Глава 40. Прочие механизмы

40.1. Решение систем линейных алгебраических уравнений

40.1.1. Общая информация

Системой линейных алгебраических уравнений (возможно использование сокращения СЛАУ) называется такая система уравнений, каждое уравнение в которой является алгебраическим уравнением первой степени (линейным уравнением). Коэффициенты при переменных, свободные члены и неизвестные считаются вещественными числами. В частности, СЛАУ используются в различных экономических задачах, таких, как расчет себестоимости выпускаемой продукции.

> **ПРИМЕЧАНИЕ**. Описание методов решений СЛАУ и их особенностей выходит за рамки данной документации. Описание различных методов решения СЛАУ следует искать в соответствующих учебных материалах.

Механизм решения СЛАУ, используемый в системе программ «1С:Предприятие», использует два алгоритма: итерационный и прямого решения. При выборе конкретного механизма применяется ряд оптимизаций, разработанных фирмой «1С».

Рассмотрим применение реализованного механизма на примере простой СЛАУ:

Копировать в буфер обмена

```
(1)   X1 +   X2 =  5
(2) 2*X1 + 3*X2 = 13
```

Цифры в скобках определяют порядковый номер уравнения.

Для решения этой системы будет использоваться объект РасчетСистемыЛинейныхУравнений. В качестве входных данных используется две таблицы:

1. Таблица, содержащая значения свободных коэффициентов СЛАУ.
2. Таблица, содержащая значения коэффициентов при переменных СЛАУ.

Эти таблицы могут быть получены как «вручную», через заполнение таблицы значений, так на основе запроса к базе данных. В текущем примере данные будут подготовлены в таблицах значений. Нумерация уравнений идет сверху вниз, а переменных в системе - слева направо.

Подготовим таблицу свободных коэффициентов:

Копировать в буфер обмена

```
ДанныеУзлов = Новый ТаблицаЗначений;
ДанныеУзлов.Колонки.Добавить("НомерУравнения");
ДанныеУзлов.Колонки.Добавить("СвобКоэффициент");
Строка = ДанныеУзлов.Добавить();
Строка.НомерУравнения = 1;
Строка.СвобКоэффициент = 5;
Строка = ДанныеУзлов.Добавить();
Строка.НомерУравнения = 2;
Строка.СвобКоэффициент = 13;
```

Подготовим таблицу коэффициентов при переменных:

Копировать в буфер обмена

```
ДанныеСвязей = Новый ТаблицаЗначений;
ДанныеСвязей.Колонки.Добавить("НомерУравнения");
ДанныеСвязей.Колонки.Добавить("НомерПеременной");
ДанныеСвязей.Колонки.Добавить("Коэффициент");
Строка = ДанныеСвязей.Добавить();
Строка.НомерУравнения = 1;
Строка.НомерПеременной = 1;
Строка.Коэффициент = 1;
Строка = ДанныеСвязей.Добавить();
Строка.НомерУравнения = 1;
Строка.НомерПеременной = 2;
Строка.Коэффициент = 1;
Строка = ДанныеСвязей.Добавить();
Строка.НомерУравнения = 2;
Строка.НомерПеременной = 1;
Строка.Коэффициент = 2;
Строка = ДанныеСвязей.Добавить();
Строка.НомерУравнения = 2;
Строка.НомерПеременной = 2;
Строка.Коэффициент = 3;
```

Теперь необходимо передать сформированные таблицы с коэффициентами механизму расчета. Для этого необходимо заполнить свойства ИсточникДанныхУзлов (таблица свободных коэффициентов) и ИсточникДанныхСвязей (таблица коэффициентов):

Копировать в буфер обмена

```
Решатель.ИсточникДанныхУзлов = ДанныеУзлов;
Решатель.ИсточникДанныхСвязей = ДанныеСвязей;
```

Теперь следует указать, в каких колонках находятся номера уравнений и коэффициентов в переданных источниках. Для таблицы свободных коэффициентов следует указать только колонку, где находятся номера уравнений (с помощью свойства КолонкаУравненияВУзлах), а для таблицы коэффициентов при переменных надо указать, где находятся и номера уравнений (свойство КолонкаУравненияВСвязях) и номера переменных (КолонкаПеременныеВСвязях):

