Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Институт электронного обучения
Специальность 230101 Вычислительные машины, комплексы, системы и сети
Кафедра вычислительной техники

ДИПЛОМНЫЙ ПРОЕКТ/РАБОТА

Тема работы
Автоматизация обмена данными между информационными системами на платформе 1С Предприятие

УДК 004.773:004.771

Студент

Группа	ФИО	Подпись	Дата
З-8302	Филиппов Антон Павлович		

Руководитель

Должность	ФИО	Ученая степень, звание	Подпись	Дата
ассистент	Полищук В.Ю.	К.Т.Н.		

КОНСУЛЬТАНТЫ:

По разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Конотопский В.Ю.	К.Э.Н.		

По разделу «Социальная ответственность»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Извеков В.Н.	К.Т.Н.		

ДОПУСТИТЬ К ЗАЩИТЕ:

Зав. кафедрой	ФИО	Ученая степень, звание	Подпись	Дата
ВТ	Марков Н.Г.	д.т.н., профессор		

Томск 2016

Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Институт электронного обучения
Специальность 230101 Вычислительные машины, комплексы, системы и сети
Кафедра Вычислительной техники

УТВЕРЖДАЮ:
Зав. кафедрой
_____ _____ Н.Г.Марков
(Подпись) (Дата) (Ф.И.О.)

ЗАДАНИЕ
на выполнение выпускной квалификационной работы

В форме:

дипломного проекта/работы

Студенту:

Группа	**ФИО**
З - 8302	Филиппов Антон Павлович

Тема работы:

Автоматизация обмена данными между информационными системами на платформе 1С:Предприятие	
Утверждена приказом директора (дата, номер)	

Срок сдачи студентом выполненной работы:	

ТЕХНИЧЕСКОЕ ЗАДАНИЕ:

Исходные данные к работе *(наименование объекта исследования или проектирования; производительность или нагрузка; режим работы (непрерывный, периодический, циклический и т. д.); вид сырья или материал изделия; требования к продукту, изделию или процессу; особые требования к особенностям функционирования (эксплуатации) объекта или изделия в плане безопасности эксплуатации, влияния на окружающую среду, энергозатратам; экономический анализ и т. д.).*	Задачи: - Необходимо проанализировать конфигурации информационных систем и определить состав объектов для обмена данными. - Изучить основные технологи, механизмы и инструменты интеграции данных. - Создать правила переноса данными. - Разработать механизм обмена данными. - Автоматизировать процесс обмена между информационными системами. ИС. Исходными данными к работе являются результаты производственной практики. Обмен данными реализован посредством платформы 1С:Предприятие 8 с использованием

	специализированной конфигурации «Конвертация данных»
Перечень подлежащих исследованию, проектированию и разработке вопросов *(аналитический обзор по литературным источникам с целью выяснения достижений мировой науки техники в рассматриваемой области; постановка задачи исследования, проектирования, конструирования; содержание процедуры исследования, проектирования, конструирования; обсуждение результатов выполненной работы; наименование дополнительных разделов, подлежащих разработке; заключение по работе).*	Целью данного проекта является разработка и автоматизация обмена данными между территориально распределенными информационными системами. В процессе выполнения проекта были использованы программные продукты, Платформа 1С: Предприятие 8 и конфигурация 1С: Конвертация данных 2.0.
Перечень графического материала *(с точным указанием обязательных чертежей)*	

Консультанты по разделам выпускной квалификационной работы

(с указанием разделов)

Раздел	Консультант
Социальная ответственность	Извеков Владимир Николаевич
Финансовый менеджмент, ресурсоэффективность и ресурсосбережение	Конотопский Владимир Юрьевич

Дата выдачи задания на выполнение выпускной квалификационной работы по линейному графику	

Задание выдал руководитель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
ассистент	Полищук В.Ю.	К.Т.Н.		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
З-8302	Филиппов Антон Павлович		

Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Институт электронного обучения
Специальность 230101 Вычислительные машины, комплексы, системы и сети
Уровень образования среднемедиальное
Кафедра вычислительной техники
Период выполнения весенний семестр 2016 учебного года

Форма представления работы:

дипломный проект/работа

(бакалаврская работа, дипломный проект/работа, магистерская диссертация)

КАЛЕНДАРНЫЙ РЕЙТИНГ-ПЛАН
выполнения выпускной квалификационной работы

Срок сдачи студентом выполненной работы:	

Дата контроля	Название раздела (модуля) / вид работы (исследования)	Максимальный балл раздела (модуля)
10.03.16	*Анализ ИС и определение состава объектов для обмена данными*	*10*
21.03.16	*Изучение механизмов обмена данными на платформе 1С*	*10*
04.04.16	*Создание правил переноса данных*	*20*
15.04.16	*Разработка механизма обмена данными на платформе 1С*	*20*
29.04.16	*Автоматизация обмена данными*	*20*
16.05.16	*Финансовый менеджмент, ресурсоэффективность и ресурсосбережение*	*10*
30.05.16	*Социальная ответсвенность*	*10*

Составил преподаватель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата

СОГЛАСОВАНО:

Зав. кафедрой	ФИО	Ученая степень, звание	Подпись	Дата

РЕФЕРАТ

Выпускная квалификационная работа состоит из 101 с., 29 рисунков., 19 табл., 18 источников, 2 прил. объемом 3 и 7 стр. соответственно.

Ключевые слова: Информационная система, автоматизация, конфигуратор, обмен данными, интеграция.

Объектом исследования является обмен данными между разными территориально распределенными информационными системами на платформе 1С: Предприятие.

Цель работы автоматизация процесса обмена данными, определенной структуры, между информационными системами «Продажи Блюд» и «1С: Управление небольшой фирмой», работающими на платформе 1С: Предприятие 8 и конфигурации 1С: Конвертация данных 2.0.

В процессе исследования проанализированы различные механизмы обмена данными, обладающими определенной структурой, между системами, применительно к платформе 1С. Изучены основные особенности обмена данными, между информационными системами.

Степень внедрения: предложенная разработка была успешно внедрена в компании ООО «Золото Сибири», в дальнейшем планируется техническое сопровождение программного модуля.

Область применения: применяется для рационального и эффективного использования времени работы персонала.

СОДЕРЖАНИЕ

ТЕРМИНОЛОГИЯ

Сокращения/Термин	Определение
ИС	Информационная система либо Интегрирующая среда(определяется по контексту)
«Продажи Блюд»	Конфигурация специально разработанная для автоматизации учета в сфере общественного питания
«УНФ»	«Управление небольшой фирмой» типовая конфигурация, разработанная фирмой 1С. Для автоматизации управленческого и бухгалтерского учета
МД	Метаданные
РИБ	Распределенная информационная база
Источник	Конфигурация «Продажи Блюд»
Приемник	Конфигурация УНФ
ПКО	Правила конвертации объектов
ПКС	Правила конвертации свойств
ПВД	Правила выгрузки данных
XML	Extensible Markup Language (Расширяемый язык разметки)
JSON	Текстовый формат обмена данными, основанный на JavaScript
Номенклатура	Перечень продуктов, готовых блюд.
Заказы	Документ предназначен для регистрации розничных продаж готовой продукции
Расходная накладная	Документ предназначен для хранения информации по отражению в учете различных

	операции по реализации (отгрузке) запасов и услуг стороннему контрагенту(клиенту, покупателю)
Справочник	Объект метаданных Источника или Приемника
Документ	Объект метаданных Источника или приемника
MD82exp.epf	Содержит набор правил для выгрузки структуры метаданных конфигураций.
Паттерн	Шаблон. Применительно к данной теме шаблон интеграции данных
БД	База данных
ПО	Программное обеспечение

ТЕХНИЧЕСКОЕ ЗАДАНИЕ

Предмет разработки

Предметом разработки является программный модуль обмена данными между двумя удаленными информационными системами.

Назначение

Автоматизация процесса переноса повторяющихся данных из одной информационной системы (ИС) в другую, с учетом особенностей структур данных этих систем.

Требования к конвертации объектов

Объект МД	Источник	Приемник
Справочник	Номенклатура	Номенклатура
Справочник	Клиенты	кпб_РозничныеПокупатели
Справочник	ЕденицыИзмерения	ЕденицыИзмерения
Справочник	Склады	СтруктурныеЕденицы
Документ	Заказы	Расходная Накладная
Документ	Перемещения	ПеремещениеЗапасов
Документ	АктСписания	СписаниеЗапасов
Документ	ПриходнаяНакладная	ПриходнаяНакладная

Требования к конвертации объектов и реквизитов

Заполнение реквизитов Справочника «Номенклатура» в Приемнике

Реквизит	Приемник
ЕдиницаИзмерения	Справочники.КлассификаторЕдиницИзмерения.шт
НоменклатурнаяГруппа	Справочники.НоменклатурныеГруппы.ОсновнаяГруппа
ТипНоменклатуры	Перечисления.ТипыНоменклатуры.Запас
СтавкаНДС	Справочники.СтавкиНДС.НайтиПоНаименованию("Без НДС")
МетодОценки	Перечисления.МетодОценкиЗапасов.FIFO
СпособПополнения	Перечисления.СпособыПополненияЗапасов.Производство
СчетУчетаЗапасов	ПланыСчетов.Управленческий.ТоварыПродукция
СчетУчетаЗатрат	ПланыСчетов.Управленческий.НезавершенноеПроизводство
НаправлениеДеятельности	Справочники.НаправленияДеятельности.ОсновноеНаправление
Склад	Справочники.СтруктурныеЕдиницы.ОсновнойСклад
Родитель	Справочники.Номенклатура.НайтиПоКоду("ФР-00000001")

Заполнение реквизитов Документа «РасходнаяНакладная» в Приемнике

Реквизит	Приемник
ВидОперации	Перечисления.ВидыОперацийРасходнаяНакладная.ПродажаПокупателю
Контрагент	Справочники.Контрагенты.НайтиПоКоду("ФР-000003")
Договор	Объект.Контрагент.ДоговорПоУмолчанию
Организация	Справочники.Организации.ОсновнаяОрганизация
Комментарий	"Заказ №" + Объект.НомерВходящегоДокумента
ВидЦен	Справочники.ВидыЦен.Оптовая
ВалютаДокумента	Константы.ВалютаУчета.Получить()
Курс	1
Кратность	1
НалогообложениеНДС	Перечисления.ТипыНалогообложенияНДС.НеОблагается НДС
Подразделение	Справочники.СтруктурныеЕдиницы.ОсновноеПодразделение
Родитель	Справочники.Номенклатура.НайтиПоКоду("ФР-00000001")

ВВЕДЕНИЕ

Система «1С: Предприятие» разработана для автоматизации различной деятельности предприятий. Предоставляет широкие возможности для решения задач оперативного управления и планирования, ведения управленческого и бухгалтерского учета, расчета заработной платы, а также комплексной автоматизации организационно - хозяйственной деятельности финансовых и торговых организаций, производственных предприятий, предприятий сферы услуг и бюджетных учреждений. Система «1С: Предприятие» содержит прикладные решения, разработанные на основе технологической платформы, методическую поддержку и методологию. Разработка и модификация прикладного решения осуществляется в режиме запуска системы «1С: Предприятие» «Конфигуратор». В этом режиме разработчик определяет общую архитектуру и структуру данных прикладного решения, создает экранные формы и макеты, алгоритмы поведения задаются с помощью встроенного языка в модулях объектов.

При организации обмена данными между различными конфигурациями 1С следует принять во внимание, что версии конфигураций могут не совпадать, конфигурация может быть не типовой, также следует учитывать тип передаваемых данных и наличие стандартных средств, позволяющих обмениваться данными по стандартным правилам между определенными конфигурациями.

Актуальность работы обусловлена отсутствием программных средств, обеспечивающих обмен данными, обладающими разной структурой, между двумя информационными системами, а именно: «Продажи Блюд» и «1С: Управление небольшой фирмой».

Целью данного проекта является автоматизация процесса обмена данными, определенной структуры, между информационными системами «Продажи Блюд» и «1С: Управление небольшой фирмой», работающими на

платформе 1С: Предприятие 8 и конфигурации 1С: Конвертация данных 2.0.

При работе над проектом необходимо решить следующие задачи:

- Проанализировать конфигурации информационных систем и определить состав объектов для обмена данными.
- Изучить основные технологи, механизмы и инструменты интеграции данных.
- Создать правила переноса данных между информационными системами.
- С учетом правил разработать механизм обмена данными между ИС.
- Автоматизировать процесс обмена данными между информационными системами.

1. ТЕХНОЛОГИИ ИНТЕГРАЦИИ

Интеграция данных – это процесс объединения данных из разных источников, и предоставление этих данных в унифицированном виде.

Интеграция корпоративных данных и приложений это задача, давно стоящая перед многими различными организациями, однако лишь за последнее десятилетие появилась техническая возможность ее реализации.

Методы интеграции данных:

1. Консолидация. Консолидация — процесс, направленный в одну сторону, то есть данные загруженные в хранилище из нескольких источников не выгружаются обратно в распределенную систему. Консолидированные могут использоваться в качестве основы приложений бизнес-аналитики (business intelligence, bi),olap-приложений.[1]

2. Федерализация данных. Физического перемещения в федеративных источниках данных не происходит. Они остаются у владельцев, а доступ осуществляется при необходимости.

3. Распространение данных. Приложения распространения данных копируют данные из одного места в другое. Как правило эти приложения зависят от определенных событий и работают в оперативном режиме, а также производят перемещение данных к местам назначения. Обновления в источнике могут передаваться в приемник синхронно или асинхронно. Метод распространения, независимо от типа синхронизации, гарантирует доставку данных в систему назначения. Это является главным отличительным признаком технологии распространения данных.

4. Сервисно - ориентированная архитектура или SOA (Service Oriented Architecture), применяемая при интеграции приложений, также имеет возможность применения для интеграции данных. При запросе происходит обращение к определённому сервису-посреднику, который связан с источником, в котором находится её информация и конкретный адрес. из Из

нескольких источников интеграция данных объединяет информацию таким образом, чтобы ее можно было представить клиенту в виде сервиса.

Сервис — это извлечение некоторой бизнес-сущности (или сущностей), которое может быть выполнено сервисом интеграции через серию запросов и других сервисов [7]. Подход SOA, в первую очередь, концентрируется на определении и совместном использовании самых важных бизнес-функций в организации. Следовательно, сервис-ориентированные интерфейсы основаны на небольшом количестве запросов на нужную информацию, которую необходимо предоставить потребителю.

