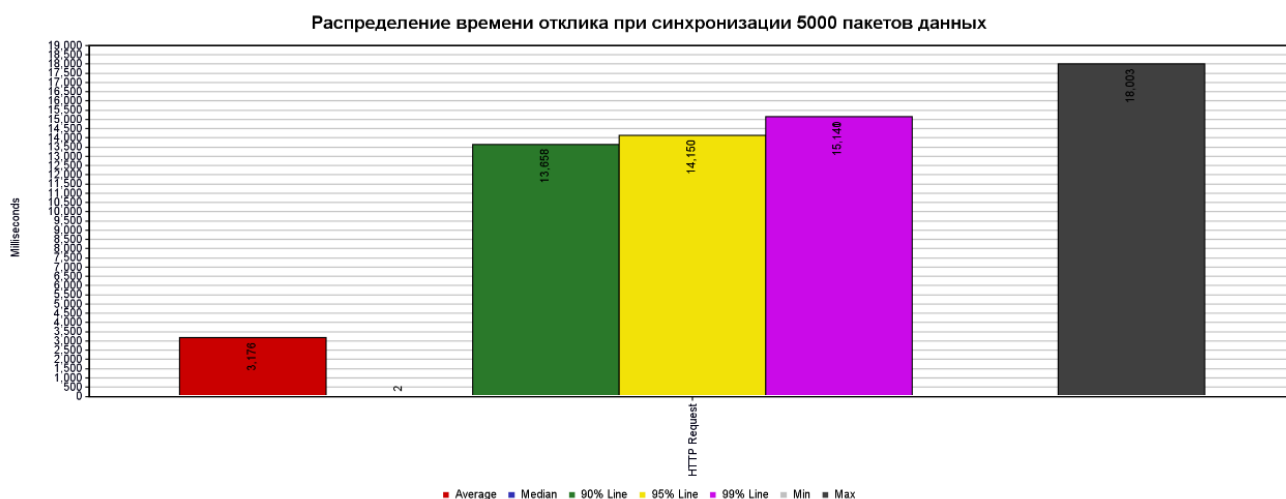


Рисунок 37 – Результаты теста на 5000 запросов

Тест на 5000 запросов – это проверка системы на прочность. Здесь нагрузка на компьютер была максимальной. Главный результат этого теста в том, что даже при такой огромной нагрузке количество ошибок осталось равным 0%. Все 5000 пакетов данных были корректно приняты, разобраны и сохранены в базу данных SQL.

Проведенные тесты подтвердили, что созданный механизм синхронизации работает надежно. Система без проблем выдерживает поток до 5000 сообщений за один цикл. Средняя скорость работы в 2 – 4 секунды полностью позволяет обновлять информацию на сайте быстро, в режиме реального времени. Выбранные технологии (Java и SQL Server) обеспечивают нужный запас прочности, и сайт не «зависнет» даже в моменты очень активных продаж или отгрузок на предприятии.

4.3 Метрики программного кода

Для объективной оценки качества, сложности и объема разработанного программного обеспечения применяются методы статического анализа кода – метрики программного обеспечения. В рамках данного проекта оценка проводилась по двум наиболее признанным методикам: метрике SLOC и метрике Холстеда. Объектом анализа выступила ключевая процедура системы-источника, отвечающая за событийную регистрацию объектов.

4.3.1 Метрики SLOC

Количество строк кода (SLOC) позволяет определить физический объем программного модуля [21]. Данная метрика важна для оценки трудоемкости поддержки кода и его читаемости. Программная реализация анализируемой процедуры приведена в листинге 7.

Листинг 7 – Процедура «ЗарегистрироватьИзменение»

```
Процедура ЗарегистрироватьИзменение (Источник, ТипСобытия) Экспорт
    Менеджер =
    РегистрыСведений.ОчередьСинхронизации.СоздатьМенеджерЗаписи ();
    Менеджер.ОбъектID =
    Строка (Источник.Ссылка.УникальныйИдентификатор ());
    Менеджер.ТипОбъекта = Источник.Метаданные ().Имя;
    Менеджер.ДатаСобытия = ТекущаяДата ();
    Если Источник.ПометкаУдаления Тогда
        Менеджер.ТипСобытия =
    Перечисления.ТипыСобытийСинхронизации.Удаление;
    Иначе
        // Проверка на существование успешной отправки в очереди
        Запрос = Новый Запрос;
        Запрос.Текст = "ВЫБРАТЬ ПЕРВЫЕ 1 ОбъектID ИЗ
    РегистрыСведений.ОчередьСинхронизации
        | ГДЕ ОбъектID = &ID И Состояние =
    ЗНАЧЕНИЕ (Перечисление.СтатусыСинхронизации.Отправлено) ";
        Запрос.УстановитьПараметр ("ID", Менеджер.ОбъектID);
        Если Запрос.Выполнить ().Пустой () Тогда
            Менеджер.ТипСобытия =
    Перечисления.ТипыСобытийСинхронизации.Создание;
        Иначе
            Менеджер.ТипСобытия =
    Перечисления.ТипыСобытийСинхронизации.Обновление;
        КонецЕсли;
    КонецЕсли;
```

Менеджер.Состояние =
 Перечисления.СтатусыСинхронизации.В_Очереди;
 Менеджер.Записать ();
 КонецПроцедуры

Количественные метрики процедуры приведены в таблице 31.

Таблица 31 – Количественные метрики программного кода (SLOC)

Характеристика	Значение	Пояснение
SLOC (Physical)	25	Общее количество физических строк в исследуемом коде, включая пустые строки и комментарии.
SLOC (Logical)	19	Количество исполняемых команд
Количество пустых строк	1	Используются для визуального разделения логических блоков кода (между объявлением переменных и началом условий).
Количество комментариев	1	Строки, содержащие текстовые пояснения к логике работы событийного триггера.
Процент комментариев	4%	Показатель задокументированности кода в рамках анализируемой процедуры.
Среднее число строк для функций	25	Суммарный объем кода, приходящийся на одну программную единицу.
Среднее число строк исходного кода	24	Количество строк, содержащих непосредственно программные инструкции (Physical SLOC за вычетом пустых строк).

Анализ количественных показателей по метрике SLOC подтверждает компактность и высокую эффективность программного кода. Процедура не перегружена лишними операторами и имеет четкую структуру, что облегчает её дальнейшую поддержку и модификацию.

4.3.2 Метрики Холстеда

Метрика Холстеда относится к методам анализа сложности программного обеспечения, основанным на подсчете количества уникальных синтаксических элементов в исходном коде. Данная методика позволяет оценить не только объем программы, но и интеллектуальные усилия, необходимые для её разработки и понимания.

