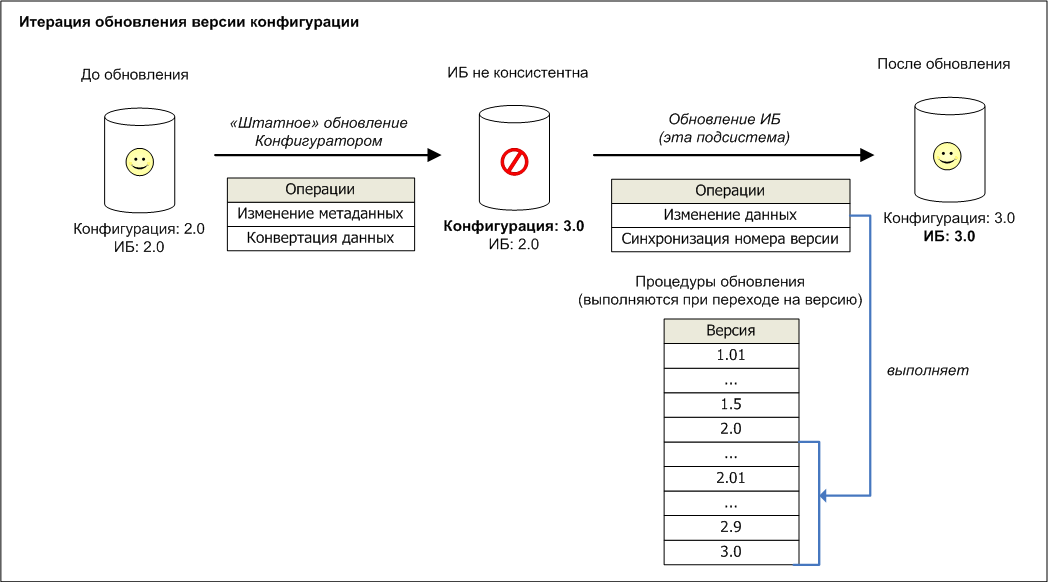
**3.31. Обновление версии ИБ**

Подсистема «Обновление версии ИБ» предоставляет программный интерфейс для выполнения процедур-обработчиков по первоначальному заполнению и обновлению данных информационной базы (ИБ) при изменении версии конфигурации, а также позволяет выводить отчет об изменениях в новой версии конфигурации.

Принципиальная схема обновления конфигурации информационной базы представлена на рисунке.



При запуске клиентского приложения подсистема «Обновление версии ИБ» проверяет, не изменилась ли конфигурация. Если версия конфигурации отличается от версии, сохраненной в информационной базе, то при запуске системы с административными правами выполняется обновление информационной базы. В этом случае, подсистема «Обновление версии ИБ» последовательно выполняет процедуры – обработчики обновления в интервале от номера версии информационной базы до номера версии конфигурации включительно и записывает в информационную базу текущую версию конфигурации.

После обновления информационной базы подсистема «Обновление версии ИБ» выводит форму Описание обновлений, в которой администратор может ознакомиться с описанием изменений в текущей версии конфигурации.

Если у пользователя недостаточно прав (нет прав МонопольныйРежим и Администрирование) или при обновлении происходит ошибка, то запуск системы останавливается.

Если в конфигурации предусмотрены отложенные процедуры – обработчики обновления, то в клиент-серверной версии они запускаются в фоне с помощью регламентного задания Отложенное обновление ИБ, параллельно с началом работы пользователей с новой версией программы. В файловом режиме работы отложенные обработчики выполняются сразу, в основном цикле обновления. Отложенные обработчики обновления могут выполняться одновременно в несколько потоков, сокращая длительность обновления.

**Настройка**

**Подготовка к использованию подсистемы**

Для библиотеки или конечной конфигурации создать отдельный общий модуль с именем ОбновлениеИнформационнойБазы<Сокращение>, где сокращение – короткое имя библиотеки или конфигурации. Пример: общий модуль ОбновлениеИнформационнойБазыБСП.

Затем добавить имя созданного модуля в процедуру ПриДобавленииПодсистем общего модуля ПодсистемыКонфигурацииПереопределяемый.

Из общего модуля ОбновлениеИнформационнойБазыБСП скопировать определения следующих процедур (очистив их содержимое) в созданный общий модуль:

Процедура ПриДобавленииПодсистемы(Описание) Экспорт

КонецПроцедуры

Процедура ПриДобавленииОбработчиковОбновления(Обработчики) Экспорт

КонецПроцедуры

Процедура ПередОбновлениемИнформационнойБазы() Экспорт

КонецПроцедуры

Процедура ПослеОбновленияИнформационнойБазы(Знач ПредыдущаяВерсия, Знач ТекущаяВерсия,

Знач ВыполненныеОбработчики, ВыводитьОписаниеОбновлений, МонопольныйРежим) Экспорт

КонецПроцедуры

Процедура ПриПодготовкеМакетаОписанияОбновлений(Знач Макет) Экспорт

КонецПроцедуры

Для конечных конфигураций также добавить три процедуры:

* ● ПриДобавленииОбработчиковПереходаСДругойПрограммы,
* ● ПриОпределенииРежимаОбновленияДанных,
* ● ПриЗавершенииПереходаСДругойПрограммы:

Процедура ПриДобавленииОбработчиковПереходаСДругойПрограммы(Обработчики) Экспорт

КонецПроцедуры

Процедура ПриОпределенииРежимаОбновленияДанных(РежимОбновленияДанных, СтандартнаяОбработка) Экспорт

КонецПроцедуры

Процедура ПриЗавершенииПереходаСДругойПрограммы(Знач ПредыдущееИмяКонфигурации, Знач ПредыдущаяВерсияКонфигурации, Параметры) Экспорт

КонецПроцедуры

Пример реализации см. в общем модуле \_ДемоОбновлениеИнформационнойБазыБСП демонстрационной конфигурации.

Затем в процедуру ПриДобавленииПодсистемы вписать имя и версию библиотеки или конфигурации, а также зависимости от других библиотек (если предусмотрены). По указанным зависимостям вычисляется порядок вызова обработчиков обновления данных библиотек. Если зависимости не указаны, порядок вызова обработчиков библиотек определяется порядком добавления имен модулей библиотек в общем модуле ПодсистемыКонфигурацииПереопределяемый. При этом библиотека с именем СтандартныеПодсистемы будет вызываться первой, а библиотека у которой имя совпадает со значением свойства Метаданные.Имя всегда будет вызываться последней.

// См. описание этой же процедуры в модуле ОбновлениеИнформационнойБазыБСП.

Процедура ПриДобавленииПодсистемы(Описание) Экспорт

Описание.Имя = "БиблиотекаСтандартныхПодсистемДемо";

Описание.Версия = "2.1.3.24";

// Требуется библиотека стандартных подсистем.

Описание.ТребуемыеПодсистемы.Добавить("СтандартныеПодсистемы");

КонецПроцедуры

При этом в коде процедуры ПриДобавленииПодсистемы не следует получать имя и версию напрямую из свойств конфигурации Метаданные.Имя и Метаданные.Версия. В противном случае при доработке конфигураций потребуется снимать с поддержки и вносить изменения в модуль обновления поставщика.