5. Гибридный метод, использующий консолидацию, и одновременно поддерживает федеративный подход к интеграции данных. Данные, которые относятся к определенному первичному приложению (например, заказы), могут быть федерализированы, а общие данные о клиентах (имя, адрес и т.д.) могут быть консолидированы. За счет метода распространения данных такой гибридный подход может быть расширен. В том случае, когда клиент во время транзакции обновляет свой адрес и имя, эти изменения могут быть отправлены в консолидированный склад данных, откуда они распространяются в другие первичные системы.

Широкое применение и развитие этих методов привело к формированию программных технологий на основе которой строится интеграция информации.

Технологии интеграции данных:

- Интеграция приложений предприятия (Enterprise Application Integration, сокр. EAI);
- Интеграция информации предприятия (Enterprise Information Integration, сокр. EII)
- Программное обеспечение для извлечения, преобразования и загрузки данных (Extract, Transform and Load, сокр. ETL).

EAI - это технология, задача которой, объединить в единый процесс несколько приложений и осуществления между ними преобразования форматов данных.

ETL - это технология переноса данных из автоматизированной информационной системы или приложения в другие информационные системы[5]. Процесс ETL реализуется путем разработки приложения ETL, либо создания комплекса встроенных программных процедур. Приложения ETL извлекают информацию из исходной базы данных источника, преобразуют ее в формат, поддерживаемый базой данных назначения, а затем загружают преобразованные данные.

EII - это технология, работающая в режиме реального времени, для интеграции несопоставимых типов данных из большого количества источников как за пределами организации, так и внутри нее. Инструменты EII используют технологию поиска информации и обеспечивают универсальный уровень доступа к данным (pull technology).

Рассмотрим место этих технологий в уже существующей архитектуре.

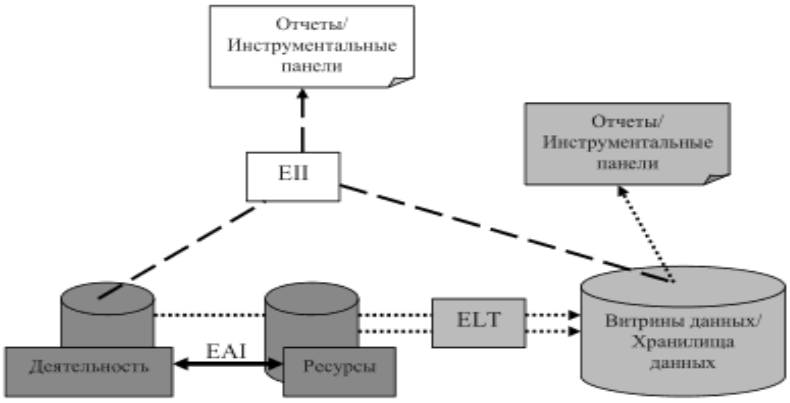

Рисунок 1.1. Архитектура использования технологий.

При необходимости связать приложения в реальном времени для автоматизации бизнес-процессов наиболее применима технология EAI. А также EAI применяется, когда необходимо, чтобы изменения, внесенные в одно приложение (обычно это небольшой набор записей), были отражены во

всех других.

Технология ETL является наиболее полезной тогда, когда необходимо создать хранилище, которое содержит проверенные надежные данные для исторического анализа, например, для анализа многомерных запросов или временных рядов. ETL применяется для осуществления проверки качества и удаления дублирующихся данных. Также ETL используются для создания отдельных специализированных хранилищ данных, обслуживающих бизнес-процесс или конкретный отдел.

Технология EII обеспечивает непрерывную интеграцию данных и универсальную систему доступа к ним. Кроме традиционных реляционных баз данных, инструменты EII могут работать с XML и LDAP-файлами (Lightweght Directory Access Protocol - облегченный протокол доступа к каталогам) и другими не реляционными данными. Эти инструменты также способны представить реляционные данные в формате web-сервисов или формате XML. В случае возникновения необходимости добавить к справочным данным хранилища дополнительные детали, например, сопоставление исторических данных с текущей ситуацией особенно полезны инструменты EII.

Внедрение этих технологий требует от IT-персонала глубокого понимания требований, которые предъявляются к данным для принятия стратегических и тактических решений. Применительно к технологии ETL это означает, что необходимые данные извлекаются, преобразуются и загружаются в пригодном для использования виде EII – сервером или аналитиками. При применении технологии EII, данные должны быть пригодны для использования в аналитических отчетах.

Важно учитывать, что внедрение этих технологий в уже сложившуюся архитектуру предполагает от IT-персонала, создания стратегии управления приложениями и данными, которая будет поддерживать этот процесс в активном состоянии. Осознание того, что с самого начала создаются

контрольные журналы и повышается важность механизмов архивирования, является обязательной составляющей этой стратегии. Это необходимо для обеспечения надежности и слаженности приложений и интегрированных данных. И наконец, в условиях конкретной инфраструктуры очень важен постоянный мониторинг эффективности и производительности этих технологий. В большей степени их производительность будет зависеть от размеров и детальности данных, скорости архивирования, а также от эффективного функционирования системы в условиях полной нагрузки.

1.1. Основные трудности разработки интегральных решений:

- Ненадежность сети передачи данных. Все интеграционные решения предусматривают передачу информации между устройствами, чаще всего общающиеся информационные системы разделены территориально, что требует перемещать данные по сегментам локальных сетей, телефонным линиям, через коммутаторы, маршрутизаторы, спутниковые каналы связи и общедоступные сети. Доставка информации связана с задержкой и риском потери данных.
- Низкая скорость передачи данных. Процесс доставки данных с помощью компьютерной сети занимает гораздо больше времени чем вызов локального метода. Таким образом для создания распределенного приложения необходимо применять иные принципы проектирования, нежели создание приложения, которое выполняется в пределах одного компьютера.
- Различия между приложениями. Существующие различия между объединяемыми системами (язык программирования, платформа, формат данных) должны учитываться при внедрении интеграционного решения.
- Неизбежность изменений. Интеграционное решение должно иметь возможность адаптации к изменению объединяемых им приложений. Преобразования в одной системе, как правило, влекут за собой

непредсказуемые последствия для других систем, поэтому при интеграции приложений важно уменьшить их взаимозависимость.

1.2. Способы решения трудностей интегральных решений:

1. Передача файла (File Transfer). Одно приложение создает файл, а другое приложение его считывает. Приложения должны согласовать расположение, имя файла, время записи/считывания, формат, и процедуру удаления.

2. Общая база данных (Shared Database). Одной физической базе данных соответствует несколько приложений которые используют общую логическую структуру данных. Проблемы передачи информации между приложениями устраняются наличием единого хранилища данных.

3. Удаленный вызов процедуры (Remote Procedure Invocation). С помощью удаленного вызова процедуры приложение предоставляет доступ к части своей функциональности. Приложения взаимодействуют между собой синхронно в режиме реального времени.

4. Обмен сообщениями (Messaging). Приложение размещает сообщение в общем канале, которое затем считывается другим приложением. Приложения должны согласовать канал, а также формат сообщения. Между приложениями взаимодействие осуществляется в асинхронном режиме.

Каждый из выше описанных методов имеет свои собственные недостатки и преимущества. Что бы использовать преимущества того или иного подхода, на практике приложения могут быть интегрированы разными способами.

1.3. Критерии оценки качества интеграции

- Связывание приложений. Зависимости между интегрированными приложениями должны быть минимизированы. Сильное связывание

подразумевает большое количество допущений между интегрированными приложениями. Нарушение некоторых допущений может произойти в результате изменения функционала единственного приложения. Следствием этого может стать нестабильная работа интеграционного решения. Таким образом, необходимую функциональность должны обеспечивать интерфейсы объединяемых приложений, одновременно допуская возможность изменения внутренней реализации.

- Изменение приложений. Количество изменений и кода, требуемого при разработке интеграционного решения, вносимые в объединяемые приложения должны быть минимальны. При этом подобная минимизация не должна препятствовать реализации необходимой функциональности конечного решения.
- Выбор технологии. К выбору метода интеграции, который будет использоваться в проекте следует подойти очень основательно. Стоит отметить, что удорожанию проекта может привести использование специализированного аппаратного и (или) программного обеспечения. А также следует учитывать, что начинать интеграцию «с нуля» чревато всевозможными рисками.
- Формат данных. В интегрированных приложениях должны использоваться одинаковые форматы данных. В случае, когда это невозможно с помощью встроенных средств приложений или путем внесения в них изменений, то для унификации формата данных используются внешние трансляторы.
- Своевременность доставки данных. Периодический обмен небольшими объемами данных является наиболее простым способом обеспечения своевременной доставки информации между приложениями. Но при большом количестве передаваемых данных, такой подход не будет эффективным. Негативные последствия, возникающие в результате задержки

обмена данными, могут привести к рассогласованию приложений, а также усложнить интеграционное решение

- Общая функциональность. Помимо обмена данными, многие интеграционные решения предполагают использование общего функционала приложений. Вызов функции удаленного приложения имеет принципиальное отличия от вызова локальной функции, которые способны повлиять на интеграционное решение.
- Удаленное взаимодействие. Удаленные процедуры не стоит вызывать синхронно. Использование асинхронного взаимодействия существенно повышает эффективность интеграционного решения, но в то же время, делает его более сложным в обслуживании, проектировании и разработке.
- Надежность. Вызов локальной функции гораздо более надежен, чем удаленное взаимодействие. Вызов удаленной процедуры связан с определенными рисками, а именно с необходимостью корректной работы удаленного приложения и функционирования сети. Асинхронный подход обеспечивает надежное взаимодействие между интегрируемыми приложениями.

Не существует универсального способа интеграции приложений, который в равной степени удовлетворял бы всем рассмотренным выше критериям. Но для каждого конкретного интеграционного решения есть свой оптимальный способ. У каждого способа интеграции имеются свои преимущества и недостатки.

1.4. Основные преимущества и недостатки интегральных решений

Передача файла (File Transfer). Файлы — это универсальный механизм, реализованный в операционной системе для хранения данных, а также поддерживающийся любым языком программирования. Из этого

следует, что самым простым способом интеграции данных может быть основан на использовании файлов[3].

Не так давно наиболее распространенным стандартом обмена считался простой текстовый файл, но на сегодняшний день современные интеграционные решения в основном используют обмен данными через XML - файл и в свою очередь набирает популярность формат JSON.

При организации обмена следует учитывать с какой периодичностью следует создавать и считывать файлы С точки зрения использования ресурсов излишне частая работа с файлами является неэффективной. Чаще всего, периодичность создания файлов определяется исходя из бизнес-логики организации.

Основное преимущество при использовании обмена файлами заключается в том, что разработчику интеграционного решения не требуется обладать дополнительной специфичной информацией о внутренней реализации объединяемых приложений. Соответственно основной задачей разработчиков, если это требуется, будет преобразование форматов файлов. В результате получаем слабое связывание этих приложений, а общедоступными интерфейсами являются файлы, создаваемые этими приложениями. Отсутствие необходимости применения дополнительных пакетов интеграции является важным преимуществом передачи файлов. Однако в результате этого увеличивается нагрузка на разработчиков интеграционного решения. При интеграции приложений должны использоваться одинаковые правила именования файлов, а приложение, создающее файл, должно обеспечить уникальность его имени.

Также необходимо использовать механизм периодического удаления старых файлов. В случае если у интегрируемых приложений нет доступа к одному диску, то также необходимо использовать механизм переноса файлов из одного месторасположения в другое.

В результате достаточно редкого обмена информацией возникает проблема рассинхронизации интегрируемых систем, это является одним из недостатков метода передачи файлов. Например, обмен данными между приложениями происходит раз в сутки, следовательно, при изменении каких-то данных в одной системе, в другой до ближайшего обмена будут использоваться старые данные. Тогда первую очередь периодичность создания файлов зависит от потребности в наличии актуальной информации приложений, обрабатывающих эти файлы.

Общая база данных (Shared Database). Несколько приложений или информационных систем используют одну базу данных. Основным недостатком является настолько тесная связь между интегрированными приложениями, что иногда невозможно заметить границу между ними (чаще всего так интегрируется программное обеспечение одного производителя). Однако слишком тесная связь превращает интегрированные приложения в некую «суперсистему». Самостоятельная замена и модернизация отдельных компонентов которой является достаточно сложной. Эта проблема решается с применением механизмов серверов баз данных, но не всегда эффективно.

Во избежание этой проблемы информацию необходимо хранить в центральной базе данных, которая доступна для всех интегрируемых приложений. А также общая база обеспечивает полную согласованность хранящихся в ней данных. При использовании данных различными источниками контроль транзакций БД будет пресекать попытки одновременного изменения информации. Использование реляционной модели данный является наиболее простым подходом к реализации общей базы данных, поддерживающей язык запросов SQL. На сегодняшний день этот язык поддерживается практически всеми платформами для разработки приложений и позволяет решить проблему различия в форматах файлов. На этапах проектирования и реализации интеграционного решения применение общей

БД способствует устранению проблемы семантического несоответствия. Разработка схемы базы данных представляет собой одну из главных трудностей этого подхода. Использование унифицированной БД не должно приводить к уменьшению производительности особо важного приложения, иначе руководство предприятия, скорее всего, настоит на пересмотре проекта интеграции.

Коммерческое программное обеспечение также является одной из проблем в процессе реализации общей базы данных. В большинстве случаев коммерческие приложения работают со встроенной схемой данных, возможность адаптации которой достаточно трудно выполнима. Задача усложняется еще тем, что поставщик новой версии приложения может изменить схему данных.

Общая база данных может стать узким метом интеграционного решения при увеличении числа обращений к ней. Следовательно, это может привести к увеличению вероятности блокировки данных и снижению производительности объединяемых приложений. Эти проблемы следует учитывать при проектировании интеграционного решения.

Удаленный вызов процедуры (Remote Procedure Invocation). Инкапсуляция является одним из мощнейших механизмов структурирования приложений, а также предусматривает доступ к данным с помощью программного кода. Общая база данных представляет собой не инкапсулированную структуру, которая затрудняет поддержку интегрированных с ее помощью приложений. Изменения в приложении могут повлечь за собой изменения в базе данных, что, в свою очередь, скажется на всех оставшихся приложениях.

Решением этой проблем может является создание механизма, позволяющего одному приложению использовать функцию другого приложения, передавая все необходимые данные для обработки. Такой

механизм называется удаленный вызов процедуры. Для получения или изменения информации, которая находится в другой системе, приложение обращается с помощью вызова определенной функции. Таким образом, каждое приложение самостоятельно может изменять формат и обеспечивать целостность своих данных, при этом другие приложения не будут задействованы.

Удаленный вызов процедуры поддерживается множеством технологий, таких как RPC, SOAP CORBA, COM, .NET Remoting и Java RMI. Реализация удаленного вызова процедуры часто предполагает наличие дополнительных механизмов, например, поддержки транзакций.