Основу метриков Холстеда составляют четыре измеряемые характеристики:

- n1 (NUOprtr), число уникальных операторов программы, включая знаки операций, ключевые слова и вызовы функций;
- n2 (NUOprnd), число уникальных операндов программы, включая переменные и строковые константы;
- N1 (Noprtr), общее число операторов в программе;
- N2 (Noprnd), общее число операндов в программе.

Операторы и операнды приведены в таблицах 32 и 33 соответственно.

Таблица 32 – Операторы процедуры «ЗарегистрироватьИзменение»

Операторы	Число вхождений
=	10
.	12
()	7
Если...Тогда...Иначе...КонецЕсли	2
Новый	1
УстановитьПараметр	1
ЗНАЧЕНИЕ()	1
& (параметр запроса)	1
Итого уникальных (n1)	8
Всего вхождений (N1)	35

Таблица 33 – Операнды процедуры «ЗарегистрироватьИзменение»

Операнды	Число вхождений
Менеджер	8
Источник	2
ТипСобытия	4
Запрос	3
ОбъектID	2
ТипОбъекта	1
ДатаСобытия	1
ПометкаУдаления	1
Текст	1
Состояние	1
РегистрыСведений	2
Перечисления	3
Текст SQL-запроса	1
«ID»	2
Итого уникальных (n2)	14
Всего вхождений (N2)	32

На основе полученных данных выполнен расчет результирующих характеристик сложности.

Расчет длины программы производится по формуле:

$$N=N1+N2=35+32=67,$$

где $N1$ – общее число операторов в программе;

$N2$ – общее число операндов в программе.

Словарь программы рассчитывается по формуле:

$$n=n1+n2=8+14=22,$$

где $n1$ – число уникальных операторов программы;

$n2$ – число уникальных операндов программы.

Объем программы рассчитывается по формуле:

$$V=N \cdot \log_2 n = 67 \times \log_2 22 = 298.82,$$

где N – длина программы;

n – словарь программы.

Сложность программы вычисляется по формуле:

$$D = \frac{n1}{2} \times \frac{N2}{n2} = \frac{8}{2} \times \frac{32}{22} = 9.14,$$

где $n1$ – число уникальных операторов программы;

$N2$ – общее число операндов в программе;

$n2$ – число уникальных операндов программы.

Пояснение ко всем полученным результатам приведено в таблице 34

Таблица 34 – Сводная таблица с характеристиками сложности кода

Характеристика	Результат	Пояснение
Длина программы	67	Процедура компактная, легко читается
Словарь программы	22	Умеренное разнообразие элементов. Код не перегружен
Объем программы	298.82	Низкая трудоёмкость. Программа не требует значительных усилий для понимания
Сложность программы	9.14	Низкая сложность. Код легко поддерживать и модифицировать

Анализ кода показал, что модуль компактен и удобен для поддержки.

Заключение

В рамках выпускной квалификационной работы был спроектирован и реализован механизм синхронизации полиструктурированных данных в режиме, близком к реальному времени. На примере предприятия, осуществляющего оптово-розничную торговлю строительными материалами, была обоснована необходимость перехода от пакетного обмена данными к событийному. В ходе выполнения работы были определены функциональные и нефункциональные требования, а также установлены границы взаимодействия системы с помощью диаграммы вариантов использования.

Процесс разработки был детально спланирован в среде ProjectLibre. На основе декомпозиции проекта на 30 задач была вычислена общая трудоёмкость, составившая 294 рабочих часов. Суммарные затраты на реализацию проекта были определены в размере 323 704 рублей. Календарные сроки выполнения работ были установлены с 8 апреля по 29 мая текущего года.

На этапе проектирования информационного обеспечения были разработаны логические и физические модели баз данных для системы-источника и системы-приемника. Особое внимание было уделено созданию спецификаций сопоставления данных и правил их преобразования из внутренних структур 1С в формат JSON. Динамика и логика работы механизма были зафиксированы с помощью диаграмм последовательности, деятельности и состояний языка UML.

Программная реализация системы включала в себя создание двух основных контуров. В системе-источнике на платформе «1С:Предприятие 8.3» был реализован триггерный механизм захвата изменений через подписки на события и организована асинхронная очередь по паттерну Transactional Outbox. Принимающая сторона была выполнена в виде веб-портала мониторинга на языке Java с использованием фреймворка Spring Boot, а также СУБД Microsoft SQL Server. Клиентская часть была оснащена динамическим парсером на

JavaScript, обеспечивающим визуализацию полиструктурированных данных без жёсткой привязки к макетам страниц.

Для подтверждения качества разработанного решения было проведено функциональное, сквозное и нагрузочное тестирование с применением инструмента Apache JMeter. Эффективность и сопровождаемость программного кода были подтверждены результатами анализа метрик SLOC, Холстеда. Всё это показало, что механизм синхронизации работает стабильно, правильно обрабатывает полиструктурные данные и подходит для использования в реальной работе предприятия.

Список использованных источников

1. Лекция 2. Источники и типы данных (структурированные, неструктурированные) // Farabi University [Электронный ресурс]. URL: https://farabi.university/storage/files/24393343766902655fa519b210296830_Лекция%202.pdf (дата обращения: 11.04.2026).
2. Visual Paradigm // Рувики: интернет-энциклопедия [Электронный ресурс]. URL: https://ru.ruwiki.ru/wiki/Visual_Paradigm (дата обращения: 13.04.2026).
3. «1С:Предприятие 8.3» что это такое и как работают программы // Блог 1АБ [Электронный ресурс]. URL: <https://1ab.ru/blog/> (дата обращения: 11.04.2026).
4. Основы Spring Boot: создаём веб-приложение на Java // Skillbox Media [Электронный ресурс]. URL: <https://skillbox.ru/media/code/zapuskajem-pervoe-vebprilozhenie-na-spring-boot/> (Дата обращения: 12.04.2026).
5. UML – диаграмма вариантов использования (use case diagram) // Хабр [Электронный ресурс]. URL: <https://habr.com/ru/articles/47940/> (Дата обращения: 13.04.2026).
6. Режим времени, приближенный к реальному // Мультиязычный правовой портал [Электронный ресурс]. URL: <https://multilang.pravo.by/ru/term/index/41815> (Дата обращения: 15.04.2026).
7. ProjectLibre // Руниверсалис [Электронный ресурс]. URL: <https://руни.рф/ProjectLibre> (дата обращения: 16.04.2026).
8. Что такое диаграмма Ганта: основные возможности, преимущества и программы // РБК [Электронный ресурс]. URL: <https://companies.rbc.ru/news/RcHHj2PJnw/chto-takoe-diagramma-ganta-osnovnyie-vozmozhnosti-preimuschestva-i-programmyi/> (Дата обращения: 17.04.2026).