Копировать в буфер обмена

```
Решатель.КолонкаУравненияВСвязях = "НомерУравнения";
Решатель.КолонкаПеременныеВСвязях = "НомерПеременной";
Решатель.КолонкаУравненияВУзлах = "НомерУравнения";
```

Последним шагом является создание описания решаемой системы уравнений. Для этого следует использовать свойство РасчетСистемыЛинейныхУравнений.ОписанияСистем. Для каждой добавляемой системы необходимо указать, какой столбец отвечает за свободные коэффициенты системы, а какой - за коэффициенты при переменных решаемой системы:

Копировать в буфер обмена

```
ОписаниеСЛАУ = Решатель.ОписанияСистем.Добавить();
ОписаниеСЛАУ.КолонкаКоэффициентовВУзлах = "СвобКоэффициент";
ОписаниеСЛАУ.КолонкаКоэффициентовВСвязях = "Коэффициент";
```

Теперь все параметры заданы и можно решить заданную систему уравнений:

Копировать в буфер обмена

```
Результат = Решатель.РассчитатьСистемыЛинейныхУравнений();
```

В результате решения нашей СЛАУ получится таблица значений, состоящая из двух колонок:

- Колонка `НомерУзла` содержит номер переменной СЛАУ.
- Колонка `Решение1` содержит значение решения для каждой переменной.

Для используемого примера решение будет следующим: $X_1 = 2$, $X_2 = 3$.

Итоговый текст примера:

Копировать в буфер обмена

```
Решатель = Новый РасчетСистемЛинейныхУравнений;
ДанныеУзлов = Новый ТаблицаЗначений;
ДанныеУзлов.Колонки.Добавить("НомерУравнения");
ДанныеУзлов.Колонки.Добавить("СвобКоэффициент");
Строка = ДанныеУзлов.Добавить();
Строка.НомерУравнения = 1;
Строка.СвобКоэффициент = 5;
Строка = ДанныеУзлов.Добавить();
Строка.НомерУравнения = 2;
Строка.СвобКоэффициент = 13;
ДанныеСвязей = Новый ТаблицаЗначений;
ДанныеСвязей.Колонки.Добавить("НомерУравнения");
ДанныеСвязей.Колонки.Добавить("НомерПеременной");
ДанныеСвязей.Колонки.Добавить("Коэффициент");
Строка = ДанныеСвязей.Добавить();
Строка.НомерУравнения = 1;
Строка.НомерПеременной = 1;
Строка.Коэффициент = 1;
Строка = ДанныеСвязей.Добавить();
Строка.НомерУравнения = 1;
Строка.НомерПеременной = 2;
Строка.Коэффициент = 1;
Строка = ДанныеСвязей.Добавить();
Строка.НомерУравнения = 2;
Строка.НомерПеременной = 1;
Строка.Коэффициент = 2;
Строка = ДанныеСвязей.Добавить();
Строка.НомерУравнения = 2;
Строка.НомерПеременной = 2;
Строка.Коэффициент = 3;
Решатель.ИсточникДанныхУзлов = ДанныеУзлов;
Решатель.ИсточникДанныхСвязей = ДанныеСвязей;
ОписаниеСЛАУ = Решатель.ОписанияСистем.Добавить();
ОписаниеСЛАУ.КолонкаКоэффициентовВУзлах = "СвобКоэффициент";
ОписаниеСЛАУ.КолонкаКоэффициентовВСвязях = "Коэффициент";
Решатель.КолонкаУравненияВСвязях = "НомерУравнения";
Решатель.КолонкаПеременныеВСвязях = "НомерПеременной";
Решатель.КолонкаУравненияВУзлах = "НомерУравнения";
Результат = Решатель.РассчитатьСистемыЛинейныхУравнений();
```

Рассмотрим, как можно одновременно решать несколько СЛАУ.