Отличительной особенностью удаленного вызова процедуры является сильной связывание приложений. Обычно это связано с использованием правил, применяемых в разработке конкретного приложения.

Обмен сообщениями (Messaging). Система обмена сообщениями обеспечивает слабое связывание интегрируемых приложений. Изменение формата сообщения осуществляется во время передачи, без уведомления, как получателя, так и отправителя. Система обмена сообщениями использует различные способы доставки информации: широковещательную рассылку всем получателям, маршрутизацию сообщений одному получателю или нескольким. Необходимость изменения данных диктуется различными моделями, находящимися в основе консолидируемых приложений.

Периодический обмен небольшими частями данных создает причины для использования общих функций приложений. А также вызывающей стороне не требуется время на ожидание посланного сообщения.

Возможность преобразования сообщений позволяет обеспечить гораздо более слабую связность приложений, чем этого можно было бы достичь с помощь удаленного вызова процедур. Вместе с тем подобная независимость означает дополнительную нагрузку на разработчиков интеграционного

решения, которым приходится создавать множество строк связующего кода.

Взаимное отношение описанных выше подходов можно представить в графическом виде. Стрелками указаны преимущества одного подхода над другим.

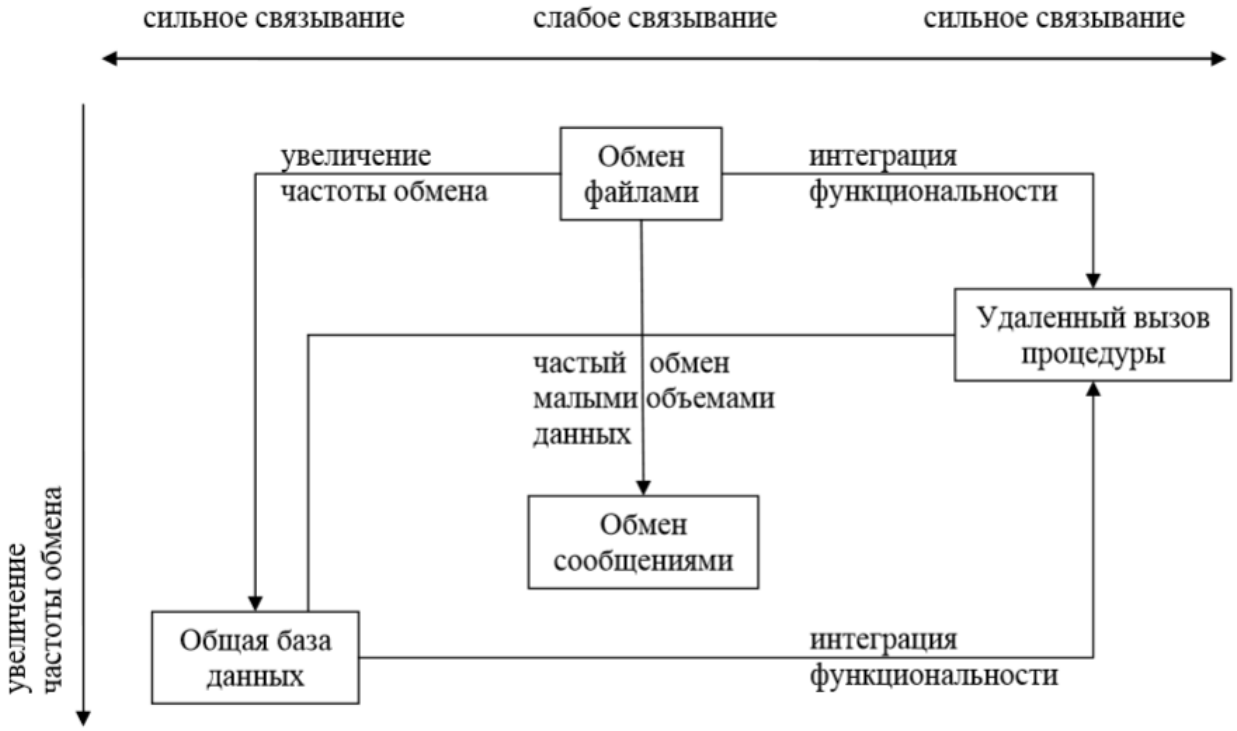

Рисунок 1.2 — Взаимное отношение подходов интеграции

2. ШАБЛОНЫ ИНТЕГРАЦИИ

Для интеграции были созданы паттерны, то есть шаблоны интеграционных решений. Шаблоны (паттерны) проектирования — это проверенный способ представления накопленного опыта и знаний в таких областях, как архитектура программных приложений, объектно-ориентированное проектирование и создание интеграционных решений. Каждый шаблон включает в себя обсуждение исходных условий, описание некой проблемы проектирования, а также предоставление конкретного решения [2]. Каждый шаблон содержит в себе опыт старших архитекторов и разработчиков для получения оптимального решения конкретной задачи интеграции.

Шаблоны проектирования делятся на три группы:

1. Шаблоны проектирования распределения обязанностей и объектов информационных систем или взаимодействия отдельных классов;
2. Архитектурные шаблоны;
3. Шаблоны интегрирования ИС.

2.1. Структурные паттерны интеграции

1. Взаимодействие "точка - точка". При таком взаимодействии одна система является активной, другая пассивной. Активная система использует интерфейс другой системы, а пассивная система, предоставляет интерфейсы для использования другими системами.

Схема такого взаимодействия представлена на рисунке.

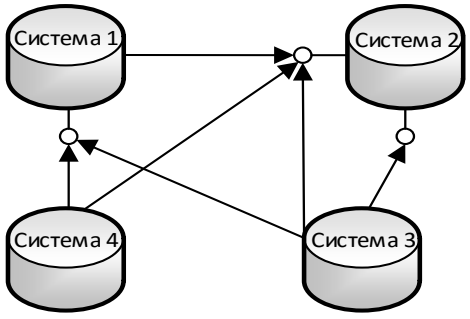

Рисунок 2.1. – Схема взаимодействия «точка-точка»

2. Взаимодействие «звезда» (интегрирующая среда). Данный способ предполагает наличие центрального компонента (интегрирующей среды), который с помощью общей информационной системы управляет взаимодействием подсистем.

Интегрирующей среда - это совокупность программных и организационных компонентов, цель которых - обеспечить взаимодействие подсистем и образовать единую систему [2]. Таким образом присутствие интегрирующей среды говорит о целостности единой системы, а не о наборе ее отдельных приложений.

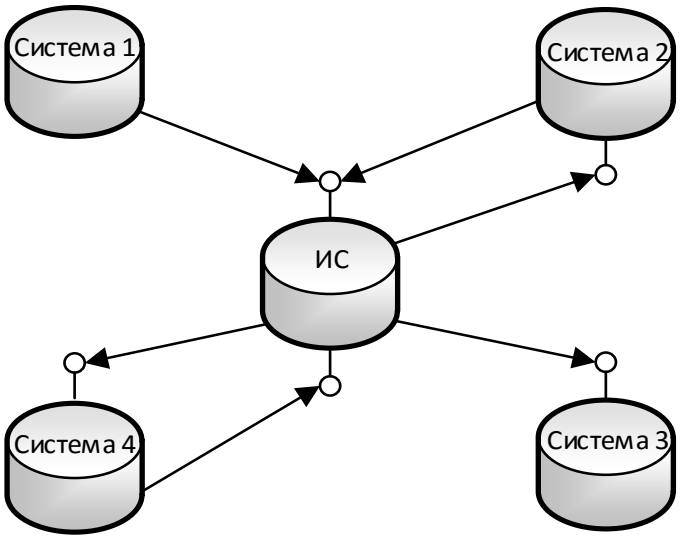

Рисунок 2.2. – Схема взаимодействия «звезда»

Для доступа активных систем интегрирующая среда имеет универсальный интерфейс. А также, интегрирующая среда может использовать интерфейсы пассивных систем.

Интегрирующая система включает в себя реализацию основных уровней интегрирующей среды, представлена на рисунке 2.3

Основные уровни интегрирующей среды

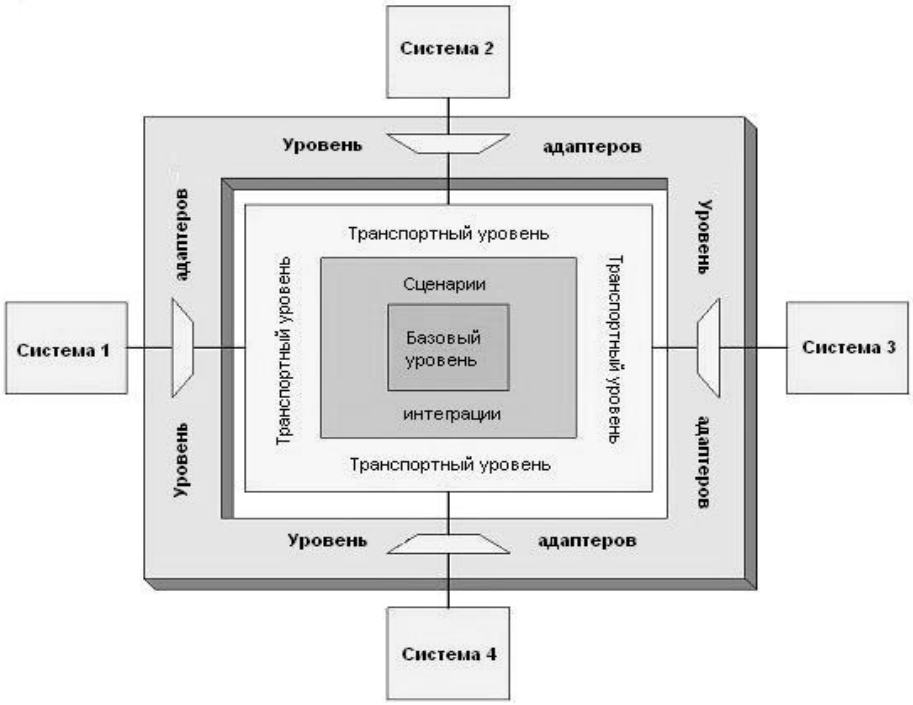

Рисунок 2.3 – Уровни интегрирующей среды

• Базовый уровень является ядром интегрирующей среды. На этом уровне имеется платформа для вызовов сценариев транзакции, мониторинга состояния интегрирующей среды и службы протоколирования, а также базовый функционал по взаимодействию приложений.

• Уровень сценариев интеграции предполагает наличие графической схемы обмена сообщениями между системами, алгоритмов преобразования и маршрутизации сообщений;

• Транспортный уровень интегрирующей среды обеспечивает физическую доставку сообщений между составляющими системы.

• Уровень адаптеров обеспечивает взаимодействие с системой с помощью API. Генерацию и передачу сообщений базовому уровню используя транспортный уровень.

3. Смешанный способ взаимодействия. В данном способе объединены первые два подхода к взаимодействию систем. При этом

интерфейсы могут использоваться в обход интегрирующей среды или непосредственно напрямую. Данный способ имеет следующие преимущества: унификация интерфейсов, централизация управления процессами взаимодействия систем, а также возможность напрямую использовать интерфейсы между системами.

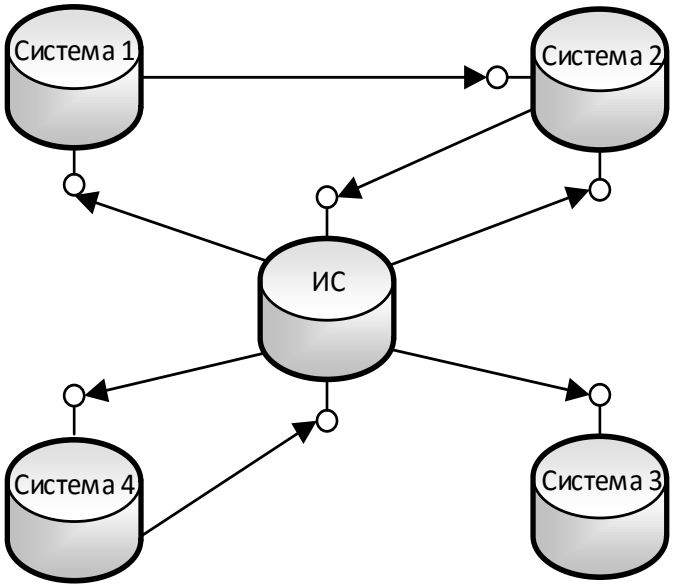

Рисунок 2.4 – Смешанный способ взаимодействия

2.2. Паттерны по методу интеграции

1. Интеграция систем по данным (data-centric). При интеграции приложений по данным, главным фактором при разработке информационной системы является общая база данных, предназначенная для многопользовательского доступа. Смысл такого подхода в объединении приложений в систему вокруг объединяемых данных под управлением СУБД. Интегрирующей средой является, как правило, реляционная СУБД с SQL интерфейсом доступа к данным. Все функции прикладной обработки находятся в клиентских программах. Основной недостаток такого подхода в необходимость передачи большого количества данных.

2. Функционально-центрический (function-centric) подход. Сервисы являются основным фактором функционально-центрического подхода. В виде

сервисов используются различного вида функции, такие как контроль информационной безопасности, прикладная обработка, служба единого времени и централизованный файловый доступ. Все сервисы интегрированы, т.е. функции, реализуемые сервисами общедоступны, достоверны, и непротиворечивы. Идея данного подхода состоит в том, что вокруг интегрированных сервисов, приложения объединяются в систему со стандартизованным интерфейсом. Интегрирующей средой является монитор транзакций со стандартным API или сервер приложений. Общая архитектура системы состоит из трех частей: сервер базы данных, клиентское приложение, функциональные сервисы.

3. Объектно-центрический (object-centric). Объектно-центрический подход, основывается на положениях объектного взаимодействия CORBA, COM/DCOM, DOTNET и является сложением типов объектно-центрического объединения систем по данным. Идея интеграции в том, что система объединяется вокруг распределенных, общедоступных объектов со стандартными интерфейсами.

Особенностями данного подхода:

- унифицированный язык спецификации интерфейсов объектов;
- отделение реализации компонентов от спецификации их интерфейсов;
- общий механизм поддержки взаимодействия объектов (брокер объектных запросов, играющий роль «общей шины», поддерживающей взаимодействие объектов).

4. Интеграция на основе единой модели понятий предметной области (concept-centric). В таком подходе нужно решить задачу, для которой необходима интеграция в пределах единой системы интегрирующих средств разного рода. Эта проблема часто встречается для любой ИС большого

масштаба, в которой используются различные системы с собственными серверами приложений и прочими видами ПО промежуточного уровня.