9. Технологии интеграции информационных систем. Часть 1. Файловый обмен, общая БД, удалённый вызов процедур // Хабр [Электронный ресурс]. URL: <https://habr.com/ru/articles/> (дата обращения: 20.04.2026).

10. Проектирование интеграций API: принципы, архитектура и лучшие практики // КТ Team Blog [Электронный ресурс]. URL: <https://blog.kteam.ru/> (дата обращения: 23.05.2026).

11. Клеппман, М. Высоконагруженные приложения. Программирование, масштабирование, поддержка / М. Клеппман. – Санкт-Петербург: Питер, 2023. – 640 с.

12. Паттерн Transactional Outbox: от теории до продакшена // Хабр [Электронный ресурс]. URL: <https://habr.com/ru/articles/> (дата обращения: 23.04.2026).

13. Понимание диаграмм деятельности в UML. Полное руководство // Visual Paradigm Guides [Электронный ресурс]. URL: <https://guides.visual-paradigm.com/ru/understanding-activity-diagrams-in-uml-a-comprehensive-guide/> (дата обращения: 25.04.2026).

14. Архитектура приложения: что это, какие есть виды и как проектировать // Mindsw.io [Электронный ресурс]. URL: <https://mindsw.io/cases/chto-takoe-arkhitektura-prilozheniya/> (дата обращения: 25.04.2026).

15. Для чего нужны документы в 1С и как они обеспечивают учёт // Integration-software [Электронный ресурс]. URL: <https://integration-software.ru/blog/dokument-v-1s-8-3-nastroyka-provedenie-i-formirovanie> (дата обращения: 01.05.2026).

16. Репозиторий кода дипломной работы // GITHUB. [Электронный ресурс]. URL: https://github.com/juliafedulova/diplom_sync (дата обращения: 06.05.2026)..

17. Парсинг JSON с помощью Jackson // Хабр [Электронный ресурс]. URL: <https://habr.com/ru/companies/otus/articles/687004/> (дата обращения: 06.05.2026).

18. Тест-дизайн: классы эквивалентности, граничные значения, BDD // Wiki ПМИФи [Электронный ресурс]. URL: <https://wiki.pmifi.ru/disciplines/testing/lectures/design> (дата обращения: 25.05.2026).

19. Тестирование в массы. Часть II. Техники тест-дизайна // Directum Club [Электронный ресурс]. URL: <https://club.directum.ru/post/374562> (дата обращения: 25.05.2026).

20. End-to-end-тестирование: цели и особенности внедрения на IT-проекте // In-Aim [Электронный ресурс]. URL: <https://www.in-aim.ru/blog/end-to-end-testirovanie-tseli-i-osobennosti-vnedreniya-na-it-proekte/> (дата обращения: 26.05.2026).

21. SLOC – количество строк кода: раскрываем скрытые показатели качества // MetriDev [Электронный ресурс]. URL: <https://www.metridev.com/en/metrics/sloc-lines-of-code-unveiling-the-hidden-quality-metrics/> (дата обращения: 28.05.2026).

Приложение А
(справочное)
Глоссарий

Полиструктурированные данные – данные, которые не имеют фиксированного набора полей. Состав атрибутов может меняться в зависимости от типа объекта или конкретной ситуации.

Синхронизация данных – процесс приведения данных в двух или более независимых источниках к одинаковому состоянию.

Режим, близкий к реальному времени (near-real-time) – режим обработки данных, при котором задержка между событием и реакцией системы минимальна, но не равна нулю.

Триггер – программный механизм, который автоматически срабатывает при наступлении определённого события (например, при записи, изменении или удалении данных).

Очередь сообщений – промежуточное хранилище, в которое помещаются данные для последующей обработки. Обеспечивает асинхронный обмен между системами.

Маппинг – процесс сопоставления полей одной структуры данных полям другой структуры (например, из 1С в SQL).

REST API – архитектурный стиль построения веб-сервисов, при котором каждый запрос содержит всю необходимую информацию для его обработки.

JSON (JavaScript Object Notation) – текстовый формат обмена данными, основанный на синтаксисе JavaScript. Широко используется для передачи данных между сервером и клиентом.

Spring Boot – фреймворк для создания веб-приложений на языке Java, упрощающий настройку и развёртывание проектов.

Приложение Б
(обязательное)
Листинг системы-источника 1С

Листинг Б.1 – Процедура ВыполнитьОбменДаннымиПроцедура

ВыполнитьОбменДанными() Экспорт

```

ТекДата = ТекущаяДата();
ВыборкаУзлов = Справочники.УзлыСинхронизации.Выбрать();

Пока ВыборкаУзлов.Следующий() Цикл
    Если НЕ ВыборкаУзлов.Активен Тогда Продолжить; КонецЕсли;

    Запрос = Новый Запрос;
    Запрос.Текст = "ВЫБРАТЬ ПЕРВЫЕ 10 * ИЗ
РегистрСведений.ОчередьСинхронизации
    | ГДЕ Состояние =
ЗНАЧЕНИЕ(Перечисление.СтатусыСинхронизации.В_Очереди)
    | ИЛИ (Состояние =
ЗНАЧЕНИЕ(Перечисление.СтатусыСинхронизации.Ошибка)
    | И ДатаСледующейПопытки <= &ТекДата И
ДатаСледующейПопытки <> ДАТАВРЕМЯ(1,1,1))";

    Запрос.УстановитьПараметр("ТекДата", ТекДата);
    Выборка = Запрос.Выполнить().Выбрать();

    Пока Выборка.Следующий() Цикл

        Ответ = ""; Успешно = Ложь; ПакетJSON = "";
        ДопПуть = "/" + Выборка.ТипОбъекта;
        Если Выборка.ТипСобытия =
Перечисления.ТипыСобытийСинхронизации.Создание Тогда
            МетодHTTP = "POST";
            ПакетJSON =
СформироватьПолиструктурныйПакет(Выборка.ОбъектID,
Выборка.ТипОбъекта, "save");
            ИначеЕсли Выборка.ТипСобытия =
Перечисления.ТипыСобытийСинхронизации.Обновление Тогда
                МетодHTTP = "PUT"; ДопПуть = ДопПуть + "/" +
Выборка.ОбъектID;
                ПакетJSON =
СформироватьПолиструктурныйПакет(Выборка.ОбъектID,
Выборка.ТипОбъекта, "save");
                ИначеЕсли Выборка.ТипСобытия =
Перечисления.ТипыСобытийСинхронизации.Удаление Тогда
                    МетодHTTP = "DELETE"; ДопПуть = ДопПуть + "/" +
Выборка.ОбъектID;
                    ПакетJSON = "Команда DELETE (тело пустое)";
                    КонецЕсли;
                    Успешно = ОтправитьПакетНаСервер(ВыборкаУзлов.Ссылка,
ПакетJSON, Ответ, МетодHTTP, ДопПуть);
                    МенеджерЛога =
РегистрыСведений.ЖурналИнтеграции.СоздатьМенеджерЗаписи();
                    МенеджерЛога.Период = ТекущаяДата();
                    МенеджерЛога.Узел = ВыборкаУзлов.Ссылка;

