1С:Технология разработки решений 1cFresh

Москва
Фирма «1С»
2015

Фирма «1С», Москва, 123056, а/я 64
Отдел продаж: Селезневская ул., 21,
телефон: +7 (495) 737-92-57, факс: +7 (495) 681-44-07,
e-mail: 1c@1c.ru
URL: http://www.1c.ru, http://www.1cFresh.com, http://www.v8.1c.ru, http://users.v8.1c.ru

Группа разработки программ — А. Алексеев, А. Безбородов, Д. Бескоровайнов, П. Василец, А. Виноградов, А. Волков, И. Гольштейн, Е. Горностаев, Г. Дамье, А. Даровских, О. Дерут, Н. Евграфов, Б. Евтифеев, Д. Зарецкий, Д. Ивашов, И. Коваленко, С. Копиенко, Н. Корсаков, С. Кравченко, В. Кудрявцев, П. Кукушкин, А. Кунченко, А. Лакутин, М. Лейбович, Г. Леонтьев, А. Лехан, А. Макеев, А. Медведев, Е. Митрошкин, С. Мурзин, С. Нуралиев, Л. Онучин, М. Отставнов, Д. Павленко, А. Плякин, А. Рукин, Д. Русанов, М. Саблин, Е. Силин, Д. Службин, А. Смирнов, В. Соколов, П. Солодкий, В. Сосновский, Д. Сысоенков, А. Топорков, В. Тунегов, А. Трубкин, В. Филиппов, А. Цилябин, В. Черемисинов, П. Чиков, А. Чичерин, А. Шевченко, А. Щербинин.

Группа разработки технологии 1cFresh — Р. Алейников, А. Андреев, А. Боженов, К. Борзов, Г. Давидян, Д. Давыдов, А. Зайцева, И. Карвецкий, В. Ксенз, К. Конаныхин, Н. Корсаков, А. Лайер, О. Литвиненко, В. Мазаев, М. Максаймер, Н. Мохнатов, А. Плякин, М. Попов, С. Седов, С. Трусов, А. Тюшкин, О. Шамарин, А. Шамин, Н. Шаргунова.

Группа тестирования — А. Зайцева

Документация — Р. Алейников, А. Андреев, А. Боженов, Д. Давыдов, Н. Корсаков, О. Литвиненко, А. Плякин, А. Сережин, О. Шамарин, В. Фигурнов.

Корректура — О. Литвиненко

Наименование книги:	1С:Технология разработки решений 1cFresh
Номер издания:	82.008.13.01
Дата выхода:	23.12.2015

Оглавление

Введение

Данный документ описывает требования к реализации прикладного решения и технологию модификации прикладного решения на базе платформы «1С:Предприятие» версии 8, призванные обеспечить работоспособность прикладного решения в модели сервиса.

Для использования данного документа необходимо ознакомиться с документом «1С:Технология публикации решений 1cFresh», в котором приведены общие сведения о технологии 1cFresh, развертывании и настройке инфраструктуры сервиса в соответствии технологией 1cFresh, обслуживания развернутого сервиса и использования приложений в сервисе.

Для работы с документом желательно иметь:

- представление о работе механизма разделения данных (глава 24 книги «1С:Предприятие 8.3. Руководство разработчика»);
- навыки использования механизма разделения данных системы «1С:Предприятие»;
- навыки применения Библиотеки стандартных подсистем фирмы 1С (далее — БСП), включая навык внедрения этой библиотеки в состав прикладного решения.

Данный документ не является заменой документации по системе «1С:Предприятие» и БСП.

Глава 1. Методика модификации прикладного решения

Прикладное решение, которое может функционировать в модели сервиса, должно соответствовать требованиям, описанным в этой главе.

1.1. Общие требования

1. Необходимо проверить работоспособность прикладного решения в веб-клиенте. Это основной режим работы в модели сервиса.

   При проверке желательно проверять работу решения во всех поддерживаемых «1С:Предприятием» веб-браузерах, так как некоторые веб-браузеры вносят дополнительные ограничения. Подробнее об ограничениях веб-клиента можно прочитать в Приложении 7.3 книги «1С:Предприятие 8.3. Руководство разработчика» (http://its.1c.ru/db/v83doc#bookmark:dev:TI000001245).

2. Нужно избегать режимов работы, которые работают только в тонком клиенте.

3. Необходимо обеспечить, чтобы использование прикладного решения было возможно без использования расширения работы с файлами. При этом допускается уменьшение сервисных возможностей при выполнении операций, связанных с файлами. Если клиентское приложение обнаруживает установленное расширение работы с файлами, оно должно это учитывать.

4. В реализации учетных алгоритмов нужно избегать использования метода ТекущаяДата(), так как для каждой области данных может быть установлен свой часовой пояс. Необходимо отдельно проработать, какие дата и время будут использоваться в каждой ситуации. Например, при выполнении учетных операций обычно следует использоваться метод ТекущаяДатаСеанса(). Более подробно этот вопрос описан в стандартах разработки прикладных решений (http://its.1c.ru/db/v8std/browse/13/-1).

5. Прикладное решение должно одинаково функционировать в файловом и клиент-серверном вариантах.

6. Необходимо помнить, что сервер «1С:Предприятия» может располагаться как на ОС Windows, так и на ОС Linux. Также заранее не известна СУБД, используемая в сервисе. Из этого вытекает запрет на использование в процессе разработки каких-либо решений, характерных для конкретной СУБД и ОС.

   В частности, не следует реализовывать механизмы работы с кластером серверов из прикладного решения (например, реализация аналога консоли управления кластером на встроенном языке), т. к. подсистема COM недоступна на ОС Linux.

7. Необходимо учитывать, что компьютеру (или компьютерам), на котором выполняется сервер «1С:Предприятия», по соображениям безопасности может быть ограничен доступ к внешним ресурсам. Следует считать, что имеется доступ только в каталог временных файлов операционной системы.

8. Не следует использовать в прикладном решении стороннее программное обеспечение, требующее отдельной установки и администрирования. Рекомендуется использовать штатные возможности системы «1С:Предприятие».

9. Средства администрирования также недоступны для разработчиков прикладных решений (в общем случае) и пользователей абонента сервиса.

10. При разработке прикладных решений необходимо избегать длительных серверных вызовов. При длительности серверного вызова, превышающей максимально допустимую, соединение веб-браузера (или тонкого клиента) с информационной базой будет разорвано. Максимально допустимая длительность серверного вызова равна:

    - примерно 20 секундам для наиболее распространенных браузеров и веб-серверов;
    - 8 секунд для некоторых браузеров.

    Поэтому для прикладных решений, предназначенных для работы в модели сервиса, все длительные (более 2–3 секунд) серверные вызовы следует выполнять асинхронно, с помощью фонового задания, как это описано в стандарте разработки

Длительные операции
(http://its.1c.ru/db/v8std/content/2149184291/1).

11. При разработке прикладного решения рекомендуется следовать стандартам и методикам разработки конфигураций для платформы «1С:Предприятие»
(http://its.1c.ru/db/v8std/browse/13/-1).

1.2. Особенности внедрения БСП

После того как прикладное решение доработано с учетом вышеописанных требований, необходимо выполнить ряд действий для того, чтобы решение смогло работать в модели сервиса. Для этого нужно интегрировать в это решение БСП версии 2.0 и выше.

Общая инструкция по внедрению БСП приведена в документации БСП (http://its.1c.ru/db/bspdoc/browse/13/-1).

1.2.1. Внедряемые подсистемы БСП

При объединении необходимо обязательно выбрать для внедрения следующие подсистемы:

1. Базовая функциональность.
2. Завершение работы пользователей.
3. Контактная информация.
4. Настройка порядка элементов.
5. Обмен данными.
6. Обновление версии ИБ.
7. Получение файлов из Интернета.
8. Пользователи.
9. Префиксация объектов.
10. Работа с почтовыми сообщениями.
11. Работа в модели сервиса.

1.2.2. Компоненты подсистемы БСП «Работа в модели сервиса»

Среди компонент подсистемы БСП «Работа в модели сервиса» можно выделить следующие виды:

- **обязательные** — они должны быть обязательно внедрены в прикладное решение, предназначенное для работы в модели сервиса;
- **«парные» к подсистемам БСП** — они обеспечивают работу в модели сервиса подсистем БСП с соответствующими именами (без суффикса в модели сервиса). Эти компоненты должны быть внедрены, если прикладное решение использует такие подсистемы БСП. Например, если к внедрению отмечена подсистема **Валюты**, следует также отметить и подсистему **Валюты в модели сервиса**;
- **необязательные** — решения о целесообразности их внедрения необходимо принимать индивидуально.

Перечислим все виды подсистем:

1. Обязательные:
   а) Базовая функциональность в модели сервиса.
   б) Выгрузка загрузка данных.
   в) Обмен данными в модели сервиса.
   г) Обмен сообщениями.
   д) Обновление версии ИБ в модели сервиса.
   е) Очередь заданий.
   ж) Пользователи в модели сервиса.
   з) Резервное копирование областей данных.
   и) Удаленное администрирование.
2. **«Парные» к подсистемам БСП**
   а) Валюты в модели сервиса.
   б) Дополнительные отчеты и обработки в модели сервиса.
   в) Управление доступом в модели сервиса.
   г) Файловые функции в модели сервиса.
3. **Необязательные**:
   а) Поставляемые данные — эта компонента должна внедряться в том случае, если прикладное решение использует какие-

либо поставляемые данные, например, используется подсистема Валюты в модели сервиса.

Более подробные сведения о зависимостях подсистем БСП приведены в инструкции по внедрению БСП (http://its.1c.ru/db/bspdoc/content/182/1, раздел «Перенос объектов метаданных из файла поставки библиотеки в конфигурацию», подраздел «Зависимости между подсистемами библиотеки»).

1.2.3. Доработка прикладного решения

После окончания объединения прикладного решения и БСП необходимо доработать программные модули в соответствии с документацией по БСП: разделы «Инструкция по внедрению библиотеки» и «Настройка и использование подсистем при разработке конфигурации» (http://its.1c.ru/db/bspdoc/content/182/1 и http://its.1c.ru/db/bspdoc/content/186/1).

1.3. Библиотека технологий сервиса

«1С:Библиотека технологий сервиса» (БТС) предназначена для реализации в прикладных решениях на платформе «1С:Предприятие 8» функционала, необходимого для работы через Интернет в модели сервиса в соответствии с технологией 1cFresh. Библиотека состоит из набора подсистем, часть которых может работать не только в модели сервиса, но и локально.

1.3.1. Подсистемы БТС

В состав БТС включены следующие подсистемы, которые ранее входили в «1С:Библиотеку стандартных подсистем» (БСП):

- **Информационный центр** (исключена из состава БСП с версии 2.1.5);
- **Выгрузка / загрузка областей данных** (исключена из состава БСП с версии 2.2.1, ранее называлась «Выгрузка / загрузка данных»);
- **Дополнительные отчеты и обработки в модели сервиса** (исключена из состава БСП с версии 2.2.2).

Также в составе БТС поставляются служебные подсистемы, расширяющие поведение БСП и предназначенные для

использования вместе с одноименными подсистемами БСП (а не вместо них):

- **Базовая функциональность в модели сервиса** (эту подсистему необходимо внедрять в прикладное решение при внедрении любой другой подсистемы БТС);
- **Обмен данными в модели сервиса**;
- **Обновление версии ИБ в модели сервиса**;
- **Пользователи в модели сервиса**;
- **Удаленное администрирование в модели сервиса**;
- **Управление доступом в модели сервиса**;
- **Файловые функции в модели сервиса**.

Каждая из этих подсистем обязательна к внедрению в прикладное решение, если внедряется соответствующая подсистема БСП.

ПРИМЕЧАНИЕ. Цель выделения части функционала для разработки прикладных решений в модели сервиса из инструментария «1С:Библиотека стандартных подсистем» в отдельную библиотеку — это обеспечение оперативности внедрения и обновления подсистем, необходимых для работы прикладных решений в модели сервиса. В дальнейшем планируется выделить из инструментария БСП и другие подсистемы, необходимые для работы прикладных решений в модели сервиса.