**Размещение в командном интерфейсе**

Если в конфигурации не используется подсистема «Настройки программы», то в командном интерфейсе администратора необходимо разместить следующие объекты метаданных:

* ● константа ДетализироватьОбновлениеИБВЖурналеРегистрации,
* ● общая форма ОписаниеИзмененийПрограммы,
* ● обработка РезультатыОбновленияПрограммы.

См. пример размещения в демонстрационной конфигурации в группе Обновление версии программы формы ПоискИУстановкаОбновлений обработки ПанельАдминистрированияБСП.

**Создать общий макет ОписаниеИзмененийСистемы**

Описание изменений в новой версии готовится разработчиками конфигурации к выпуску каждой версии конфигурации в общем табличном макете ОписаниеИзмененийСистемы. Табличный макет может включать в себя столько разделов, сколько версий содержится в истории версий продукта.

Каждый такой раздел состоит из двух областей:

* ● В область ШапкаР\_П\_В\_С помещается текст заголовка Новое в версии Р.П.В.С.
* ● Область ВерсияР\_П\_В\_С содержит описание изменений данной версии; может состоять из подразделов или содержать гиперссылки.

Разделы между собой отделяются отступом, который определяется областью Отступ.

Для специальной обработки нажатий на гиперссылки, содержащихся в тексте макета, предназначена процедура-обработчик ПриНажатииНаГиперссылкуВДокументеОписанияОбновлений общего модуля ОбновлениеИнформационнойБазыКлиентПереопределяемый.

Пример заполнения макета можно посмотреть в демонстрационной конфигурации.

Важно!

При первом внедрении макет ОписаниеИзмененийСистемы следует создать до выполнения первого запуска конфигурации в режиме 1С:Предприятие.

**Настройка прав доступа пользователей**

Для настройки прав доступа пользователей к данным и функциям подсистемы «Обновление версии ИБ» следует использовать роли, указанные ниже.

Таблица 3.67.

| № | Роли и их назначение |
| --- | --- |
| 1. | АдминистраторСистемы (из подсистемы «Базовая функциональность»)  Запуск программы для обновления версии ИБ |
| 2. | БазовыеПраваБСП (из подсистемы «Базовая функциональность»)  Просмотр описания изменения системы |

**Настройка обмена данными**

Для настройки обмена данными следует руководствоваться общими правилами.

В планы обмена распределенной информационной базы (РИБ) рекомендуется включать все объекты метаданных подсистемы, за исключением:

* ● константа КоличествоПотоковОбновленияИнформационнойБазы.

В планах обмена распределенной информационной базы (РИБ) рекомендуется отключать регистрацию изменений для следующих объектов метаданных подсистемы (см. также раздел «[Особенности создания начального образа подчиненного узла распределенной ИБ](https://its.1c.ru/db/content/bsp301doc/src/3.30.%20обмен%20данными.htm?_=1542292879#_особенности_создания_начального)»):

* ● константа ДетализироватьОбновлениеИБВЖурналеРегистрации,
* ● константа СведенияОбОбновленииИБ,
* ● константа ОтложенноеОбновлениеЗавершеноУспешно,
* ● константа СведенияОБлокируемыхОбъектах,
* ● регистр сведений ВерсииПодсистем.

Из остальных планов обмена (не РИБ) рекомендуется исключать следующие объекты метаданных:

* ● константа ДетализироватьОбновлениеИБВЖурналеРегистрации,
* ● константа КоличествоПотоковОбновленияИнформационнойБазы,
* ● константа СведенияОбОбновленииИБ,
* ● регистр сведений ВерсииПодсистем.

**Использование при разработке конфигурации**

**Разработка обработчиков обновления**

Для подключения своих обработчиков обновления необходимо в модуль обновления информационной базы библиотеки или конфигурации, созданный на предыдущем шаге, добавить описание обработчиков обновления в процедуру ПриДобавленииОбработчиковОбновления.

Для каждого обработчика обновления нужно добавить фрагмент кода по шаблону:

Обработчик = Обработчики.Добавить();

Обработчик.Версия = "<номер версии>";

Обработчик.Процедура = "<полное имя экспортной процедуры>";

Обработчик.НачальноеЗаполнение = {Истина|Ложь};

Обработчик.РежимВыполнения = {"Монопольно"|"Оперативно"|"Отложенно"};

Строка таблицы значений Обработчик имеет различный состав полей, который зависит от вида обработчика обновления: монопольный, оперативный или отложенный. Свойства, общие для всех видов обработчиков обновления:

* ● Версия (Строка) – номер версии конфигурации, при обновлении на которую должна быть вызвана процедура обновления, указанная в параметре Процедура.
* ● номер версии конфигурации указывается в формате Р.П.В.С (Р – старший номер редакции; П – младший номер редакции; В – номер версии; С – номер сборки);
* ● если в качестве версии указан символ «\*», то обработчик обновления должен выполняться каждый раз при обновлении информационной базы, независимо от номера версии конфигурации;
* ● если свойство Версия не задано, то должно быть установлено в Истина свойство НачальноеЗаполнение (см. далее).
* ● Процедура (Строка) – имя процедуры обновления. Процедура обновления должна располагаться в серверном общем модуле и должна быть экспортной. Например, ОбновлениеИБ.ПерейтиНаВерсию\_1\_2\_3\_4.
* ● НачальноеЗаполнение (Булево) – если Истина, то процедура обновления будет вызвана при первом запуске на пустой информационной базе (версия «0.0.0.0»), созданной из файла поставки конфигурации и не содержащей данных. Это обработчики первоначального заполнения базы. По умолчанию – Ложь.
* ● РежимВыполнения (Строка) – принимает одно из значений: Монопольно, Оперативно и Отложенно. Если свойство не задано, то обработчик – монопольный.

Например, для выполнения двух экспортных процедур ВыполнятьВсегдаПриСменеВерсии и ПерейтиНаВерсию\_1\_0\_0\_5 общего модуля \_ДемоОбновлениеИнформационнойБазыБСП при переходе с версии 1.0.0.1 на версию 1.0.0.5 необходимо разместить следующий фрагмент кода в процедуре ПриДобавленииОбработчиковОбновления:

Процедура ПриДобавленииОбработчиковОбновления(Обработчики) Экспорт

// Подключаются процедуры - обработчики обновления конфигурации.

Обработчик = Обработчики.Добавить();

Обработчик.Версия = "1.0.0.0";

Обработчик.Процедура = "\_ДемоОбновлениеИнформационнойБазыБСП.ПерейтиНаВерсию\_1\_0\_0\_0";

Обработчик = Обработчики.Добавить();

Обработчик.Версия = "1.0.0.5";

Обработчик.Процедура = "\_ДемоОбновлениеИнформационнойБазыБСП.ПерейтиНаВерсию\_1\_0\_0\_5";

Обработчик = Обработчики.Добавить();

Обработчик.Версия = "\*";

Обработчик.Процедура = "\_ДемоОбновлениеИнформационнойБазыБСП.ВыполнятьВсегдаПриСменеВерсии";

КонецПроцедуры

**Как выбрать способ обработки данных: монопольный, оперативный или отложенный**

В зависимости от значения свойства РежимВыполнения можно выбрать наиболее эффективный способ обработки данных.