```

```

        МенеджерЛога.ОбъектID = Выборка.ОбъектID;
        МенеджерЛога.ОбъектПредставление =
Выборка.ОбъектПредставление;
        Если Успешно Тогда
            МенеджерЛога.Статус =
Перечисления.РезультатыИнтеграции.Успех;
        Иначе
            МенеджерЛога.Статус =
Перечисления.РезультатыИнтеграции.Ошибка;
            КонецЕсли;
            МенеджерЛога.ТекстСообщения = ПакетJSON;
            МенеджерЛога.ОтветСервера = Ответ;
            МенеджерЛога.Записать();
            МенеджерЗаписи =
РегистрыСведений.ОчередьСинхронизации.СоздатьМенеджерЗаписи();
            ЗаполнитьЗначенияСвойств(МенеджерЗаписи, Выборка);

        Если Успешно Тогда
            МенеджерЗаписи.Состояние =
Перечисления.СтатусыСинхронизации.Отправлено;
            МенеджерЗаписи.ТекстОшибки = "";
            МенеджерЗаписи.КоличествоПопыток = 0; // Сброс
счетчика
            МенеджерЗаписи.ДатаСледующейПопытки = '00010101';
        Иначе
            // Если произошла ошибка
            НоваяПопытка = Выборка.КоличествоПопыток + 1;
            МенеджерЗаписи.КоличествоПопыток = НоваяПопытка;
            МенеджерЗаписи.ТекстОшибки = Ответ;

            Если НоваяПопытка >= 5 Тогда
                МенеджерЗаписи.Состояние =
Перечисления.СтатусыСинхронизации.ЛимитПопытокИсчерпан;
                МенеджерЗаписи.ДатаСледующейПопытки =
'00010101';
            Иначе
                (1 мин, 5 мин, 30 мин, 2 часа)
                Пауза = ?(НоваяПопытка = 1, 60, ?(НоваяПопытка
= 2, 300, ?(НоваяПопытка = 3, 1800, 7200)));
                МенеджерЗаписи.Состояние =
Перечисления.СтатусыСинхронизации.Ошибка;
                МенеджерЗаписи.ДатаСледующейПопытки = ТекДата
+ Пауза;

            КонецЕсли;
        КонецЕсли;

        МенеджерЗаписи.Записать();

        КонецЦикла;
    КонецЦикла;
КонецПроцедуры
Листинг А.2 – Процедура СобратьПакетЗаказПокупателя
Процедура СобратьПакетЗаказПокупателя(ДокОбъект, Данные)

```

```

        Данные.Вставить ("number",           ДокОбъект.Номер);
        Данные.Вставить ("date",             Формат (ДокОбъект.Дата,
"ДФ=yyyy-MM-ddTHH:mm:ss"));
        Данные.Вставить ("client",           Строка (ДокОбъект.Клиент));
        Данные.Вставить ("manager",          Строка (ДокОбъект.Менеджер));
        Данные.Вставить ("warehouse",        Строка (ДокОбъект.Склад));
        Данные.Вставить ("status",           Строка (ДокОбъект.Статус));
        Данные.Вставить ("payment",
Строка (ДокОбъект.СтатусОплаты));
        Данные.Вставить ("total",             ДокОбъект.СуммаЗаказа);
        Данные.Вставить ("needs_delivery",    ДокОбъект.НужнаДоставка);

// ТЧ Товары
МассивТоваров = Новый Массив;
Для Каждого СтрТ Из ДокОбъект.Товары Цикл
    Товар = Новый Структура ("name, qty, unit, price, total",
Строка (СтрТ.Номенклатура), СтрТ.Количество,
Строка (СтрТ.ЕдиницаИзмерения), СтрТ.Цена, СтрТ.Сумма);

    // Сборка полиструктурных свойств
    Свойства = Новый Соответствие;
    Для Каждого СтрСвойство Из ДокОбъект.СвойстваСтрок Цикл
        Если СтрСвойство.КлючСвязи = СтрТ.КлючСвязи Тогда
            Свойства.Вставить (Строка (СтрСвойство.Свойство),
ФорматироватьЗначениеПолиструктуры (СтрСвойство.Значение));
        КонецЕсли;
    КонецЦикла;

    Товар.Вставить ("props", Свойства);
    МассивТоваров.Добавить (Товар);
КонецЦикла;
Данные.Вставить ("items", МассивТоваров);
КонецПроцедуры

```

Листинг Б.2 – Процедура СобратьПакетРеализация Процедура

СобратьПакетРеализация (ДокОбъект, Данные)

```

        Данные.Вставить ("number",           ДокОбъект.Номер);
        Данные.Вставить ("date",             Формат (ДокОбъект.Дата,
"ДФ=yyyy-MM-ddTHH:mm:ss"));
        Данные.Вставить ("warehouse",        Строка (ДокОбъект.Склад));
        Данные.Вставить ("base_order",
Строка (ДокОбъект.ЗаказОснование));
        Данные.Вставить ("client",           Строка (ДокОбъект.Клиент));
        Данные.Вставить ("total",            ДокОбъект.СуммаДокумента);
        Данные.Вставить ("delivery_needed",  ДокОбъект.НужнаДоставка);

        Если ДокОбъект.НужнаДоставка Тогда
            Логистика = Новый Структура;
            Логистика.Вставить ("driver",
ПарситьСтрокуВОбъект (ДокОбъект.ДанныеВодителя));
        КонецЕсли;