Решением этой проблемы интеграции является разработка компонентов общесистемного языка взаимодействия(ОЯВ), основой которых является единая модель понятий, описывающая объекты предметной области, а также их взаимодействие. Обычно, общесистемный язык взаимодействия является языком сообщений высокого уровня и имеет естественно-языковую лексику на основе бизнес объектов и достаточно простой синтаксис. Единая понятийная модель является базой метаданных, в которой хранится описания бизнес-объектов каждого из компонентов и связи между ними. В базе данных должно поддерживаться постоянное соответствие между интегрируемыми компонентами и их описаниями. Язык взаимодействия и хранящиеся в базе метаданных описания строятся как независимые от конкретного интегрирующего ПО. Преобразование сообщений на ОЯВ в вызовы функций той или иной интегрирующей среды обеспечивается дополнительной интегрирующей оболочкой с единым интерфейсом, который предназначен только для обмена сообщениями на ОЯВ. Единицей информационного обмена в рассматриваемом подходе являются сообщения, поэтому целесообразно строить такое программное обеспечение на основе программных продуктов класса MOM (Message Oriented Middleware – это системное программное обеспечение промежуточного слоя, ориентированное на обмен сообщениями [2]).

2.3. Паттерны интеграции по типу обмена данными

1. Файловый обмен. Данный тип интеграции основывается на концепции «точка-точка», системы экспортируют общие данные в формате пригодном для импорта в другие системы. На сегодняшний день в качестве

единого формата обмена файлов чаще всего выбирают XML, как наиболее распространенный и поддерживаемый в мире.

XML (eXtensile Markup Language) - расширяемый язык гипертекстовой разметки, используемый в интернете. Язык XML использует структуру тегов и определяет содержание гипертекстового документа, а также позволяет автоматизировать обмен данными.

Практически все системы позволяют производить выгрузку-загрузку данных в формате XML. Существует большое количество программ, которые позволяют в удобной форме сконвертировать XML данные на основе технологии XSLT.

XSLT (eXtensible Stylesheet Language for Transformations) – технология, используется для преобразования XML документов. С ее помощью можно создать правила обмена, которые позволяют преобразовать документ в другую форму, структуру или формат, например, в HTML или текстовый документ.

Схема такого взаимодействия представлена на рисунке

Рисунок 2.5. — Схема шаблона файлового обмена.

Недостатком этого подхода является необходимость создания дополнительного механизма, который будет отвечать за периодическое проведение операций выгрузки-загрузки данных, корректности этих операций, а также за процессом преобразования и соблюдения формата обмена.

2. Общая база данных. Позволяет получить интегрированную систему приложений, которая использует единые данные в режиме реального времени. Изменения, выполненные в одном приложении, автоматически

отражаются в другом. За корректность данных отвечает многопользовательская СУБД.

Схема взаимодействия типа «Общая база данных»

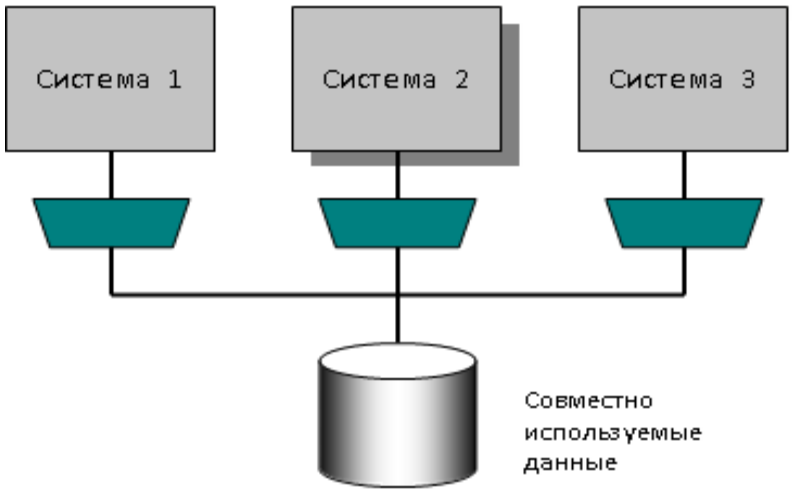

Рисунок 2.6. — Схема шаблона общей базы данных

3. Удаленный вызов процедур. В данном случае приложения интегрируются на уровне функций. А также посредством вызова функций происходит изменение данных в другой системе.

Схема взаимодействия типа «Удаленный вызов процедур»

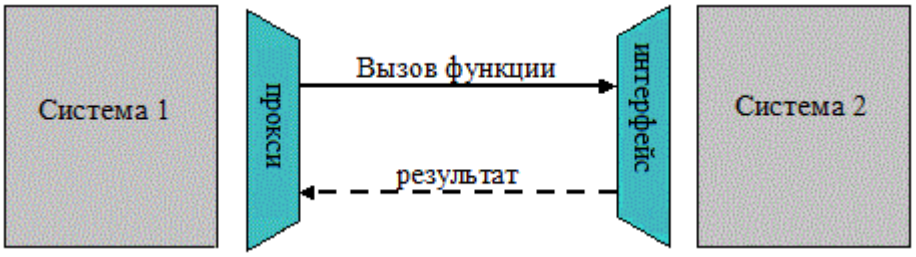

Рисунок 2.7. — Схема удаленный вызов процедуры.

Каждая из систем самостоятельно заботится о поддержке данных в корректном состоянии, что является довольно сложно задачей.

4. Обмен сообщениями. В данном типе интеграции используется асинхронный обмен сообщениями предназначенный для интеграции независимых приложений с минимальными доработками либо вовсе без них. При этом за логику процесса отвечает шина данных в отличие от других типов

интеграции. Такой подход позволяет легко интегрировать новые системы, а также изменять логику процесса интеграции.

Схема взаимодействия типа «Обмен сообщениями»

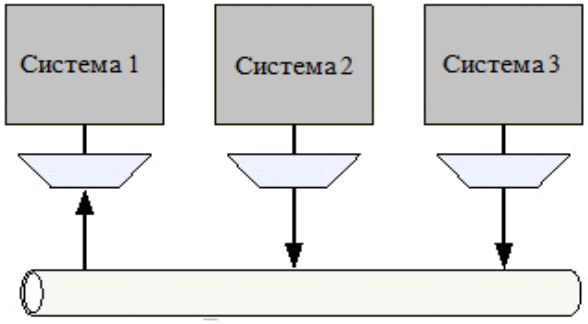

Рисунок 2.8. — Схема взаимодействия шаблона обмен сообщениями

3. МЕХАНИЗМЫ ОБМЕНА ДАННЫМИ 1с

Механизмы обмена данными платформы 1с - это набор инструментов системы, используемых для создания обмена данными между различными информационными базами, а также информационными базами и внешними источниками данных[3].

Механизмы обмена данными могут быть условно разделены на два уровня:

Рисунок 3.1. Общая структура обмена данными 1С.

3.1. Универсальный механизм обмена данными

Система 1С:Предприятие в процессе обмена данными между БД не ограничивает структуру конкретных объектов и идентичность конфигурации. Обмен данными также может быть организован с другими информационными системами, а не только с ИС 1С:Предприятие.

Недостаток этого способа заключается в том, что он не является готовым решением и требует настройки структуры и детальной проработки

передаваемых данных и состава передаваемой информации. В отличие от РИБ данный вариант не имеет готовых решений в случае некорректной работы. В следствие того, что в процессе работы универсального механизма обмена данными неизбежно возникают нештатные ситуации, необходим специалист, обеспечивающий поддержку и сопровождение системы для быстрого восстановления ее работы.

В качестве ИС, с которыми организуется обмен, могут выступать другие информационные системы 1С:Предприятие. Также информационные системы, которые обмениваются между собой данными могут иметь различные конфигурации. Помимо этого, для организации обмена со сторонним программным обеспечением, не основанным на платформе 1С используется универсальный механизм обмена данными.

Факторы использования универсального механизма обмена данными:

- Формат обмена основывается на языке XML, который общепринятым средством представления данных.
- Средства обмена данными, используются для организации разнообразных схем обмена данными, благодаря своей высокой гибкости и модульной организации.
- протоколы, которые предлагаются механизмами обмена данными, достаточно просты и могут быть использованы во внешних программных ПО.

Составляющими универсального механизма обмена являются:

- средства чтения и записи документов XML;
- XML-сериализация;
- планы обмена.

3.1.1. Средства чтения и записи документов xml.

Обмен данными между участниками обмена осуществляется с помощью сообщений в формате XML, следовательно, базовый уровень обмена данными

образуется с помощью средств записи и чтения документов XML.

К средствам чтения и записи документов XML, предоставляемым системой 1С: Предприятие 8 относятся объекты: ЧтениеXML, ЗаписьXML и ПреобразованиеXSL.

Используя внешнее соединение и механизмы работы с XML можно организовывать интеграцию с прикладными системами по принятым в этих системах форматам. Для этого применяются механизмы XSL-преобразования.

3.1.2. Xml-сериализация

XML-сериализация – это процесс преобразования последовательности данных формата XML в данные 1С:Предприятия 8 и напротив, процесс преобразования данных 1С:Предприятия 8 в последовательность данных формата XML.

3.2. Распределенные информационные базы

Предназначен для создания территориально распределенных систем на основе абсолютно одинаковых и идентичных конфигураций 1С:Предприятия 8. В случае необходимости обмена данными между бизнес-единицами и подразделениями занимающимися разными видами деятельности и требующие использования специализированных конфигураций 1С:Предприятие 8 с учетом отраслевой специфики, этот способ может оказаться неприемлемым. Однако при этом данный способ также имеет ряд преимуществ:

- Позволяет организовывать обмен данными без дополнительного программирования, а также различные распределенные системы.
- Позволяет обеспечить идентичность конфигураций информационных баз, входящих в состав распределенной системы.
- Внести изменения в конфигурацию возможно только в

центральном узле распределенной системы.

- Передача данных возможна между любыми узлами, а не только через центральную БД.
- Более высокий уровень надежности хранения и восстановления информации.

Распределенная информационная база является иерархической структурой, которая состоит из отдельных информационных баз системы 1С: Предприятие 8 — узлов РИБ, между которыми организован обмен данными с целью синхронизации конфигурации и данных.

РИБ — состоит из информационных баз системы 1С: Предприятие 8 (узлов распределенной информационной базы), в которых используется синхронизация данных и конфигурации. РИБ имеет иерархическую структуру. У каждого узла распределенной информационной базы может быть один главный и произвольное число подчиненных узлов. Корневым узлом РИБ является главный узел, или узел, у которого нет главного узла. Каждый узел обменивается данными только со своим главным и подчиненными узлами.

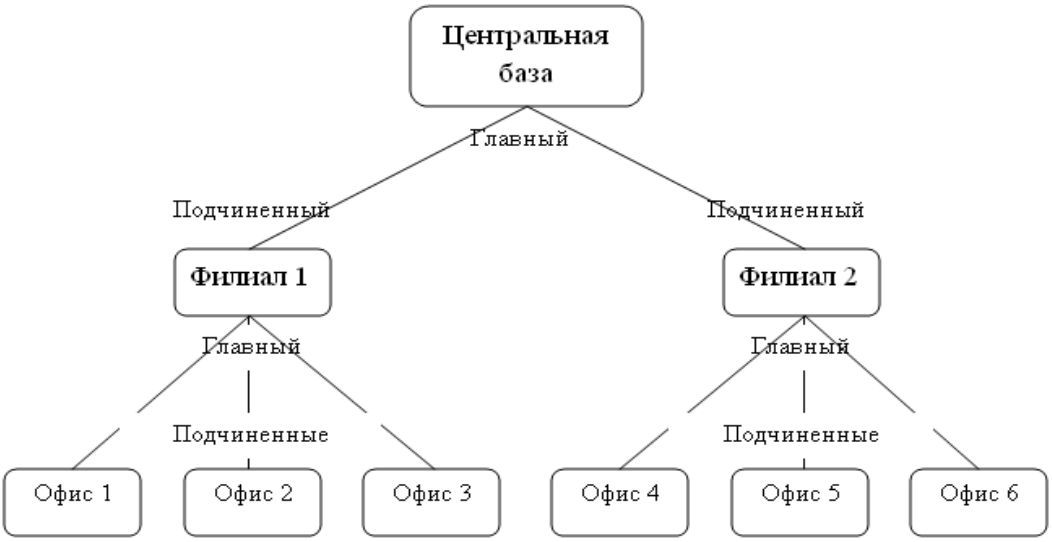

Рисунок 3.2. Схема распределенного обмена данными.

Изменения конфигурации доступны только в корневом узле РИБ с

последующим распространением по иерархии от главного узла к его подчиненным. Таким образом, механизм управления РИБ обеспечивает наличие во всех узлах одной и той же конфигурации.

Изменение данных допускается в любом узле распределенной информационной базы. Синхронизация данных достигается путем распространения изменений данных, произведенных в одном узле, во все структуры РИБ.

3.2.1. Планы обмена

В управлении распределенными информационными базами планы обмена занимают центральное место. И у плана обмена должно быть установлено свойство «Распределенная информационная база» для того, чтобы он оказался пригоден для организации РИБ. Содержимое сообщений, которые передаются между узлами РИБ, должно быть регламентированным, принятым для распределенной информационной базы протоколом обмена.

Составом плана обмена определяется перечень данных, с которыми производиться обмен в рамках распределенной информационной базы. Вхождение объекта метаданных в состав плана обмена показывает, что изменения данных, соответствующих объекту метаданных, могут регистрироваться для узлов данного плана обмена. В отличие от универсальных механизмов обмена данными, номенклатура данных, обмен которыми может производиться в рамках распределенной информационной базы, строго ограничена составом соответствующего плана обмена.

Для регистрации изменений данных в распределенной информационной базе задействована служба регистрации изменений. Элементы данных помещаются в сообщение с использованием механизмов XML-сериализация. Помимо изменений данных, между узлами распределенной информационной базы передаются изменения конфигурации, а также некоторая дополнительная служебная информация. Регистрация изменений конфигурации и их передача

в распределенной информационной базе осуществляются полностью автоматически и недоступны для пользователя и разработчика конфигураций.

Формирование и прием сообщения обмена данными в РИБ производятся «в одно действие», то есть все содержимое сообщения формируется путем вызова одного метода встроенного языка. Аналогично и считывание содержимого сообщения производится путем вызова одного метода. Для того чтобы управлять составом данных, помещаемых в сообщение, а также считываемых из сообщения и помещаемых в базу данных, на уровне отдельных элементов, в модуле плана обмена могут быть определены обработчики событий:

- ПриОтправкеДанныхПодчиненному,
- ПриОтправкеДанныхГлавному,
- ПриПолученииДанныхОтПодчиненного,
- ПриПолученииДанныхОтГлавного.