1. Монопольно – если обработчик обновления необходимо выполнять монопольно, в условиях отсутствия активных сеансов работы пользователей, регламентных заданий, внешних соединений и подключений по веб-сервисам. В противном случае обновление версии программы прерывается.

Монопольные обработчики предназначены для обновления тех данных, обработка которых должна быть обязательно завершена к моменту входа пользователей в программу. Для сокращения времени простоя (ожидания обработки данных), рекомендуется большие объемы данных обновлять отложено (см. ниже). Примеры монопольных обработчиков: обработка небольшого объема данных текущего периода, активных позиций номенклатуры и т. п.

Если хотя бы один обработчик обновления конфигурации – монопольный, то все оперативные обработчики (см. далее) выполняются в монопольном режиме.

2. Оперативно – если обработчик обновления необходимо выполнять не монопольно: при активных сеансах работы пользователей, регламентных заданий, внешних соединений и подключений через веб-сервисы.

Оперативные обработчики следует применять в редких случаях, когда важно сократить время ожидания пользователей на обновление информационной базы – например, для обработки неразделенных данных в модели сервиса, инициализации настроек пользователей и т. п. Как правило, их следует применять только при выпуске исправительных релизов.

Подробнее об оперативном обновлении см. раздел «[Оперативное обновление на исправительные релизы конфигураций](https://its.1c.ru/db/content/bsp301doc/src/3.31.%20%D0%BE%D0%B1%D0%BD%D0%BE%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5%20%D0%B2%D0%B5%D1%80%D1%81%D0%B8%D0%B8%20%D0%B8%D0%B1.htm?_=1542292879#_оперативное_обновление_на)».

3. Отложенно – если обработчик обновления необходимо выполнять в фоне после того, как завершено выполнение монопольных (оперативных) обработчиков, и пользователям уже разрешен вход в программу.

Отложенные обработчики предназначены для обработки той части данных ИБ, которые не препятствуют пользователям начинать свою работу с новой версией программы, не дожидаясь завершения обработки этих данных. Примеры отложенных обработчиков: обработка больших архивов данных за закрытые/прошлые периоды, неактивных позиций номенклатуры, различных данных, отключенных в данный момент функциональными опциями и т. п.

Подробнее об отложенной обработке данных см. раздел «[Отложенное обновление больших архивов данных](https://its.1c.ru/db/content/bsp301doc/src/3.31.%20%D0%BE%D0%B1%D0%BD%D0%BE%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5%20%D0%B2%D0%B5%D1%80%D1%81%D0%B8%D0%B8%20%D0%B8%D0%B1.htm?_=1542292879#_отложенное_обновление_больших)».

**Общие рекомендации по реализации обработчиков обновления**

* ● Код обработчиков обновления выполняется на сервере, поэтому они не должны содержать никакой логики по интерактивному взаимодействию с пользователем.
* ● В случае критической ошибки при обновлении в коде обработчика необходимо вызвать исключение, которое приведет к остановке всей процедуры обновления. Остановка обновления информационной базы приведет к невозможности запуска до тех пор, пока причины ошибки не будут устранены.
* ● Код обработчика обновления должен быть рассчитан на неоднократное выполнение, чтобы его повторное выполнение не приводило, например, к дублированию данных в информационной базе.
* ● На одну и ту же версию конфигурации может быть написано сколь угодно много обработчиков. Например, это могут быть обработчики от разных подсистем. При этом порядок их вызова для одной версии может быть произвольным.
* ● Порядок вызова обработчиков обновления в пределах одной версии является случайным, т. е. нельзя ставить в зависимость работоспособность одного обработчика обновления от выполнения другого обработчика. Если подобные зависимости появляются, то такие обработчики необходимо объединять в один.

Кроме того, обработчик обновления не должен содержать лишних, избыточных действий с данными – должен выполняться максимально быстро. Для этого в большинстве случаев необходимо отключать бизнес-логику при обработке данных, а также отключать регистрацию изменений на узлах планов обмена, чтобы избежать отправки всего объема обработанных данных во всех узлы. Таким образом:

* ● в распределенной информационной базе (РИБ) обработка данных должна выполняться независимо в каждом из узлов;
* ● при обмене между произвольными конфигурациями (программами) обработка данных не должна приводить к их выгрузке в базы-получатели.

Исключение составляют случаи создания ссылочных объектов, которые должны быть перенесены механизмами обмена данными в другие узлы РИБ с тем же значением реквизита Ссылка.

Таким образом, в коде обработчика обновления вместо кода вида:

ДокументОбъект.Записать();

должно быть:

ДокументОбъект.ОбменДанными.Загрузка = Истина; // отключить всю бизнес-логику при записи

ДокументОбъект.ДополнительныеСвойства.Вставить("ОтключитьМеханизмРегистрацииОбъектов");

ДокументОбъект.ОбменДанными.Получатели.АвтоЗаполнение = Ложь;

ДокументОбъект.Записать();

Для сокращения объема кода рекомендуется использовать процедуру ЗаписатьДанные общего модуля ОбновлениеИнформационнойБазы:

ОбновлениеИнформационнойБазы.ЗаписатьДанные(ДокументОбъект);

Для удобной отладки монопольных и оперативных обработчиков обновления можно воспользоваться параметром запуска РежимОтладки (обновление будет выполнено без использования фоновых заданий) или флажком Выполнять обновление версии ИБ без установки монопольного режима и без фонового задания (режим отладки) в обработке ОбновлениеВерсииИБ.

**Отложенное обновление больших архивов данных**

В тех случаях, когда исчерпаны все остальные средства по оптимизации обработчиков обновления и можно выделить некоторые действия по обработке данных, выполнение которых не требуется обязательно для начала работы пользователей с программой, рекомендуется перенести эту обработку на более поздний момент времени и выполнять ее отложенно.

Отложенная обработка данных не блокирует вход пользователей в программу и позволяет избежать ситуаций, когда обновление больших баз занимает существенное время (сутки и более), что нарушает график работы компании (из-за большого времени простоя информационной системы).

Важно!

Отложенная обработка данных возможна только в клиент-серверном варианте работы. В файловом режиме работы отложенные обработчики обновления выполняются сразу, до начала работы пользователей с новой версией программы.

Рекомендуется реализовать отложенные обработчики обновления для обработки больших архивов данных за закрытые/прошлые периоды, неактивных позиций номенклатуры, закрытых договоров, различных данных, отключенных в данный момент функциональными опциями и т. п. В большинстве случаев отложенно следует обновлять документы, регистры, бизнес-процессы и задачи, которые имеют тенденцию накапливаться со временем.