```

```

        Логистика.Вставить ("transport",
ПарситьСтрокуВОбъект (ДокОбъект.ТранспортнаяНакладная));
        Логистика.Вставить ("priority",
Строка (ДокОбъект.ПриоритетДоставки));
        Данные.Вставить ("delivery_info", Логистика);
    КонецЕсли;

    // ТЧ Товары
    МассивТоваров = Новый Массив;
    Для Каждого Стр Из ДокОбъект.Товары Цикл
        ТоварJSON = Новый Структура ("name, qty, unit, price, sum",
Строка (Стр.Номенклатура), Стр.Количество,
Строка (Стр.ЕдиницыИзмерения), Стр.Цена, Стр.Сумма);
        МассивТоваров.Добавить (ТоварJSON);
    КонецЦикла;
    Данные.Вставить ("items", МассивТоваров);
КонецПроцедуры

```

Приложение В
(обязательное)
Листинг серверной части веб-портала

Листинг В.1 – Метод processIncomingJson (класс SyncService, главный диспетчер)

```

@Transactional
public void processIncomingJson(String rawJson) throws Exception {
    JsonNode root = mapper.readTree(rawJson);
    JsonNode meta = root.path("meta_info");

    String type = meta.path("type").asText();
    String action = meta.path("action").asText("save");
    String guid = meta.path("id").asText();

    if ("delete".equals(action)) {
        executeDelete(type, guid);
        return;
    }

    if ("ЗаказПокупателя".equals(type)) {
        saveOrder(root);
    } else if ("ПоступлениеТоваров".equals(type)) {
        saveIncomingGoods(root);
    } else if ("РеализацияТоваровИУслуг".equals(type)) {
        saveInvoice(root);
    } else if ("ПоступлениеДенежныхСредств".equals(type)) {
        savePaymentIn(root);
    } else if ("РасходДенежныхСредств".equals(type)) {
        savePaymentOut(root);
    }
}

```

Листинг В.2 – Метод executeDelete (класс SyncService, удаление документа)

```

@Transactional
public void executeDelete(String type, String guid) {
    if ("ЗаказПокупателя".equals(type)) {
        orderItemRepo.deleteByOrderId(guid);
        orderRepo.deleteById(guid);
    } else if ("ПоступлениеТоваров".equals(type)) {
        incomingItemRepo.deleteByParentDocId(guid);
        incomingRepo.deleteById(guid);
    } else if ("РеализацияТоваровИУслуг".equals(type)) {
        invoiceItemRepo.deleteByInvoiceId(guid);
        invoiceRepo.deleteById(guid);
    } else if ("ПоступлениеДенежныхСредств".equals(type)) {
        paymentInItemRepo.deleteByPaymentId(guid);
        paymentInRepo.deleteById(guid);
    } else if ("РасходДенежныхСредств".equals(type)) {
        paymentOutRepo.deleteById(guid);
    }
}

```

Листинг В.3 – Метод saveOrder (класс SyncService, сохранение заказа покупателя с полиструктурными свойствами)

```

java
private void saveOrder(JsonNode node) {
    String guid = node.path("meta_info").path("id").asText();
    orderItemRepo.deleteByOrderId(guid);

    SalesOrder order = new SalesOrder();
    order.setId(guid);
    order.setNumber(node.path("number").asText());
    if (node.has("date") && !node.path("date").isNull()) {
order.setDate(LocalDateTime.parse(node.path("date").asText()));
    }
    order.setClient(node.path("client").asText());
    order.setManager(node.path("manager").asText());
    order.setWarehouse(node.path("warehouse").asText());
    order.setStatus(node.path("status").asText());
    order.setPaymentStatus(node.path("payment").asText());

    order.setNeedsDelivery(node.path("needs_delivery").asBoolean());
    if (node.has("total")) order.setTotalAmount(new
BigDecimal(node.path("total").asText()));
    orderRepo.save(order);

    JsonNode itemsNode = node.path("items");
    if (itemsNode.isArray()) {
        for (JsonNode i : itemsNode) {
            SalesOrderItem item = new SalesOrderItem();
            item.setOrder(order);
            item.setProductName(i.path("name").asText());
            item.setUnit(i.path("unit").asText());
            if (i.has("qty")) item.setQuantity(new
BigDecimal(i.path("qty").asText()));
            if (i.has("price")) item.setPrice(new
BigDecimal(i.path("price").asText()));
            if (i.has("total")) item.setTotal(new
BigDecimal(i.path("total").asText()));

            item.setCustomPropertiesJson(i.path("props").toString());
            orderItemRepo.save(item);
        }
    }
}

```

Приложение Г
(обязательное)
Листинг клиентской части веб-портала

```

document.addEventListener("DOMContentLoaded", function() {
    document.querySelectorAll('.js-json-formatter, .js-tech-
parser').forEach(function(element) {
        let rawData = element.textContent.trim();
        if (!rawData || rawData === 'null' || rawData === '{}' ||
rawData === 'нет данных') {
            element.innerHTML = '<span class="no-data">нет
данных</span>';
            return;
        }
        try {
            let json = JSON.parse(rawData);
            let isTech = element.classList.contains('js-tech-
parser');
            let html = `<ul style="list-style:none; padding:0;
margin:0;">`;

            for (let key in json) {
                let val = json[key];
                let checkStr = String(val).toLowerCase().trim();
                if (checkStr === "true") {
                    val = "Да";
                } else if (checkStr === "false") {
                    val = "Нет";
                }
                if (typeof val === 'object' && val !== null) val =
JSON.stringify(val);

                html += `<li style="margin-bottom:4px; font-
size:0.85rem; display:flex; align-items:center;">
                    <span class="${isTech ? 'tech-label' :
'property-label'}">${key}</span>
                    <span style="margin-
left:8px;">${val}</span>
                </li>`;
            }
            element.innerHTML = html + '</ul>';
            element.classList.remove('no-data');
        } catch (e) {
            console.log("Ошибка в данных: " + rawData);
        }
    });

    document.querySelectorAll('.js-bank-render, .js-details-
render').forEach(function(container) {
        let raw = container.getAttribute('data-json');
        if (!raw || raw === 'null' || raw === '{}') return;
        try {
            let json = JSON.parse(raw);
            let html = '';
            for (let key in json) {