Для одновременного решения нескольких СЛАУ надо выполнить предварительное действие - привести системы к одинаковой размерности (как по числу переменных, так и по количеству уравнений). В простейшем случае это решается формированием недостающих уравнений или введением в уравнения переменных с нулевыми коэффициентами. Таким образом, перед решением нескольких систем, у нас должна быть единая нумерация, как самих уравнений, так и их переменных.

Затем необходимо правильно сформировать входные данные для механизма решения СЛАУ. Как уже было отмечено ранее, на вход системе решения поступает таблица, содержащая свободные коэффициенты (свойство `РасчетСистемыЛинейныхУравнений.ИсточникДанныхУзлов`) и таблица, содержащая коэффициенты при переменных системы (свойство `РасчетСистемыЛинейныхУравнений.ИсточникДанныхСвязей`). Для каждой дополнительной системы в каждый из источников данных добавляется новая колонка. Имена колонок, отвечающих за коэффициенты одной системы в каждом источнике, указываются с помощью свойства `РасчетСистемыЛинейныхУравнений.ОписанияСистем`.

Рассмотрим пример из двух СЛАУ:

Копировать в буфер обмена

```
(1)     2*X1 + 3*X2 = 10         4*X1 + 15*X2 = 40
(2)     3*X1 + 2*X2 = 15         5*X1 +  3*X2 = 10
```

Имеется две системы из двух уравнений, в каждой из которых по две переменные. Для указания свободных коэффициентов должна быть сформирована следующая таблица:

НомерУравнения	СК1	СК2
1	10	40
2	15	10

В примере, приведенном выше, столбец таблицы значений для указания свободных коэффициентов назывался `СвобКоэффициент`. Однако теперь систем больше одной и нельзя указать две колонки в таблице значений с одинаковым именем. Поэтом колонки будут называться `СК1` и `СК2`.

Аналогичным образом будет выглядеть таблица для коэффициентов при переменных СЛАУ:

НомерУравнения	НомерПеременной	К1	К2
1	1	2	4
1	2	3	15
2	1	3	5
2	2	2	3

Теперь вместо имени `Коэффициент` колонки будут называться по «номерам» СЛАУ: `К1` (для первой СЛАУ) и `К2` (для второй СЛАУ).

Указание системы на имена колонок, которые содержат номера уравнений и переменных не изменилось:

Копировать в буфер обмена

```
Решатель.КолонкаУравненияВСвязях = "НомерУравнения";
Решатель.КолонкаПеременныеВСвязях = "НомерПеременной";
Решатель.КолонкаУравненияВУзлах = "НомерУравнения";
```

Но описание системы уравнений несколько изменится. Теперь вместо описания одной системы будет описание двух систем:

```
ОписаниеСЛАУ = Решатель.ОписанияСистем.Добавить();
ОписаниеСЛАУ.КолонкаКоэффициентовВУзлах = "СК1";
ОписаниеСЛАУ.КолонкаКоэффициентовВСвязях = "К1";
ОписаниеСЛАУ = Решатель.ОписанияСистем.Добавить();
ОписаниеСЛАУ.КолонкаКоэффициентовВУзлах = "СК2";
ОписаниеСЛАУ.КолонкаКоэффициентовВСвязях = "К2";
```

Если подходить к вопросу формально, то каждую СЛАУ определяют имена колонок свободных коэффициентов и коэффициентов при переменных в таблицах, переданных на вход механизму решения СЛАУ. Поэтому при решении нескольких систем, которые отличаются только свободными коэффициентами, нет необходимости для каждой системы повторять коэффициенты при переменных. В таком случае будет достаточно создать колонки только для уникальных параметров - свободных коэффициентов. Исходя из формального описания СЛАУ (с точки зрения механизма решения), в рассмотренном примере можно задать еще две СЛАУ, если скомбинировать коэффициенты при переменных и свободные коэффициенты из разных систем. Очевидно, что делать такую комбинацию имеет смысл только в том случае, если это имеет прикладной смысл.

В результате решения двух СЛАУ получится таблица значений, состоящая из двух колонок:

- Колонка `НомерУзла` содержит номер переменной СЛАУ.
- Колонка `Решение1` содержит значение решения для каждой переменной из первой системы.
- Колонка `Решение2` содержит значение решения для каждой переменной из второй системы.