1.3.2. Подсистема «Информационный центр»

Подсистема БТС «Информационный центр» предоставляет возможность пользователю прикладного решения получить методическую поддержку непосредственно из прикладного решения, а именно:

- отображение ссылок на полезные ресурсы и статьи;
- отправка писем в техническую поддержку и просмотр переписки по ним;
- поиск информации на сайте ИТС;
- отображение недоступности ресурсов сервиса;
- просмотр и добавление идей (пожеланий), голосование за предложенную идею, комментирование пожеланий;
- отображение информационных ссылок в формах прикладного решения на любые внешние ресурсы.

1.3.3. Подсистема «Выгрузка / загрузка областей данных»

Подсистема БТС «Выгрузка / загрузка областей данных» предоставляет возможность выполнять выгрузку данных приложения в XML-файлы, которые в дальнейшем могут быть загружены в другую информационную базу или область данных. С использованием механизмов этой подсистемы реализуются:

- перенос данных пользователя из локальной версии конфигурации в приложение в сервисе;
- перенос данных пользователя из сервиса в локальную версию конфигурации;
- создание резервных копий приложений в сервисе (и восстановление приложений из резервных копий).

Подсистема «Выгрузка / загрузка областей данных» является обязательной для внедрения в конфигурации, которые должны поддерживать возможность развертывания в модели сервиса. При внедрении этой подсистемы обязательным является внедрение подсистемы БСП «Выгрузка / загрузка данных».

1.3.4. Внедрение БТС

Внедрение БТС может осуществляться:

- с помощью функции «Сравнить, объединить с конфигурацией из файла...»;
- с помощью внешней обработки «Помощник внедрения БТС», поставляемой в составе БТС.

Подробные сведения о порядке внедрения БТС и настройки подсистем БТС содержатся в документации, поставляемой в составе БТС.

1.4. Использование механизма разделения данных

1.4.1. Общие сведения о механизме разделения данных

При работе в модели сервиса используется механизм разделения данных, который позволяет разделить все хранимые данные на отдельные части. При использовании механизма разделения данных

все данные, хранящиеся в информационной базе, и объекты метаданных, соответствующие этим данным, могут быть разделены на следующие виды:

- **Общие (неразделенные) данные** — данные, которые являются общими для всех областей данных и не разделяются общими реквизитами-разделителями. Из сеансов, в которых установлены значения разделителей, общие данные доступны только для чтения;
- **Разделенные данные** — данные, относящиеся к области данных, разделяются общими реквизитами-разделителями.

При использовании механизма разделения данных:

- изменяется поведение объектов конфигурации, входящих в состав общего реквизита (далее — разделителя);
- для каждого разделителя в информационной базе определяется текущее значение разделителя и признак использования данного разделителя;
- данные информационной базы, доступные для выбранных значений разделителей, и данные неразделенных объектов конфигурации называются **областью данных**.

Во всех операциях чтения записей базы данных «1С:Предприятие» автоматически отбирает только те записи, в которых значения используемых разделителей совпадают с текущими значениями разделителей.

Подробнее о разделении данных можно прочитать в документации к системе «1С:Предприятие» (книга «1С:Предприятие 8.3. Руководство разработчика»):

- раздел 5.5.5 «Общие реквизиты» (http://its.1c.ru/db/v83doc/content/63/1),
- глава 24 «Механизм разделения данных» (http://its.1c.ru/db/v83doc/content/82/1).

Об особенностях использования разделения данных в модели сервиса написано в документации по БСП: раздел «Настройка и использование подсистем при разработке конфигурации», подраздел «Работа в модели сервиса» (http://its.1c.ru/db/bspdoc/content/226/1).

1.4.2. Основные и вспомогательные разделенные данные

При использовании БСП версии 2.1.3 и старше разделенные данные, хранящиеся в информационной базе, и объекты метаданных, соответствующие этим данным, могут быть одного из двух следующих видов:

- **Основные данные** — это разделенные данные, обладающие прикладным характером (представляющие ценность для пользователей). Большинство объектов метаданных, относящихся к разделенным данным — это основные данные. Основные данные доступны только из разделенных сеансов.
- **Вспомогательные данные** — это данные, логически являющиеся частью данных области, но доступные как из разделенных, так и из неразделенных сеансов. Вспомогательными могут быть только служебные данные (не имеющие ценности для пользователей).

1.4.3. Реквизиты-разделители, используемые при работе в модели сервиса

При использовании подсистемы «Работа в модели сервиса» БСП версии 2.1.3, и старше:

- **основные разделенные данные** должны разделяться общим реквизитом-разделителем ОбластьДанныхОсновныеДанные;
- **вспомогательные разделенные данные** должны разделяться общим реквизитом-разделителем ОбластьДанныхВспомогательныеДанные.

При использовании подсистемы «Работа в модели сервиса» БСП версии младшей, чем версия 2.1.3, разделенные данные должны разделяться общим реквизитом-разделителем ОбластьДанных.

Корректное функционирование подсистем БСП при добавлении других разделителей не гарантируется.

1.4.4. Общий реквизит ОбластьДанныхОсновныеДанные

Общий реквизит ОбластьДанныхОсновныеДанные (при использовании БСП версии младшей, чем версия 2.1.3 — ОбластьДанных) имеет тип Число(7), использование разделяемых данных — Независимо. Значение этого разделителя должно быть

установлено для всех пользователей информационной базы, кроме администратора информационной базы и пользователя, от имени которого компоненты сервиса осуществляют доступ к информационной базе с помощью Web-сервисов.

Каждое значение этого разделителя будет определять область данных (приложение). Одной областью данных владеет один абонент. Доступ к данным области имеют только пользователи этого абонента. Другие пользователи, в том числе администраторы и операторы сервиса, этого доступа не имеют.

У общего реквизита ОбластьДанныхОсновныеДанные (при использовании БСП версии младшей, чем версия 2.1.3 — ОбластьДанных):

- свойство Автоиспользование установлено в значение Использовать. Это удобно, так как большинство объектов метаданных являются основными разделенными данными. Данное свойство реквизита указывает, что вновь создаваемые объекты конфигурации по умолчанию будут отнесены к основным разделенным данным;
- свойство Разделение пользователей установлено в значение Разделять. Это сделано потому, что у каждого абонента может быть несколько пользователей, которые работают с приложением. При этом необходимо обеспечить, чтобы список пользователей абонента был недоступен за «пределами» приложения (области данных с уникальным значением разделителя);
- свойство Разделение аутентификаций установлено в значение Разделять. Это сделано потому, что один пользователь абонента может иметь доступ к разным приложениям. При этом требуется, чтобы пользователь абонента во всех приложениях, доступ к которым ему предоставлен, использовал один и тот же логин.

ПРИМЕЧАНИЕ. Подробнее про свойство Разделение пользователей можно прочитать в разделе 24.2.3 книги «1С:Предприятие 8.3. Руководство разработчика» (http://its.1c.ru/db/v83doc#bookmark:dev:TI000000881), а про свойство Разделение аутентификаций — в разделе 24.2.4 книги

«1С:Предприятие 8.3. Руководство разработчика» (http://its.1c.ru/db/v83doc#bookmark:dev:TI000000882).

1.4.5. Общий реквизит ОбластьДанныхВспомогательныеДанные

Общий реквизит ОбластьДанныхВспомогательныеДанные имеет тип Число(7), использование разделяемых данных — Независимо и совместно. Данный разделитель не используется для разделения пользователей и аутентификации.

Для этого реквизита свойство Автоиспользование установлено в значение Не использовать, поэтому вновь создаваемые объекты метаданных не будут включаться в состав данного разделителя.

1.4.6. Объекты, исключаемые из разделяемых данных

После внедрения БСП необходимо исключить из состава разделителя ОбластьДанныхОсновныеДанные (ОбластьДанных при внедрении версии БСП младше, чем 2.1.3):

1. Все регламентные задания, не входящие в состав БСП. Принадлежность разделителю регламентных заданий, входящих в состав БСП, изменять не требуется.
2. Объекты, хранящие общие настройки информационной базы (то есть, являющиеся общими для всех областей данных).
3. Прикладные данные, общие для всех областей.
4. Данные, являющиеся «частью» конфигурации, т. е. объекты конфигурации, которые содержат только предопределенные данные и не предполагают добавления других данных иначе, кроме как разработчиками прикладного решения. Фактически эти данные меняются только вместе с изменением конфигурации. Например: ПланВидовХарактеристик.ВидыДоступа в БСП.

Состав разделителя в части объектов БСП должен в точности соответствовать составу разделителя самой БСП. В противном случае не гарантируется работоспособность получившейся конфигурации.

1.5. Обеспечение работы приложения в модели сервиса и в локальном варианте

Прикладное решение должно обеспечивать работу как в модели сервиса, так и в локальном варианте. Например, конфигурация «Бухгалтерия предприятия» должна работать как на компьютере конечного пользователя (локальный вариант), так и в сервисе. В этом случае одной конфигурацией будет пользоваться большое количество потребителей (абонентов), но при этом конфигурация, использующаяся в обоих случаях, будет одной и той же.

1.5.1. Условное разделение

Для реализации такого сценария использования для разделителя ОбластьДанныхОсновныеДанные включено условное разделение. Подробнее про условное разделение можно прочитать в разделе 24.2.5 книги «1С:Предприятие 8.3. Руководство разработчика» (http://its.1c.ru/db/v83doc#bookmark:dev:TI000000883).

Если в прикладном решении есть какой-то специфичный для модели сервиса код, его выполнение необходимо предварять проверкой включения разделения. Для проверки включения разделения:

- на стороне клиента, в программном коде, который исполняется при начале работы системы (в соответствующих обработчиках) следует использовать СтандартныеПодсистемыКлиентПовтИсп.ПараметрыРаботыКлиентаПриЗапуске(). Включение или выключение разделение описывается свойством РазделениеВключено структуры, которая возвращается при вызове функции;
- в остальных случаях следует использовать функцию ОбщегоНазначенияПовтИсп.РазделениеВключено().

Если один и тот же программный код прикладного решения должен выполняться и в сеансе с используемым разделителем и в сеансе с неиспользуемым разделителем (включая условно выключенный разделитель), то для проверки того, что в сеансе используется разделитель, необходимо:

- на стороне сервера использовать функцию ОбщегоНазначенияПовтИсп.ДоступноИспользованиеРазделенныхДанных();

- на стороне клиента, в программном коде, который исполняется при начале работы системы (в соответствующих обработчиках) следует использовать СтандартныеПодсистемыКлиентПовтИсп.ПараметрыРаботыКлиентаПриЗапуске(). Использование разделителей описывается свойством ДоступноИспользованиеРазделенныхДанных структуры, которая возвращается при вызове функции;
- в остальных случаях следует использовать СтандартныеПодсистемыКлиентПовтИсп.ПараметрыРаботыКлиента(). Использование разделителей описывается свойством ДоступноИспользованиеРазделенныхДанных структуры, которая возвращается при вызове функции.

1.5.2. Функциональные опции

Если в прикладном решении существуют возможности, которые должны присутствовать в интерфейсе только в режиме с используемым или неиспользуемым разделителем, то необходимо воспользоваться функциональными опциями. Эти опции не могут быть одновременно установлены или сняты. Функциональные опции входят в состав БСП (подробнее см. документацию к БСП). При этом в БСП состав данных функциональных опций не заполнен. Его необходимо заполнить после внедрения БСП. В состав данных функциональных опций не следует включать объекты БСП.

1.5.3. Специфичные действия при условном включении разделения

Для осуществления специфичных действий при условном включении разделения предназначена переопределяемая процедура РаботаВМоделиСервисаПереопределяемый.ПриВключенииРазделенияПоОбластямДанных(). В частности, в эту процедуру необходимо поместить методы по включению регламентных заданий, используемых только в разделенном режиме и выключению заданий используемых только в неразделенном режиме. Более подробно о действиях, связанных с использованием условного разделения в БСП, написано в соответствующем разделе документации по БСП. При этом специальных переопределяемых процедур, которые вызываются при выключении разделения, не предусмотрено.

1.5.4. Отсутствие модальных форм и синхронных вызовов

Для корректной работы приложения в модели сервиса в приложении не должны использоваться модальные формы и синхронные клиентские вызовы.

