Использование 2iS:Интеграция как шины веб-сервисов

Краткое описание

Используемые сокращения:

1. Сторона 1 = инфобаза, предоставляющая веб-сервис = поставщик сервиса
2. Сторона 2 = инфобаза, из которой выполняется вызов веб-сервиса = потребитель сервиса

Интеграция может использоваться как шина веб-сервисов при организации вызова веб-сервиса одной инфобазы (стороны 1) из другой инфобазы (на стороне 2). При этом не требуется вносить изменения в конфигурацию стороны 1: подключение к стороне 1 и возврат результата берет на себя Интеграция, сам веб-сервис также хранится в Интеграции.

Примеры использования веб-сервисов:

1. Получение списка номенклатуры
2. Получение долга контрагента
3. Получение остатка товара
4. Получение курса валюты

В каждом из примеров для реализации "классического" веб-сервиса разработчику пришлось бы создавать и описывать этот веб-сервис на стороне 1, что повлекло бы внесение изменений в конфигурацию инфобазы, поставляющей веб-сервис.
С использованием Интеграции у разработчика появляется возможность реализовывать веб-сервисы для любой инфобазы без изменения конфигурации этой инфобазы.

Описание тестового примера

Далее описывается процесс и особенности создания веб-сервиса на примере.

Стороной 1 (поставщиком веб-сервиса) будет выступать инфобаза с конфигурацией "Бухгалтерия предприятия, ред. 3.0", далее БП.
Стороной 2 (потребителем веб-сервиса) будет выступать инфобаза с конфигурацией "Управление торговлей, ред. 11", далее УТ.

Для УТ будет создана внешняя обработка, демонстрирующая вызов веб-сервиса:

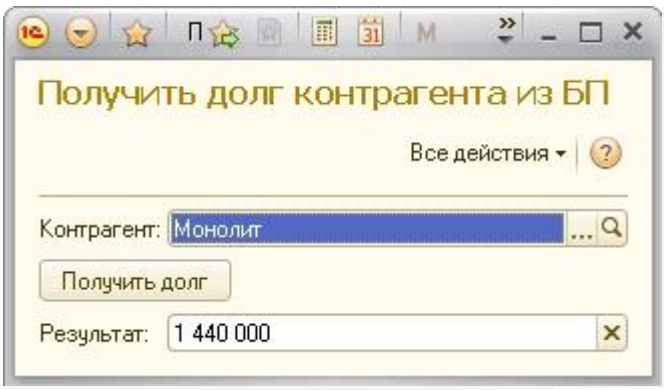


Рисунок RP3-000729. Обработка для УТ
Рисунок 1.

Сценарий использования:

1. Пользователь УТ выбирает контрагента в своей инфобазе

2. После этого пользователь нажимает кнопку **Получить долг**
3. Осуществляется вызов веб-сервиса инфобазы БП через Интеграцию
4. На стороне БП вычисляется свернутый остаток на счетах 60 и 62 для переданного* контрагента
5. Интеграция передает в УТ результат выполнения веб-сервиса
6. В поле **Результат** отображается результат выполнения веб-сервиса

*В рассматриваемом примере идентификация контрагента в базе БП будет осуществляться по его уникальному идентификатору, поэтому входным параметром для веб-сервиса будет являться уникальный идентификатор (в виде строки) контрагента из базы УТ. Разработчик может использовать любой иной способ сопоставления передаваемых между инфобазами значений (при этом желательно, чтобы сопоставление осуществлялось на вызывающей стороне).

Однако, прежде чем вызывать и использовать веб-сервис, его сначала необходимо создать. Делается это путем создания в Интеграции *автозадания* по определенным правилам. Поскольку *реальный* (т.е. на уровне объектов метаданных конфигурации) веб-сервис при этом не создается, будем называть создаваемый веб-сервис *виртуальным*.

Создание веб-сервиса (часть 1)

Опишем обязательные и рекомендуемые признаки виртуального веб-сервиса:

1. Автозадание должно быть с включенным флагом **Веб-сервис**
    1. Флаг позволяет управлять доступностью автозадания через веб-сервис: если флаг выключен, то веб-сервис будет недоступен для вызова
2. Имя веб-сервиса должно быть задано в реквизите автозадания **Имя в веб-сервисе**. Рекомендуется использовать латиницу.
3. Флаг **Мультисеансовость** при его установке допускает одновременный вызов (выполнение) веб-сервиса из разных сеансов
    1. В рассматриваемом нами примере (по получению долга контрагента) данный флаг будет установлен.
4. В реквизите **Инфобаза** группы свойств *Инфобаза-исполнитель* необходимо указать ту инфобазу (и параметры подключения к ней), в которой предполагается выполнение создаваемого веб-сервиса
    1. В рассматриваемом нами примере такой инфобазой будет выступать инфобаза БП
    2. Пользователю инфобазы, под которым будет выполняться подключение к инфобазе, рекомендуется установить режим запуска *Обычное приложение* (для удобства отладки)
        1. Для этого в демо-базе БП у пользователя *Любимов (администратор)* был изменен режим запуска

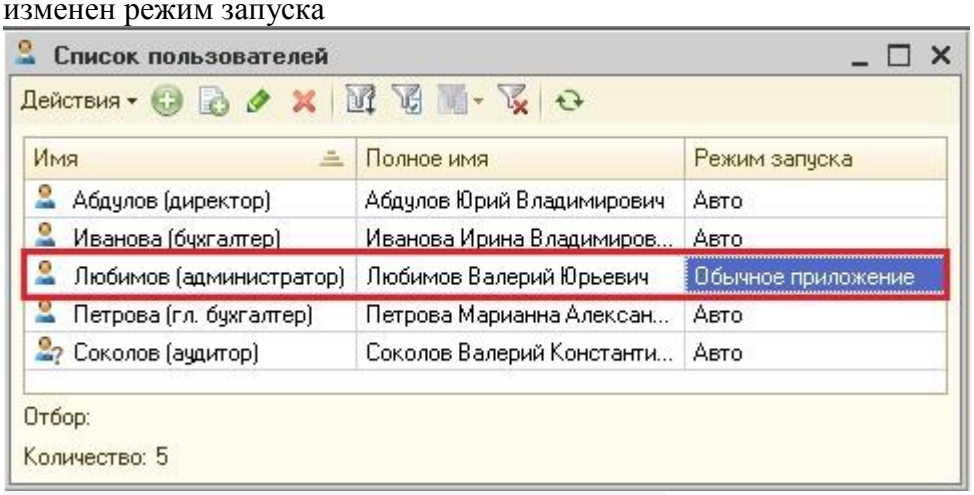


Рисунок RP3-000730. Любимов (режим запуска)

Рисунок 2.

5. Вид автозадания (реквизит **Вид**) должен быть создан по определенным правилам (см. в главе "Создание веб-сервиса, часть 3")

Пример созданного виртуального веб-сервиса в Интеграции:

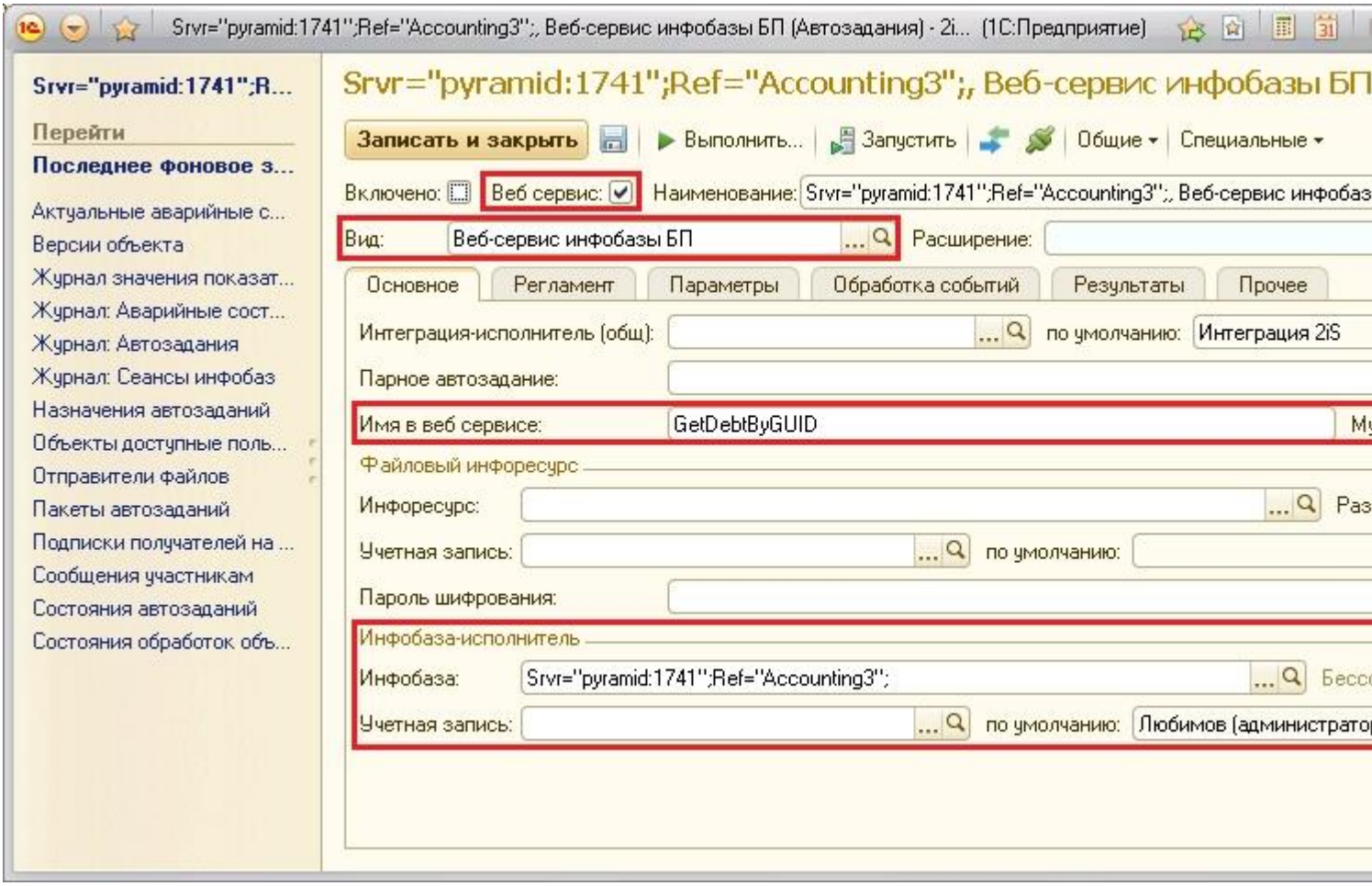


Рисунок RP3-000731. Веб-сервис в Интеграции
Рисунок 3.

Создание веб-сервиса (часть 2)

Для виртуального веб-сервиса (автозадания) в Интеграции будет необходимо создать *сервис*, который будет выполняться этим автозаданием.
Данный сервис будет представлять из себя внешнюю обработку, выполняемую в базе БП.

Данная обработка будет:

1. Принимать на вход строку с уникальным идентификатором контрагента
2. Получать из регистра бухгалтерии инфобазы БП долг контрагента
3. Возвращать долг контрагента

Для того, чтобы созданная внешняя обработка могла использоваться в Интеграции как сервис, она должна удовлетворять набору требований:

1. В тексте модуля обработки должна содержаться экспортная функция `мВыполнитьСервис()`
2. В реквизитах обработки должны быть описаны все входные параметры веб-сервиса
    1. В рассматриваемом нами примере у виртуального веб-сервиса будет один входной параметр (GUID контрагента), поэтому у обработки будет один строковый реквизит **СтрокаGUID**
3. Экспортная функция `мВыполнитьСервис()` должна возвращать значение, содержащее результат выполнения веб-сервиса

1. В рассматриваемом нами примере виртуальный веб-сервис должен возвращать только одно значение (долг контрагента), поэтому упомянутая экспортная функция будет возвращать строковое представление числа (долга контрагента)
2. Если веб-сервис должен возвращать несколько значений, то их следует поместить в коллекцию (структура, массив, таблица значений и др.)
    1. Данный пример приведен в главе "Усложненный пример веб-сервиса"

На рисунке показан пример созданной внешней обработки:

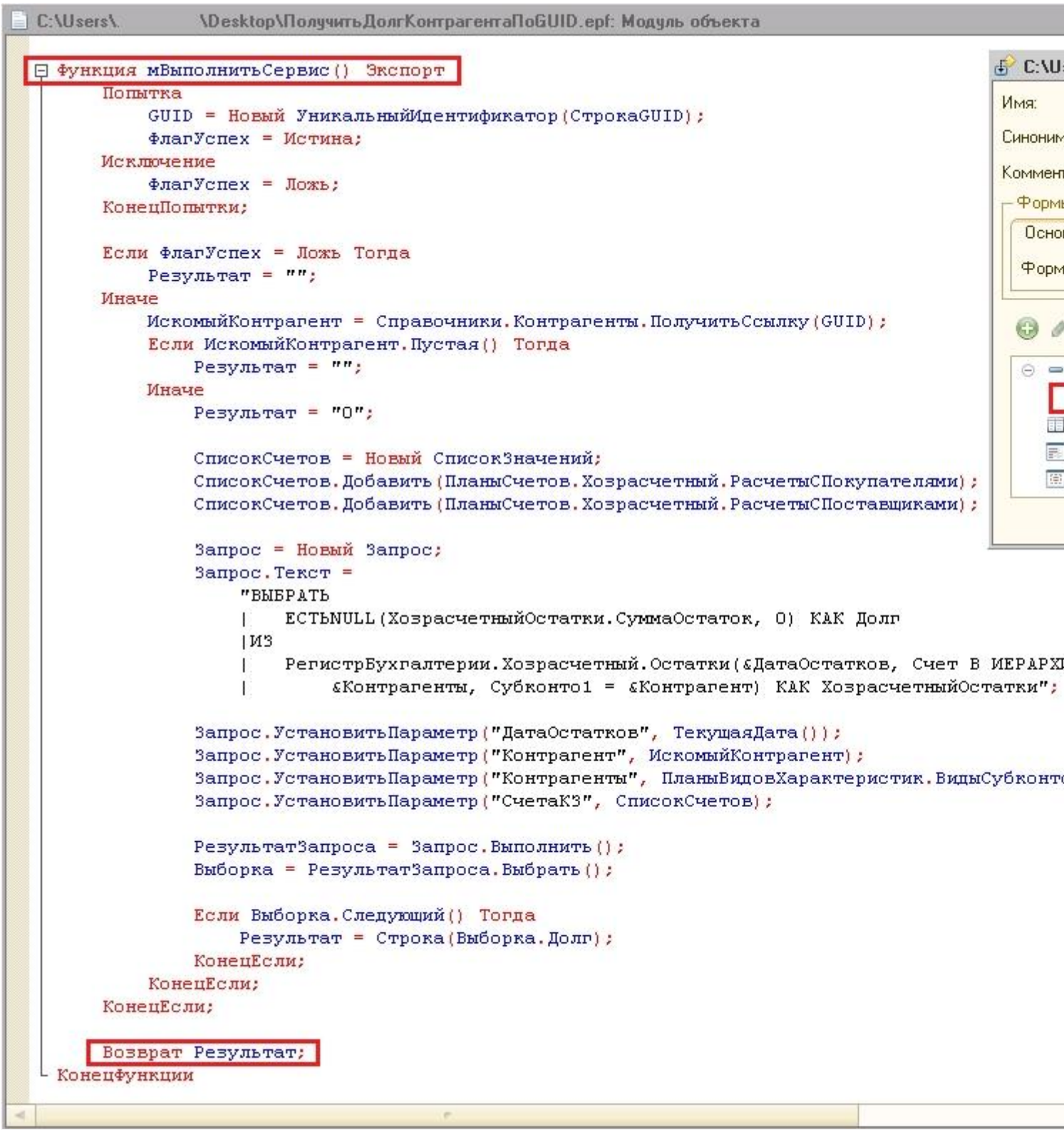


Рисунок RP3-000732. Внешняя обработка для БП
Рисунок 4.