Механизм отложенной обработки данных имеет два режима выполнения, которые настраиваются отдельно для каждой библиотеки и основной конфигурации:

* ● Последовательно (по умолчанию) – отложенные обработчики обновления выполняются последовательно в интервале от номера версии информационной базы до номера версии конфигурации включительно (по возрастанию номеров версий, которые указаны в обработчиках). До тех пор пока один из обработчиков не завершил обработку своей порции данных, следующий не запускается. Данный режим обновления подходит для конфигураций (и библиотек), в которых отложенные обработчики для новых версий обрабатывают те же данные, что обрабатывали обработчики более старых версий. Кроме того, за счет последовательного выполнения к ним предъявляются минимальные требования по «устойчивости» к обрабатываемым данным: при обновлении «через» несколько версий они гарантированно могут рассчитывать на определенное начальное состояние обрабатываемых данных, которое осталось после выполнения обработчиков предыдущей версии.
* ● Параллельно – отложенный обработчик после обработки первой порции данных передает управление следующему обработчику, а после выполнения последнего обработчика цикл повторяется заново, пока все данные не будут обработаны. Таким образом одновременно обновляются объекты информационной базы сразу всех типов, в отличие от последовательного режима, при котором объекты разных типов обрабатываются по очереди и многократно (при обновлении «через» несколько версий).

Разработка отложенных обработчиков обновления для обоих режимов существенно различается. Особенности разработки обработчиков для отложенного режима обновления описаны отдельно.

Для того чтобы указать, что обработчик обновления должен выполняться отложенно, необходимо свойству РежимВыполнения присвоить значение Отложенно, указать уникальный идентификатор и задать комментарий, который кратко поясняет пользователю, какие данные и как он обрабатывает.

* Свойства, специфичные для обоих видов отложенных обработчиков обновления:
* ● Комментарий (Строка) – комментарий заполняется обязательно и не должен совпадать с комментариями к другим обработчиками обновления. В нем рекомендуется описывать не только суть выполняемого действия, но и масштаб временно неработоспособного функционала, например:
* ● Подготовка индекса для поиска отчетов, предусмотренных в программе. Поиск отчетов временно недоступен.
* ● Реструктуризация дополнительных реквизитов и сведений. Рекомендуется воздержаться от их редактирования до завершения обработки.
* ● Первоначальный расчет количества нерассмотренных писем по папкам. До завершения обработки всех писем их количество может выводиться некорректно.
* ● Заполняются движения по новому регистру Движения Номенклатура-Контрагент по документам Расчет себестоимости товаров. После выполнения обработки появится возможность формировать отчеты по товарам.
* ● Идентификатор (УникальныйИдентификатор) – идентификатор отложенного обработчика, который необходимо заполнять для разрешения конфликтов при переименованиях или переносе в другой модуль процедуры обновления. В таких случаях по идентификатору будет определен новый путь к обработчику, и он успешно завершит обработку данных.
* ● БлокируемыеОбъекты (Строка) – полные имена объектов через запятую, которые следует блокировать в пользовательском интерфейсе от редактирования до завершения процедуры обработки данных. Подробнее см. [Блокировка необработанных данных в пользовательском интерфейсе](https://its.1c.ru/db/content/bsp301doc/src/3.31.%20%D0%BE%D0%B1%D0%BD%D0%BE%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5%20%D0%B2%D0%B5%D1%80%D1%81%D0%B8%D0%B8%20%D0%B8%D0%B1.htm?_=1542292879#_блокировка_необработанных_данных).
* ● ПроцедураПроверки (Строка) – имя функции, которая дополнительно для переданного объекта определяет, завершена ли для него процедура обработки данных. Если переданный объект обработан, то следует вернуть значение Истина. Вызывается из процедуры ПроверитьОбъектОбработан общего модуля ОбновлениеИнформационнойБазы. Подробнее см. [Блокировка необработанных данных в пользовательском интерфейсе](https://its.1c.ru/db/content/bsp301doc/src/3.31.%20%D0%BE%D0%B1%D0%BD%D0%BE%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5%20%D0%B2%D0%B5%D1%80%D1%81%D0%B8%D0%B8%20%D0%B8%D0%B1.htm?_=1542292879#_блокировка_еще_необработанных).

Например:

Обработчик = Обработчики.Добавить();

Обработчик.Версия = "1.2.3.4";

Обработчик.Процедура = "Заказы.ЗаполнитьСтатусЗаказовПокупателей";

Обработчик.РежимВыполнения = "Отложенно";

Обработчик.Идентификатор = Новый УникальныйИдентификатор("83d5c5dd-1462-4d72-ab98-f8f5dcc0664d");

Обработчик.Комментарий = НСтр("ru = 'Заполняет значение нового реквизита СтатусЗаказа у документов ""Заказ покупателя"" прошлых периодов.'");

Синтаксис процедуры-обработчика отложенного обновления:

Процедура ЗаполнитьСтатусыЗаказовПокупателей(Параметры) Экспорт

где Параметры – Структура со свойствами:

* ● ОбработкаЗавершена (Булево) – для того чтобы обработчик был вызван повторно для обработки следующей порции данных, следует записать в него значение Ложь;
* ● ПрогрессВыполнения (Структура) – необходимо заполнять для отображения прогресса обработки данных:
* ● ВсегоОбъектов (Число) – общее количество объектов, которое необходимо обработать;
* ● ОбработаноОбъектов (Число) – сколько объектов уже обработано.

Кроме того, в структуру Параметры можно добавить произвольное количество свойств произвольных типов, значения которых будут автоматически запоминаться между вызовами процедуры – обработчика отложенного обновления. Таким образом можно передавать контекст отложенной обработки между ее вызовами (например, дату, по которую был обработан архив документов). При этом не рекомендуется сохранять в свойствах структуры большие объемы данных, а использовать только примитивные типы (например, Дата).

Отложенную обработку данных необходимо выполнять порциями, чтобы не создавать длительную нагрузку на сервер предприятия и СУБД. По умолчанию размер порции – 1000 (документов, записей и т. п.). Размер порции можно увеличить для небольших объектов и уменьшить для документов, в которых большие табличные части (в среднем).

Также рекомендуется начинать обработку с самых свежих данных.

Например:

Запрос = Новый Запрос;

Запрос.Текст =

"ВЫБРАТЬ ПЕРВЫЕ 1000

| ЗаказПокупателя.Ссылка

|ИЗ

| Документ.ЗаказПокупателя КАК ЗаказПокупателя

|ГДЕ

| ЗаказПокупателя.СтатусЗаказа = &ПустаяСсылка

|

|УПОРЯДОЧИТЬ ПО

| ЗаказПокупателя.Дата УБЫВ";

Отложенные обработчики обновления должны самостоятельно заботиться о целостности обновляемых данных: при чтении данных с последующим изменением требуется выполнять эти действия в транзакции и устанавливать исключительную управляемую блокировку.

Например:

НачатьТранзакцию();

Попытка

Блокировка = Новый БлокировкаДанных;

ЭлементБлокировки = Блокировка.Добавить("Документ.\_ДемоЗаказПокупателя");

ЭлементБлокировки.УстановитьЗначение("Ссылка", ЗаказПокупателя.Ссылка);

Блокировка.Заблокировать();

ДокументОбъект = ЗаказПокупателя.Ссылка.ПолучитьОбъект();

// Если объект ранее был удален или обработан другими сеансами, пропускаем его.

Если ДокументОбъект = Неопределено Тогда

ОтменитьТранзакцию();

Возврат;

КонецЕсли;

Если ДокументОбъект.СтатусЗаказа = Перечисления.\_ДемоСтатусыЗаказовПокупателей.ПустаяСсылка() Тогда

ОтменитьТранзакцию();

Возврат;

КонецЕсли;

// Обрабатываем документ

// ...