1.6. Регламентные задания

Прикладное решение не должно содержать предопределенных разделенных регламентных заданий. При наличии большого количества областей данных в одной информационной базе разделенные регламентные задания могут вызвать перегрузку рабочих процессов, обслуживающих данную информационную базу. Подробнее о разделенных регламентных заданиях можно прочитать в приложении 7.5.9 книги «1С:Предприятие 8.3. Руководство разработчика» (http://its.1c.ru/db/v83doc#bookmark:dev:TI000001256).

Все регламентные задания (не входящие в состав БСП), которые существуют в прикладном решении, должны быть исключены из состава разделителя. Для исполнения регламентных заданий, которые должны выполняться в разделенном режиме, необходимо использовать подсистему Очередь заданий из БСП (раздел «Работа в модели сервиса»).

1.7. Монопольный режим и блокировка области данных

Алгоритмы прикладного решения, которые требуют установки монопольного режима, необходимо проанализировать и, при необходимости, переработать. Это связано с тем, что блокировка всех областей данных воспрепятствует работе всех пользователей сервиса, что, как правило, недопустимо.

Следует иметь в виду, что в платформе «1С:Предприятие 8», начиная с версии 8.3.3, метод встроенного языка УстановитьМонопольныйРежим() выполняется следующим образом:

- при использовании в разделенном сеансе метод устанавливает или снимает монопольный режим области данных, определяемой текущими значениями разделителей;
- при использовании в неразделенном сеансе метод устанавливает или снимает монопольную блокировку всей информационной базы.

При установке монопольного режима области данных работа пользователей с другими областями данных информационной базы не блокируется.

Для установки или снятия монопольного режима области данных рекомендуется использовать специальные методы БСП:

- ОбщегоНазначения.ЗаблокироватьИБ()
- ОбщегоНазначения.РазблокироватьИБ()

ВНИМАНИЕ! Метод ОбщегоНазначения.ЗаблокироватьИБ() отменяет открытую транзакцию, если метод ИнформацияОбОшибке() возвращает информацию об ошибке. Если метод ИнформацияОбОшибке() возвращает пустой объект, то транзакция фиксируется.

СОВЕТ. Не рекомендуется всегда блокировать область данных целиком. Целесообразно в каждом случае принимать решение по блокировке конкретных данных.

1.8. Отчеты

Чтобы статистика о выполнении отчетов в сервисе была правильной, необходимо, чтобы отчеты были разработаны в соответствии со стандартом выполнения длительных операций (http://its.1c.ru/db/v8std/content/2149184291/1) или использовали формы, которые автоматически формирует платформа.

1.9. Подготовка областей данных

После того как прикладное решение размещено в сервисе, менеджер сервиса выполняет предварительную подготовку областей данных для использования абонентами сервиса. Для этого менеджер сервиса отправляет специальное сообщение в прикладное решение (с помощью подсистемы обмена сообщениями БСП). Количество

подготавливаемых областей для информационной базы определяется в свойствах информационной базы в менеджере сервиса.

Предварительно подготовленные области предназначены для того, чтобы уменьшить время начала работы для нового абонента, особенно в тех случаях, когда действия, выполняемые во время первого запуска приложения, занимают существенное время.

1.9.1. Подготовка областей данных для БСП младше 2.1.2

Если в прикладное решение внедрена версия БСП младше 2.1.2, то при подготовке области в качестве источника данных используется специальная область данных, которая называется **эталонной**.
В качестве эталонной области используется область со значением разделителя, равным 0. Если в прикладное решение внедрена версия БСП 2.1.2 (или старше), то в качестве источника данных для подготовки области используются специальные файлы начальных данных, загружаемые в менеджер сервиса.

Процесс подготовки новой области состоит из следующих действий:

- при необходимости создать новую область данных в информационной базе (целевая база) менеджер сервиса передает в информационную базу команду создания новой области. Возможна ситуация, когда область необходимо создать не пустой, а с конкретными данными. В этом случае информационной базе дополнительно передается ссылка на данные, которые необходимо загрузить в новую область данных. Область резервируется со своим номером в регистре сведений целевой информационной базы ОбластиДанных;
- в целевой информационной базе регламентное задание выбирает все новые области данных и приступает к их подготовке:
  - если для области указана ссылка на загружаемые данные, система загружает эти данные и помечает область как используемую;
  - если ссылка не указана, то выполняется копирование эталонной области в том случае, если включено копирование области данных из эталонной (константа КопироватьОбластьДанныхИзЭталонной установлена в значение Истина) или выполняется подготовка области в

демонстрационной информационной базе (константа РежимИспользованияИнформационнойБазы установлена в значение Демонстрационный);

- завершающим этапом подготовки области является запуск механизма обновления информационной базы с принудительным выполнением обязательных обработчиков. После завершения подготовки, информационная база отправляет в менеджер сервиса сообщение с результатом подготовки.

Если в прикладное решение внедрена версия БСП младше 2.1.2, необходимо первый запуск информационной базы выполнить с указанием параметра запуска ИнициализироватьРазделеннуюИБ. Кроме того, такой запуск необходимо выполнять на абсолютно пустой информационной базе. После выполнения такой подготовки информационная база готова к размещению в сервисе. Если будет обнаружено, что выполняется не первый запуск прикладного решения, подготовка информационной базы к работе в сервисе не будет выполнена.

1.9.2. Подготовка областей данных для БСП версии 2.1.2 и выше

Если в прикладное решение внедрена БСП версии 2.1.2 и выше, выполнять запуск с параметром ИнициализироватьРазделеннуюИБ не требуется. Вместо этого перед подключением ИБ к менеджеру сервиса следует установить значение константы ИспользоватьСинхронизациюДанных равным Истина. Аналогичное действие требуется выполнить при первом обновлении прикладного решения до версии, содержащей внедренную версию БСП 2.1.2

1.9.3. Предупреждение при запуске от имени неразделенного пользователя

Если подготовка информационной базы выполнена, то при запуске прикладного решения от имени неразделенного пользователя в заголовке основного окна приложения будет присутствовать текст: Не установлены разделители.

Глава 2. Проверка функционирования в модели сервиса

После окончания разработки прикладного решения необходимо проверить его функционирование в модели сервиса 1cFresh.
В частности, следует проверить:

- корректность взаимодействия прикладного решения с компонентами сервиса — менеджером сервиса и агентом сервиса;
- обмен данными опубликованного в сервисе прикладного решения с другими приложениями сервиса, локальными информационными базами и автономным рабочим местом — если для прикладного решения предусмотрены такие возможности;
- корректность обновления информационных баз прикладного решения с предыдущих версий, если для прикладного решения предусмотрено такое обновление.

2.1. Тестовый стенд для проверки

Для проверки функционирования прикладного решения в модели сервиса 1cFresh необходимо создать тестовый стенд, на котором будет осуществляться проверка. На этом стенде следует установить:

1. Компоненты сервиса 1cFresh, непосредственно взаимодействующие с прикладным решением, а именно:
    - менеджер сервиса;
    - агент сервиса.
2. Другие компоненты сервиса 1cFresh (при желании).
3. Информационные базы прикладных решений, обмен данными с которыми необходимо проверить.
4. Приложения, необходимые для развертывания и функционирования компонент сервиса 1cFresh, в частности:
    - операционную систему (или операционные системы);

- сервер «1С:Предприятия 8»;
- толстый клиент «1С:Предприятия 8» той же версии, что сервер «1С:Предприятия 8»;
- сервер СУБД;
- веб-сервер для публикации информационных баз.

5. Для проверки использования прикладного решения с протоколом HTTPS и OpenID-аутентификацией необходимо установить фронтенд-сервер, например, NGINX.

Этот набор компонентов является минимально необходимой конфигурацией сервиса, требуемой для проверки работоспособности прикладного решения в модели сервиса. Эти компоненты можно развернуть на одном компьютере.

О том, как осуществляется развертывание компонент сервиса, написано в документах:

- «1С:Технология публикации решений 1cFresh», главы 3-8;
- Демонстрационный пример № 1 развертывания сервиса 1cFresh (на Linux и Windows).
- Демонстрационный пример № 2 развертывания сервиса 1cFresh (на Windows).
- Демонстрационный пример № 3 развертывания сервиса 1cFresh (на Linux).

2.2. Действия для проверки

Для проверки работоспособности прикладного решения в модели сервиса 1cFresh должны быть успешно выполнены следующие операции:

- регистрация конфигурации и версии конфигурации прикладного решения в менеджере сервиса;
- развертывание в сервисе информационной базы прикладного решения в рабочем и демонстрационном вариантах, и регистрация их в менеджере сервиса;
- добавление приложений (областей данных) в рабочий и демонстрационный варианты информационной базы прикладного решения;
- регистрация в сервисе абонента, и добавление пользователю-владельцу абонента приложения прикладного решения;
- запуск рабочего варианта приложения пользователем сервиса в веб-клиенте и в тонком клиенте;

- запуск демонстрационного варианта приложения;
- возможность работы пользователей с рабочим и демонстрационным вариантами приложения;
- создание резервной копии приложения;
- обновление информационной базы прикладного решения;
- обмен данными (для тех приложений, с которыми заявлена такая возможность):
    - с другими приложениями сервиса;
    - с локальными версиями приложений.

Подробнее о перечне требований, необходимых для обеспечения работоспособности прикладного решения в модели сервиса 1cFresh и о методике проверки этих требований написано в документе «Рекомендации по подготовке конфигураций к работе в режиме сервиса 1cFresh».

2.3. Нагрузочное тестирование

Важным этапом проверки работоспособности прикладного решения в модели сервиса 1cFresh является нагрузочное тестирование. Для выполнения нагрузочного тестирования необходимо описать основные ключевые операции прикладного решения в виде тестовых обработок Тест-центра (http://v8.1c.ru/expert/tc/tc_overview.htm), , а основные бизнес-процессы прикладного решения — в виде тестовых сценариев Тест-центра.

При выполнении нагрузочных тестов не должно наблюдаться таких проблем, как взаимоблокировки, превышение стандартных таймаутов и т. д., а производительность прикладного решения должна быть на приемлемом уровне. Типовые требования — при нагрузочном тестировании на 600 одновременно работающих пользователях общий APDEX не должен быть ниже 0,85, APDEX каждой ключевой операции не ниже 0,5.

Если при нагрузочном тестировании будут обнаружены взаимоблокировки, превышение стандартных таймаутов или низкая производительность, то конфигурацию прикладного решения необходимо оптимизировать. Это позволит обеспечить возможность одновременной работы большого количества пользователей с информационной базой, что необходимо для использования прикладных решений в модели сервиса.

Глава 3. Использование поставляемых данных

3.1. Поставляемые данные

Часть нормативно-справочной информации (НСИ), входящая в состав прикладного решения, может быть получена из внешних источников. Такие данные называются **поставляемыми данными**. Обновление таких данных выполняется с помощью специальной подсистемы менеджера сервиса. Требования к общим данным определяются БСП.

С точки зрения прикладного решения работа с общими данными не требует дополнительных усилий. В тоже время прикладной разработчик может разработать собственные поставляемые данные. Разработчик поставляемых данных выбирает, в каких объектах прикладного решения (разделенных или неразделенных) будут храниться «его» поставляемые данные. Технология разработки поставляемых данных описана ниже, а также в документации к БСП.

3.2. Разработка новых видов поставляемых данных

Нормативно-справочная информация, которая предоставляется прикладным решениям, с помощью подсистемы поставляемых данных менеджера сервиса, хранится в виде файлов. Содержание файлов зависит от вида поставляемых данных и, в общем случае, может быть произвольным. Поставляемые данные характеризуются **видом поставляемых данных** и произвольным **набором характеристик**. Характеристики могут быть **ключевыми**, **обычными** и **дополнительными**.

В системе не может быть двух файлов с одинаковым видом данных и набором ключевых характеристик. Дополнительные характеристики представляют собой произвольный набор пар Наименование-Значение. Состав дополнительных характеристик у различных данных одного вида может различаться. Длина значения

дополнительных характеристик неограниченна. Набор характеристик определяется разработчиком самостоятельно.

Комбинация вида поставляемых данных и комбинации всех значений характеристик называется дескриптором (описанием) поставляемых данных.