Также стоит отметить, в каком случае рекомендуется решать несколько СЛАУ одним вызовом, а когда - разными вызовами. Если несколько СЛАУ имеют одинаковую размерность и очевидную зависимость по коэффициентам, то рекомендуется решать несколько СЛАУ одним вызовом. Например, если решается три системы с вариантами прогноза: пессимистический, реалистический, оптимистический. Если системы СЛАУ не имеют очевидной связи (просто две разные системы), то лучше каждую систему решать обособленно. Такое различие объясняется тем, что платформа анализирует переданные системы, заранее предполагая их схожесть. На основании проведенного анализа выполняются некоторые внутренние оптимизации. Если системы не имеют связи - могут быть приняты неверные решения.

Дополнительно рассмотрим некоторые параметры объекта `РасчетСистемыЛинейныхУравнений`:

- Свойство `ТребуемаяТочность` - максимальная погрешность искомого решения. Вычисляется как разница между левой и правой частью каждого уравнения решаемой системы при подстановке текущего решения. Задание чрезмерной точности приведет к существенному увеличению времени решения. Для получения в результате точного i-го знака после запятой, рекомендуется устанавливать точность в виде значения 10 в степени `-(i+1)`. Таким образом, для получения точного 5 знака следует установить для свойства `ТребуемаяТочность` значение 10^{-6}.

  Рекомендуется при изменении желаемой точности решения сначала изменять свойство `ТребуемаяТочность`. Если результата не достигается, значит причиной может служить недостаточное количество итераций, используемых для нахождения решения. В этом случае можно увеличить значение свойства `КоличествоИтераций`.

- Свойство `КоличествоИтераций` определяет количество итераций, которые будет использовать итерационный алгоритм решения СЛАУ. Данная настройка вторична по отношению к свойству `ТребуемаяТочность`. Рекомендуется подбирать минимально возможное значение для получения в результате требуемой точности решения (свойство `ТребуемаяТочность`). Увеличение значения свойства выше требуемого не имеет смысла, т. к. алгоритм прекратит выполнение расчета после достижения требуемой точности.

- Свойство `ИспользованиеВычислительныхРесурсов` позволяет указать, сколько вычислительных потоков будут принимать участие в решении СЛАУ. Значение по умолчанию обеспечивает максимальное использование доступных вычислительных мощностей. Значение данного свойства не должно превышать количества доступных ядер процессора (включая ядра, доступные в результате использования технологии hyper-threading). Рекомендуется изменять данное свойство только в том случае, если требуется уменьшить количество вычислительных потоков механизма решения относительно значения по умолчанию.

  Если все ресурсы процессора выделять на решение СЛАУ нет возможности, но и выделение для решения только одного вычислительного потока является недостаточным, то рекомендуется в качестве значения свойства `ИспользованиеВычислительныхРесурсов` установить число, равное числу решаемых одновременно систем (тогда каждая система будет рассчитываться независимо), либо кратное ему (тогда каждую систему будет решать количество потоков, равное значению краткости). При этом полученное значение не должно превышать общее количество ядер процессора.

Как было сказано в начале раздела, механизм решения СЛАУ использует комбинацию двух алгоритмов: итерационного и прямого. Каждый из этих алгоритмов обладает своими плюсами и минусами:

- Итерационный алгоритм:
  - Скорость работы зависит от требуемой точности решения. Это является одновременно и плюсом и минусом данного алгоритма. Плюсом это является в том смысле, что позволяет ограничиться требуемой точностью, а не находить решение с избыточной точностью (для данной предметной областью). Минусом это является в том смысле, что неоправданное повышение точности решения может либо фатально сказаться на времени решения, либо вовсе не позволит это решение получить.
  - Плюсы алгоритма:
    - На крупных матрицах работает быстрее (асимптотика $O(n^2)$).
  - Минусы алгоритма:
    - Для получения большой точности может потребоваться значительное число итераций (как следствие, время расчета увеличивается).
    - Есть теоретические ограничения по сходимости алгоритма.
- Прямой алгоритм:
  - Плюсы алгоритма:
    - Гарантированно точное решение (ограничивается точностью математических расчетов процессора компьютера).
    - На небольших матрицах работает быстрее.
  - Минусы алгоритма:
    - На крупных матрицах работает существенно медленнее (асимптотика $O(n^3)$).