ОбновлениеИнформационнойБазы.ЗаписатьДанные(ДокументОбъект);

ЗафиксироватьТранзакцию();

Исключение

ОтменитьТранзакцию();

ВызватьИсключение;

КонецПопытки;

При возникновении исключения в обработчике отложенного обновления следует всегда пробрасывать его в вызывающий код (см. блок Исключение – КонецПопытки в примере выше). В противном случае может возникнуть ситуация, когда обработчик будет выполняться бесконечно долго (если параметру ОбработкаЗавершена присвоено значение Ложь), а данные ИБ не будут обновлены.

При возникновении исключения в обработчике он помечается как ошибочный, и через определенный интервал времени выполняются еще две попытки его запуска. Тем самым обработчик сможет корректно обновить данные, временно заблокированные другими сеансами. Но если обработчик продолжает и дальше прерываться с исключением (допущена ошибка в самом обработчике), то он пропускается и ставится в очередь выполнения только при очередном обновлении версии программы. Это позволяет обновить данные при переходе на следующий исправительный релиз, в котором исправлена ошибка в обработчике.

В общем случае при обработке данных конкретной таблицы (документа, регистра и т. п.) некоторая часть ее данных требуется пользователям сразу к моменту начала работы в новой версии программы, а все остальное может быть обработано отложенно. В таких случаях рекомендуется реализовать два обработчика обновления: монопольный и отложенный.

В редких случаях, когда в новой версии конфигурации появился монопольный (или оперативный) обработчик обновления, данные которого обрабатывались в предыдущих версиях отложенно, следует:

* ● пересмотреть проектное решение и сделать такой обработчик отложенным,
* ● либо «старые» отложенные обработчики обновления сделать монопольными (оперативными).

В противном случае возникнет ситуация, когда данные будут обработаны в неправильном порядке: сначала выполнится монопольный (оперативный) обработчик, который рассчитывается на то, что данные были обработаны ранее отложенно.

**Блокировка необработанных данных в пользовательском интерфейсе**

Если отложенный обработчик обрабатывает не только архивные данные, но и данные текущего периода, то для предотвращения некорректной работы программы рекомендуется блокировать еще не обработанные данные от изменения пользователями программы, а также отчеты и обработки, которые могут некорректно работать до завершения отложенного обновления этих данных.

Для этого необходимо:

* ● в свойстве БлокируемыеОбъекты отложенного обработчика обновления следует указать те объекты метаданных, с которыми он работает (читает или записывает), а также связанные с ними отчеты и обработки. Если в конфигурации используется подсистема Варианты отчетов, то отчеты, написанные на базе системы компоновки данных (СКД), в списке блокируемых объектов указывать не требуется - для них автоматически будут определены читаемые объекты и если выводимые в отчете данные содержат не обновленную информацию, то будет показано соответствующее предупреждение;
* ● в свойстве ПроцедураПроверки указать имя экспортной функции, которая дополнительно проверяет, нужно ли блокировать конкретный объект. Например:

Обработчик = Обработчики.Добавить();

Обработчик.Версия = "1.2.3.4";

Обработчик.Идентификатор = Новый УникальныйИдентификатор("b3be66c5-708d-42c8-a019-818036d09d06");

Обработчик.Процедура = "Заказы.ЗаполнитьСтатусыЗаказовПокупателей";

Обработчик.Комментарий = НСтр("ru = 'Заполнение значения нового реквизита ""Статус заказа"" у документов ""Демо: Заказ покупателя"" прошлых периодов.

|До завершения обработки ""Статус заказа"" данных документов будет отображаться некорректно.'");

Обработчик.РежимВыполнения = "Отложенно";

Обработчик.ПроцедураПроверки = "Заказы.ЗаказПокупателяОбработан";

Обработчик.БлокируемыеОбъекты = "Документ.ЗаказПокупателя,Отчет.СтатусыЗаказовПокупателей";

Затем для всех объектов, указанных в свойстве БлокируемыеОбъекты, необходимо добавить вызов процедуры ОбновлениеИнформационнойБазы.ПроверитьОбъектОбработан в обработчике события модуля формы объекта ПриСозданииНаСервере и в обработчике события модуля объекта ПередЗаписью:

Процедура ПриСозданииНаСервере(Отказ, СтандартнаяОбработка)

Если Параметры.Свойство("АвтоТест") Тогда // Возврат при получении формы для анализа.

Возврат;

КонецЕсли;

ОбновлениеИнформационнойБазы.ПроверитьОбъектОбработан(Объект, ЭтотОбъект);

…

КонецПроцедуры

Процедура ПередЗаписью(Отказ, РежимЗаписи, РежимПроведения)

Если ОбменДанными.Загрузка Тогда

Возврат;

КонецЕсли;

ОбновлениеИнформационнойБазы.ПроверитьОбъектОбработан(ЭтотОбъект);

…

КонецПроцедуры

Функция проверки, указанная в свойстве ПроцедураПроверки, возвращает Истина для объектов, если объект уже обработан и с ним могут работать пользователи, или Ложь, если объект следует заблокировать от редактирования пользователями. Она принимает на вход параметр типа Структура, свойства которого идентифицируют запрашиваемый объект:

* ● Данные (ЛюбаяСсылка, НаборЗаписей, Объект, ДанныеФормыСтруктура) – объект, который нужно проверить, обработан ли он данным отложенным обработчиком обновления или еще нет.
* ● МетаданныеОбъекта (ОбъектМетаданных) - объект метаданных, соответствующий параметру Данные.
* ● ПолноеИмя (Строка) - полное имя объекта метаданных.
* ● Отбор (ЛюбаяСсылка, Структура) – если Данные – это ссылочный объект, то значение ссылки; если регистр подчиненный регистратору – значение отбора по регистратору. Если Данные – это независимый регистр сведений, то в этом параметре предается структура, соответствующая установленным отборам по измерениям.
* ● ЭтоНовый (Булево) – если Данные – это ссылочный объект, то признак нового объекта. Для других типов – всегда Ложь.

Ее реализация должна быть достаточно простой, чтобы не сильно замедлять открытие формы, например:

Функция ЗаказПокупателяОбработан(Параметры) Экспорт

Если ТипЗнч(Параметры.Данные) = Тип("ДокументСсылка.\_ДемоЗаказПокупателя") Тогда

СтатусЗаказа = ОбщегоНазначения.ЗначениеРеквизитаОбъекта(Параметры.Данные, "СтатусЗаказа");

Иначе

СтатусЗаказа = Параметры.Данные.СтатусЗаказа;

КонецЕсли;

Возврат СтатусЗаказа <> Перечисления.\_ДемоСтатусыЗаказовПокупателей.ПустаяСсылка();

КонецФункции

Пример отложенного обработчика обновления с блокировкой необработанных данных в пользовательском интерфейсе см. в демонстрационной конфигурации: процедура ПриДобавленииОбработчиковОбновления общего модуля \_ДемоОбновлениеИнформационнойБазыБСП.