В том случае, если необходимо разработать новый вид поставляемых данных, разработчику необходимо выполнить следующее:

- Разработать механизм получения нового вида поставляемых данных для менеджера сервиса. Для этого необходимо реализовать обработку, реализующую специальный интерфейс.
- Доработать прикладное решение таким образом, чтобы оно содержало в себе механизмы работы с новым видом поставляемых данных. В частности — обработка полученного файла с данными и размещение этих данных (при необходимости) в структурах данных прикладного решения.

Общая схема работы выглядит следующим образом:

- Менеджер сервиса выполняет получение новых поставляемых данных с помощью специализированной обработки.
- После сохранения файла с данными (в информационную базу или в файловую систему), менеджер сервиса рассылает об этом уведомления. Для получения уведомления, прикладное решение должно подписаться на получение обновлений о появлении новых поставляемых данных. Рассылка уведомлений выполняется через механизм обмена сообщениями БСП. Менеджер сервиса следит за тем, чтобы не произошло одновременного обращения за новыми данными от множества подписчиков (перегрузка сервиса).

  Уведомление о доступности новых данных содержит полный набор характеристики и может быть использовано прикладным решением для определения необходимости загрузки файла данных.

- При получении сообщения, прикладное решение выполняет получение новой версии поставляемых данных с помощью подсистемы Сервис передачи файлов, предоставляемой БСП. Получение данных прикладным решением должно выполняться не ранее времени, которое менеджер сервиса устанавливает в посылаемое сообщение. Если в новых поставляемых данных нет

необходимости — прикладное решение может проигнорировать сообщение.

Кроме того, прикладное решение имеет возможность самостоятельно, используя Web-сервис поставляемых данных, запросить у менеджера сервиса необходимые ему дескрипторы данных, а затем и сами данные.

Далее будут подробно рассмотрены этапы разработки нового вида поставляемых данных.

3.3. Разработка механизма получения данных

Для получения новой версии поставляемых данных в менеджере сервиса необходимо создать специальную обработку, которая обязана предоставлять определенный интерфейс и которая описывается особым манифестом.

Обработки получения поставляемых данных могут быть следующих типов:

- **Интерфейсные**. Данный тип обработки обладает формой, предназначенной для того, чтобы оператор имел возможность выполнить некоторые интерактивные действия, связанные с получением поставляемых данных, например, указать файл на диске, содержащий новую версию поставляемых данных или указать пароль для доступа к ресурсу, откуда получаются поставляемые данные.
- **Фоновые**. Данный тип обработки не обладает пользовательским интерфейсом, однако обладает возможностью выполняться в не интерактивном режиме. Примером такой обработки является обработка, ежедневно загружающая курсы валют с сайта РБК (http://www.rbc.ru).
- **Интерфейсные и фоновые**. Данный тип обработки сочетает в себе обе возможности: наличие пользовательского интерфейса и возможность не интерактивного запуска.

Независимо от типа, обработка должна обладать формой настроек, в которой пользователь менеджера сервиса задает параметры, которые будут использоваться при получении данных.

Система поддерживает механизм обновления версий обработок. Так, обработки с одинаковым именем считаются разными версиями

одной обработки, и при добавлении новая обработка замещает старую. Разработчик может предусмотреть обновление структуры данных, используемых обработкой, при выполнении при обновления. Для этого в вызове функции ПриДобавлении() (вызываемой как при начальном добавлении обработки, так и при обновлении ее на новую версию) следует ориентироваться на значение параметра Настройки. При начальном добавлении этот параметр имеет значение Неопределенно. При обновлении там содержится ранее сохраненное значение настроек. Обработка может, например, хранить там структуру с текущей версией в одном из полей.

Для того чтобы система могла подключить и использовать обработку, в ее модуле должен присутствовать текстовый макет с именем МанифестОбработки, содержащий данные в формате XDTO-объекта SuppliedDataProcessor.

Пример манифеста:

```
<?xml version="1.0" ?>
<SuppliedDataProcessor xmlns="http://www.1c.ru/SaaS/SuppliedData"
Version="1.0.1" >
  <Name>Получение курсов валют</Name>
  <Version>1.0</Version>
  <Description>
    Обработка предназначена для автоматической загрузки
    курсов валют с сайта http://cbrates.rbc.ru
  </Description>
  <Type>Background</Type>
  <JobSchedule xmlns:v8de="http://v8.1c.ru/8.1/data/enterprise"
BeginDate="0001-01-01" EndDate="0001-01-01"
   BeginTime="18:00:00" EndTime="00:00:00"
CompletionTime="00:00:00"
   CompletionInterval="0" RepeatPeriodInDay="0" RepeatPause="0"
   WeekDayInMonth="0" DayInMonth="0" WeeksPeriod="1"
DaysRepeatPeriod="1">
    <v8de:WeekDays/>
    <v8de:Months/>
  </JobSchedule>
  <DataTypes>
    <DataType Code="КурсыВалютЗаДень" Description="Курсы валют за
день" Timeout="0.3" SuppressNotifications="true">
      <Property Code="Дата" Description="Дата" IsKey="true" />
    </DataType>
    <DataType Code="КурсыВалют" Description="Курсы валют за все
время" Timeout="20" />
  </DataTypes>
</SuppliedDataProcessor>
```

В манифесте указываются следующие свойства:

- Name. Тип: Строка(150). Отображаемое пользователю название обработки (пользователь может в дальнейшем при желании переименовать ее)
- Version. Тип: Строка (10). Отображаемая пользователю версия обработки
- Description. Тип: Строка неограниченной длины. Отображаемая пользователю строка с описанием обработки
- Type. Описывает тип обработки. Может принимать одно из следующих значений: Interactive (тип обработки — интерфейсная), Background (тип обработки — фоновая) или InteractiveAndBackground (интерфейсная и фоновая обработка).
- JobSchedule. Содержит рекомендуемое расписание запуска обработки для получения внешних данных. Пользователь может переопределить его. Для обработок с типом Interactive данный элемент игнорируется.
- DataTypes — список видов данных, обрабатываемых обработкой. Состоит из элементов DataType. Может быть добавлено несколько различных видов данных, например Курсы валют за все время и Курсы валют за день, или Справочник индексов и Справочник ОКАТО.
- DataType — описание вида данных. Содержит следующие атрибуты:
  - Code — код вида данных;
  - Description — наименование вида данных;
  - Timeout — рекомендуемая задержка в секундах между последовательными обращениями потребителей за данным видом данных;
  - SuppressNotifications — признак того, что данные указанного вида не будут автоматически отправлены в прикладное решение при его подключении к менеджеру сервиса.
  - Кроме того, вид данных может включать описание характеристик данных (элемент Property) в качестве дочерних элементов.
- Property — описание обычных и ключевых характеристик вида данных. Содержит следующие атрибуты:
  - Code — код характеристики;
  - Description — наименование характеристики;
  - IsKey — признак того, что характеристика является ключевой.

Манифест обработки анализируется при добавлении обработки в систему и при сбросе расписания с пользовательских настроек на стандартные. Виды и характеристики данных будут зарегистрированы в менеджере сервиса при добавлении обработки в систему.

3.3.1. Программный интерфейс обработки

В модуле обработки должны быть реализованы следующие экспортируемые процедуры:

ПриДобавлении

Описание:

Вызывается при добавлении обработки в систему или при обновлении обработки на новую версию. При необходимости процедура может записать в аргумент Настройки произвольную сериализуемую информацию. В этом случае она будет сохранена в информационной базе. Если добавление обработки невозможно — процедура должна вызывать исключение.

При обновлении версии обработки также вызывается данная процедура. В нее передается ранее установленное значение настроек. В этой процедуре можно выполнить обновление настроек в соответствии с новой версией обработки.

Синтаксис:

ПриДобавлении(Настройки)

Параметры:

Настройки

Произвольные данные, допускающие помещение в объект ХранилищеЗначения. Содержат настройки обработки получения поставляемых данных.

ПолучитьДанные

Описание:

Вызывается для импорта данных в фоновом режиме. Для обработок с типом Интерфейсная может отсутствовать. Внутри для работы с данными может осуществляться вызов процедур модуля

ПоставляемыеДанныеОбработка (см. ниже). Типичная процедура вызывает процедуру ПолучитьДескрипторы(), чтобы убедиться в том, что система не содержит уже актуальные данные. Если данные содержатся — процедура завершает свою работу, в противном случае — получает данные из внешнего источника, формирует на их основе файл и сохраняет в базу с помощью вызова СохранитьПоставляемыеДанные().

Синтаксис:

ПолучитьДанные(Знач Настройки)

Параметры:

Настройки	*По значению*

Настройки обработки, выполненные в форме настройки. Изменения настроек, сделанные в процедуре, игнорируются.

3.3.2. Формы

Обработка должна содержать несколько форм:

- Форма обработки по умолчанию — используется системой в качестве формы настроек. Должна присутствовать в обработках любого типа.
  Форма должна содержать реквизит НастройкиОбработки, тип Произвольный. В данный реквизит помещается объект настроек, которые будут использоваться при работе обработки (например, при получении поставляемых данных). При закрытии формы с КодВозвратаДиалога.ОК читается текущее название реквизита и сохраняется в базе.
- Форма ПолучитьДанные — форма реализует интерактивное получение поставляемых данных. Должна присутствовать только в обработках типа Интерфейсная или Интерфейсная и фоновая. Вызывается при нажатии кнопки Выполнить интерактивно интерфейса менеджера сервиса.
  Форма должна содержать реквизит НастройкиОбработки, тип Произвольный. В данный реквизит помещается объект настроек, которые будут использоваться при работе формы.

Наличие других форм определяется разработчиком вида поставляемых данных и собственно обработки.

3.3.3. Вспомогательные методы

Для работы с поставляемыми данными можно использовать методы общего модуля ПоставляемыеДанныеОбработка подсистемы ПоставляемыеДанные.

ПолучитьДескрипторы

Описание:

Может использоваться для определения наличия в системе актуальных данных. Если актуальные данные присутствуют — выполнение обработки можно завершить или предпринять другие действия.

В качестве фильтра можно использовать произвольную комбинацию характеристик. Если фильтр не указан — возвращаются все дескрипторы данных заданного типа. Допускается указывать произвольный набор ключевых и не ключевых характеристик. Если обработке необходимо получить файл данных, соответствующих дескриптору, то следует использовать функции модуля ФайлыМС.

Пример использования:

```
Функция ПолучитьФайлКурсов()

  Дескрипторы =
ПоставляемыеДанныеОбработка.ПолучитьДескрипторы("КурсыВалют");
  Если Дескрипторы.Количество() = 0 Тогда
    Возврат Неопределено;
  КонецЕсли;

  Результат = ФайлыМС.Получить(Дескрипторы[0].ИдентификаторФайла,
Неопределено);

  ИмяФайла = ПолучитьИмяВременногоФайла();

  Данные = ПолучитьИзВременногоХранилища(Результат.Адрес);
  Данные.Записать(ИмяФайла);

  Возврат ИмяФайла;

КонецФункции
```

Синтаксис:

ПолучитьДескрипторы(Знач ВидДанных, Знач Фильтр, Знач ВВидеXDTO)

Параметры:

ВидДанных	*По значению*

Тип: Строка. Содержит код вида поставляемых данных.

Фильтр	*По значению*

Тип: Массив. Каждый элемент массива должен содержать структуру, состоящую из следующих элементов:

- Код — строка, код характеристики;
- Значение — строка.

Значение по умолчанию: Неопределено.

ВВидеXDTO	*По значению*

Тип: Булево. Указывает, необходимость возвращать значение в виде объекта XDTO.

Значение по умолчанию: Ложь.

Возвращаемое значение:

ОбъектXDTO типа ArrayOfDescriptor или массив структур с полями ВидДанных, ДатаДобавления, ИдентификаторФайла, Характеристики, где Характеристики — массив структур с полями Код, Значение, Ключевая.

СохранитьПоставляемыеДанные

Описание:

Экспортируемая процедура, вызываемая внешними обработками для сохранения в базе поставляемых данных. Если вид данных неизвестен или среди характеристик есть дубли или заполнены не все ключевые характеристики, вызывается исключение. Характеристики с незарегистрированными в манифесте кодами считаются дополнительными.

Синтаксис:

СохранитьПоставляемыеДанные(Знач АдресВХранилище, Знач ИмяФайлаДанных, Знач ВидДанных, Знач Характеристики, Знач ЗапретУведомления)

Параметры:

АдресВХранилище *По значению*

Тип: Строка или ДвоичныеДанные или Файл.