Свойство `РасчетСистемЛинейныхУравнения.ГраницаИзмененияАлгоритма` определяет, какому алгоритму будет отдан приоритет после выполнения предварительного анализа СЛАУ, которые переданы для решения одним вызовом. Чем меньше значение - тем более вероятно использование итерационного алгоритма, чем больше значение - тем выше вероятность использования собственного алгоритма. Следует понимать, что смещая приоритет в сторону того или иного алгоритма, усиливаются не только положительные, но и отрицательные стороны того алгоритма, в сторону которого смещается выбор. Значение по умолчанию обеспечивает разумный баланс работы механизма, и изменять его следует только при очень хорошем понимании общей схемы работы и последствий принимаемого решения.

40.1.2. Компоненты связности графа

Если решаемая задача может быть смоделирована с помощью графа, то может возникнуть необходимость декомпозировать получившийся граф. Целью декомпозиции является повышение скорости получения решения. Это можно сделать, оперируя понятием «компоненты связности графа».

Платформа позволяет выделять следующие компоненты связности (это понятие представлено перечислением СпособПолученияКомпонентСвязностиРасчетаСистемЛинейныхУравнений):

- КомпонентыСлабойСвязности. Под компонентами слабой связности понимаются максимальные по включению компоненты графа (подграфы), такие, что из любой вершины компоненты (подграфа) существует путь в любую другую вершину этой компоненты (подграфа). Ориентация ребер графа игнорируется (ориентированные ребра рассматриваются как неориентированные).

- КомпонентыСильнойСвязности. Под компонентами сильной связности понимаются максимальные по включению компоненты графа (подграфы), такие, что из любой вершины компоненты (подграфа) существует путь в любую другую вершину этой компоненты (подграфа). Путь строится с учетом ориентации ребер.

- КомпонентыСильнойСвязностиБезТребованияСвязиВнутриКомпонент. Под этим термином понимаются такие компоненты связности, что для любых двух вершин A и B из двух любых разных компонент гарантируется, что одновременно не существует пути из A в B и из B в A. При этом для любых двух вершин A и B из одной компоненты каких-либо гарантий наличия путей не дается. В данном варианте система выполняет расчет компонент связности, наиболее оптимальных для анализа и отладки механизма решения систем линейных уравнений.

Рассмотрим примеры выделения компонент различной связности. Дана система уравнений:

Копировать в буфер обмена

```
(0)   X0 + 5*X1 = 1
(1) 6*X0 + 2*X1 + X2 = 1
(2) 3*X2 = 1
(3) 4*X3 = 1
```

Эту систему можно представить следующим графом:

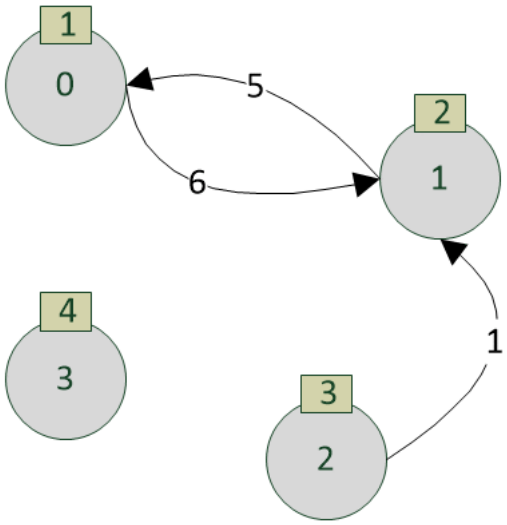

Рис. 794. Начальный граф

Если получить компоненты слабой связности для данного графа, то будет выделено две компоненты:

- Компонента 1: {0, 1, 2}.
- Компонента 2: {3}.