```

```

        if (json.hasOwnProperty(key) && json[key] !==
null) {
            let label = labels[key] || key;
            let val = json[key];
            if (typeof val === "boolean") {
                val = val ? "Да" : "Нет";
            }

            let isDetail =
container.classList.contains('js-details-render');
            html += `
```

```

        html += `<div class="col-md-5">
                    <div class="card h-100 shadow-sm card-
transport">
                        <div class="card-body">
                            <h6 class="text-warning fw-
bold mb-3 small">ТРАНСПОРТ</h6>
                            ${Object.entries(json.transport).map(([k, v]) => `<div
class="small">· ${k}: ${formatVal(v)}</div>`).join('')}
                            </div>
                        </div>
                    </div>`;
    }

    if (json.priority) {
        html += `<div class="col-md-2">
                    <div class="card h-100 shadow-sm bg-
white text-center d-flex align-items-center justify-content-center
p-3">
                        <div class="x-small-title mb-
2">Приоритет</div>
                        <div class="badge bg-danger p-2
px-3">${formatVal(json.priority)}</div>
                    </div>
                </div>`;
    }

    container.innerHTML = html;
} catch (e) { console.error("Ошибка парсинга логистики",
e); }
    });
});
/

```

Приложение Д
(рекомендуемое)

Результаты сквозного тестирования

← →

☆

Заказ покупателя 000000025 от 07.06.2026 17:45:41

🔗 ⓘ

Провести и закрыть

Записать

Провести

Еще

Номер:

000000025

Дата:

07.06.2026 17:45:41

📅

Клиент:

ИП Волков С.П.

✕ ⓘ

Нужна доставка:

☐

Менеджер:

Кузнецова Татьяна Борисовна

✕ ⓘ

Склад:

Магазин-склад "ТехноСтрой"

✕ ⓘ

Статус:

Отгружен

✕

Статус оплаты:

Не оплачен

✕

Сумма заказа:

4 370,00

👤

Состав заказа

Добавить

⬆ ⬇ ⬆

Поиск (Ctrl+F)

✕

Еще

N	Номенклатура	Количество	Единица измерения	Цена	Сумма	Сидра
1	Кирпич облицовочный красный	60,000	шт	24,50	1 470,00	
2	Цемент М-500 ПЦ (50 кг)	5,000	упак	580,00	2 900,00	

Свойства позиции

Добавить

⬆ ⬇ ⬆

Поиск (Ctrl+F)

✕

Еще

N	Свойство	Значение
1	Зона выгрузки на объекте	Сектор А (Фундамент)

Панель мониторинга предприятия

Заказы

Закупки

Отгрузки

Приход денег

Расход денег

№ Заказа	Дата	Клиент	Менеджер	Склад	Доставка	Статус	Оплата	Сумма
000000025	07.06.2026 17:45	ИП Волков С.П.	Кузнецова Татьяна Борисовна	Магазин-склад "ТехноСтрой"	Самовывоз	Отгружен	Не оплачен	4370,00 Р

Состав заказа:

Товар	Кол-во	Ед.	Цена	Сумма	Характеристики
Кирпич облицовочный красный	60,000	шт	24,50	1470,00	Зона выгрузки на объекте: Сектор А (Фундамент)
Цемент М-500 ПЦ (50 кг)	5,000	упак	580,00	2900,00	нет данных

Рисунок Д.1 – Создание документа «Заказ покупателя»

← →

☆

Заказ покупателя 000000025 от 07.06.2026 17:45:41

🔗 ⓘ

Провести и закрыть

Записать

Провести

Еще

Номер:

000000025

Дата:

07.06.2026 17:45:41

📅

Клиент:

ИП Волков С.П.

✕ ⓘ

Нужна доставка:

☐

Менеджер:

Кузнецова Татьяна Борисовна

✕ ⓘ

Склад:

Магазин-склад "ТехноСтрой"

✕ ⓘ

Статус:

Отгружен

✕

Статус оплаты:

Не оплачен

✕

Сумма заказа:

4 370,00

👤

Состав заказа

Добавить

⬆ ⬇ ⬆

Поиск (Ctrl+F)

✕

Еще

N	Номенклатура	Количество	Единица измерения	Цена	Сумма	Сидра
1	Кирпич облицовочный красный	60,000	шт	24,50	1 470,00	
2	Цемент М-500 ПЦ (50 кг)	5,000	упак	580,00	2 900,00	

Свойства позиции

Добавить

⬆ ⬇ ⬆

Поиск (Ctrl+F)

✕

Еще

N	Свойство	Значение
1	Дополнительная маркировка	Наслейка "Хрупкое"
2	Зона выгрузки на объекте	Сектор А (Фундамент)
3	Класс опасности груза	Класс Б (средняя опасность)
4	Особые условия транспортировки	Запрещена перевалка (не перегружать)

Панель мониторинга предприятия

Заказы

Закупки

Отгрузки

Приход денег

Расход денег

№ Заказа	Дата	Клиент	Менеджер	Склад	Доставка	Статус	Оплата	Сумма
000000025	07.06.2026 17:45	ИП Волков С.П.	Кузнецова Татьяна Борисовна	Магазин-склад "ТехноСтрой"	Самовывоз	Отгружен	Не оплачен	4370,00 Р

Состав заказа:

Товар	Кол-во	Ед.	Цена	Сумма	Характеристики
Кирпич облицовочный красный	60,000	шт	24,50	1470,00	Особые условия транспортировки: Запрещена перевалка (не перегружать) Класс опасности груза: Класс Б (средняя опасность) Зона выгрузки на объекте: Сектор А (Фундамент) Дополнительная маркировка: Наслейка "Хрупкое"
Цемент М-500 ПЦ (50 кг)	5,000	упак	580,00	2900,00	Тип упаковки: На поддонах

Рисунок Д.2 – Обновление документа «Заказ покупателя»

Заказ покупателя

Реализация товаров

Поступление товаров

Номенклатура

Контрагенты

Склады

Свойства доставки

Свойства номенклатуры

Транспортные средства

← → ☆

Поступление товаров 000000004 от 04.06.2026 15:46:33

🔍 ⓘ ✕

Провести и закрыть

Записать

Провести

Еще

Номер:

000000004

Дата:

04.06.2026 15:46:33

📅

Поставщик:

ОАО "ЦементМастер"

✕ ⓘ

Склад:

Транзитная площадка (Юг)

✕ ⓘ

Организация:

Учредитель (Инвестиции)

✕ ⓘ

Сумма документа:

1 252,00

Статус оплаты:

Оплачен

✕

Состав поступления

Добавить

⬆ ⬇ ⬆

Поиск (Ctrl+F)

✕

Еще

N	Номенклатура	Количество	Единицы измерения	Цена	Сумма
1	Кирпич облицовочный красный	4,000	шт	13,00	52,00

Технические параметры

Добавить

⬆ ⬇ ⬆

Поиск (Ctrl+F)