Если параметр имеет тип:

- Строка, то он хранит адрес хранилища данных с помещенным туда файлом.
- ДвоичныеДанные, то параметр содержит непосредственно данные файла.
- Файл, то параметр содержит объект типа Файл, указывающий на нужный файл.

ИмяФайлаДанных *По значению*

Тип: Строка. Содержит отображаемое пользователю имя файла данных.

ВидДанных *По значению*

Тип: Строка. Код вида сохраняемых данных.

Характеристики *По значению*

Тип: Массив. Элемент коллекции должен иметь свойства:

- Код. Тип: Строка. Код характеристики поставляемых данных.
- Значение. Тип: Строка. Собственно значение.

ЗапретУведомления *По значению*

Тип: Булево. Посылать или нет потребителям уведомление о доступности новых данных. При первоначальной загрузке большого количества данных бывает полезно отключить уведомления.

Значение по умолчанию: Неопределено. В этом случае необходимость отправки уведомления будет определяться свойством SupressNotifications вида данных, указанного в параметре ВидДанных.

3.4. Доработка прикладного решения

Для получения поставляемых данных, прикладное решение должно зарегистрировать обработчики поставляемых данных определенного вида с помощью процедуры ПоставляемыеДанныеПереопределяемый.ПолучитьОбработчикиПоставляемыхДанных().

В эту процедуру передается таблица значений (параметр Обработчики), каждая строка которой описывает обработчик для одного вида поставляемых данных. При необходимости зарегистрировать обработчики для нескольких видов поставляемых данных, в таблице значений необходимо добавить несколько строк.

Таблица значений имеет следующие колонки:

- ВидДанных, Тип: Строка. Идентификатор вида данных, обрабатываемый обработчиком.
- КодОбработчика. Тип: Строка(20). Идентификатор обработчика, должен быть уникальным для каждого обработчика. Используется при восстановлении обработки в случае сбоя.
- Обработчик. Тип: ОбщийМодуль. Общий модуль, который должен содержать следующие процедуры:
  - ДоступныНовыеДанные();
  - ОбработатьНовыеДанные();
  - ОбработкаДанныхОтменена().

Рекомендуется добавлять в общий модуль ПоставляемыеДанныеПереопределяемый вызов регистрирующей процедуры, расположенной в прикладном модуле следующим образом:

Общий модуль ПоставляемыеДанныеПереопределяемый:

```
Процедура ПолучитьОбработчикиПоставляемыхДанных(Обработчики)
Экспорт

  КурсыВалютВМоделиСервиса.ЗарегистрироватьОбработчикиПоставляемыхД
анных(Обработчики);

КонецПроцедуры
```

Общий модуль ПоставляемыеДанныеПереопределяемый:

```
Процедура ЗарегистрироватьОбработчикиПоставляемыхДанных(Знач
Обработчики) Экспорт

  Обработчик = Обработчики.Добавить();
  Обработчик.ВидДанных = "КурсыВалютЗаДень";
  Обработчик.КодОбработчика = "КурсыВалютЗаДень";
  Обработчик.Обработчик = КурсыВалютВМоделиСервиса;

КонецПроцедуры
```

В процедуру ДоступныНовыеДанные() зарегистрированного модуля передается дескриптор обновившихся поставляемых данных. Если поступившие данные нужно загрузить, следует установить параметр процедуры Загружать в значение Истина. Данные будут загружены в фоновом задании, в момент, указанный в дескрипторе (каждый дескриптор содержит свое время загрузки для предотвращения перегрузки менеджера сервиса одновременным обращением множества потребителей).

Когда файл получен, будет вызвана вторая процедура зарегистрированного общего модуля — ОбработатьНовыеДанные(). В эту процедуру будет передан дескриптор загруженных данных и путь к файлу данных на диске. Прикладное решение должно прочитать файл и поместить находящиеся там данные в соответствующие структуры информационной базы.

В случае если к моменту запланированного вызова процедуры ОбработатьНовыеДанные() окажется, что файл поставляемых данных стал недоступен, или метод ОбработатьНовыеДанные() три раза вызвал исключение, вызывается процедура ОбработкаДанныхОтменена(), в которой разработчик имеет возможность обработать данную ситуацию (например, очистить временные данные, используемые при обработке).

Кроме работы по оповещению, система позволяет вручную получить дескриптор по заданным значениям вида данных и характеристик. Для этого предназначена функция ПоставляемыеДанные.ПолучитьДескрипторы(). Она может быть использована при начальном заполнении информационной базы при подключении ее к менеджеру сервиса.

Для хранения поставляемых данных на стороне информационной базы, предназначен справочник ПоставляемыеДанные и

программный интерфейс доступа к нему (непосредственная работа со справочником не рекомендуется).

Для восстановления после ошибки выполнения длительных операций обновлений данных в приложениях, подсистема содержит регистр сведений ОбластиТребующиеОбработкиПоставляемыхДанных и программный интерфейс доступа к нему. Если распространение поставляемых данных по приложениям занимает продолжительное время, то разработчик должен использовать этот программный интерфейс, чтобы в случае сбоя иметь возможность продолжить с того же места.

При подключении базы к менеджеру сервиса, туда автоматически передаются все поставляемые данные, для типов которых не установлен флаг SuppressNotifications.

В том случае, если после обновления конфигурации, подключенной к менеджеру сервиса, ей стали требоваться поставляемые данные нового вида, следует учитывать, что менеджер сервиса не передаст их автоматически в том случае, если данные были загружены в менеджер заранее.

Например, если конфигурация не работала с валютами, все уведомления о доступности новых курсов ею игнорировались. Для того чтобы корректно обработать ситуацию добавления поддержки валют в такую конфигурацию, необходимо в обработчике обновления информационной базы вызвать процедуру ПоставляемыеДанные.ЗапроситьВсеДанные(). Вызов метода приведет к повторной отправке всех имеющихся в менеджере сервиса поставляемых данных в прикладное решение, аналогично тому, как это выполняется при первоначальном подключении прикладного решения к менеджеру сервиса. Следует установить для такого вызов параметр Опциональный, чтобы избежать его выполнения при первоначальном заполнении информационной базы.

Глава 4. API для тарифов

В менеджере сервиса реализован веб-сервис TariffControl, позволяющий внешним веб-сервисам осуществлять удаленный доступ к хранимым менеджером сервиса данным о тарифах и лицензиях. В этой главе мы расскажем о том, как использовать этот сервис.

О том, как настроить менеджер сервиса для того, чтобы обеспечить возможность использования веб-сервиса TariffControl, рассказано в книге «1С:Технология публикации решений 1cFresh».

4.1. Использование веб-сервиса TariffControl

4.1.1. Получение информации о соответствии тарифов

В менеджере сервиса и в лицензируемых сервисах (например, «ИТС») существуют свои тарифы. Вся информация о тарифах абонента, пользователь которого работает в лицензируемом сервисе, хранится только в менеджере сервиса, а лицензируемые сервисы должны предоставлять тот или иной уровень доступа к своим данным на основе своих тарифов. Для решения этой задачи менеджер сервиса хранит у себя соответствие между своими тарифами и тарифами лицензируемого сервиса и по запросу может предоставить эту информацию лицензируемым сервисам.

Получить информацию о соответствии тарифов можно, как по пользователю, так и по абоненту при помощи операции GetTariffs() (подробнее см. 4.3). При получении информации по пользователю, возвращается информация по всем абонентам, в которых зарегистрирован пользователь.

Операция возвращает массив идентификаторов тарифов лицензируемого сервиса, на которые подписан указанный абонент или абоненты, в которых зарегистрирован указанный пользователь.

ВНИМАНИЕ! Для повышения производительности менеджера сервиса и лицензируемых сервисов, настоятельно рекомендуется в лицензируемых сервисах реализовывать кэш результатов вызова операций по получению соответствий между тарифами. Длительность хранения данных в кэше предлагается ограничивать 1–5 минутами (значения менее минуты снизят производительность, а значения более пяти минут вызовут ощутимые несоответствия после внесения изменений в тарифы).

4.1.2. Работа с лицензиями

Лицензируемые сервисы (например, электронная подпись в облаке) могут ограничивать доступ к своим ресурсам на основе тарифов менеджера сервиса. Для реализации этой возможности необходимо воспользоваться операциями:

- AcquireLicenses (занимает лицензии),
- ReleaseLicenses (освобождает лицензии),

веб сервиса TariffControl (подробнее см. 4.3 «Веб сервис TariffControl»).

В общем виде все эти операции на входе принимают описание лицензии, включающее в себя:

- идентификатор вида ограничения тарифа (код элемента справочника ВидыОграниченийТарифов). Значение этого параметра соответствует коду вида ограничения тарифа, заданного в справочнике Виды ограничения тарифов менеджера сервиса;
- количество захватываемых или освобождаемых лицензий;
- идентификатор выполняемой операции (аналог идентификатора транзакции, в которой выполняется операция по выделению или освобождению ограничиваемого тарифами ресурса);
- количество секунд, относительно момента времени вызова операции, после которого операция считается просроченной и может быть удалена, если константа ЗапретитьАвтоматическоеВосстановлениеЛицензий установлена в Ложь.
- описание объекта ограничения (зависит от конкретной операции, подробнее см. п. 4.3);

На выходе эти операции возвращают результат выполнения или текст ошибки, если не удалось выполнить операцию.

4.2. XDTO пакет TariffControl

Некоторые параметры операций веб сервиса TariffControl требуют наличия нестандартных типов, которые описаны в созданном для этих целей XDTO пакете TariffControl:

- имя: TariffControl_1_0_0_1;
- пространство имен: http://www.1c.ru/1cFresh/TariffControl/1.0.0.1;
- типы значений:
  - Result – перечисление вариантов результата выполнения операций:
    - Failed – операция не выполнена, описание причины в ResultMessage.
    - Succeeded – операция выполнена успешно.
    - SynchronizationRequired – операция не выполнена, необходимо выполнить синхронизацию и затем повторить выполнение операции.
  - UUID – уникальный идентификатор:
    - Образец: [A-Fa-f0-9]{32}|(\{|\()?[A-Fa-f0-9]{8}-([A-Fa-f0-9]{4}-){3}[A-Fa-f0-9]{12}(\}|\))?
- типы объектов:
  - Application – описание приложения:
    - ID (integer) – номер области данных.
  - Configuration – описание конфигурации:
    - ID (string) – код конфигурации.
    - Subscriber (integer) – код абонента.
  - License – описание лицензии:
    - Type (string) – идентификатор вида ограничения тарифа (код элемента справочника ВидыОграниченийТарифов.
    - Count (integer) – количество используемых лицензий.
    - OperationID (uuid) – идентификатор операции при помощи которой была выдана лицензия.

        - Commited (boolean) – признак того, выполнялось ли подтверждение CommitAccuireLicenses() после получения лицензии. Используется при синхронизации данных.
        - Timeout (integer) – количество секунд, относительно момента вызова операции, после которого не подтвержденная лицензия считает просроченной и может быть удалена, если значение константы ЗапретитьАвтоматическоеВосстановлениеЛицензий равно Ложь.
        - Context (LicenseContext) – привязка лицензии.
    - LicenseList [0; -1] – список выданных лицензий:
        - License (License) – описание выданной лицензии.
    - TariffList [0; -1] – список тарифов:
        - Tariff (string) – имя тарифа в терминах внешнего сервиса.
    - LicenseContext – один из перечисленных ниже элементов:
        - Application (Application) – параметры привязки по приложению.
        - Configuration (Configuration) – параметры привязки по конфигурации.
    - TariffContext один из перечисленных ниже элементов:
        - Subscriber (integer) – код абонента,
        - User (string) – логин пользователя.

4.3. Веб сервис TariffControl

Для получения информации по тарифам и управления лицензиями создан веб сервис TariffControl со следующей структурой:

- имя: TariffControl_1_0_0_1;
- подсистема: Обслуживание.УправлениеТарифами.ВнешнийИнтерфейс;
- пространство имен: http://www.1c.ru/1cFresh/TariffControl/1.0.0.1;
- имя файла публикации: tariffcontrol_1_0_0_1.1cws;
- пакеты XDTO: TariffControl_1_0_0_1.