**Особенности параллельного режима отложенного обновления**

Для установки параллельного режима выполнения отложенных обработчиков конфигурации (библиотеки) следует в процедуре ПриДобавленииПодсистемы общего модуля ОбновлениеИнформационнойБазы<Сокращение> указать:

Описание.РежимВыполненияОтложенныхОбработчиков = "Параллельно";

Данный режим обновления можно включить с определенной версии, чтобы не пересматривать реализацию старых отложенных обработчиков. Для этого необходимо заполнить свойство ПараллельноеОтложенноеОбновлениеСВерсии:

Описание.ПараллельноеОтложенноеОбновлениеСВерсии = "2.3.3.20";

Свойства, специфичные для отложенных обработчиков обновления с режимом выполнения Параллельно:

* ● ПроцедураЗаполненияДанныхОбновления (Строка) – указывается процедура, которая регистрирует данные, подлежащие обновлению данным обработчиком. Регистрация выбранных данных выполняется при помощи процедур ОтметитьКОбработке и ОтметитьРегистраторыКОбработке общего модуля ОбновлениеИнформационнойБазы. Например:

Процедура ЗарегистрироватьДанныеКОбработкеДляПереходаНаНовуюВерсию(Параметры) Экспорт

Запрос = Новый Запрос;

Запрос.Текст =

"

...

";

Результат = Запрос.Выполнить().Выгрузить();

МассивСсылок = Результат.ВыгрузитьКолонку("Ссылка");

ОбновлениеИнформационнойБазы.ОтметитьКОбработке(Параметры, МассивСсылок);

КонецПроцедуры

Эти объекты метаданных следует также включить в состав плана обмена ОбновлениеИнформационнойБазы.

* ● ПроцедураПроверки (Строка) – всегда указывается «ОбновлениеИнформационнойБазы.ДанныеОбновленыНаНовуюВерсиюПрограммы».
* ● ЗапускатьТолькоВГлавномУзле (Булево) – указать Истина, если обработчик обновления может выполняться только в главном узле РИБ. По умолчанию Ложь. Может понадобиться, если, например, нужно сгенерировать новые ссылочные объекты в информационной базе. Данные, созданные таким обработчиком обновления, будут отправлены во все дочерние узлы РИБ.
* ● ЗапускатьИВПодчиненномУзлеРИБСФильтрами (Булево) – указать Истина, если обработчик обновления может выполняться только в подчиненном узле РИБ с фильтрами. По умолчанию Ложь. В подчиненном узле РИБ с фильтрами может не быть всех требуемых данных для корректного выполнения обработчика обновления, поэтому данные обрабатываются централизованно в главном узле, после чего отправляются в подчиненные. Включать возможность запуска обработчика нужно после предварительного анализа читаемых и обрабатываемых данных.
* ● ЧитаемыеОбъекты (Строка) – список полных имен объектов через запятую, которые обработчик обновления читает при обработке данных.
* ● ИзменяемыеОбъекты (Строка) – список полных имен объектов через запятую, которые обработчик обновления изменяет при обработке данных. С помощью значений свойств ЧитаемыеОбъекты и ИзменяемыеОбъекты контролируются возможные конфликты между обработчиками,  изменяющими или читающими одни и те же объекты.
* ● ОчередьОтложеннойОбработки (Число) – если обработчик обрабатывает данные независимо от других, то следует указывать значение очереди – «1». Если же при обработке данных обработчик опирается на результат выполнения другого обработчика, то ему следует указывать более высокое значение очереди. Это необходимо для оптимальной обработки данных, чтобы обработчик не отрабатывал вхолостую.
* ● Приоритеты (ТаблицаЗначений) – таблица зависимостей между обработчиками, изменяющими или читающими одни и те же объекты:
* ● Порядок (Строка) – порядок выполнения обработчика относительно другого. Может принимать значения До, После или Любой;
* ● Идентификатор (УникальныйИдентификатор) – идентификатор процедуры, с которой указывается взаимосвязь;
* ● Процедура (Строка) – полное имя процедуры, с которой указывается взаимосвязь. Допускается указывать только одно из свойств – Процедура или Идентификатор.

Например:

Обработчик.ЧитаемыеОбъекты = "Документ.ЗаказПокупателя";

Обработчик.ИзменяемыеОбъекты = "Документ.ЗаказПокупателя";

Обработчик.ПриоритетыВыполнения = ОбновлениеИнформационнойБазы.ПриоритетыВыполненияОбработчика();

Приоритет = Обработчик.ПриоритетыВыполнения.Добавить();

Приоритет.Порядок = "До";

Приоритет.Идентификатор = Новый УникальныйИдентификатор("b3be66c5-708d-42c8-a019-818036d09d06");

Приоритет.Процедура = "Документы.ЗаказПокупателя.ОбработатьДанныеДляПереходаНаНовуюВерсию"

В итоге, синтаксис добавления обработчика для параллельного режима отложенного обновления будет иметь следующий вид:

Обработчик = Обработчики.Добавить();

Обработчик.Версия = "1.2.3.4";

Обработчик.Идентификатор = Новый УникальныйИдентификатор("b3be66c5-708d-42c8-a019-818036d09d06");

Обработчик.Процедура = "Документы.ЗаказПокупателя.ОбработатьДанныеДляПереходаНаНовуюВерсию";

Обработчик.Комментарий = НСтр("ru = 'Заполнение значения нового реквизита ""Статус заказа"" у документов ""Заказ покупателя"" прошлых периодов.

|До завершения обработки ""Статус заказа"" данных документов будет отображаться некорректно.'");

Обработчик.РежимВыполнения = "Отложенно";

Обработчик.ОчередьОтложеннойОбработки = 1;

Обработчик.ПроцедураЗаполненияДанныхОбновления = "Документы.ЗаказПокупателя.ЗарегистрироватьДанныеКОбработкеДляПереходаНаНовуюВерсию";

Обработчик.ЧитаемыеОбъекты      = "Документ.ЗаказПокупателя";

Обработчик.ИзменяемыеОбъекты    = "Документ.ЗаказПокупателя";

Обработчик.ПроцедураПроверки    = "ОбновлениеИнформационнойБазы.ДанныеОбновленыНаНовуюВерсиюПрограммы";

Обработчик.БлокируемыеОбъекты   = "Документ.ЗаказПокупателя";

**Рекомендации по многопоточному выполнению обработчиков обновления**

Для выполнения длительных отложенных параллельных обработчиков обновления рекомендуется включать многопотоный режим. Для этого в процедуре ПриОпределенииНастроек общего модуля ОбновлениеИнформационнойБазыПереопределяемый следует установить параметры:

* ● МногопоточноеОбновление – разрешает многопоточное обновление, если установлено значение Истина;
* ● КоличествоПотоковОбновленияИнформационнойБазыПоУмолчанию – количество одновременно выполняющихся обработчиков обновления по умолчанию. Рекомендуемое значение равно наиболее вероятному количеству ядер процессора сервера 1С:Предприятие у пользователей. Например:

Процедура ПриОпределенииНастроек(Параметры) Экспорт

Параметры.Вставить("МногопоточноеОбновление", Истина);

Параметры.Вставить("КоличествоПотоковОбновленияИнформационнойБазыПоУмолчанию", 8);

КонецПроцедуры

Если включена возможность многопоточного обновления, то помимо конкурентной работы в сеансах пользователяей и регламентных заданий одновременно выполняются сразу несколько обработчиков обновленияю. Поэтому перед каждым выпуском прикладной конфигурации рекомендуется тестировать обновление «большой» реалистично наполненной информационной базы с минимальной поддерживаемой версии в режиме обработки данных с количеством потоков, установленным по умолчанию (8).