✕

Еще

N	Параметр	Значение
1	Тип упаковки	На поддонах
2	Имеется сертификат	Да

🚚 Панель мониторинга предприятия

Заказы

Закупки

Отгрузки

Приход денег

Расход денег

№ Поступление товаров	Дата закупки	Поставщик	Организация	Склад приемки	Оплата	Сумма
000000004	04.06.2026 15:46	ОАО "ЦементМастер"	Учредитель (Инвестиции)	Транзитная площадка (Юг)	Оплачен	1252,00 Р

📦 Принятые товары по накладной:

Товар	Кол-во	Единица измерения	Цена	Сумма	Параметры качества
Кирпич облицовочный красный	4,000	шт	13,00	52,00 Р	Имеется сертификат Да Тип упаковки На поддонах

Рисунок Д.3 – Создание документа «Поступление товаров»

← → ☆

Поступление товаров 000000004 от 04.06.2026 15:46:33 *

🔍 ⓘ ✕

Провести и закрыть

Записать

Провести

Еще

Номер:

000000004

Дата:

04.06.2026 15:46:33

📅

Поставщик:

ОАО "ЦементМастер"

✕ ⓘ

Склад:

Транзитная площадка (Юг)

✕ ⓘ

Организация:

Учредитель (Инвестиции)

✕ ⓘ

Сумма документа:

52,00

Статус оплаты:

Оплачен

✕

Состав поступления

Добавить

⬆ ⬇ ⬆

Поиск (Ctrl+F)

✕

Еще

N	Номенклатура	Количество	Единицы измерения	Цена	Сумма
1	Кирпич облицовочный красный	4,000	шт	13,00	52,00
2	Цемент М-500 ПЦ (50 кг)	3,000	упак	400,00	1 200,00

Технические параметры

Добавить

⬆ ⬇ ⬆

Поиск (Ctrl+F)

✕

Еще

N	Параметр	Значение
4	Меры безопасности при разгрузке	Использовать перчатки (защита рук)
5	Брак при приеме	Нет

🚚 Панель мониторинга предприятия

Заказы

Закупки

Отгрузки

Приход денег

Расход денег

№ Поступление товаров	Дата закупки	Поставщик	Организация	Склад приемки	Оплата	Сумма
000000004	04.06.2026 15:46	ОАО "ЦементМастер"	Учредитель (Инвестиции)	Транзитная площадка (Юг)	Оплачен	1252,00 Р

📦 Принятые товары по накладной:

Товар	Кол-во	Единица измерения	Цена	Сумма	Параметры качества
Кирпич облицовочный красный	4,000	шт	13,00	52,00 Р	Дата ветконтроля 2026-06-01 Имеется сертификат Да Тип упаковки На поддонах
Цемент М-500 ПЦ (50 кг)	3,000	упак	400,00	1200,00 Р	Брак при приеме Нет Меры безопасности при разгрузке Использовать перчатки (защита рук)

Рисунок Д.4 – Обновление документа «Поступление товаров»

← →

☆ Реализация товаров 000000005 от 09.06.2026 12:44:51

🔍

🔍

Провести и закрыть

Записать

Провести

Еще

Основные данные

Номер: 000000005

Дата: 09.06.2026 12:44:51

Менеджер: Кузнецова Татьяна Борисовна

Заказ основание: Заказ покупателя 000000025 от 07.06.2026 17:45:41

Нужна доставка: ☐

Клиент: ИП Волков С.П.

Сумма документа: 4 370,00

Склад: Магазин-склад "ТехноСтрой"

Добавить

↑

↓

Поиск (Ctrl+F)

Еще

N	Номенклатура	Количество	Единицы измерения	Цена	Сумма
1	Кирпич облицовочный красный	60,000	шт	24,50	1 470,00
2	Цемент М-500 ПЦ (50 кг)	5,000	упак	580,00	2 900,00

Панель мониторинга предприятия

Заказы

Закупки

Отгрузки

Приход денег

Расход денег

№ Реализации	Дата	Заказ основание	Клиент	Склад	Менеджер	Доставка	Сумма итого
000000005	09.06.2026 12:44	Заказ покупателя 000000025 от 07.06.2026 17:45:41	ИП Волков С.П.	Магазин-склад "ТехноСтрой"	нет данных	Самовывоз	4370,00 Р

≡ Спецификация отгрузки:

Наименование	Кол-во	Ед.	Цена	Итого
Кирпич облицовочный красный	60,000	шт	24,50	1470,00
Цемент М-500 ПЦ (50 кг)	5,000	упак	580,00	2900,00

Рисунок Д.5 – Создание документа «Реализация товаров»

← →

☆ Реализация товаров 000000005 от 09.06.2026 12:44:51

🔍

🔍

Провести и закрыть

Записать

Провести

Еще

Основные данные

Номер: 000000005

Дата: 09.06.2026 12:44:51

Менеджер: Кузнецова Татьяна Борисовна

Заказ основание: Заказ покупателя 000000025 от 07.06.2026 17:45:41

Нужна доставка: ☒

Клиент: ИП Волков С.П.

Сумма документа: 4 370,00

Склад: Магазин-склад "ТехноСтрой"

Добавить

↑

↓

Поиск (Ctrl+F)

Еще

N	Номенклатура	Количество	Единицы измерения	Цена	Сумма
1	Кирпич облицовочный красный	60,000	шт	24,50	1 470,00
2	Цемент М-500 ПЦ (50 кг)	5,000	упак	580,00	2 900,00

← →

Результаты ответственного за доставку

🔍

ФИО: Морозов Артем Александров

Телефон: +7 (900) 666-77-88

Статус инструкции по ТБ: Спец.инструкция по работе

Примеры СМЗ: Комплект палатки (ваз, жилет, обувь)

Завершить ввод

← →

Спецификация транспортной логистики

🔍

Номер ТП: 34353531135

Дата ТП: 07.06.2026

Вид перевозки: Грузовой темный

Перевозчик: ИП Грозаво (Иванов)

Транспорт: Volvo FH16 (Дальнобой)

Завершить ввод

Панель мониторинга предприятия

Заказы

Закупки

Отгрузки

Приход денег

Расход денег

№ Реализации	Дата	Заказ основание	Клиент	Склад	Менеджер	Доставка	Сумма итого
000000005	09.06.2026 12:44	Заказ покупателя 000000025 от 07.06.2026 17:45:41	ИП Волков С.П.	Магазин-склад "ТехноСтрой"	нет данных	Дл	4370,00 Р

🔍 Данные о доставке:

ВОДИТЕЛЬ

СМЗ Комплект полный (ваз, жилет, обувь)
Инструкция: Спец.инструкция по работе
Тек: +7 (900) 666-77-88
Водитель: Морозов Артем Александров

ТРАНСПОРТ

Авто: Volvo FH16 (Дальнобой)
Вид: Грузовой темный
Компания: ИП Грозаво (Иванов)
Дата: 02.06.2026
Номер: 34353531135

ПРИОРИТЕТ

Напом.