Создание веб-сервиса (часть 3)

Возвращаемся к созданию автозадания (виртуального веб-сервиса) в Интеграции. Как было сказано в части 1, *вид автозадания* должен удовлетворять определенным требованиям:

1. Флаг **Выполняется в интеграции** должен быть снят
    1. Флаг **Требует инфобазу** будет взведен при этом автоматически
2. Для вида автозадания необходимо создать и указать **Сервис** (об этом ниже)
    1. Реквизит **Конфигурация метаданных** будет заполнен автоматически при выборе сервиса

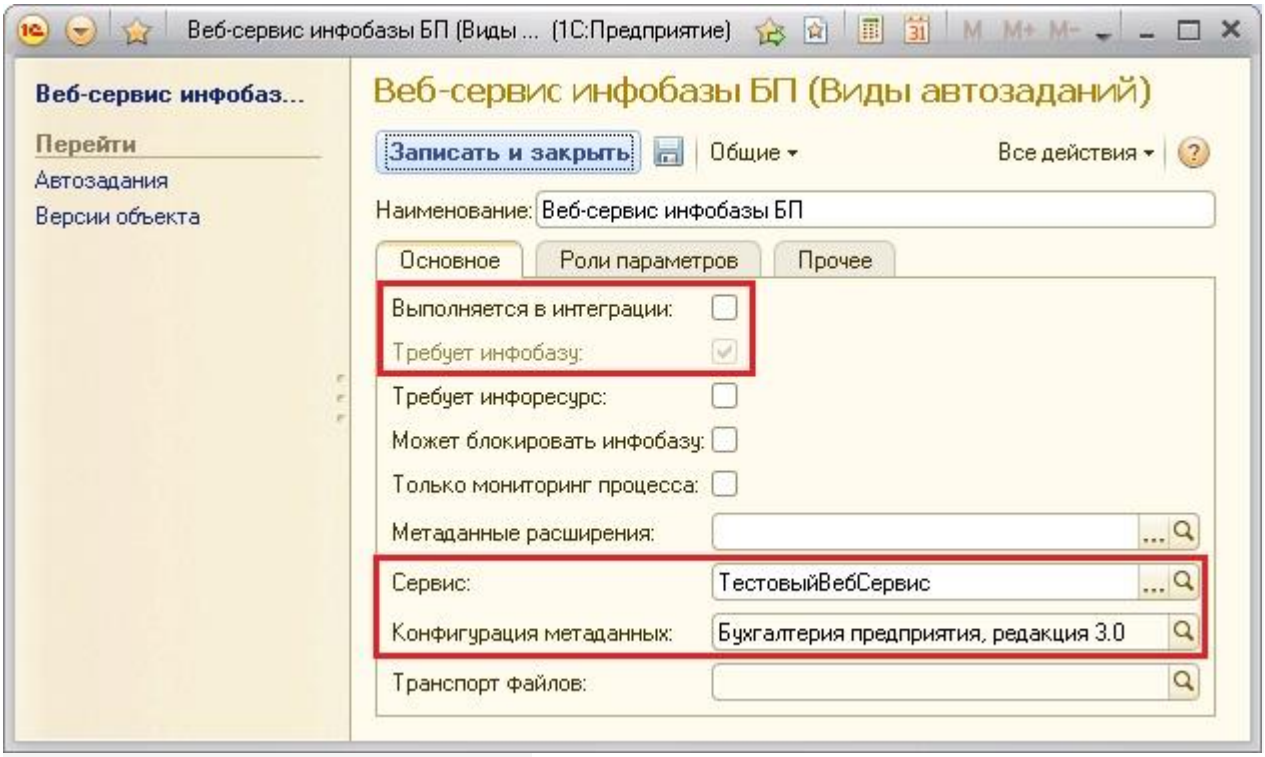


Рисунок RP3-000733. Вид автозадания
Рисунок 5.

Создаваемый *сервис*, в свою очередь, также должен удовлетворять определеным требованиям:

1. **Тип сервиса** = Объект
2. **Конфигурация метаданных** (вкладка *Прочие*) = конфигурация инфобазы, в которой даный сервис будет выполняться
    1. В рассматриваемом примере это *Бухгалтерия предприятия, редакция 3.0*

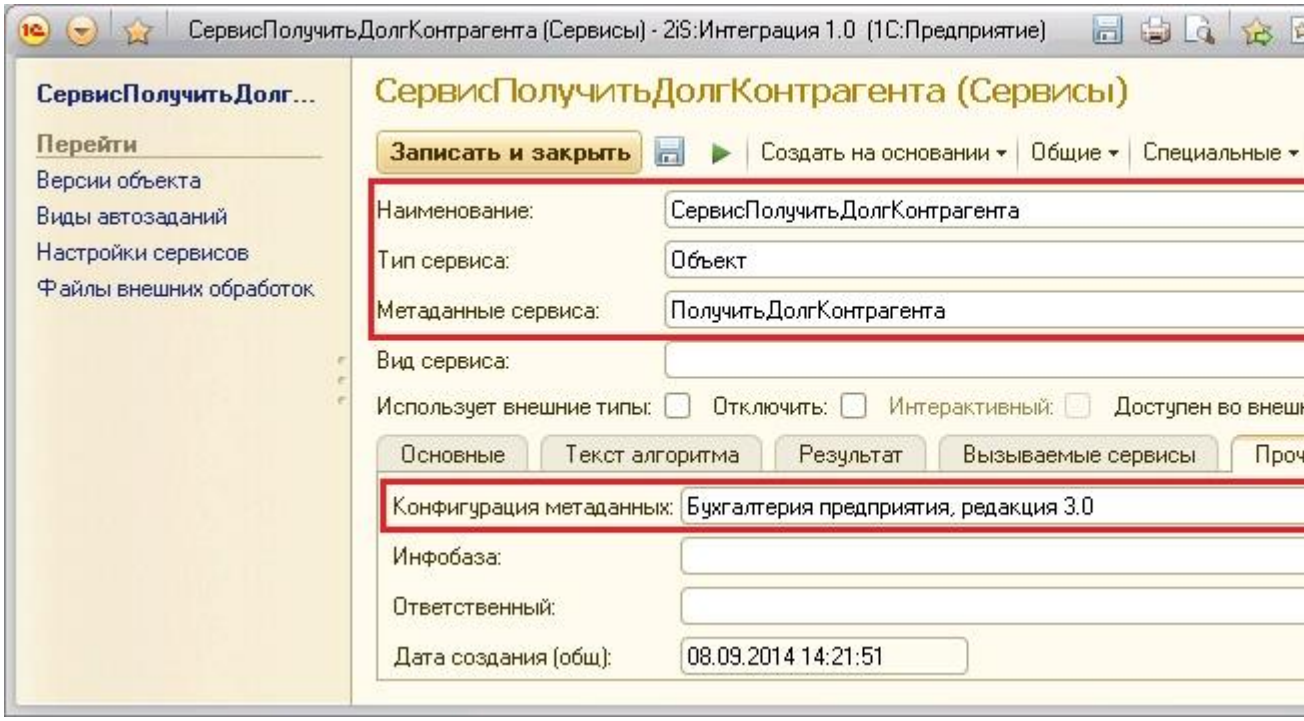


Рисунок RP3-000734. Сервис автоздания
Рисунок 6.