Для анализа и устранения выявленных проблем производительности (ожидания на блокировках, таймауты и взаимоблокировки) рекомендуется воспользоваться конфигурацей «Центр управления производительностью».

**Параллельное отложенное обновление в РИБ с фильтрами**

Разработка отложенного обновления имеет ряд особенностей в конфигурациях, предусматривающих работу в распределенной информационной базе (РИБ), когда данные синхронизируются не полностью (например, по организациям и т.п.). Такая организация работы называется РИБ с фильтрами.

В таком режиме обработчики отложенного обновления выполняются только в главном узле, т.к. в подчиненном узле может не быть всех требуемых для корректного обновления данных. Исключение составляют обработчики, которым установлен признак ЗапускатьИВПодчиненномУзлеРИБСФильтрами. При разработке отложенных обработчиков данный признак рекомендуется использовать в тех случаях, когда точно известно, что обновляемые данные самодостаточны, т.е. могут быть корректно обновлены в любом узле РИБ.

До того момента, как обновление конфигурации будет загружено в подчиненный узел, пользователи могут продолжать вводить данные на старой версии конфигурации подчиненного узла, которые будут отправлены в главный узел и могут привести к появлению некорректных (необновленных) данных. Чтобы исключить эту ситуацию, после первой синхронизации с каждым подчиненным узлом все параллельные отложенные обработчики в главном узле перезапускаются для обработки полученных из подчиненного узла данных.

При этом стоит учитывать, что, не смотря на то, что сами отложенные обработчики в подчиненном узле РИБ не выполняются, процедуры регистрации обновляемых данных для них будут выполнены. Это необходимо для того, чтобы корректно работал механизм защиты от изменения пользователем необновленных данных.

**Оперативное обновление на исправительные релизы конфигураций**

При разработке исправительных релизов конфигураций рекомендуется сохранять возможность оперативного (динамического) обновления таким образом, чтобы при обновлении на такие версии не требовалось обязательного прекращения работы пользователей и других активных сеансов. Для этого исправительный релиз конфигурации должен удовлетворять следующим условиям:

* ● в него могут быть внесены только «незначительные» изменения, которые позволяют динамически обновлять конфигурацию базы данных (т. е. когда не требуется реструктуризация информационной базы);
* ● обработчики обновления версии ИБ не должны требовать установки монопольного режима для своего выполнения;
* ● изменения в конфигурации должны ограничиваться только теми изменениями, которые не потребуют выполнения обязательных обработчиков обновления, предусмотренных в БСП (свойство Версия = «\*»).

Для обязательных обработчиков обновления (свойство Версия = «\*») дополнительно предусмотрена возможность программно определять, действительно ли для их выполнения требуется монопольный режим. Для этого следует установить свойство обработчика МонопольныйРежим в значение Истина. В этом случае:

* ● Такой обработчик вызывается дважды, в него передается параметр Параметры типа Структура, в котором имеется свойство МонопольныйРежим (Булево).
* ● При первом вызове в режиме проверки свойство МонопольныйРежим содержит значение Ложь. При этом код обработчика не должен модифицировать данные ИБ.
* ● Если в ходе выполнения обработчика возникает необходимость внесения изменений в ИБ, обработчик должен установить значение свойства в значение Истина и прекратить свое выполнение.
* ● При втором вызове в режиме выполнения свойство МонопольныйРежим содержит значение Истина. В этом случае код обработчика может модифицировать данные ИБ. Изменение значения свойства в этом случае игнорируется.

Оперативные обработчики обновления должны самостоятельно заботиться о целостности обновляемых данных: если выполняется более одного изменения данных, требуется выполнять действия в транзакции, а если производится чтение данных с последующим изменением, то требуется устанавливать исключительную управляемую блокировку.

Пример обязательного обработчика обновления с проверкой необходимости монопольного режима см. в демонстрационной конфигурации: процедура ВыполнятьВсегдаПриСменеВерсии общего модуля \_ДемоОбновлениеИнформационнойБазыБСП.

Для гарантированной проверки возможности оперативного обновления на новую версию конфигурации следует выполнить динамическое обновление конфигурации базы данных при активном сеансе «1С:Предприятия» и выполнить запуск нового сеанса администратора. Для этого необходимо:

1. Установить конфигурацию предыдущего релиза.

2. Запустить ИБ предыдущего релиза конфигурации под каким-либо пользователем.

3. Открыть ИБ предыдущего релиза конфигуратором и нажать F7.

4. Проверить, что конфигуратор предлагает обновиться динамически. Если не предлагает, см. ниже «Проблема 1».

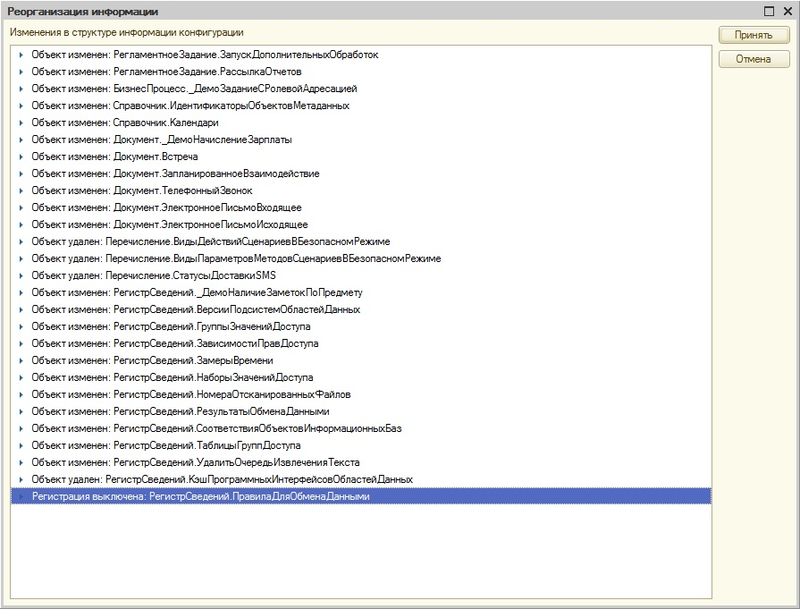
5. Обновить динамически, затем выполнить запуск под администратором по F5.

6. Проверить, что запуск прошел без ошибок и выводится окно с описанием изменений в новой версии. Если возникли ошибки, см. ниже «Проблема 2».

Проблема 1. Не проходит динамическое обновление конфигурации.

Для расследования нужно:

* ● завершить сеанс пользователя (запущенный на шаге 2) и нажать F7 еще раз;
* ● будет выведен список объектов конфигурации, препятствующих динамическому обновлению:



Для дальнейшего расследования причин (разработчиками) нужно сравнить тестируемую конфигурацию с предыдущей версией и выяснить, какие именно изменения блокируют динамическое обновление.

Проблема 2. Ошибки при первом запуске программы администратором.

Помимо «просто» ошибок в обработчиках обновления, может выдаваться ошибка вида:

* ● Невозможно выполнить обновление информационной базы:
* ● невозможно установить монопольный режим,
* ● версия конфигурации не предусматривает обновление без установки монопольного режима.