3.2.2. Решение коллизий

На основе отношения «главный - подчиненный» в РИБ организована типовая процедура разрешения коллизий, автоматически выполняемая при приеме сообщения. Принято считать, что произведенное в главном узле изменение элемента данных, имеет высокий приоритет по отношению к изменению в подчиненном узле. В том случае, когда сообщение от подчиненного узла, содержит в себе элемент данных, изменения которого зарегистрированы для этого узла, то никаких действий не предпринимается. То есть запись регистрации изменений не будет удалена и этот элемент данных не будет помещен в БД.

Если пришедшее от главного узла сообщение, содержит в себе элемент данных, изменения которого зарегистрированы, то запись регистрации изменения будет удалена, а элемент данных будет записан в БД.

3.2.3. Сообщение обмена данными в РИБ

Сообщения обмена используются для передачи изменений данных и конфигурации в РИБ которые предоставляются инфраструктурой сообщений. Если в случае РИБ структура и состав данных, помещаемых в тело сообщения, четко определены, то в случае применения универсальных механизмов обмена данными разработчик конфигурации сам определяет, что и как помещается в тело сообщения.

4. КОНФИГУРАЦИЯ 1С: КОНВЕРТАЦИЯ ДАННЫХ

Для решения задачи обмена данными фирмой «1С» выпущена специализированная конфигурация «Конвертация данных». А также выпущены готовые правила переноса данных из однотипных конфигураций, например, из «1С: Бухгалтерии 7.7» в «1С: Бухгалтерию 8», но пользователям нетиповых или изменённых типовых конфигураций придётся создавать правила переноса данных самостоятельно [4].

Перечень решаемых задач:

- Синхронизация справочной информации (создание новых, обновление существующих элементов справочников, удаление, сохранение или изменение иерархии, перенос истории изменения значений периодических реквизитов);
- Синхронизация документов и операций (создание, изменение документов или преобразование одних видов документов в другие, слияние или размножение);
- Создание достаточных начальных условий по учётным регистрам для ведения хозяйственной деятельности (перенос остатков товаров и пр.).
- Структуры хранения данных в «1С: Предприятии» разных версий и/или конфигураций различаются, поэтому перенос данных — это не простое копирование файлов или таблиц, а их преобразование. Чтобы преобразование было однозначным и корректным, для переноса данных необходимо создать и настроить правила.

Создание и настройка правил переноса данных между различными информационными базами возможны, если известна структура хранения данных в базе-источнике и базе-приемнике. Описание структуры метаданных конфигураций должно быть унифицировано. Конфигурация «Конвертация данных» служит для создания и настройки правил переноса данных на основе описаний структуры метаданных конфигурации источника и приемника.

4.1. Структура правил конвертации данных

Для хранения информации о настроенных правилах обмена в конфигурации Конвертация данных предназначены следующие объекты:

- Справочник Конвертации
- Справочник Правила Конвертации объектов
- Правила конвертации свойств
- Правила конвертации значений
- Правила выгрузки данных

Справочник Конвертации – каждый элемент справочника имеет реквизиты Источник и Приемник, которые ссылаются на справочник Конфигурации. Таким образом в настройках конвертации прежде всего определяется между какими конфигурациями осуществляется обмен.

Справочник Правила конвертации объектов (ПКО) подчинен справочнику конвертации. Основное назначение правил конвертации объектов – сопоставление объектов источника объектам приемника определенного типа. В качестве источника данных может выступать как прикладной объект конфигурации источника, так и произвольные данные. Таким образом основными реквизитами правила конвертации объектов являются реквизиты Объект – Источник и Объект- Приемник.

Справочник Правила конвертации свойств (ПКС) подчинен справочнику Правила конвертации объектов. Правила конвертации свойств предназначены для сопоставления реквизитов объекта – источника реквизитам объекта приемника. В качестве источника данных может выступать свойство объекта источника или произвольные данные.

Правила конвертации значений предназначены для конвертации предопределенных значений объектов: справочников, перечислений, планов видов характеристик, план счетов, план видов расчета.

Правила выгрузки данных. Любые правила обмена должны содержать правила выгрузки данных, поскольку именно эти правила определяют какая именно информация должна быть выгружена. Каждое правило выгрузки данных определяет перечень объектов источника подлежащих выгрузке и передает эти правила конвертации объектов.

4.2. Обработчики событий

Механизм обработчиков событий является одним из ключевых в технологии обмена данными. Грамотное и умелое использование механизма позволяет решать практически любые задачи по преобразованию данных. С помощью технологии обработчиков легко реализуется отбор данных, преобразование данных разных типов, сложные выборки, настройка параметров и многие другие задачи.

4.2.1. Обработчики "Правил конвертации объектов":

Перед выгрузкой. Событие выполняется перед выгрузкой каждого объекта в файл обмена, не зависимо от того как выгружается объект по правилу выгрузки данных либо потому что на него есть ссылки. Событие вызывается, когда узел объекта приемника еще не создан и недоступен. Возможен отказ от выгрузки, например, в случае невыполнения каких-либо условий.

При выгрузке. Событие выполняется при выгрузке каждого объекта в файл обмена, не зависимо от того как выгружается объект по правилу выгрузки данных либо потому что на него есть ссылки. Событие вызывается, когда узел объекта приемника уже создан и доступен для изменения.

После выгрузки. Событие выполняется после выгрузки объекта в промежуточную структуру данных, но до выгрузки его непосредственно в файл обмена. Возможен отказ от записи объекта в файл. Может быть

использован для выгрузки дополнительной информации по выгруженному объекту в файл обмена.

После выгрузки в файл. Событие выполняется после выгрузки объекта в файл обмена. Может быть использован для выгрузки дополнительной информации по выгруженному объекту в файл обмена. При этом информация будет записана в файл обмена после выгруженного объекта.

Поля поиска. Событие выполняется при поиске элемента ссылочного типа. Если установлен поиск по уникальному идентификатору и программа нашла элемент, то поиск прекращается. Если поиск по уникальному идентификатору не дал положительного результата и указано, что нужно продолжить поиск в этом случае или поиск по уникальному идентификатору не проводился, то программа пытается найти элементы по свойствам поиска. В обработчике нужно установить список полей через запятую по которым нужно проводить поиск. Если очередная попытка дала положительный результат, то поиск прекращается. Поиск возможен только по тем полям у которых на этапе выгрузка был установлен флаг поиска данных.

Перед загрузкой. Событие выполняется перед загрузкой объекта. Возможен отказ от загрузки, например, в случае невыполнения каких-либо условий. Загружаемый в БД объект еще не инициализирован.

При загрузке. Событие выполняется после попытки идентификации. В случае успешной синхронизации доступен найденный объект. В противном случае его значение «Неопределено». Можно произвести произвольную инициализацию, заполняемого данными объекта.

После загрузки. Событие выполняется после прочтения и установки атрибутов объекта из файла, но до его записи в информационную базу. Возможна модификация загруженного объекта.

4.2.2. Обработчики "правила выгрузки данных":

Перед обработкой. Событие выполняется перед обработкой каждого правила выгрузки один раз. Используется для установки дополнительных параметров перед выгрузкой данных, для переопределения выборки объектов, подлежащей выгрузке, либо для выгрузки дополнительной информации.

Перед выгрузкой объекта. Событие выполняется при получении очередного объекта из выборки, до передачи этого объекта правилу конвертации.

После выгрузки объекта. Событие выполняется после выгрузки объекта по правилу конвертации. Может использоваться для выгрузки дополнительной связанной с объектом информации.

После обработки. Событие выполняется после выгрузки всех данных определенному правилу выгрузки. Для каждого правила выгрузки данных выполняется только один раз, если по этому правилу происходила выгрузка данных.

5. РЕАЛИЗАЦИЯ ПРОЕКТА

5.1. Описание процесса обмена данных

Проанализировав выше описанные подходы, шаблоны, методы и технологии интеграции решений, а также рассмотрев инструменты для обмена данными на платформе 1С:Предприятие можно сделать выбор в пользу метода обмена файлами. А использование конфигурации 1С: Предприятие Конвертация данных является одним из наиболее эффективных на данный момент способом решения задач обмена данными на платформе 1с.

Необходимо обеспечить обмен данными между двумя различными информационными системами, которые имеют условные названия Источник и Приемник. Указанные системы имеют разную структуру метаданных конфигурации.

Для наглядного описания процесса обмена данными используем методологию функционального моделирования IDEF0. IDEF0 применяется для создания функциональной модели, отображающей структуру системы, а также потоки информации.

На рисунке 5.1 представлена диаграмма обмена данными верхнего уровня A0.

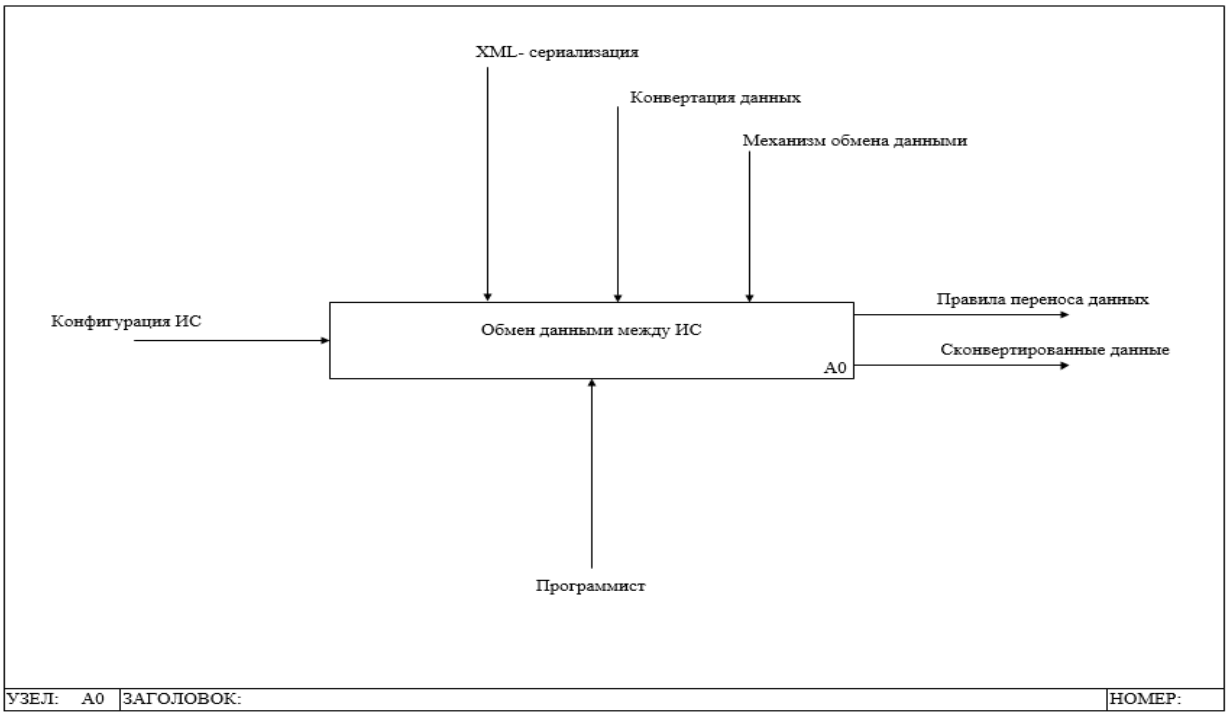

Рисунок. 5.1 Диаграмма обмена данными A0

На второй диаграмме расписан нижний уровень процесса обмена данными, который состоит из следующих блоков:

- Выгрузка и загрузка файлов описания метаданных конфигураций.
- Создание Конфигураций в «Конвертации данных».
- Создание самой конвертации.
- Создание правил конвертации объектов.
- Создание правил конвертации свойств и значений.
- Создание правил выгрузки данных.
- Описание обработчиков событий.
- Процедура выгрузки и загрузки данных из одной конфигурации в другую.

На рисунке 5.2 представлена диаграмма процесса уровня A1.

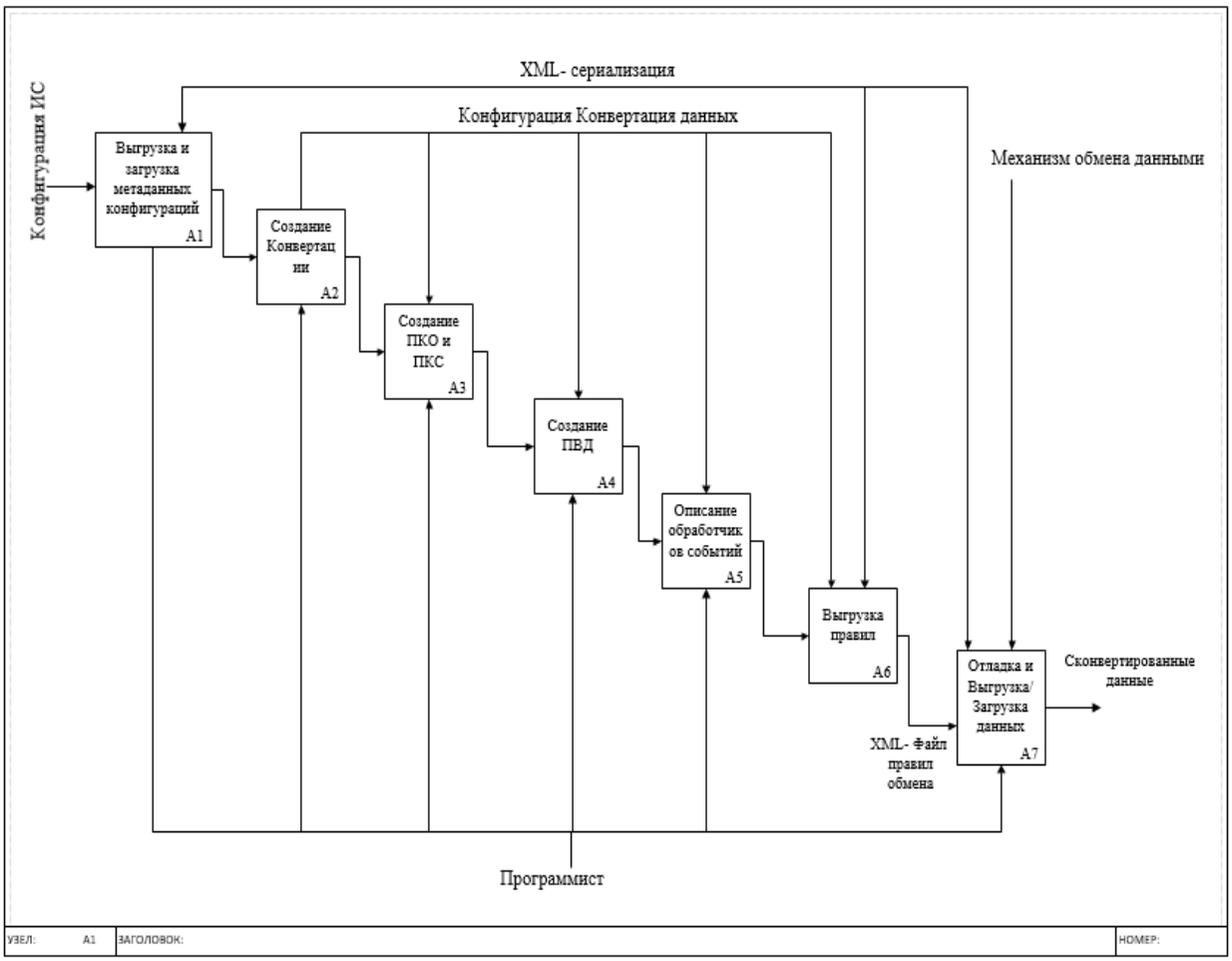

Рисунок 5.2. диаграмма уровня A1.