3. Для сервиса необходимо создать метаданные сервиса (и поместить в реквизит **Метаданные сервиса**):
    1. Тип данных = Файлы внешних обработок
    2. В открывшейся форме нового элемента справочника необходимо задать ему наименование и воспользоваться командой *Загрузить файл*, где указать файл внешней обработки, созданный в предыдущей главе

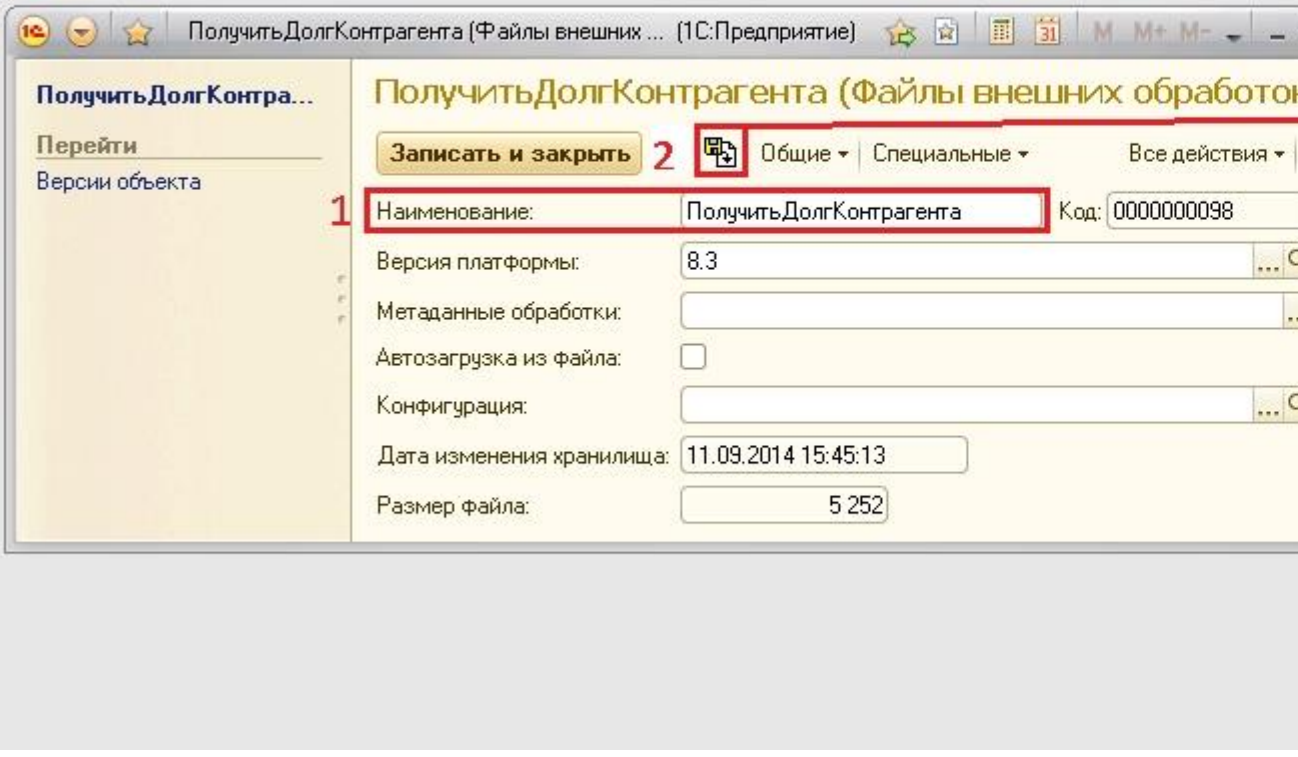


Рисунок RP3-000735. Создание метаданных сервиса
Рисунок 7.

4. После этого необходимо описать входные параметры сервиса.
   Для автоматического заполнения таблицы параметров сервиса рекомендуется использовать команду *Обновить из метаданных*:

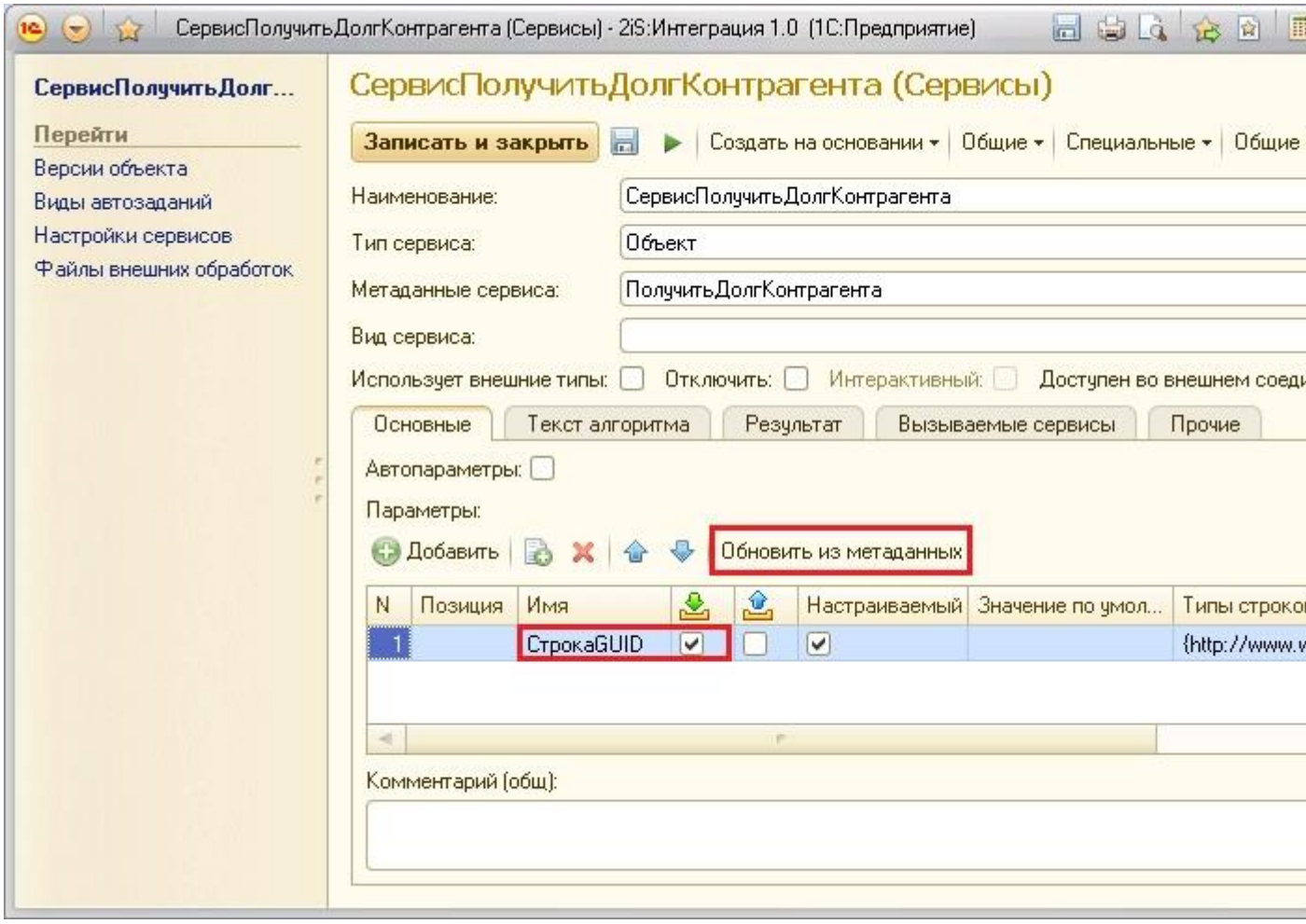


Рисунок RP3-000736. Парметры сервиса
Рисунок 8.

1. Поскольку в рассматриваемом примере сервис возвращает одно значение простого типа (число), то выходные параметры сервиса описывать не требуется, т.к. результат выполнения сервиса всегда помещается в параметр, создаваемый Интеграцией по умолчанию: **выхРезультат**.
2. Пример использования выходных параметров сервиса для возврата сложного результата рассмотрен в главе "Усложненный пример веб-сервиса".

На этом процесс создания виртуального веб-сервиса завершается.

Вызов веб-сервиса

В предыдущих главах руководства был описан процесс создания внешней обработки, выполняющей логику веб-сервиса в инфобазе БП, а также подключение данной обработки к Интеграции в виде виртуального веб-сервиса.

В настоящей главе будет приведен пример вызова и использования подключенного таким образом виртуального веб-сервиса из инфобазы УТ.

Для целей рассматриваемого примера по получению долга контрагента была создана внешняя обработка, внешний вид которой можно наблюдать на рисунке 1.