Она означает:

* ● что один из обработчиков обновления требует монопольного режима для своего выполнения;
* ● либо в конфигурацию были внесены изменения, которые потребовали выполнения обязательных обработчиков обновления, предусмотренных в БСП (для версии «\*»).

Дальнейшее расследование следует выполнять под отладкой либо если была включена константа Детализировать обновление ИБ в Журнале регистрации, то информацию о проблемном обработчике можно посмотреть в журнале регистрации.

**Переход с другой конфигурации**

Кроме переходов между версиями одной конфигурации подсистема позволяет также выполнять прикладной код при переходе с других конфигураций в режиме обновления. Например, можно предусмотреть переход с базовой версии конфигурации на версию ПРОФ, с ПРОФ на КОРП или другие переходы, при которых меняется не только номер версии, но и имя конфигурации.

При необходимости выполнить прикладной код при таком переходе в процедуре ПриДобавленииОбработчиковПереходаСДругойПрограммы общего модуля, имя которого задано в процедуре ПодсистемыКонфигурацииПереопределяемый.ПриДобавленииПодсистем, следует определить обработчики перехода в виде:

  Обработчик = Обработчики.Добавить();

  Обработчик.ПредыдущееИмяКонфигурации = "УправлениеТорговлей";

  Обработчик.Процедура = "ОбновлениеУПП.ЗаполнитьУчетнуюПолитику";

Для ветвления логики перехода в зависимости от версии исходной конфигурации внутри обработчика можно воспользоваться функцией ВерсияИБ общего модуля ОбновлениеИнформационнойБазы.

Пример обработчика перехода см. в общем модуле \_ДемоОбновлениеИнформационнойБазыБСП демонстрационной конфигурации.

**Особенности функционирования механизма при работе в модели сервиса**

При работе в модели сервиса переход с других конфигураций в режиме обновления не поддерживается (при такой попытке будет вызвано исключение).

Схема выполнения обработчиков обновления версии ИБ в модели сервиса состоит из двух частей:

* ● Выполняется обновление общих (неразделенных) данных. При необходимости выполняется блокировка всей информационной базы (используется монопольный режим). Порядок выполнения обработчиков обновления:
* ● выполняются обработчики с признаком ОбщиеДанные = Истина;
* ● устанавливаются блокировки на все области данных;
* ● в очередь заданий добавляются задания на выполнение во всех областях данных обработчиков обновления с признаком ОбщиеДанные = Ложь.
* ● Выполняется обновление разделенных данных (во всех областях данных). Выполняется механизмом очереди заданий. При необходимости используется блокировка области данных. Особенности выполнения:
* ● при обновлении существующих областей данных пропускаются обязательные обработчики обновления (Версия = «\*»). Чтобы эти обработчики были выполнены, необходимо их повторно зарегистрировать в обработчике обновления общих данных, у которого свойство УправлениеОбработчиками равно Истина;
* ● при подготовке новых областей данных обязательные обработчики обновления разделенных данных всегда выполняются.

Свойства, специфичные только для обработчиков обновления в модели сервиса:

* ● ОбщиеДанные (Булево) – если Истина, то обработчик обновления будет выполняться из неразделенного сеанса до выполнения любых обработчиков обновления разделенных данных с признаком ОбщиеДанные = Ложь. Такой обработчик может обращаться только к неразделенным данным. Допустимо указывать только для обработчиков с режимом выполнения Монопольно и Оперативно. Если указать значение Истина для обработчика с режимом выполнения Отложенно, будет выдано исключение. По умолчанию Ложь.
* ● УправлениеОбработчиками (Булево) – если Истина, в процедуру обработчика обновления будет передан дополнительный параметр типа Структура со свойством РазделенныеОбработчики. Свойство содержит дополнительную таблицу обработчиков, в которую нужно добавить обработчики разделенных данных, как и в основную таблицу Обработчики. По умолчанию Ложь.

Пример обработчика обновления общего (неразделенного) классификатора банков:

Обработчик = Обработчики.Добавить();

Обработчик.Версия = "1.2.3.4";

Обработчик.Процедура = "КлассификаторБанков.ЗаполнитьИНН";

Обработчик.ОбщиеДанные = Истина;

**Подключение обработчиков обновления в доработанных конфигурациях**

При доработке типовых конфигураций часто возникает необходимость добавлять свои документы и справочники. В дальнейшем это приводит также к необходимости создавать свои обработчики обновления. Для этих целей следует использовать следующий подход.

Считаем, что исходная типовая конфигурация становится библиотекой со своими собственными номером версии и обработчиками обновления, а доработанная конфигурация выступает в качестве основной конфигурации с собственным именем и своей нумерацией версий, разработанной на базе этой библиотеки.

Например, в типовой конфигурации БухгалтерияПредприятияКОРП в общем модуле ПодсистемыКонфигурацииПереопределяемый в процедуре ПриДобавленииПодсистем предусмотрена ссылка на общий модуль, который содержит описание типовой конфигурации:

МодулиПодсистем.Добавить("ОбновлениеИнформационнойБазыБП");

В общем модуле ОбновлениеИнформационнойБазыБП в процедуре ПриДобавленииПодсистемы указано имя и версия типовой конфигурации, например:

Описание.Имя = "БухгалтерияПредприятияКОРП";

Описание.Версия = "3.0.38.31";

В этом случае можно смело изменить имя и синоним конфигурации на свои собственные, например, БухгалтерияПредприятияКОРП\_СRM, и ввести свою нумерацию версий конфигурации. Затем создать общий модуль вида ОбновлениеИнформационнойБазы<Сокращение>, например, ОбновлениеИнформационнойБазыCRM и подключить его в конце процедуры ПриДобавленииПодсистем общего модуля ПодсистемыКонфигурацииПереопределяемый:

МодулиПодсистем.Добавить("ОбновлениеИнформационнойБазыCRM");

В общем модуле ОбновлениеИнформационнойБазыCRM в процедуре ПриДобавленииПодсистемы указать имя доработанной конфигурации, версию и зависимость от типовой конфигурации, например:

Описание.Имя = "БухгалтерияПредприятияКОРП\_СRM";

Описание.Версия = "1.0.1.1";

Описание.ТребуемыеПодсистемы.Добавить("БухгалтерияПредприятияКОРП");

Обработчики обновления для доработанной конфигурации следует размещать в этом же модуле и привязывать их к новой системе нумерации версий: «1.0.0.2» и т. д.

В дальнейшем, при каждом обновлении доработанной конфигурации на новую версию типовой конфигурации необходимо будет увеличивать номер версии доработанной конфигурации, для того чтобы сработали все обработчики обновления.

Подробнее о настройке модулей ПодсистемыКонфигурацииПереопределяемый и ОбновлениеИнформационнойБазы<Сокращение> см. раздел «[Подготовка к использованию подсистемы](https://its.1c.ru/db/content/bsp301doc/src/3.31.%20%D0%BE%D0%B1%D0%BD%D0%BE%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5%20%D0%B2%D0%B5%D1%80%D1%81%D0%B8%D0%B8%20%D0%B8%D0%B1.htm?_=1542292879#_подготовка_к_использованию)».