Операции этого веб-сервиса следующие:

1. Result AcquireLicenses(License, ResultMessage) – получить лицензии указанного типа, где:
   - License (License) – описание параметров получения лицензии.
   - ResultMessage (string, выходной) – описание ошибки.
   - Возвращаемое значение (Result).
2. Result CommitAcquireLicenses(OperationID, ResultMessage) – зафиксировать ранее выполненное получение лицензии, где:
   - OperationID (uuid) – идентификатор фиксируемой операции.
   - ResultMessage (string, выходной) – описание ошибки.
   - Возвращаемое значение (Result).
3. dateTime GetLastAccessTime(ResultMessage) – получить дату последнего обращения при помощи GetLastAccessTime(), используемую для определения необходимости выполнения синхронизации менеджера сервиса с лицензируемым сервисом, где:
   - ResultMessage (string, выходной) – описание ошибки.
   - dateTime – дата в UTC+0.
4. TariffList GetTariffs(Context, ResultMessage) – получить идентификатор тарифа стороннего сервиса по имени учетной записи пользователя или коду абонента, где:
   - Context (TariffContext) – параметры пользователя или абонента;
   - ResultMessage (string, выходной) – описание ошибки.
   - Возвращаемое значение (TariffList) – массив идентификаторов тарифов, на которые подписан пользователь во внешнем сервисе.
5. Result IsSyncFinished(ResultMessage) – определить, закончилась ли синхронизация, начатая при помощи вызова Synchronize(), где:
   - ResultMessage (string, выходной) – описание ошибки.
   - Возвращаемое значение (Result).

6. Result IsSyncRequired(LastCall, ResultMessage) – требуется ли синхронизация, где:
   - LastCall (integer) – номер последнего успешно выполненного вызова.
   - ResultMessage (string, выходной) – описание ошибки.
   - Возвращаемое значение (Result).
7. Result ReleaseLicenses(OperationID, ResultMessage) – освободить ранее занятые лицензии, соответствующие указанному идентификатору, где:
   - OperationID (uuid) – идентификатор операции, которая ранее получила лицензию.
   - ResultMessage (string, выходной) – описание ошибки.
   - Возвращаемое значение (Result).
8. Result RollbackAcquireLicenses(OperationID, ResultMessage) – отменить получение лицензии с указанным идентификатором:
   - OperationID (uuid) – идентификатор операции, которая ранее получила лицензию.
   - ResultMessage (string, выходной) – описание ошибки.
   - Возвращаемое значение (Result).
9. Result Synchronize(Licenses, Replace, Finish, ResultMessage) – синхронизировать лицензии менеджера сервиса с занятыми ресурсами удаленной системы, где:
   - Licenses (LicenseList) – массив структур, описывающих занятые лицензии.
   - Replace (boolean) – перед синхронизацией удалять все записи из менеджера сервиса.
   - Finish (boolean) – снимать признак необходимости синхронизации после ее завершения.
   - ResultMessage (string, выходной) – описание ошибки.
   - Возвращаемое значение (boolean) – Истина, операция выполнена успешно.

ЗАМЕЧАНИЕ. Более подробные сведения о веб-сервисе TariffControl можно получить, выведя WSDL-описание веб-сервиса (перейдя в веб-обозревателе по адресу типа https://имя-сайта/a/adm/ws/TariffControl_1_0_0_1?wsdl).

4.4. Алгоритмы

4.4.1. Получение лицензии

Менеджер сервиса	Сторонняя система
	НачатьТранзакцию(). Операция = Новый GUID. ОперацииСЛицензиями += {Операция, ПолучениеЛицензии}. НомерВызова = Текущий() + 1. МС.AccuireLicenses(Операция, Тип, Количество, НомерВызова, Таймаут).
НачатьТранзакцию(). ЗанятыеЛицензии.Количество += Количество. ДатаАктуальности = ТекущаяДата() + Таймаут. ОперацииСЛицензиями += {Операция, ПолучениеЛицензии, Количество, ДатаАктуальности}. МС.НомерВызова = НомерВызова. ЗафиксироватьТранзакцию().	
	Ресурс = Новый GUID. ОперацииСЛицензиями = {Операция, СозданиеРесурса, Ресурс}. НовыйРесурс(Ресурс). НомерВызова = Текущий() + 1. МС.CommitAccuireLicenses(Опер ация, НомерВызова).
НачатьТранзакцию(). ОперацииСЛицензиями = {Операция, ЛицензияПолучена}. МС.НомерВызова = НомерВызова.	

Менеджер сервиса	Сторонняя система
ЗафиксироватьТранзакцию().	
	ОперацииСЛицензиями = {Операция, РесурсСоздан}. ЗафиксироватьТранзакцию().

4.4.1.1. Обработка ошибок получения во время выполнения

Менеджер сервиса	Сторонняя система
	НачатьТранзакцию(). АварияНаЛюбомШаге(). НомерВызова = Текущий() + 1. МС.RollbackAccuireLicenses(Операция, НомерВызова).
НачатьТранзакцию(). ЗанятыеЛицензии.Количество -= НайтиЗанятыеЛицензии(Операция). ОперацииСЛицензиями -= {Операция}. МС.НомерВызова = НомерВызова. ЗафиксироватьТранзакцию().	
	Удалить(Ресурс). ОтменитьТранзакцию().

4.4.2. Освобождение лицензии

Менеджер сервиса	Сторонняя система
	НачатьТранзакцию(). Операция = НайтиОперациюПоРесурсу(Ресурс). ОперацииСЛицензиями = {Операция, УдалениеРесурса, Ресурс}. Удалить(Ресурс). НомерВызова = Текущий() + 1. МС.ReleaseLicenses(Операция, НомерВызова, Таймаут).

Менеджер сервиса	Сторонняя система
НачатьТранзакцию(). ЗанятыеЛицензии.Количество -= НайтиЗанятыеЛицензии(Операция). ОперацииСЛицензиями -= {Операция}. МС.НомерВызова = НомерВызова. ЗафиксироватьТранзакцию().	
	ОперацииСЛицензиями -= {Операция}. ЗафиксироватьТранзакцию().

4.4.3. Очистка просроченных операций в менеджере сервиса

В соответствии с расписанием (по умолчанию раз в час) запускается регламентное задание, которое ищет в регистре сведений ОперацииСЛицензиями записи, у которых ДатаАктуальности меньше текущей даты и состояние равно ПолучениеЛицензии.

Если найдена запись в состоянии ПолучениеЛицензии, то она удаляется и уменьшается ЗанятыеЛицензии.Количество на указанное значение в соответствующей записи ОперацииСЛицензиями.

4.4.4. Отслеживание потери синхронизации

Лицензируемый сервис должен периодически вызывать операцию GetLastAccessTime() и:

- сравнивать разницу между текущим и предыдущим вызовом. Если разница, например, в 2 раза больше, чем интервал вызова операции GetLastAccessTime(), то необходимо начать процесс синхронизации;
- записывать у себя результат последнего вызова GetLastAccessTime().

4.4.5. Обработка переполнения счетчика вызовов

Менеджер сервиса и каждый лицензируемый сервис должны вести учет успешных вызовов, номер которых (LastCall) используется в операции IsSyncRequired() веб сервиса TariffControl. При этом, рано или поздно, значение этого счетчика может стать настолько большим, что вызовет переполнение (превысит максимально возможное значение хранилища счетчика). Чтобы этого не произошло, требуется в каждом лицензируемом сервисе:

- увеличивать счетчик на единицу, если его текущее значение меньше 2 000 000 000;
- устанавливать счетчик в значение 0, если его текущее значение равно 2 000 000 000.

2 000 000 000 – это наибольшее округленное значение для знакового целочисленного типа размером в 4 байта, которым располагает любой современный лицензируемый сервис.

Пример увеличения значений счетчика с учетом переполнения:

```
Если Счетчик < 2000000000 Тогда
  Счетчик = Счетчик + 1;
Иначе
  Счетчик = 0;
КонецЕсли;
```

4.4.6. Синхронизация данных

Для возможности синхронизации менеджера сервиса и удаленного сервиса, работающего с веб сервисом TariffControl, удаленный сервис должен выполнять следующие требования:

1. Создать у себя счетчик обращений к веб сервису TariffControl и увеличивать его на единицу после каждого успешного вызова:
   - AcquireLicenses(),
   - CommitAcquireLicenses(),
   - ReleaseLicenses(),
   - RollbackAcquireLicenses(),
   - Synchronize().
2. Если перечисленные операции возвращают состояние SynchronizationRequired, то необходимо:
   - приостановить текущую работу;
   - выполнить операцию IsSynchronized();

- если результат IsSynchronized() равен SynchronizationRequired , то выполнить операцию Synchronize();
- повторить выполнение действий, вернувших состояние SynchronizationRequired.

При выполнении операции Synchronize() в менеджере сервиса удаляются все данные соответствующие лицензируемому сервису и на их место записываются новые данные, переданные в параметре Licenses операции Synchronize().

Перед выполнением синхронизации следует дождаться завершения всех активных вызовов операций веб сервиса TariffControl.

4.5. Конфигурация TariffControlTest

Для тестирования веб сервиса TariffControl и демонстрации примера реализации лицензируемого сервиса, использующего веб сервис TariffControl, разработана конфигурация TariffControlTest.

Возможности этой конфигурации:

- Получение соответствий тарифов;
- Получение и освобождение лицензий, с возможностью тестирования синхронизации лицензий.

Также конфигурация предоставляет возможности:

- пошагового выполнения всех операций для отладки и для получения представления о работе механизма работы веб-сервиса TariffControl;
- многопоточной обработки операций по работе с лицензиями для организации нагрузочного тестирования.

Глава 5. API для идей пользователей

В менеджере сервиса реализован веб-сервис UsersIdeas_1_0_0_1, позволяющий внешним веб-сервисам осуществлять удаленный доступ к хранимым менеджером сервиса данным об идеях пользователей, добавлять идеи, добавлять и удалять комментарии к идеям, и т. д. В этой главе мы расскажем о том, как использовать этот веб-сервис.

5.1. Веб-сервис UsersIdeas_1_0_0_1

Для получения информации по идеям и управления идеями, комментариями по идеям, голосами за идеи, создан веб сервис UsersIdeas_1_0_0_1 со следующей структурой:

- имя: UsersIdeas_1_0_0_1;
- подсистема: ОбслуживаниеУправлениеИнформационнымЦентромИдеиПользователей;
- пространство имен: http://www.1c.ru/1cFresh/InformationCenter/UsersIdeas/1.0.0.1;
- имя файла публикации: usersideas_1_0_0_1.1cws;
- пакеты XDTO: UsersIdeas_1_0_0_1.

5.2. Авторизация

Для использования методов веб сервиса UsersIdeas_1_0_0_1 приложение должно авторизоваться от имени служебного пользователя с ролью УдаленныйДоступ.

5.3. Операции веб-сервиса

Операции веб-сервиса UsersIdeas_1_0_0_1 следующие.

1. Boolean addIdea (UserId, HTMLText, Attachments, Subject, Name) – добавить идею.

Здесь UserId, HTMLText, Attachments, Subject, Name — входящие параметры:

- UserId (string) — идентификатор пользователя;
- HTMLText (string) — содержание идеи в виде HTML-текста;
- Attachments (тип AttachmentsList) — список вложений;
- Subject (string) – предмет идеи;
- Name (string) – наименование идеи;

Возвращаемое значение типа Boolean сообщает, добавлена идея или нет.

2. Boolean addIdeaComment (IdeaId, UserId, CommentId, Text) – добавить комментарий к идее.

   Здесь IdeaId, UserId, CommentId, Text — входящие параметры:

   - IdeaId (string) – идентификатор идеи;
   - UserId (string) – идентификатор пользователя;
   - CommentId (string) – идентификатор комментария к идее;
   - Text (string) – текст комментария.

   Возвращаемое значение типа Boolean сообщает, добавлен комментарий или нет.

3. Boolean addVote (IdeaId, UserId, Vote) – добавить голос к идее.

   Здесь IdeaId, UserId, Vote — входящие параметры:

   - IdeaId (string) – идентификатор идеи;
   - UserId (string) – идентификатор пользователя;
   - Vote (int) – голос за (1) или против (–1) идеи.