Листинг модуля формы обработки:

[начало листинга]

```
&НаКлиенте
Процедура ПолучитьДолг(Команда)
    СтрокаGUID = Строка(Контрагент.УникальныйИдентификатор());
    ДолгПолученный = ПолучитьДолгНаСервере(СтрокаGUID);
    Долг = ДолгПолученный; // вывод результата в элемент
управления формы
КонецПроцедуры

&НаСервереБезКонтекста
Функция ПолучитьДолгНаСервере(СтрокаGUID);

    // Инициализация параметров вызова веб-сервиса
    ПространствоИмен = "http://2iS.ru/Integration";
    ИмяПользователя = "ПользовательИнтеграции";
    ПарольПользователя = "ПарольПользователяИнтеграции";
    ИмяВебСервиса = "Integration2iS";
    ИмяБазыИнтеграции = "IntegrationInfobase";
    АдресВебСервиса = "http://server_address/" + ИмяБазыИнтеграции +
"/ws/" + ИмяВебСервиса; // например:
http://192.168.0.1/IntegrationInfobase/ws/Integration2iS
    Определения = Новый WSОпределения(АдресВебСервиса + "?wsdl",
ИмяПользователя, ПарольПользователя);
    ИмяТочкиПодключения = ИмяВебСервиса + "Soap";
    Прокси = Новый WSПрокси(Определения, ПространствоИмен,
ИмяВебСервиса, ИмяТочкиПодключения);
    Прокси.Пользователь = ИмяПользователя;
    Прокси.Пароль = ПарольПользователя;
    ИмяСервиса = "GetDebtByGUID";

    // Получение схемы вызова сервиса...
    ТекстСхемыXMLПараметров = Прокси.GetServiceSchema(ИмяСервиса);
    ЧтениеXML = Новый ЧтениеXML;
    ЧтениеXML.УстановитьСтроку(ТекстСхемыXMLПараметров);
    НовыйПостроительДом = Новый ПостроительDOM;
    НовыйДокументДом = НовыйПостроительДом.Прочитать(ЧтениеXML);
    НовыйПостроительСхемXML = Новый ПостроительСхемXML;
    СхемаXML =
НовыйПостроительСхемXML.СоздатьСхемуXML(НовыйДокументДом);
    НаборСхем = Новый НаборСхемXML;
    НаборСхем.Добавить(СхемаXML);
    // ... для создания фабрики, которой будут известны типы параметров
сервиса
    ФабрикаВызова = Новый ФабрикаXDTO(НаборСхем);

    // Имя входных параметров в схеме вызова сервиса жестко задано как
"InParameters"
    ТипXDTO = ФабрикаВызова.Тип(ПространствоИмен, "InParameters");
    // Инициализация параметров входа...
    ПараметрыВходаОбъектXDTO = ФабрикаВызова.Создать(ТипXDTO); // у
```

```
данного объекта XDTO свойства соответствуют параметрам сервиса (см. рис. 8)
    ПараметрыВходаОбъектXDTO.СтрокаGUID = СтрокаGUID;
    // ... и их сериализация (для передачи в веб-сервис)
    ЗаписьXML = Новый ЗаписьXML;
    ЗаписьXML.УстановитьСтроку();
    ФабрикаВызова.ЗаписатьXML(ЗаписьXML, ПараметрыВходаОбъектXDTO);
    ПараметрыXMLВход = ЗаписьXML.Закрыть();
    ПараметрыXMLВыход = Неопределено; //сюда будут помещены
сериализованные выходные параметры (в виде XML-строки)

    // Вызов виртуального веб-сервиса "GetDebtByGUID" через реальный
веб-сервис Интеграции
    РезультатВызова = Прокси.RunService(ИмяСервиса, ПараметрыXMLВход,
ПараметрыXMLВыход);

    // Получение простого результата веб-сервиса
    ЧтениеXML.УстановитьСтроку(ПараметрыXMLВыход);
    ОбъектXDTOПараметрыВыхода = ФабрикаВызова.ПрочитатьXML(ЧтениеXML);
    ПростойРезультат = ОбъектXDTOПараметрыВыхода.выхРезультат;
    Возврат ПростойРезультат;

КонецФункции;
```

[окончание листинга]

Комментарии к листингу:

1. Имя веб-сервиса "`Integration2iS`" всегда должно соответствовать имени реального (присутствующего в метаданных конфигурации) веб-сервиса Интеграции
2. Пространство имен "`http://2iS.ru/Integration`" всегда должно соответствовать пространству имен реального (присутствующего в метаданных конфигурации) веб-сервиса Интеграции

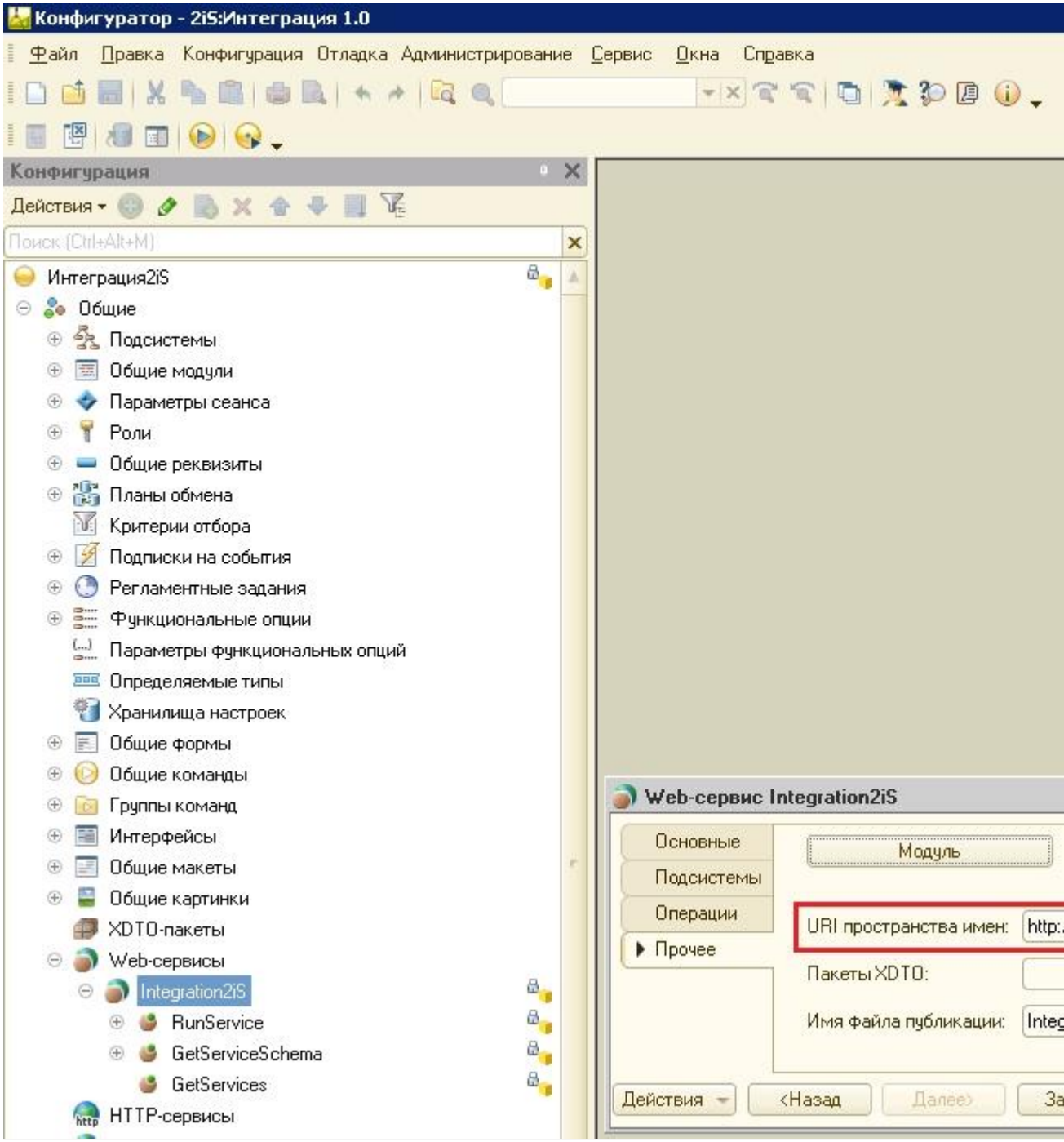
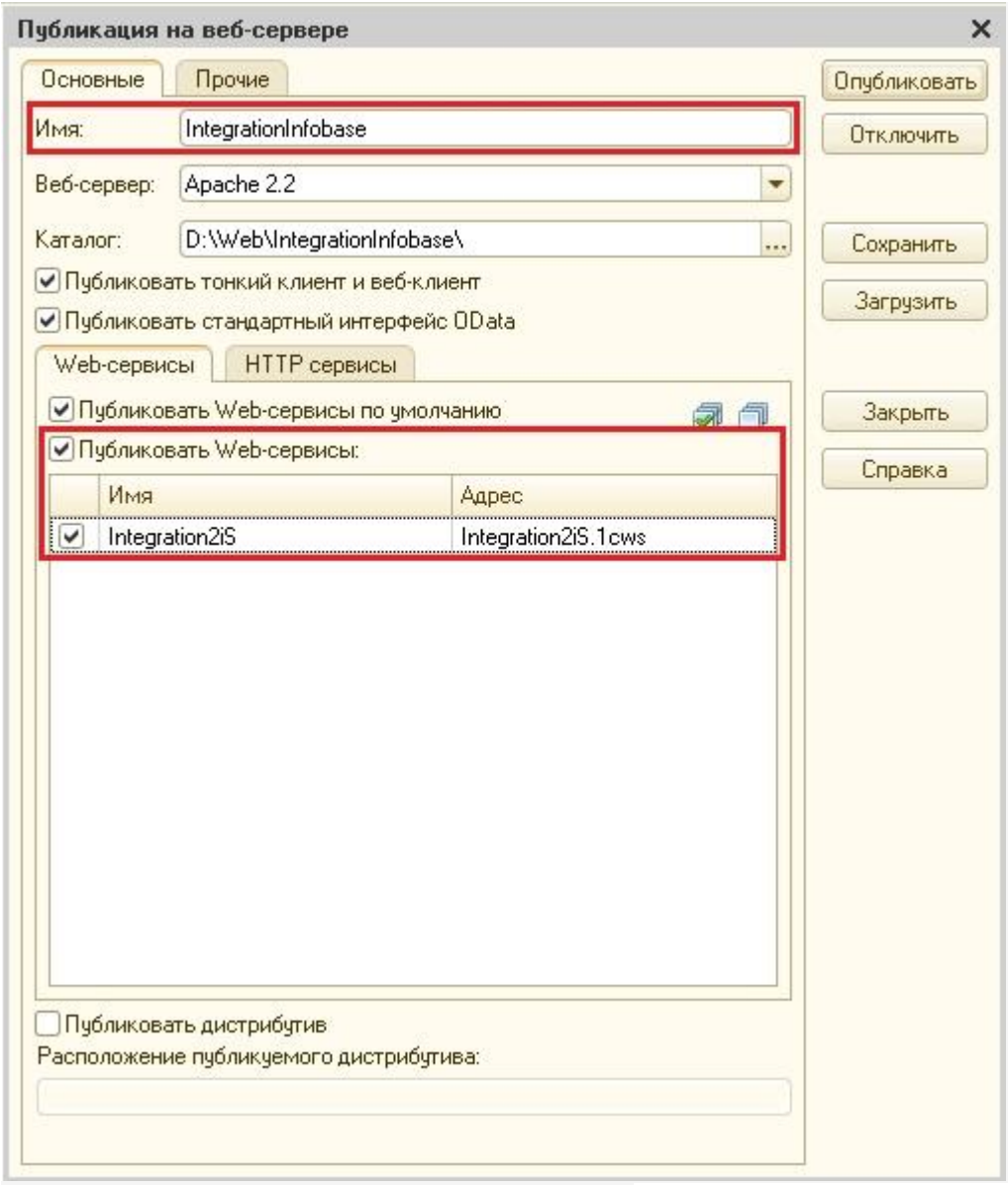
1. 