≡ Спецификация отгрузки:

Наименование	Кол-во	Ед.	Цена	Итого
Кирпич облицовочный красный	60,000	шт	24,50	1470,00
Цемент М-500 ПЦ (50 кг)	5,000	упак	580,00	2900,00

Рисунок Д.6 – Обновление документа «Реализация товаров»

← → ☆ Поступление денежных средств 000000016 от 01.06.2026 13:50:09 *

Провести и закрыть

Записать

Провести

Еще

Номер: 000000016

Дата: 01.06.2026 13:50:09

Организация: Наша компания (ТехноСтрой-М)

Клиент: ОАО "ЦементМастер"

Способ оплаты: Наличные

Сумма документа: 2 900,00

Расшифровка платежа

Добавить

↑ ↓

Поиск (Ctrl+F)

×

Еще

N	Заказ	Сумма
1	Заказ покупателя 000000022 от 31.05.2026 13:21:43	2 900,00

Свойства платежа

Добавить

↑ ↓

Поиск (Ctrl+F)

×

Еще

N	Свойство	Значение
1	Код программы лояльности	678 076 445

Панель мониторинга предприятия

Заказы

Закупки

Отгрузки

Приход денег

Расход денег

№ Документа	Дата	Организация	Клиент	Способ оплаты	Сумма
000000016	01.06.2026 13:50	Наша компания (ТехноСтрой-М)	ОАО "ЦементМастер"	Наличные	2900,00 Р

Расшифровка платежа по заказам:

Заказ покупателя	Сумма оплаты	Дополнительные характеристики платежа
Заказ покупателя 000000022 от 31.05.2026 13:21:43	2900,00 Р	<div>Код программы лояльности: 678076445</div>

Рисунок Д.7 – Создание документа «Поступление денежных»

← → ☆ Поступление денежных средств 000000016 от 01.06.2026 13:50:09 *

Провести и закрыть

Записать

Провести

Еще

Номер: 000000016

Дата: 01.06.2026 13:50:09

Организация: Наша компания (ТехноСтрой-М)

Клиент: ОАО "ЦементМастер"

Способ оплаты: Наличные

Сумма документа: 2 900,00

Расшифровка платежа

Добавить

↑ ↓

Поиск (Ctrl+F)

×

Еще

N	Заказ	Сумма
1	Заказ покупателя 000000022 от 31.05.2026 13:21:43	2 900,00

Свойства платежа

Добавить

↑ ↓

Поиск (Ctrl+F)

×

Еще

N	Свойство	Значение
1	Код программы лояльности	678 076 445
2	Промокод	AR5453B53
3	Дата сверки с банком	06.05.2026

Панель мониторинга предприятия

Заказы

Закупки

Отгрузки

Приход денег

Расход денег

№ Документа	Дата	Организация	Клиент	Способ оплаты	Сумма
000000016	01.06.2026 13:50	Наша компания (ТехноСтрой-М)	ОАО "ЦементМастер"	Наличные	2900,00 Р

Расшифровка платежа по заказам:

Заказ покупателя	Сумма оплаты	Дополнительные характеристики платежа
Заказ покупателя 000000022 от 31.05.2026 13:21:43	2900,00 Р	<div>Дата сверки с банком: 2026-05-06</div> <div>Промокод: AR5453B53</div> <div>Код программы лояльности: 678076445</div>

Рисунок Д.8 – Обновление документа «Поступление денежных средств»

←

→

☆

Расход денежных средств 000000007 от 13.05.2026 23:39:02

Провести и закрыть

Записать

Провести

Еще

Номер:

000000007

Дата:

13.05.2026 23:39:02

Организация:

Наша компания (ТехноСтрой-М)

Вид операции:

Выплата зарплаты

Получатель:

Васильев Максим Игоревич

Месяц выплаты:

Февраль

Вид выплаты:

Зарботная плата

Сумма:

4 000,00

Заказы

Закупки

Отгрузки

Приход денег

Расход денег

Панель мониторинга предприятия

№ Расхода	Дата	Организация	Получатель	Вид операции	Сумма
000000009	09.06.2026 13:26	Наша компания (ТехноСтрой-М)	Васильев Максим Игоревич	Выплата зарплаты	5,00 Р
000000008	09.06.2026 13:26	Наша компания (ТехноСтрой-М)	ОАО "ЦементМастер"	Оплата поставщику	52,00 Р
000000007	13.05.2026 23:39	Наша компания (ТехноСтрой-М)	Васильев Максим Игоревич	Выплата зарплаты	4000,00 Р

Детали операции:

МЕСЯЦ ВЫПЛАТЫ:

Февраль

ВИД ВЫПЛАТЫ:

Заработная плата

Рисунок Д.9 – Создание документа «Расход денежных средств»

Поступление денежных средств

Расход денежных средств

Банки

Банковские счета

Единицы измерения

Свойства платежа

← → ☆

Расход денежных средств 000000007 от 13.05.2026 23:39:02

🔗

Провести и закрыть

Записать

Провести

Номер:

000000007

Дата:

13.05.2026 23:39:02

📅

Организация:

Наша компания (ТехноСтрой-М)

▼ 📄

Вид операции:

Оплата поставщику

▼

Получатель:

ОАО "ЦементМастер"

▼ 📄

Документ закупки:

Поступление товаров 000000003 от 08.05.2026 17:19:33

▼ 📄

Сумма:

2 465,00

👤

🏠

Панель мониторинга предприятия

📄 Заказы

📄 Закупки

📄 Отгрузки

📄 Приход денег

📄 Расход денег

№ Расхода	Дата	Организация	Получатель	Вид операции	Сумма
000000009	09.06.2026 13:26	Наша компания (ТехноСтрой-М)	Васильев Максим Игоревич	Выплата зарплаты	5,00 Р
000000008	09.06.2026 13:26	Наша компания (ТехноСтрой-М)	ОАО "ЦементМастер"	Оплата поставщику	52,00 Р
000000007	13.05.2026 23:39	Наша компания (ТехноСтрой-М)	ОАО "ЦементМастер"	Оплата поставщику	2465,00 Р

ⓘ Детали операции:

📄 ДОКУМЕНТ ЗАКУПКИ:

Поступление товаров 000000003 от 08.05.2026 17:19:33

Рисунок Д.10 – Обновление документа «Расход денежных средств»