   Возвращаемое значение типа Boolean сообщает, добавлен голос или нет.

4. deleteIdeaComment(CommentId) – удалить комментарий к идее.

   Здесь CommentId — входящий параметр:

   - CommentId (String) – идентификатор комментария к идее.

5. IdeaPresentation getIdea(Id, UserId, PageCommentNumber, CountCommentsOnPage) – получить идею.

Здесь Id, UserId, PageCommentNumber, CountCommentsOnPage — входящие параметры:

- Id (string) — идентификатор идеи;
- UserId (string) – идентификатор пользователя;
- PageCommentNumber (int) — номер страницы комментариев;
- CountCommentsOnPage (int) – количество комментариев на странице.

Возвращаемое значение типа IdeaPresentation содержит идею, список комментариев и количество комментариев.

6. IdeaListPresentation getIdeas(StatusFilter, SubjectFilter, Sort, PageNumber, UserId, CountIdeasOnPage) – получить идеи.

   Здесь StatusFilter, SubjectFilter, Sort, PageNumber, UserId, CountIdeasOnPage — входящие параметры:

   - StatusFilter (тип StatusFilter) — фильтр по предмету;
   - SubjectFilter (тип SubjectFilterArray) — Фильтр по статусу;
   - Sort (тип SortType) – тип сортировки;
   - PageNumber (int) — номер страницы;
   - UserId (string) — идентификатор пользователя;
   - CountIdeasOnPage (int) – количество идей на странице.

   Возвращаемое значение типа IdeaListPresentation содержит представление списка идей.

7. IdeaListPresentation searchIdeas(Request, StatusFilter, SubjectsFilter, PageNumber, UserId, CountIdeasOnPage) – найти список идей по полнотекстовому поиску.

   Здесь Request, StatusFilter, SubjectFilter, Sort, PageNumber, UserId, CountIdeasOnPage — входящие параметры:

   - Request (string) — текст запроса;
   - StatusFilter (тип StatusFilter) — фильтр по предмету;
   - SubjectFilter (тип SubjectFilterArray) — Фильтр по статусу;
   - Sort (тип SortType) – тип сортировки;
   - PageNumber (int) — номер страницы;
   - UserId (string) — идентификатор пользователя;
   - CountIdeasOnPage (int) – количество идей на странице.

Возвращаемое значение типа IdeaListPresentation содержит представление списка идей.

ЗАМЕЧАНИЕ. Более подробные сведения о веб-сервисе UsersIdeas_1_0_0_1 и описания используемых им типов Idea, IdeaComment, IdeaPresentation, IdeaListPresentation, SortType, StatusFilter, StatusFilterArray, SubjectFilterArray, Attachment, AttachmentsList можно получить, выведя WSDL-описание веб-сервиса (перейдя в веб-обозревателе по адресу типа https://имя-сайта/a/adm/ws/UsersIdeas_1_0_0_1?wsdl).

Глава 6. Веб-сервис поиска обслуживающих организаций

Веб-сервис поиска обслуживающих организаций используется менеджером сервиса для получения списка обслуживающих организаций, удовлетворяющих критериям поиска и передачи этого списка сайту. Требования к веб-сервису:

- Имя веб-сервиса — PartnersServiceImplService;
- Порт — PartnersServicePort;
- URI пространства имен — http://ws.partners.onec.ru.

WSDL-файл с описанием веб-сервиса поиска обслуживающих организаций имеется в интернете, он доступен по адресу: http://partnersws.1c.ru/partws/services/PartnersService?wsdl

6.1. Типы данных

Веб-сервис должен поддерживать работу с типами данных, перечисленными ниже.

Parameter

Описание:

Универсальная структура для обмена.

Свойства типа:

`Name`

Тип: [xsd:string]. Содержит имя параметра.

`Value`

Тип: [xsd:string]. Содержит значение параметра.

Parameters

Описание:

Список элементов типа [http://ws.partners.onec.ru/Parameter].

PagingInfo

Описание:

Структура, задающая свойства выходного списка.

Свойства типа:

`Start`

Тип: [xsd:int]. Начальная страница.

`Count`

Тип: [xsd:int]. Количество на странице.

`Total`

Тип: [xsd:int]. Всего элементов.

Partner

Описание:

Структура, задающая свойства партнера.

Свойства типа:

`Parameter`

Тип: [http://ws.partners.onec.ru/Parameter]. Структура, содержащая имя логина обслуживающей организации и его значение.

Partners

Описание:

Список элементов типа [http://ws.partners.onec.ru/Partner].

WSException

Описание:

Ошибка.

Свойства типа:

`Code`

Тип: [xsd:int]. Код ошибки.

`Message`

Тип: [xsd: string]. Описание ошибки.

6.2. Методы

Веб-сервис должен содержать методы, перечисленные ниже.

Search

Описание:

Производит поиск обслуживающих организаций, удовлетворяющих критериям поиска.

Параметры:

`Parameters входной`

Тип: [http://ws.partners.onec.ru/Parameters]. Список критериев поиска и их значения.

`PagingInfo входной/выходной`

Тип: [http://ws.partners.onec.ru/PagingInfo]. Устанавливаются параметры вывода, возвращается общее количество.

`Partners выходной`

Тип: [http://ws.partners.onec.ru/Partners]. Список найденных партнеров.

`WSException выходной`

Тип: [http://ws.partners.onec.ru/WSException]. При неудачном поиске содержит код ошибки.

Возвращаемое значение:

Тип произвольный, значение игнорируется.

SetPartners

Описание:

Устанавливает список обслуживающих организаций.

Параметры:

`Partners входной`

Тип: [http://ws.partners.onec.ru/Partners]. Список обслуживающих организаций.

Глава 7. Веб-сервис внешней регистрации

7.1. Назначение веб-сервиса

Веб-сервис внешней регистрации ExternalRegistration_1_0_0_1 используется промо-сайтами, создаваемыми обслуживающими организациями, для регистрации пользователей в сервисе.

Веб-сервис внешней регистрации предоставляется информационной базой менеджера сервиса.

7.2. Публикация веб-сервиса

Для того, чтобы обслуживающая организация могла регистрировать пользователей (владельцев новых абонентов) с помощью веб-сервиса внешней регистрации, провайдер сервиса должен опубликовать веб-сервис внешней регистрации информационной базы менеджера сервиса ExternalRegistration_1_0_0_1, так, чтобы эта публикация была доступна промо-сайтам, которые будут вызывать веб-сервис внешней регистрации.

ПРИМЕЧАНИЯ. 1. Целесообразно ограничить доступ к веб-сервису внешней регистрации, чтобы к нему можно было обращаться только с IP-адресов промо-сайтов.

2. Если промо-сайты и информационная база менеджера сервиса располагаются в одной локальной сети, то в качестве публикации веб-сервиса внешней регистрации может использоваться внутренняя публикация менеджера сервиса.

7.3. Использование веб-сервиса

7.3.1. Особенности регистрации с помощью веб-сервиса

Каждый пользователь, регистрируемый с помощью веб-сервиса внешней регистрации, становится владельцем нового абонента, после

чего он может самостоятельно приглашать пользователей в сервис в качестве пользователей этого абонента.

Обслуживающей организацией для созданного абонента становится та обслуживающая организация, которая создала промо-сайт, через который был зарегистрирован пользователь.

7.3.2. Действия обслуживающей организации

Для использования веб-сервиса обслуживающая организация должна обратиться к провайдеру сервиса и сообщить ему следующие сведения.

1. Тариф, который должен по умолчанию назначаться регистрируемым пользователям (абонентам).
2. Продолжительность применения этого тарифа.
3. Наименования источников внешней регистрации (промо-сайтов, промо-страниц) — если обслуживающая организация желает развернуть несколько промо-страниц и получать аналитические сведения об их использовании.
4. Наименования источников интереса — если обслуживающая организация желает получать аналитические сведения об использовании источников интереса.

От провайдера сервиса обслуживающая организация должна получить следующие сведения.

1. Имя и пароль пользователя или пользователей веб-сервиса внешней регистрации.
2. Идентификаторы источников внешней регистрации (при необходимости получения аналитических сведений об использовании источников внешней регистрации).
3. Идентификаторы источников интереса (при необходимости получения аналитических сведений об их использовании).

7.3.3. Действия провайдера сервиса

Для того, чтобы обслуживающая организация могла регистрировать пользователей (владельцев новых абонентов) с помощью веб-сервиса внешней регистрации, администратор сервиса должен выполнить следующие действия.

1. Создать в менеджере сервиса служебного пользователя, от имени которого будет вызываться веб-сервис внешней регистрации. Этому пользователю должна быть назначена единственная роль Удаленный доступ внешняя регистрация.
2. Создать в менеджере сервиса новый элемент справочника **Разрешения внешней регистрации**, и включить туда следующие сведения:
   - наименование обслуживающей организации;
   - наименование (логин) пользователя для вызова сервиса внешней регистрации;
   - тариф по умолчанию;
   - продолжительность действия этого тарифа.
3. Если обслуживающая организация желает развернуть несколько промо-страниц и получать аналитические сведения об их использовании — включить наименования этих промо-страниц в справочник **Сайты внешней регистрации** менеджера сервиса, и записать сформированные коды созданных элементов справочника **Сайты внешней регистрации**.
4. Если обслуживающая организация желает получать аналитические сведения об использовании источников интереса — включить наименования источников интереса в справочник **Источники интереса внешней регистрации** менеджера сервиса, и записать сформированные коды созданных элементов справочника.
5. Сообщить обслуживающей организации:
   - имя и пароль пользователя веб-сервиса внешней регистрации;
   - идентификаторы источников внешней регистрации (при необходимости);
   - идентификаторы источников интереса (при необходимости).

7.3.4. Сценарий использования веб-сервиса

После того, как все указанные подготовительные действия выполнены, обслуживающая организация создала и опубликовала промо-страницы, с помощью которых пользователи будут регистрироваться в сервисе, регистрация пользователей может выполняться по следующему сценарию.

1. Будущий пользователь сервиса заходит на веб-страницу промо-сайта, заполняет в форме регистрации свои контактные данные (имя, электронную почту и телефон), и нажимает кнопку, Зарегистрироваться.
2. Промо-сайт вызывает веб-сервис внешней регистрации от имени пользователя, созданного провайдером для внешней регистрации пользователей этой обслуживающей организации.
3. Если веб-сервис был вызван с параметром SendEMail=true, то менеджер сервиса посылает на указанный адрес электронной почты электронное письмо, содержащее URL-ссылку на страницу завершения регистрации.
4. Веб-сервис возвращает URL на страницу завершения регистрации.
5. Промо-сайт выполняет переадресацию на эту страницу, и там пользователь устанавливает свой пароль и завершает регистрацию.
6. Пользователь должен открыть это письмо, перейти по полученной ссылке, установить свой пароль и завершить регистрацию.

ПРИМЕЧАНИЕ. Вызов веб-сервиса с параметром SendEMail=true предназначен для использования в тех мобильных приложениях, в которых веб-страница завершения регистрации отображается некорректно.

7.4. Описание веб-сервиса

Веб-сервис внешней регистрации предоставляется информационной базой менеджера сервиса и имеет следующие свойства.

- имя — ExternalRegistration_1_0_0_1 ;
- пространство имен: http://www.1c.ru/1cFresh/ExternalRegistration/1.0.0.1
- имя файла публикации ExternalRegistration_1_0_0_1.1cws

WSDL-описание веб-сервиса внешней регистрации доступно по адресу:

адрес-публикации-менеджера-сервиса/ws/ExternalRegistration_1_0_0_1?wsdl

Веб-сервис предоставляет один метод BeginRegistration, выполняющий запрос на внешнюю регистрацию нового пользователя сервиса.

Параметры метода BeginRegistration:

Параметр	Тип	Описание
Email	string	Адрес электронной почты регистрируемого пользователя
Name	string	Логин пользователя
PhoneNumber	string	Номер телефона регистрируемого пользователя
SendEmail	boolean	При значении параметра true менеджер сервиса посылает по указанному в параметре **Email** адресу электронное письмо с URL страницы завершения регистрации.
SiteId	int	Идентификатор сайта
SourceId	int	Номер источника интереса
ErrorInfo	string	Описание ошибки или пусто.
result	string	URL страницы завершения регистрации.