В результате анализа выявились следующие особенности:

- Структура МД ИС источника отличается от структуры МД ИС назначения.
- Документы имеют абсолютно разное логическое построение.
- Некоторые реквизиты документов составного типа.
- Реализация обмена данными подразумевает написание особых правил переноса данных.

Так как логика работы с клиентами и структура документа источника «Заказы» отличается от структуры документа назначения «Расходная накладная» необходимо в конфигурацию УНФ добавить справочник «кпб_РозничныеПокупатели», а также добавить реквизит самому документу «Расходная накладная» и вынести его на форму.

Структура справочника «кпб_РозничныеПокупатели» и документа «Расходная накладная».

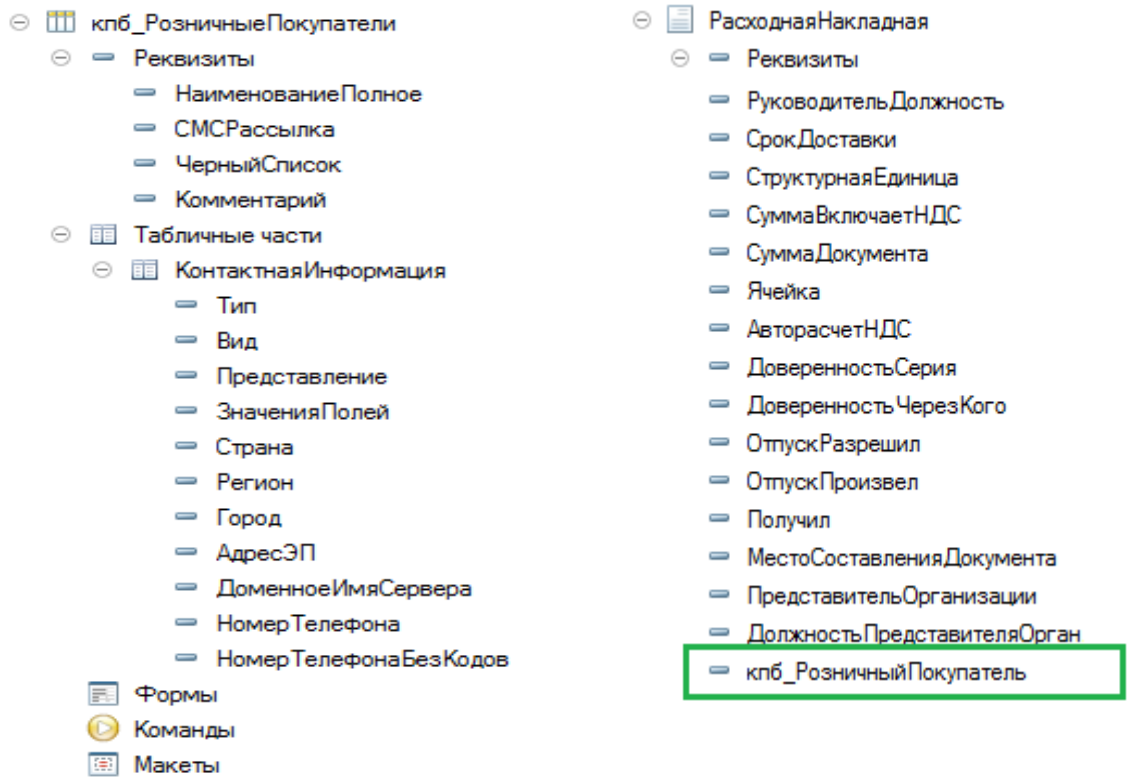

Рисунок 5.3 Структура данных справочника «кпб_Рознчные покупатели» и документа «Расходная накладная».

5.1.1. Создание файлов описания метаданных.

Для создания файлов описания метаданных конфигураций воспользуемся обработкой MD82exp.epf. Пример кода структуры метаданных Источника и Приемника представлен в Приложении А

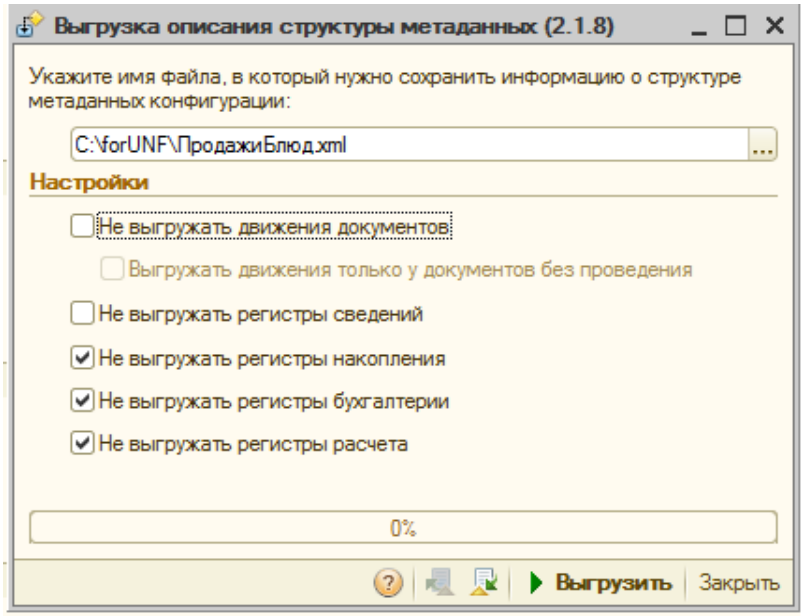

Рисунок 5.4. Выгрузка структуры МД конфигурации Источника.

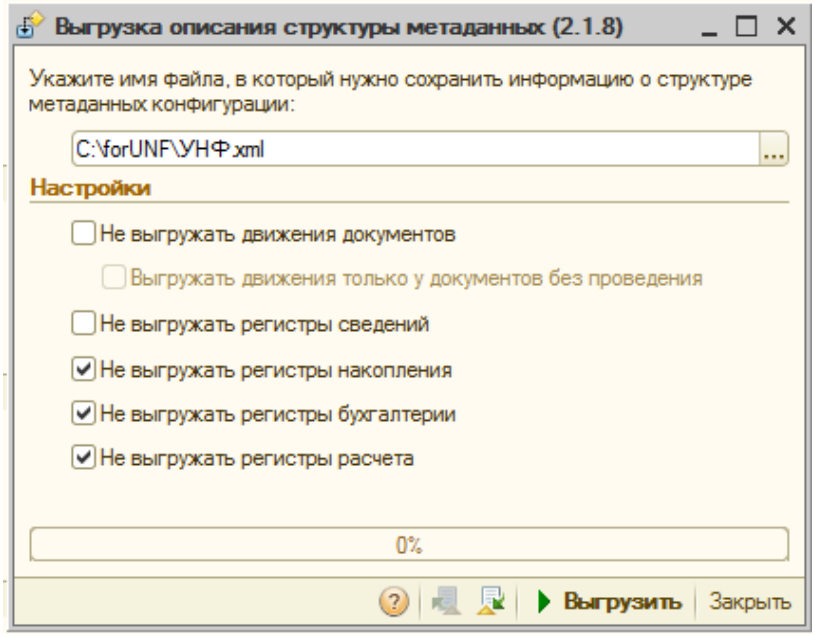

Рисунок 5.5. Выгрузка структуры МД конфигурации Приемника.

5.1.2. Создание конфигураций в конвертации данных

Добавление полученной структуры метаданных конфигураций производится встроенными средствами конфигурации «Конвертация данных».

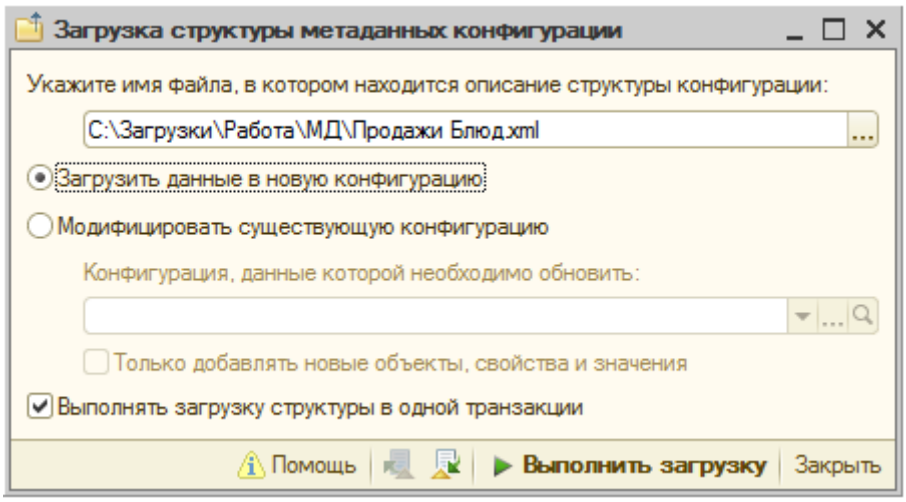

Рисунок 5.6. Загрузка структуры МД Источника в конфигурацию Конвертация данных.

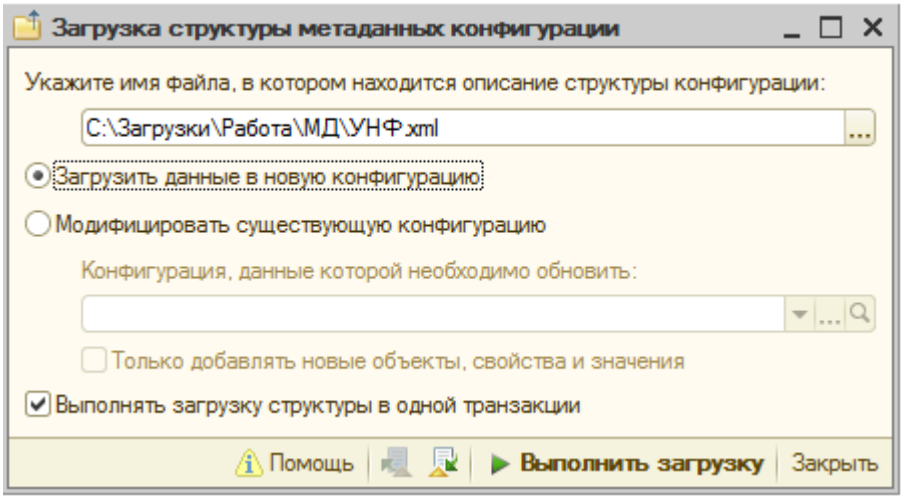

Рисунок 5.7. Загрузка структуры МД Приемника в конфигурацию Конвертация данных

В итоге после загрузки метаданных источника и приемника получаем две иерархические структуры практически идентичные дереву исходных конфигураций.

5.1.3. Создание конвертации

Создаем конвертацию. Используем ранее загруженные конфигурации. Источником является конфигурация «Продажи Блюд», Приемником – конфигурация «Управление небольшой фирмой».

5.2. Создание правил конвертации объектов и свойств.

В результате анализа конфигураций источника и назначения, для конвертации данных были определенны следующие объекты:

Таблица 1

Объект МД	Источник	Приемник
Справочник	Номенклатура	Номенклатура
Справочник	Клиенты	кпб_РозничныеПокупатели
Справочник	ЕденицыИзмерения	ЕденицыИзмерения
Справочник	Склады	СтруктурныеЕденицы
Документ	Заказы	Расходная Накладная
Документ	Перемещения	ПеремещениеЗапасов
Документ	АктСписания	СписаниеЗапасов
Документ	ПриходнаяНакладная	ПриходнаяНакладная

Правила конвертации объектов

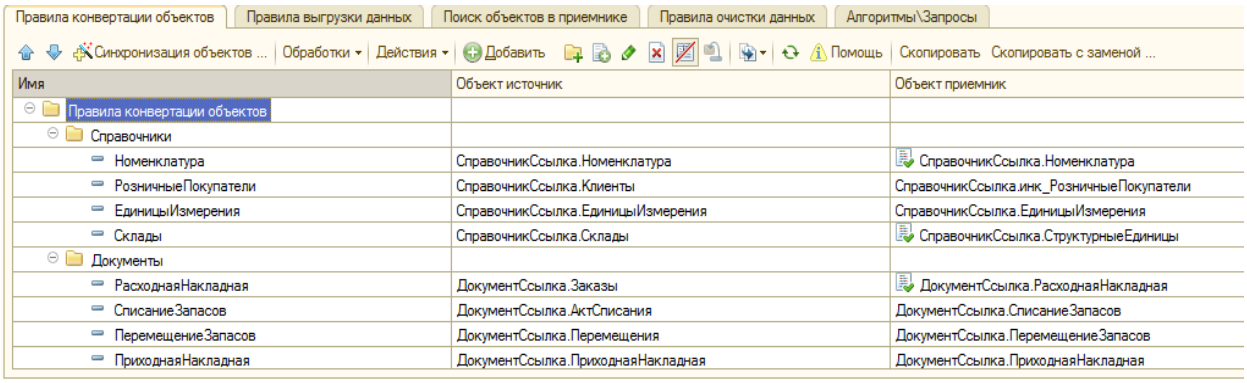

Рисунок 5.8. Правила конвертации объектов

Справочник "Номенклатура" следует перенести в УНФ, как справочник "Номенклатура", добавив к коду префикс "Н". Если родитель у группы номенклатуры в Источнике не задан, то загрузить в группу "Номенклатура".

Документ «Заказы» следует перенести в УНФ, как документ «РасходнаяНакладная». Если Цена документа «Заказы» равна нулю, тогда для каждой позиции документа скидка равна 100%.

Далее необходимо для каждого ПКО задать свойство конвертируемого объекта.