   Рисунок RP3-000738. Реальный веб-сервис Интеграции
   Рисунок 9.
3. Получение WS-опредедений `http://server address/IntegrationInfobase/ws/Integration2iS?wsdl` осуществляется по имени публикации инфобазы Интеграции (адрес сервера определяется настройками веб-сервера)



1. Рисунок RP3-000737. Публикация Интеграции
   Рисунок 10.
4. Имя вызываемого веб-сервиса "`GetDebtByGUID`" должно соответствовать реквизиту **Имя в веб-сервисе** автозадания Интеграции
5. При инициализации входного параметра (`СтрокаGUID`) необходимо указывать имя, соответствующее имени параметра сервиса (рисунок 8)
6. Вызов виртуального веб-сервиса осуществляется функцией `RunService()` веб-сервиса Интеграции
7. Результат выполнения функции `мВыполнитьСервис()` виртуального веб-сервиса содержится в XML-файле в виде параметра с именем "выхРезультат"

Усложненный пример веб-сервиса

У читателя может возникнуть вопрос: что делать,если необходимо, чтобы веб-сервис возвращал не какое-то одно значение простого типа (строка, число и др.), а коллекцию значений (список, таблица, массив)?
Интеграция позволяет легко решать подобные задачи.

Усложним рассмотренный в предыдущих главах пример: пусть веб-сервис, помимо долга контрагента, должен возвращать список договоров контрагента и валюты этих договоров.

1. В Конфигураторе: внесем изменения во внешнюю обработку (которая выполняется в инфобазе БП):
    1. Добавим в обработку реквизит **выхТаблицаЗначений** (тип = ТаблицаЗначений)
    2. И реализуем его заполнение в модуле функции `мВыполнитьСервис()` :

C:\Users\ \Desktop\ПолучитьДолгКонтрагентаПоGUID.epf: Модуль объекта

```
                Результат = Строка(Выборка.Долг);
            КонецЕсли;
        КонецЕсли;

        // (2) Более сложный случай - возврат нескольких значений
        Запрос = Новый Запрос;
        Запрос.Текст =
            "ВЫБРАТЬ
            |   ДоговорыКонтрагентов.Наименование КАК НаименованиеДоговора,
            |   ДоговорыКонтрагентов.ВалютаВзаиморасчетов.Наименование КАК Ва
            |ИЗ
            |   Справочник.ДоговорыКонтрагентов КАК ДоговорыКонтрагентов
            |ГДЕ
            |   ДоговорыКонтрагентов.Владелец = &Контрагент";

        Запрос.УстановитьПараметр("Контрагент", ИскомыйКонтрагент);

        РезультатЗапросаТЗ = Запрос.Выполнить().Выгрузить();
        Если РезультатЗапросаТЗ.Количество() <> 0 Тогда
            // возврат таблицы значений
            выхТаблицаЗначений = Новый ТаблицаЗначений;
            выхТаблицаЗначений.Колонки.Добавить("НаименованиеДоговора");
            выхТаблицаЗначений.Колонки.Добавить("ВалютаДоговора");
            Для Каждого СтрокаРезультата Из РезультатЗапросаТЗ Цикл
                СтрокаТЗ = выхТаблицаЗначений.Добавить();
                СтрокаТЗ.НаименованиеДоговора    = СтрокаРезультата.Наименова
                СтрокаТЗ.ВалютаДоговора          = СтрокаРезультата.ВалютаДого
            КонецЦикла;
        КонецЕсли;
    КонецЕсли;

    Возврат Результат;
КонецФункции
```

Рисунок RP3-000739. Внесенные изменения в реквизиты и модуль внешней обработки
Рисунок 11.

2. В Интеграции: обновим метаданные сервиса ПолучитьДолгКонтрагента с помощью команды *Загрузить файл*:

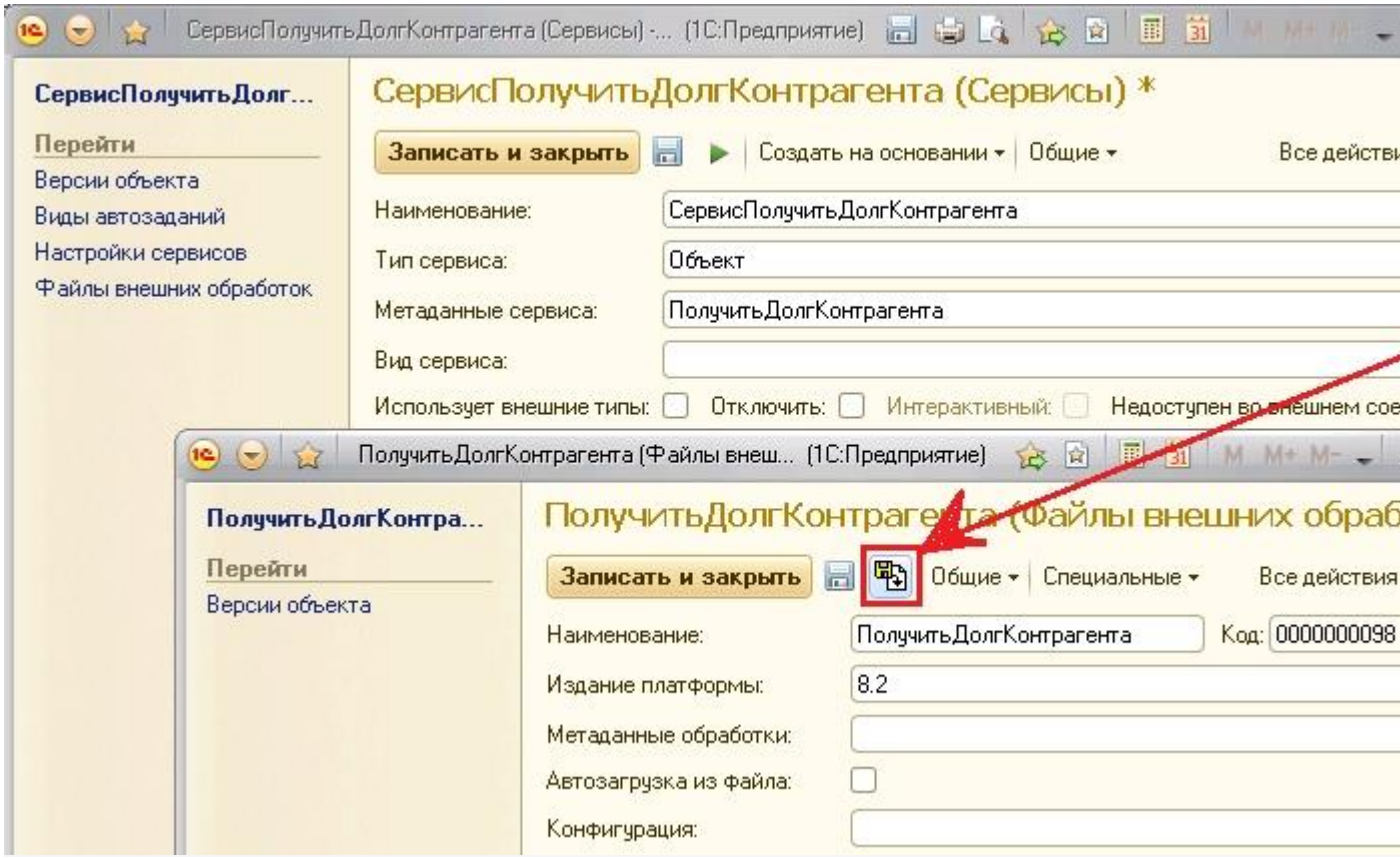


Рисунок RP3-000740. Обновление метаданных сервиса из файла внешней обработки
Рисунок 12.

3. В Интеграции: обновим параметры сервиса с помощью команды *Обновить из метаданных*:

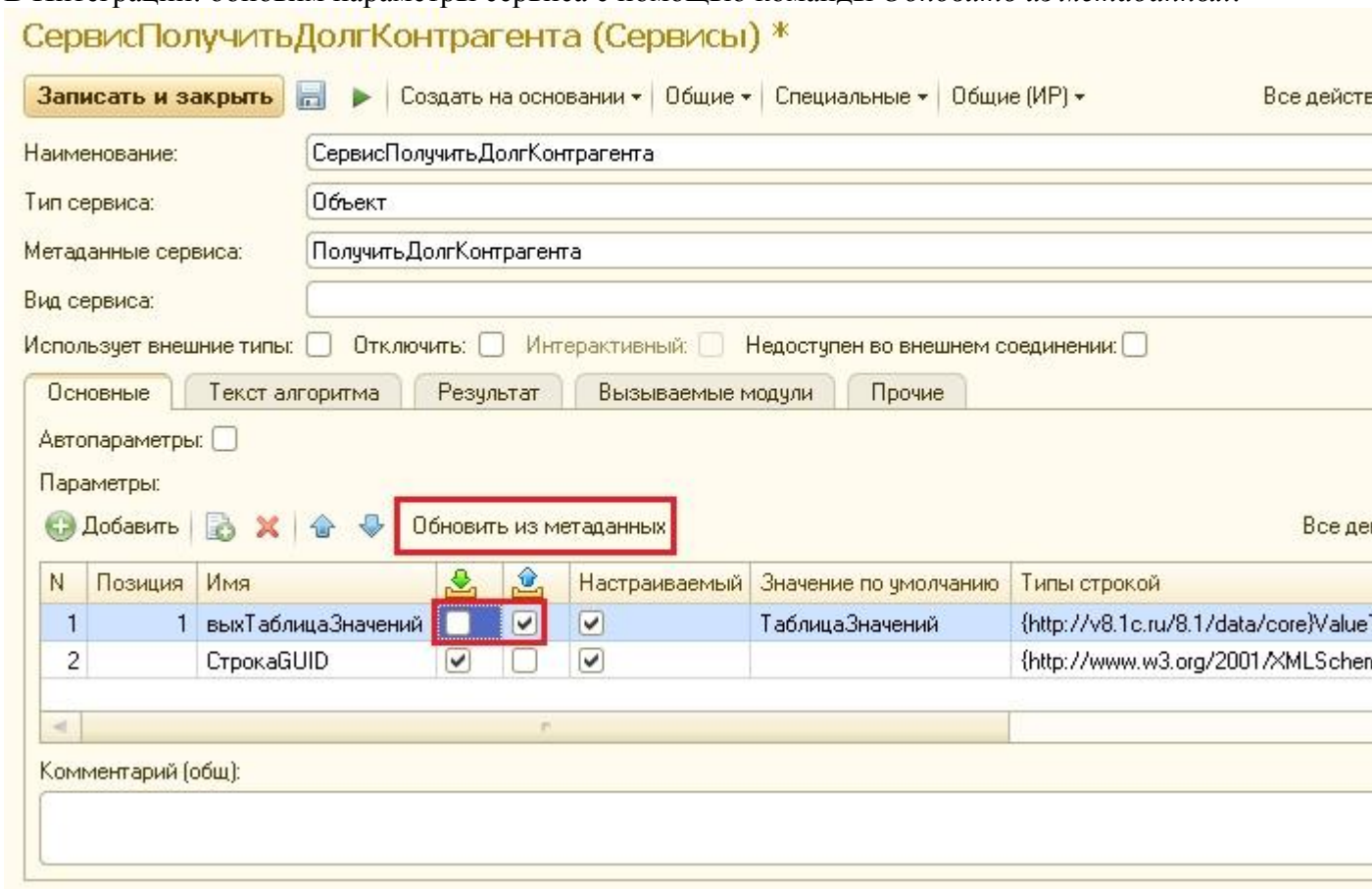


Рисунок RP3-000741. Обновление параметров сервиса
Рисунок 13.

1. По умолчанию команда *Обновить из метаданных* оставляет флажки **Вход** для всех параметров взведенными. Для нашего нового параметра **выхТаблицаЗначений** мы оставим взведенным только флажок **Выход**.

4. В Конфигураторе: добавим десериализацию выходных параметров в программный код обработки (которая запускается из инфобазы УТ и вызывает веб-сервис Интеграции) .