Здесь параметры **ErrorInfo** и **Result** — выходные, остальные параметры входные.

Значения параметров по умолчанию:

- **PhoneNumber** — пустая строка;
- **SendEmail** — false.

Глава 8. Описание дополнительной обработки обновления

Дополнительная обработка обновления информационной базы используется агентом сервиса при выполнении сценария автоматического обновления конфигурации информационной базы. Модуль объекта данной обработки должен содержать следующие экспортируемые процедуры.

8.1. Процедура ПередОбновлением

Описание:

Обработчик ПередОбновлением вызывается агентом сервиса перед выполнением фазы обновления общих данных в обновляемой информационной базе (см. раздел 21.1 в документе «1С:Технология публикации решений 1cFresh»).

Параметры:

`СтандартнаяОбработка`

Тип: Булево. Флаг выполнения вызова фазы обновления общих данных. Если значение установить равным Ложь, агент сервиса пропустит вызов фазы обновления общих данных. При этом выполнение действия сценария будет считаться успешным и выполнение задания продолжится.

8.2. Процедура ПослеОбновления

Описание:

Обработчик ПослеОбновления вызывается агентом сервиса после выполнения фазы обновления общих данных в обновляемой информационной базе.

Глава 9. Программный интерфейс работы с поставляемыми данными

9.1. Общий модуль обработчика данных

ДоступныНовыеДанные

Описание:

Функция проверяет тип и характеристики данных и при необходимости планирует загрузку данных.

Синтаксис:

ДоступныНовыеДанные(Знач Дескриптор, Загружать)

Параметры:

Дескриптор *По значению*

Тип: ОбъектXDTO. Содержит дескриптор загружаемых данных.

Загружать

Тип: Булево. Признак того, что данные должны быть загружены.

Пример:

```
Если Дескриптор.DataType = "КурсыВалютЗаДень" Тогда
  Загружать = Истина;
КонецЕсли;
```

ОбработатьНовыеДанные

Описание:

Процедура вызывается через некоторое время после вызова ДоступныНовыеДанные(), содержит дескриптор данных и путь к файлу на диске. В процедуре следует проверить тип и характеристики данных и в соответствии с ними обработать файл.

Синтаксис:

ОбработатьНовыеДанные(Знач Дескриптор, ПутьКФайлу)

Параметры:

Дескриптор	*По значению*

Тип: ОбъектXDTO. Содержит дескриптор загружаемых данных.

ПутьКФайлу

Тип: Строка. Полное имя извлеченного файла. Файл будет автоматически удален после завершения процедуры.

ОбработкаДанныхОтменена

Описание:

Процедура вызывается в том случае, если на сервере поставляемых данных отсутствует файл, на который ссылается полученный дескриптор, либо если произошло 3 исключения при вызове ОбработатьНовыеДанные().

Синтаксис:

ОбработкаДанныхОтменена(Знач Дескриптор)

Параметры:

Дескриптор	*По значению*

Тип: ОбъектXDTO. Содержит дескриптор данных, обработка которых отменена.

9.2. Программный интерфейс общего модуля ПоставляемыеДанные

ДескрипторыПоставляемыхДанныхИзМенеджера

Описание:

Функция используется для получения дескрипторов данных, удовлетворяющих заданным условиям. Для получения самого файла следует использовать функцию

РаботаВМоделиСервиса.ПолучитьФайлИзХранилищаМенеджераСервиса().

Синтаксис:

ДескрипторыПоставляемыхДанныхИзМенеджера(Знач ВидДанных, Знач Фильтр = Неопределено)

Параметры:

ВидДанных	*По значению*

Тип: Строка. Идентификатор вида поставляемых данных.

Фильтр	*По значению*

Тип: Массив. Элементы должны содержать следующие свойства:

- Код. Тип: Строка.
- Значение. Тип: Строка.

Пример:

```
Дескрипторы =
ПоставляемыеДанные.ПолучитьДескрипторы("КурсыВалют");

Если Дескрипторы.Descriptor.Количество() < 1 Тогда
  ВызватьИсключение(НСтр("ru = 'В менеджере сервиса отсутствуют
данные КурсыВалют'"));
КонецЕсли;

ИмяФайлаВыгрузки =
РаботаВМоделиСервиса.ПолучитьФайлИзХранилищаМенеджераСервиса(
Дескрипторы.Descriptor[0].FileGUID);
ЧтениеXML = Новый ЧтениеXML();
ЧтениеXML.ОткрытьФайл(ИмяФайлаВыгрузки);
```

ЗагрузитьИОбработатьДанные

Описание:

Процедура может использоваться в связке с методом ДескрипторыПоставляемыхДанныхИзМенеджера() для ручной инициации процесса обработки данных. После вызова метода система поведет себя так, как будто она только что получила уведомление о доступности новых данных, с указанным дескриптором — будет вызван ДоступныНовыеДанные(), а затем, при необходимости, ОбработатьНовыеДанные() для соответствующих обработчиков.

Синтаксис:

ЗагрузитьИОбработатьДанные(Дескриптор)

Параметры:

Дескриптор

Тип: ОбъектXDTO. Содержит дескриптор загружаемых данных.

ЗапроситьВсеДанные

Описание:

Вызов метода приведет к повторной отправке всех имеющихся в менеджере сервиса поставляемых данных в прикладное решение, аналогично тому, как это выполняется при первоначальном подключении прикладного решения к менеджеру сервиса. Исключение составляют те поставляемые данные, для которых установлено свойство SuppressNotifications.

Синтаксис:

ЗапроситьВсеДанные()

9.3. Программный интерфейс работы со справочником ПоставляемыеДанные

СохранитьПоставляемыеДанныеВКэш

Описание:

Данные сохраняются либо в том на диске, либо в поле справочника ПоставляемыеДанные в зависимости от константы ХранитьФайлыВТомахНаДиске и наличия свободных томов. Данные могут быть позднее извлечены при помощи поиска по реквизитам, либо путем указания уникального идентификатора, который передавался в поле Дескриптор.FileGUID. Если в базе уже есть данные с тем же видом данных и набором ключевых характеристик — новые данные замещают старые. При этом используется обновление существующего элемента справочника, а не удаление и создание нового.

Синтаксис:

СохранитьПоставляемыеДанныеВКэш(Знач Дескриптор, Знач ПутьКФайлу)

Параметры:

Дескриптор	*По значению*

Тип: ОбъектXDTO, Структура.

Если передано значение типа ОбъектXDTO, то оно содержит дескриптор загружаемых данных в виде объекта XDTO.

Если параметр имеет тип Структура, то оно содержит дескриптор загружаемых данных в виде структуры, которая содержит свойства ВидДанных, ДатаДобавления, ИдентификаторФайла, Характеристики. Свойство Характеристики содержит массив структур, содержащих свойства Код, Значение, Ключевая.

ПутьКФайлу	*По значению*

Тип: Строка. Полное имя файла.

УдалитьПоставляемыеДанныеИзКэша

Описание:

Удаляет файл из кеша.

Синтаксис:

УдалитьПоставляемыеДанныеИзКэша(Знач ПоставляемыеДанные)

Параметры:

ПоставляемыеДанные	*По значению*

Тип: СправочникСсылка.ПоставляемыеДанные или УникальныйИдентификатор. Если параметр имеет тип УникальныйИдентификатор, то параметр содержит идентификатор файла, используемый в подсистеме Работа с файлами в модели сервиса.

ПоставляемыеДанныеИзКэша

Описание:

Возвращает двоичные данные присоединенного файла.

Синтаксис:

ПоставляемыеДанныеИзКэша(Знач ПоставляемыеДанные)

Параметры:

ПоставляемыеДанные	*По значению*

Тип: СправочникСсылка.ПоставляемыеДанные или УникальныйИдентификатор. Если параметр имеет тип УникальныйИдентификатор, то параметр содержит идентификатор файла, используемый в подсистеме Работа с файлами в модели сервиса.

Возвращаемое значение:

Значение типа ДвоичныеДанные.

ЕстьВКеше

Описание:

Проверяет наличие данных с указанными ключевыми характеристиками в кеше.

Может использоваться для исключения повторной обработки ранее полученных данных.

Синтаксис:

ЕстьВКеше(Знач Дескриптор)

Параметры:

Дескриптор	*По значению*

Тип: ОбъектXDTO. Содержит дескриптор загружаемых данных.

Возвращаемое значение:

Значение типа Булево.

СсылкиПоставляемыхДанныхИзКэша

Описание:

Возвращает массив ссылок на данные, удовлетворяющие заданным условиям. Результаты могут быть позднее использованы в вызовах ПоставляемыеДанныеИзКэша() или

УдалитьПоставляемыеДанныеИзКэша(). Функция должна использоваться, если не требуется получения полных дескрипторов хранящихся данных, достаточно знания того, что они обладают указанными в фильтре характеристиками.

Синтаксис:

ССылкиПоставляемыхДанныхИзКэша(Знач ВидДанных, Знач Фильтр = Неопределено)

Параметры:

ВидДанных	*По значению*

Тип: Строка. Идентификатор вида поставляемых данных.

Фильтр	*По значению*

Тип: Массив. Элементы должны содержать свойства:

- Код. Тип: Строка.
- Значение. Тип: Строка.

Возвращаемое значение:

Значение типа Массив.

ДескрипторыПоставляемыхДанныхИзКэша

Описание:

Получить дескрипторы поставляемых данных по заданным условиям. Для получения самого файла необходимо вызвать метод ПолучитьПоставляемыеДанныеИзКэша(). Функция аналогична функции ДескрипторыПоставляемыхДанныхИзМенеджера(), но выполняет поиск данных в кэше.

Синтаксис:

ДескрипторыПоставляемыхДанныхИзКэша(Знач ВидДанных, Знач Фильтр, Знач ВВидеXDTO)

Параметры:

ВидДанных	*По значению*

Тип: Строка. Содержит идентификатор вида поставляемых данных.

Фильтр	*По значению*

Тип: Массив. Каждый элемент массива должен содержать структуру, состоящую из следующих элементов:

- Код — строка, код характеристики;
- Значение — строка. Значение по умолчанию: Неопределено.

ВВидеXDTO	*По значению*

Тип: Булево. Указывает, необходимость возвращать значение в виде объекта XDTO.

Значение по умолчанию: Ложь.

Возвращаемое значение:

Возвращаемое значение

ОбъектXDTO типа ArrayOfDescriptor или массив структур с полями ВидДанных, ДатаДобавления, ИдентификаторФайла, Характеристики, где Характеристики — массив структур с полями Код, Значение, Ключевая.

9.4. Программный интерфейс взаимодействия с системой восстановления после сбоев

ОбластиТребующиеОбработки

Описание:

Возвращает список областей данных, в которые еще не были скопированы поставляемые данные. В случае первого вызова функции возвращается полный набор доступных областей. При последующем вызове, при восстановлении после сбоя, будут возвращены только необработанные области. После копирования данных в область следует вызвать ОбластьОбработана().

Синтаксис:

ОбластиТребующиеОбработки(Знач ИдентификаторФайла, Знач КодОбработчика)

Параметры:

ИдентификаторФайла	*По значению*

Тип: УникальныйИдентификатор. Идентификатор файла поставляемых данных, используемый в подсистеме Работа с файлами в модели сервиса.

КодОбработчика	*По значению*

Тип: Строка. Идентификатор обработчика.

Возвращаемое значение:

Значение типа Массив.

ОбластьОбработана

Описание:

Удаляет область из списка необработанных.

Синтаксис:

ОбластьОбработана(Знач ИдентификаторФайла, Знач КодОбработчика, Знач ОбластьДанных)

Параметры:

ИдентификаторФайла	*По значению*

Тип: УникальныйИдентификатор. Идентификатор файла поставляемых данных, используемый в подсистеме Работа с файлами в модели сервиса.

КодОбработчика	*По значению*

Тип: Строка. Идентификатор обработчика.

ОбластьДанных	*По значению*

Тип: Число, Неопределено.

Если значение параметра имеет тип Число, то параметр является идентификатором обработанной области данных.

Если в качестве значения параметра передано Неопределено, то из списка необработанных областей удаляются все записи об указанных данных. Такой вариант вызова полезно использовать внутри вызова ОбработкаДанныхОтменена().