Пример свойств ПКО документа «Расходная накладная»

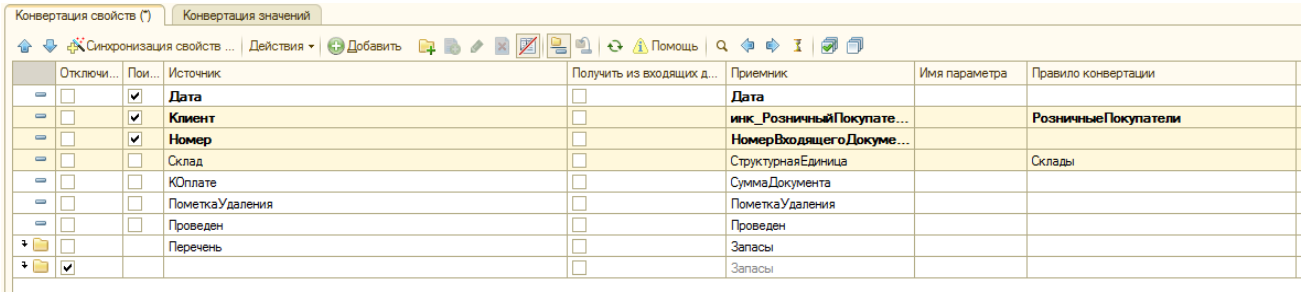

Рисунок 5.9. Свойства для конвертации объектов.

Затем согласно заданию, добавляем код в обработчики событий. Код обработчиков событий представлен в приложении Б

5.3. Разработка механизма обмена данных

Механизм обмена данными разработан используя объекты ЧтениеXML, ЗаписьXML. Первой особенностью обработки является поддержка режима отладки правил конвертации объектов, а второй поддержка выполнения загрузки данных в одной транзакции, что позволяет отменить все выполненные действия в случае ошибки и без каких-либо усилий вернуть начальное состояние базе данных. А также разработан интерфейс формы (представлена на рисунках 5.8) для удобного использования и последующего применения в подобных проектах.

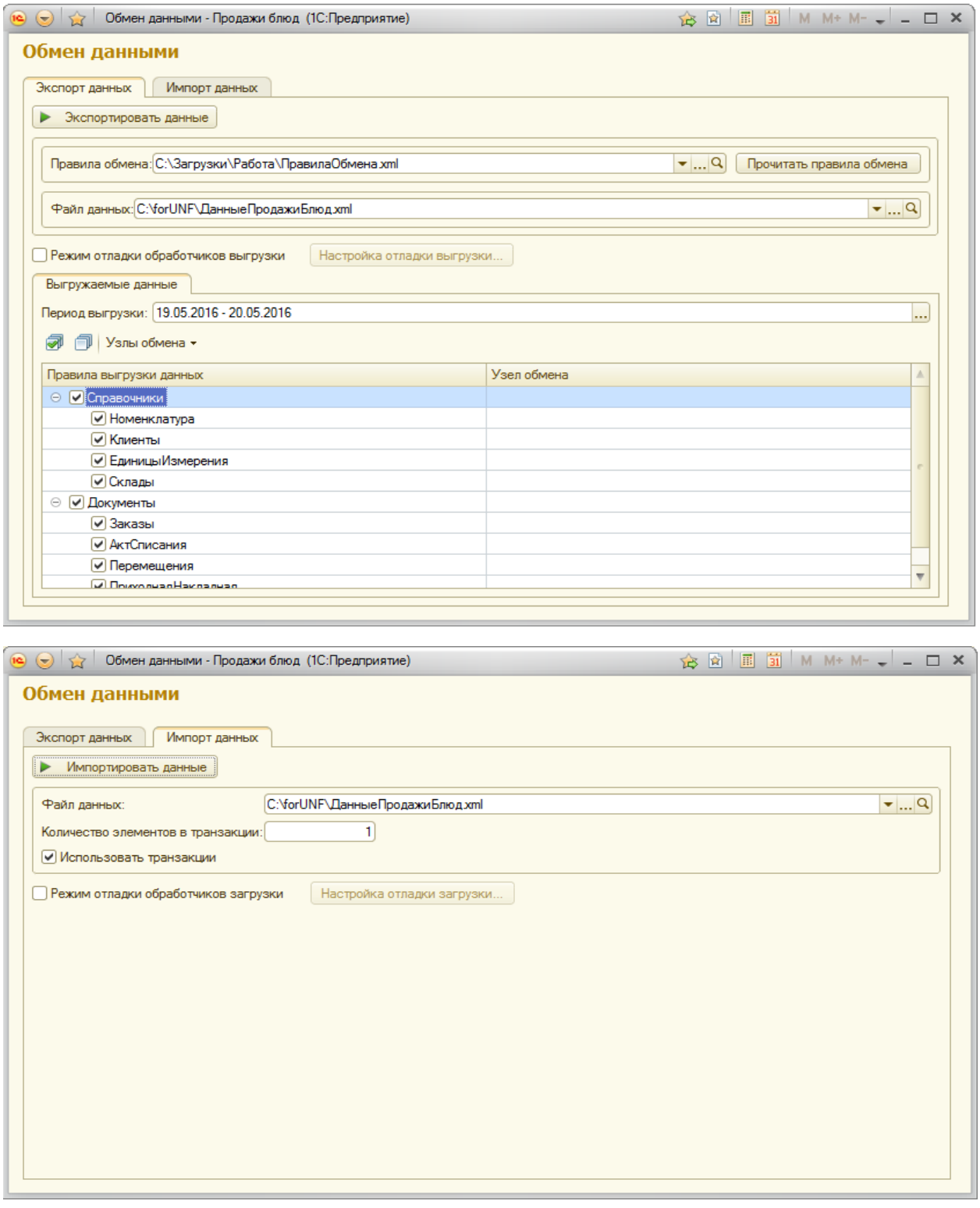

Рисунок 5.10. Форма «Обработки» обмена данными.

5.4. Автоматизация обмена данными

В платформу 1с встроен механизм заданий, он является одним из инструментов администрирования. Включает в себя фоновые и регламентные задания. Фоновые задания позволяют инициализировать выполнения процедур общих модулей асинхронно (без ожидания завершения). Регламентные задания позволяют организовать автоматический вызов процедур общих модулей по расписанию. Основное назначение этого механизма - выполнение административных действий по расписанию.

Основные возможности механизма заданий:

- Определение регламентных процедур на этапе конфигурирования системы;
- Выполнение заданных действий по расписанию;
- Выполнение вызова заданной процедуры или функции асинхронно, т.е. Без ожидания ее завершения;
- Мониторинг хода выполнения заданий;
- Управление заданиями (отмена, блокировка выполнения и др.);
- Возможность ожидания завершения одного или нескольких заданий.

Пример регламентного задания

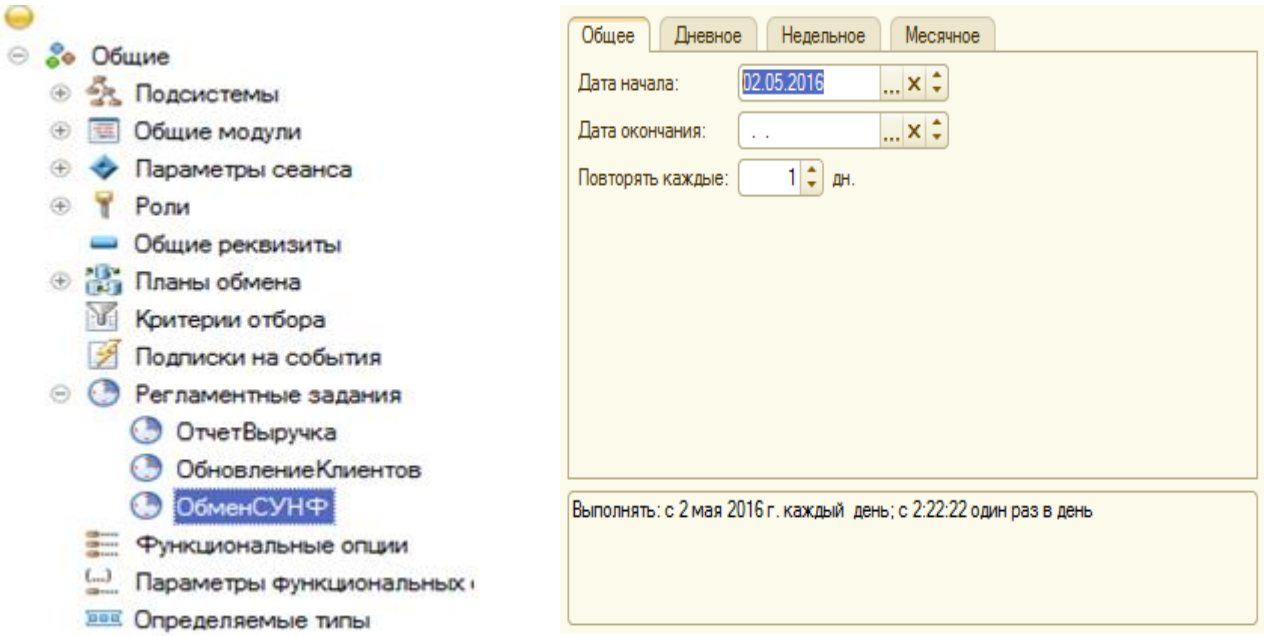

Рисунок 5.11. Регламентное задание «ОбменСУНФ»

По расписанию ежедневно запуститься процедура выгрузки данных из общего модуля «ОбменДанными». Код общего модуля «Обмен данными» представлен в Приложении Б.

5.4.1. Алгоритм общего модуля обмена данными.

Для автоматического обмена данными в информационных системах создан общий модуль «ОбменСУНФ». Код модуля запускается регламентным заданием по определенному расписанию.

Ниже на рисунке 5.12 представлена блок – схема алгоритма модуля «ОбменДанными». Из рисунка 5.12 видно, что для экспорта данных первоначально необходимо задать период и «СчетчикПопыток» (определяет количество повторений подключения к ftp-серверу в случае ошибки). Процедура «ВыполнитьОбменДанными» программно запускает обработку «ОбменДанными», затем читает правила обмена и выгружает данные на диск в виде XML-файла. После чего он архивируется и загружается на ftp-сервер. В случае если ftp-сервер недоступен, попытка соединения повторяется три раза с интервалом 180 секунд, и соответствующая запись попадает в лог-файл, а также отправляется письмом на почтовый ящик ответственного сотрудника. Если соединение с успешно тогда XML-файл архивируется и загружается на ftp-сервер, счетчик попыток при этом сбрасывается. Запись о успешном выполнение обмена подает в лог-файл и также отправляется письмом ответственному сотруднику.

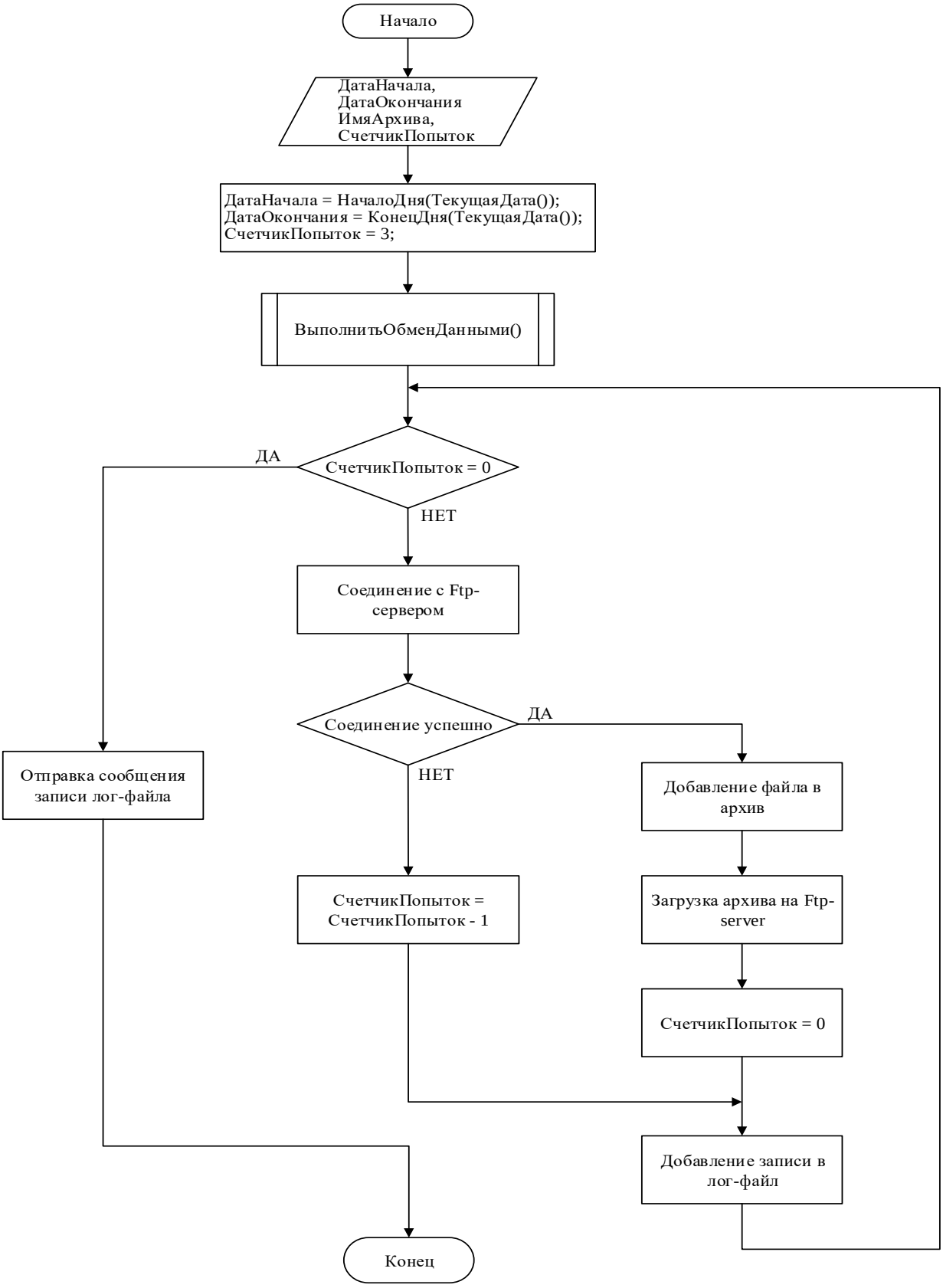

Рисунок. 5.12. Блок-схема алгоритма модуля «ОбменДанными»

ЗАКЛЮЧЕНИЕ

В результате выполнения выпускной квалификационной работы были решены следующие задачи:

- Определён состав объектов для обмена данными между информационными системами.
- Изучены основные методы, технологии и шаблоны интеграции данных.
- Изучен инструмент для написания правил обмена с помощью которого созданы правила переноса данных из одной ИС в другую.
- С помощью средств встроенного языка на платформе 1С разработана обработка обмена данными.
- На основе сформулированных правил автоматизирован процесс обмена данными между ИС, с помощью разработанных программных средств, учитывающих основные особенности передаваемых данных.