[начало листинга]

```
&НаСервереБезКонтекста
Функция ПолучитьДолгНаСервере(СтрокаGUID);

    // [пропущено]

    // Вызов виртуального веб-сервиса "GetDebtByGUID" через реальный
веб-сервис Интеграции
    РезультатВызова = Прокси.RunService(ИмяСервиса, ПараметрыXMLВход,
ПараметрыXMLВыход);

    // Десериализация выходных параметров сервиса
    ЗначенияПараметровВыхода = Новый Структура; // сюда будут помещены
десериализованные выходные параметры
    // Превращаем XML-строку в объект XDTO с правильным типом
    ЧтениеXML.УстановитьСтроку(ПараметрыXMLВыход);
    // "Правильный  тип" = тип, который известен фабрике XDTO.
    // Помним, что фабрика была создана на основе XML-схемы вызова
сервиса,
    // которая и содержит описание типов параметров этого сервиса.
    // Имя выходных параметров в этой схеме жестко задано как
"OutParameters"
    ТипXDTO = ФабрикаВызова.Тип(ПространствоИмен, "OutParameters");
    // Получаем объект XDTO
    ПараметрыВыходаОбъектXDTO = ФабрикаВызова.ПрочитатьXML(ЧтениеXML,
ТипXDTO);

    // Десериализация "сложного" результата
    СериализаторВызова = Новый СериализаторXDTO(ФабрикаВызова);
    Для Каждого СвойствоXDTO Из ПараметрыВыходаОбъектXDTO.Свойства()
Цикл
        Если ПараметрыВыходаОбъектXDTO.Установлено(СвойствоXDTO) Тогда
            ЗначениеСвойства =
ПараметрыВыходаОбъектXDTO[СвойствоXDTO.Имя];
            Если ТипЗнч(ЗначениеСвойства) = Тип("ОбъектXDTO") Тогда
                ЗначениеСвойства =
СериализаторВызова.ПрочитатьXDTO(ЗначениеСвойства);
            КонецЕсли;
            ЗначенияПараметровВыхода.Вставить(СвойствоXDTO.Имя,
ЗначениеСвойства);
        КонецЕсли;
    КонецЦикла;

    Возврат ЗначенияПараметровВыхода.выхРезультат;

КонецФункции
```

[окончание листинга]

Комментарии к листингу:

1. XML-строка со значениями выходных параметров (переменная `ПараметрыXMLВыход`) содержит значения выходных параметров, но не содержит описаний типов этих параметров
2. Поэтому для десериализации используется та же ФабрикаXDTO (переменная `ФабрикаВызова`), что и для сериализации
    1. Это обусловлено тем, что фабрика была создана по набору схем (переменная `НаборСхем`), в который входила XML-схема* вызова сервиса (переменная `СхемаXML`)
3. Использование такой фабрики позволяет:
    1. Преобразовать XML-строку выходных параметров в объект XDTO (переменная `ПараметрыВыходаОбъектXDTO`) с типами, описанными в XML-схеме* вызова сервиса
    2. Создать сериализатор XDTO (переменная `СериализаторВызова`), который преобразует объект XDTO в типы данных платформы 1С

    * Посмотреть схему вызова сервиса в Интеграци можно при помощи команды *Показать схему XML вызова*:

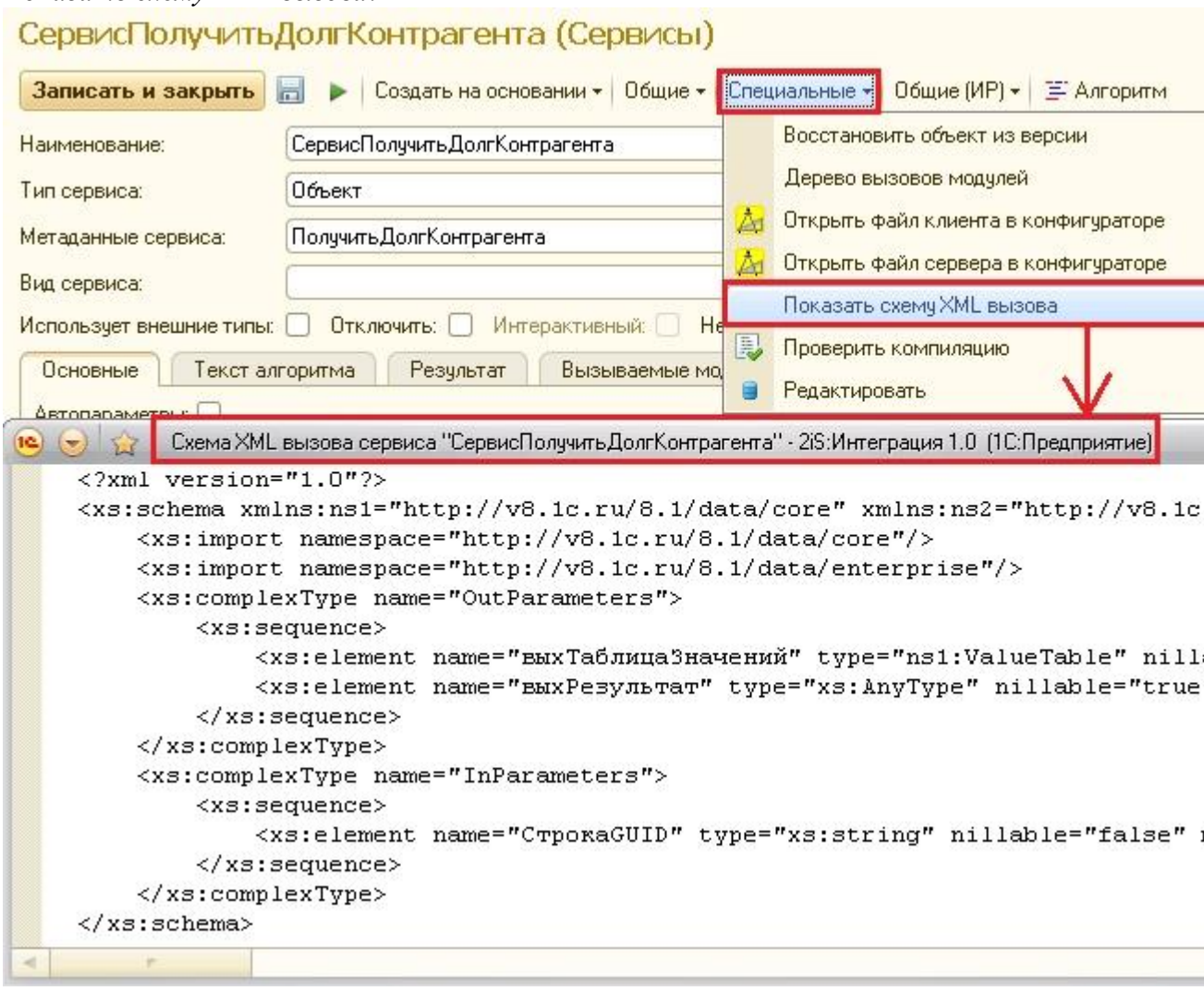


    Рисунок RP3-000743. Команда просмотра XML-схемы вызова сервиса
    Рисунок 14.
4. На выходе получаем структуру (переменная `ЗначенияПараметровВыхода`), содержащую значения выходных параметров сервиса, преобразованных в типы данных платформы 1С:

```
    // Десериализация "сложного" результата
    СериализаторВызова = Новый СериализаторXDTO(ФабрикаВызова);
    Для Каждого СвойствоXDTO Из ПараметрыВыходаОбъектXDTO.Свойства() Цикл
        Если ПараметрыВыходаОбъектXDTO.Установлено(СвойствоXDTO) Тогда
            ЗначениеСвойства = ПараметрыВыходаОбъектXDTO[СвойствоXDTO.Имя];
            Если ТипЗнч(ЗначениеСвойства) = Тип("ОбъектXDTO") Тогда
                ЗначениеСвойства = СериализаторВызова.ПрочитатьXDTO(ЗначениеСвойст
            КонецЕсли;
            ЗначенияПараметровВыхода.Вставить(СвойствоXDTO.Имя, ЗначениеСвойства);
        КонецЕсли;
    КонецЦикла;

    Возврат ЗначенияПараметровВыхода.выхРезультат;

КонецФункции
```

Табло - 1

Выражение	Значение	Тип
ПараметрыВыходаОбъектXDTO	ОбъектXDTO	ОбъектXDTO
выхРезультат	"1 440 000"	Строка
выхТаблицаЗначений	ОбъектXDTO	ОбъектXDTO
ПараметрыВыходаОбъектXDTO.выхТаблицаЗначений.Тип()	{http://v8.1c.ru/8.1/data/core}ValueTable	ТипОбъектаXDTO
ЗначенияПараметровВыхода	Структура	Структура
выхРезультат	"1 440 000"	Строка
выхТаблицаЗначений	ТаблицаЗначений	ТаблицаЗначений

ТаблицаЗначений

Количество элементов: 2

Индекс	Значение элемента	Тип элемента	НаименованиеДоговора	ВалютаДоговора
0	СтрокаТаблицыЗначений	СтрокаТаблицыЗначений	"493 от 15.01.2013"	"руб."
1	СтрокаТаблицыЗначений	СтрокаТаблицыЗначений	"Договор купли-продажи М-8900"	"USD"

[Рисунок RP3-000742. Значения выходных параметров сервиса]
Рисунок 15.

.