Разработанный программный модуль был успешно внедрен и запущен в эксплуатацию. В дальнейшем планируется его техническое сопровождение.

СПИСОК ЛИТЕРАТУРЫ

1. Когаловский М.Р. Методы интеграции данных в информационных системах
2. Гюнтер Зауфер, Мэй Сельваж, Эойн Лейн, Билл Мэтьюс. Шаблоны для информационного сервиса.
3. Гончаров Д.И. Хрусталева Е.Ю. - Технологии интеграции 1С Предприятия 8.2. – 2011.
4. Бояркин В. Э., А. Филатов А. И. - 1С Предприятие 8. Конвертация данных. Обмен данными между прикладными решениями.
5. Белошицкий Д.А., Брешенков А.В. Интеграция данных в информационных системах
6. Кнышова Е. Н. Экономика организации: учебник / Е. Н. Кнышова, Е. Е. Панфилова. – Москва: Форум Инфра-М, 2012. – 334 с.: ил. – Профессиональное образование.
7. Применение сервис ориентированной архитектуры при интеграции систем управления технологическими процессами н.и. ткаченко, н.а. спирин
8. Шульмин В. А. Экономическое обоснование в дипломных проектах: учебное пособие для вузов / В. А. Шульмин, Т. С. Усынина. – Старый Оскол: ТНТ, 2012. – 192 с.
9. Безопасность жизнедеятельности. Под ред. С.В. Белова. – М.: Высшая школа, 2002. – 357с.
10. Гигиенические требования к ВДТ, ПЭВМ и организации работы. Санитарные правила и нормы 2.2.2.542 – 96. – М., 1996
11. Долин П.А. Справочник по технике безопасности. – М.: Высшая школа, 1991.
12. Федеральный закон от 22.07.2008 N 123-ФЗ (ред. от 13.07.2015) "Технический регламент о требованиях пожарной безопасности".

13. Охрана окружающей среды. Под ред. С.В. Белова. – М.: Высшая школа, 1991.

14. Правила устройства электроустановок. Минэнерго СССР, 6-е издание – Энергоатомиздат, 1996. – 640с.

15. Федеральный закон от 22.07.2008 N 123-ФЗ (ред. от 13.07.2015) "Технический регламент о требованиях пожарной безопасности".

16. Гигиенические требования к персональным электронно-вычислительным машинам и организации работы. СанПиН 2.2.2/2.4.1340-03.

17. Гигиенические требования к микроклимату производственных помещений. СанПин 2.2.4.548-96.

18. Федеральный закон от 22.07.2008 N 123-ФЗ (ред. от 13.07.2015) "Технический регламент о требованиях пожарной безопасности".

ПРИЛОЖЕНИЕ А

Документ «Заказы» конфигурации Продажи Блюд

<CatalogObject.Объекты>

<Ref>adb74cd5-d99a-42c8-ae3d-08d0759c565c</Ref>

<IsFolder>false</IsFolder>

<DeletionMark>false</DeletionMark>

<Owner xsi:type="CatalogRef.Конфигурации">f082dbcc-2c7b-4f0a-a9b9-44ce97295d00</Owner>

<Parent>a18e052a-57ca-4ce6-bc2c-98d83664be96</Parent>

<Description>ДокументСсылка.Заказы</Description>

<Имя>Заказы</Имя>

<Синоним>Заказы</Синоним>

<Комментарий/>

<Тип>Документ</Тип>

<Иерархический>false</Иерархический>

<ВидИерархии/>

<ОграничиватьКоличествоУровней>false</ОграничиватьКоличествоУровней>

<КоличествоУровней>0</КоличествоУровней>

<СерииКодов/>

<КонтрольУникальности>true</КонтрольУникальности>

<АвтоНумерация>true</АвтоНумерация>

<Периодичность>Непериодический</Периодичность>

<Подчиненный>false</Подчиненный>

</CatalogObject.Объекты>

Документ «Расходная накладная» конфигурации УНФ

<CatalogObject.Свойства>
<Ref>56eaa474-49a9-4d28-9c51-6da3fa521893</Ref>
<IsFolder>false</IsFolder>
<DeletionMark>false</DeletionMark>
<Owner xsi:type="CatalogRef.Объекты">447e90ee-aac9-469f-8c38-9353cdb22e87</Owner>
<Parent>cbf395c1-11cc-4170-8892-7da23947ef52</Parent>
<Code>0</Code>
<Description>РасходнаяНакладная</Description>
<Синоним>Расходная накладная</Синоним>
<Комментарий/>
<Использование/>
<Индексирование>false</Индексирование>
<КвалификаторыЧисла_Длина>0</КвалификаторыЧисла_Длина>
<КвалификаторыЧисла_Точность>0</КвалификаторыЧисла_Точность>
<КвалификаторыЧисла_Неотрицательное>false</КвалификаторыЧисла_Неотрицательное>
<КвалификаторыСтроки_Длина>0</КвалификаторыСтроки_Длина>
<КвалификаторыСтроки_Фиксированная>false</КвалификаторыСтроки_Фиксированная>
<КвалификаторыДаты_Состав/>
<Авторегистрация>false</Авторегистрация>
<Вид>ЭлементСоставаПланаОбмена</Вид>
<ТипыСтрокой/>
<Типы>
<Row>

<Тип>8163a5c9-49b1-43f9-bfcc-b255a9182b64</Тип>
</Row>
</Типы>
</CatalogObject.Свойства>

ПРИЛОЖЕНИЕ Б

ПКО «Номенклатура» в обработчике событий «Поля поиска»

```
Если СвойстваПоиска["ЭтоГруппа"] = Истина Тогда
    СтрокаИменСвойствПоиска = "Наименование, ЭтоГруппа";
Иначе
    СтрокаИменСвойствПоиска = "Наименование, Родитель, ЭтоГруппа";
КонецЕсли;
```

ПКО «Номенклатура» в обработчике событий «После загрузки»

```
Если Не Объект.ЭтоГруппа Тогда
    Объект.Код = "Н" + Прав(Объект.Код,9);
    Объект.ЕдиницаИзмерения =
    Справочники.КлассификаторЕдиницИзмерения. шт;
    Объект.НоменклатурнаяГруппа =
    Справочники.НоменклатурныеГруппы.ОсновнаяГруппа;
    Объект.ТипНоменклатуры = Перечисления.ТипыНоменклатуры.Запас;
    Объект.СтавкаНДС =
    Справочники.СтавкиНДС.НайтиПоНаименованию ("Без НДС");
    Объект.МетодОценки = Перечисления.МетодОценкиЗапасов.FIFO;
    Объект.СпособПополнения =
    Перечисления.СпособыПополненияЗапасов.Производство;
    Объект.СчетУчетаЗапасов =
    ПланыСчетов.Управленческий.ТоварыПродукция;
    Объект.СчетУчетаЗатрат =
    ПланыСчетов.Управленческий.НезавершенноеПроизводство;
```

```
        Объект.НаправлениеДеятельности =
        Справочники.НаправленияДеятельности.ОсновноеНаправление;

        Объект.Склад = Справочники.СтруктурныеЕдиницы.ОсновнойСклад;

Иначе

Объект.Код = "Н" + Прав(Объект.Код,9);

Если Объект.Родитель.Пустая() Тогда

        Объект.Родитель =
        Справочники.Номенклатура.НайтиПоКоду("ФР-00000001");

КонецЕсли;

КонецЕсли;
```

ПКО «РасходнаяНакладная» в обработчике событий «После загрузки»

```
Объект.ВидОперации =
Перечисления.ВидыОперацийРасходнаяНакладная.ПродажаПокупателю;

Объект.Контрагент = Справочники.Контрагенты.НайтиПоКоду("ФР-
000003");

Объект.Договор = Объект.Контрагент.ДоговорПоУмолчанию;

Объект.Организация = Справочники.Организации.ОсновнаяОрганизация;

Объект.Комментарий = "Заказ №" + Объект.НомерВходящегоДокумента;

Объект.ВидЦен = Справочники.ВидыЦен.Оптовая;

Объект.ВалютаДокумента = Константы.ВалютаУчета.Получить();

Объект.Курс  = 1;

Объект.Кратность  = 1;

Объект.НалогообложениеНДС =
Перечисления.ТипыНалогообложенияНДС.НеОблагаетсяНДС;
```

```
Объект.Подразделение =
Справочники.СтруктурныеЕдиницы.ОсновноеПодразделение;

Если Объект.Запасы.Количество() <> 0 Тогда

    Объект.Запасы[0].Всего = Объект.СуммаДокумента;

КонецЕсли;

Для Каждого СтрокаТЧ из Объект.Запасы Цикл

    СтрокаТЧ.ЕдиницаИзмерения =
    Справочники.КлассификаторЕдиницИзмерения.НайтиПоНаименовани
    ю("шт");

    СтрокаТЧ.СтавкаНДС = СтрокаТЧ.Номенклатура.СтавкаНДС;

    Если СтрокаТЧ.Цена = 0 Тогда

        СтрокаТЧ.Цена = 1;

        СтрокаТЧ.ПроцентСкидкиНаценки = 100;

    КонецЕсли;

КонецЦикла;
```

Общий модуль запуска и контроля выполнения обмена

```
Процедура ВыполнитьОбменДанными()
    ОбработкаОбмена = Обработки.ОбменДанными.Создать();
    ОбработкаОбмена.РежимОбмена = "Экспорт";
    ОбработкаОбмена.ПравилаОбмена = "C:\forUNF\FrontUunfRules.xml";
    ОбработкаОбмена.ФайлДанных = "C:\forUNF\Datafront.xml";
    ОбработкаОбмена.ПрочитатьПравилаОбмена();

    ОбработкаОбмена.ДатаНачала = НачалоДня(ТекущаяДата() -
    60*60*12);
```

```
        ОбработкаОбмена.ДатаОкончания = КонецДня(ТекущаяДата() -
        60*60*12);
        ОбработкаОбмена.ВыполнитьЭкспортДанных();
        АдресФайла = "C:\forUNF\";
        ИмяАрхива = "Datafront_"+Формат(НачалоДня(ТекущаяДата()-
        60*60*12), "ДЛФ=Д")+".zip";
        ЗаписьZip = Новый ЗаписьZipФайла(АдресФайла+ИмяАрхива);
        ЗаписьZip.Добавить(АдресФайла+"Datafront.xml");
        ЗаписьZIP.Записать();

        Сервер = "94.158.151.212";
        Логин = "******";
        Пароль = "******";
        Каталог = "Data";
        АдресФайла = "C:\forUNF\"+ИмяАрхива;
        СчетчикПопыток = 3;

        Пока СчетчикПопыток <> 0 Цикл
            ПодключениеКФТПИЗагрузкаФайла(Логин, Пароль, ИмяАрхива,
            Сервер, Каталог, АдресФайла, СчетчикПопыток);
        КонецЦикла;

КонецПроцедуры

Функция ПодключениеКФТПИЗагрузкаФайла(Логин, Пароль,
ИмяАрхива,Сервер,Каталог,АдресФайла, СчетчикПопыток)

        Попытка
```

```
            FTPСоединение = Новый FTPСоединение(Сервер,,Логин,
Пароль,             , , 180);
            FTPСоединение.УстановитьТекущийКаталог(Каталог);
            FTPСоединение.Записать(АдресФайла, ИмяАрхива);
            СчетчикПопыток=0;

            ЛогФайл = Новый ТекстовыйДокумент;
            ПутьЛогФайла = "C:\forUNF\log.txt";
            ЛогФайл.Прочитать(ПутьЛогФайла);
            ЛогФайл.ДобавитьСтроку(Формат(ТекущаяДата(),"ДЛФ=Д")+"
            Обмен данными успешно завершен");
            ЛогФайл.Записать(ПутьЛогФайла);

            ВсегоСтрок = ЛогФайл.КоличествоСтрок();
            Строка = ЛогФайл.ПолучитьСтроку(ВсегоСтрок);

            ОтправитьПисьмо(ЛогФайл, ПутьЛогФайла, Строка);

        Исключение
            СчетчикПопыток = СчетчикПопыток - 1;
            ЛогФайл = Новый ТекстовыйДокумент;
            ПутьЛогФайла = "C:\forUNF\log.txt";
            ЛогФайл.Прочитать(ПутьЛогФайла);
            ЛогФайл.ДобавитьСтроку(Формат(ТекущаяДата(),"ДЛФ=Д")+"
            Не удалось подключится к FTPсерверу "+ОписаниеОшибки());
            ЛогФайл.Записать(ПутьЛогФайла);

            ВсегоСтрок = ЛогФайл.КоличествоСтрок();
```

```
            Строка = ЛогФайл.ПолучитьСтроку(ВсегоСтрок);

            Если СчетчикПопыток = 0 Тогда
                ОтправитьПисьмо(ЛогФайл, ПутьЛогФайла, Строка);
            КонецЕсли;
        КонецПопытки;

        Возврат СчетчикПопыток;

КонецФункции

&Насервере
Процедура ОтправитьПисьмо(ЛогФайл, ПутьЛогФайла, Строка)

        Профиль = Новый ИнтернетПочтовыйПрофиль;
        Профиль.АдресСервераSMTP = "smtp.mixfood.ru";
        Профиль.ПарольSMTP = "*********";
        Профиль.ПользовательSMTP = "report@mixfood.ru";
        Профиль.ПортSMTP = 25;
        Профиль.АутентификацияSMTP =
СпособSMTPАутентификации.Login;      Профиль.ИспользоватьSSLSMTP =
Истина;

        Письмо = Новый ИнтернетПочтовоеСообщение;
        Письмо.Тема = "Обмен с УНФ";
        Письмо.Тексты.Добавить(Строка);
        Письмо.Отправитель = "report@mixfood.ru";
        Письмо.Получатели.Добавить("a.filippov@mixfood.ru");
```

```
    Почта = Новый ИнтернетПочта;
    Попытка
        Почта.Подключиться(Профиль);
        Почта.Послать(Письмо);
    Исключение
        ЛогФайл.Прочитать(ПутьЛогФайла);
        ЛогФайл.ДобавитьСтроку(Формат(ТекущаяДата(),"ДЛФ=Д")+"
Письмо не отправленно "+ОписаниеОшибки());
        ЛогФайл.Записать(ПутьЛогФайла);
    КонецПопытки;

    Почта.Отключиться();
КонецПроцедуры
```