Или, в графическом виде:

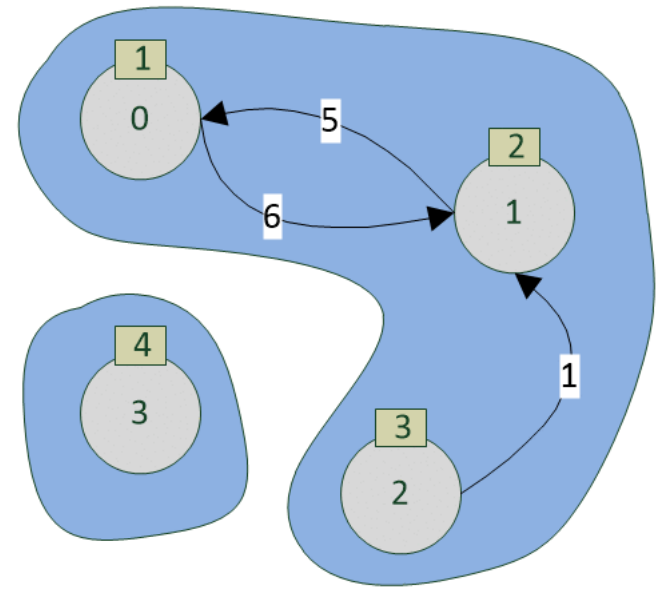

Рис. 795. Компоненты слабой связности

Если получить компоненты сильной связности для данного графа, то будет выделено три компоненты:

- Компонента 1: {0,1}.
- Компонента 2: {3}.
- Компонента 3: {2}.

Или, в графическом виде:

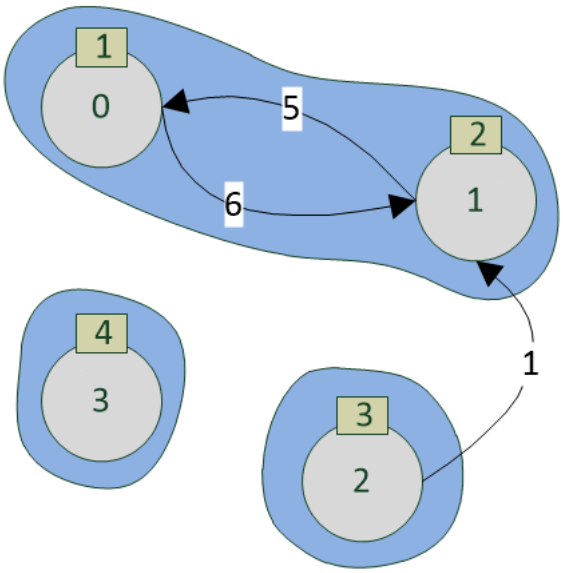

Рис. 796. Компоненты сильной связности

Выделение компонент сильной и слабой связности не зависит от того, в каком порядке задаются уравнения и коэффициенты при переменных. Выделение компоненты сильной связности без требования связи внутри компонент зависит от того, в каком порядке на вход механизму решения СЛАУ поступают определения уравнений и переменных. Так, если уравнения и коэффициенты задаются в естественном порядке (сверху вниз, слева направо), то будет выделено три компоненты (аналогично выделению компонент сильной связности):

- Компонента 1: {0, 1}.
- Компонента 2: {2}.
- Компонента 3: {3}.

Графически это изображено на рис.796.

Однако если вынести формирование параметров уравнения (2) в начало, то выделение компоненты сильной связности без требования связи внутри компонент завершится выделением двух компонент (аналогично выделению компонент слабой связности):

- Компонента 1: {0, 1, 2}.
- Компонента 2: {3}.

Графически это изображено на рис.795.

На встроенном языке вышеперечисленное будет выглядеть следующим образом:

Копировать в буфер обмена

```
Решатель = Новый РасчетСистемЛинейныхУравнений;
ДанныеУзлов = Новый ТаблицаЗначений;
ДанныеУзлов.Колонки.Добавить("НомерУравнения");
ДанныеУзлов.Колонки.Добавить("СвобКоэффициент");
Строка = ДанныеУзлов.Добавить();
Строка.НомерУравнения = 0;
Строка.СвобКоэффициент = 1;
Строка = ДанныеУзлов.Добавить();
Строка.НомерУравнения = 1;
Строка.СвобКоэффициент = 1;
Строка = ДанныеУзлов.Добавить();
Строка.НомерУравнения = 2;
Строка.СвобКоэффициент = 1;
Строка = ДанныеУзлов.Добавить();
Строка.НомерУравнения = 3;
Строка.СвобКоэффициент = 1;
ДанныеСвязей = Новый ТаблицаЗначений;
ДанныеСвязей.Колонки.Добавить("НомерУравнения");
ДанныеСвязей.Колонки.Добавить("НомерПеременной");
ДанныеСвязей.Колонки.Добавить("Коэффициент");
Строка = ДанныеСвязей.Добавить();
Строка.НомерУравнения = 0;
Строка.НомерПеременной = 0;
Строка.Коэффициент = 1;
Строка = ДанныеСвязей.Добавить();
Строка.НомерУравнения = 0;
Строка.НомерПеременной = 1;
Строка.Коэффициент = 5;
Строка = ДанныеСвязей.Добавить();
Строка.НомерУравнения = 1;
Строка.НомерПеременной = 0;
Строка.Коэффициент = 6;
Строка = ДанныеСвязей.Добавить();
Строка.НомерУравнения = 1;
Строка.НомерПеременной = 1;
Строка.Коэффициент = 2;
Строка = ДанныеСвязей.Добавить();
Строка.НомерУравнения = 1;
Строка.НомерПеременной = 2;
Строка.Коэффициент = 1;
Строка = ДанныеСвязей.Добавить();
Строка.НомерУравнения = 2;
Строка.НомерПеременной = 2;
Строка.Коэффициент = 3;
Строка = ДанныеСвязей.Добавить();
```

```
Строка.НомерУравнения = 3;
Строка.НомерПеременной = 3;
Строка.Коэффициент = 4;
Решатель.ИсточникДанныхСвязей = ДанныеСвязей;
Решатель.КолонкаУравненияВСвязях = "НомерУравнения";
Решатель.КолонкаПеременныеВСвязях = "НомерПеременной";
ОписаниеСЛАУ = Решатель.ОписанияСистем.Добавить();
ОписаниеСЛАУ.КолонкаКоэффициентовВСвязях = "Коэффициент";
СлабаяСвязность =
Решатель.ПолучитьКомпоненты Связности(СпособПолученияКомпонентСвязностиРасчетаСистемЛинейныхУравнений.КомпонентыСлабойСвязности);
СильнаяСвязность =
Решатель.ПолучитьКомпоненты Связности(СпособПолученияКомпонентСвязностиРасчетаСистемЛинейныхУравнений.КомпонентыСильнойСвязности);
СильнаяСвязностьБезТребования =
Решатель.ПолучитьКомпоненты Связности(СпособПолученияКомпонентСвязностиРасчетаСистемЛинейныхУравнений.КомпонентыСильнойСвязностиБезТре
```

В результате работы данного примера в переменных будут следующие значения:

- `СлабаяСвязность` - таблица значений, которая содержит компоненты слабой связности.
- `СильнаяСвязность` - таблица значений, которая содержит компоненты сильной связности.
- `СильнаяСвязностьБезТребования` - таблица значений, которая содержит компоненты сильной связности без требования связи внутри компонент. Данный вариант выделит компоненты аналогично компонентам сильной связности.

Если пример модифицировать, а именно:

- строки:

Копировать в буфер обмена
```
Строка = ДанныеУзлов.Добавить();
Строка.НомерУравнения = 2;
Строка.СвобКоэффициент = 1;
```

переставить сразу после строки:

Копировать в буфер обмена
```
ДанныеУзлов.Колонки.Добавить("СвобКоэффициент");
```

- строки:

Копировать в буфер обмена
```
Строка = ДанныеСвязей.Добавить();
Строка.НомерУравнения = 2;
Строка.НомерПеременной = 2;
Строка.Коэффициент = 3;
```

переставить сразу после строки:

Копировать в буфер обмена
```
ДанныеСвязей.Колонки.Добавить("Коэффициент");
```

то после выполнения примера, таблица `СильнаяСвязностьБезТребования` будет содержать компоненты, аналогичные компонентам слабой связности.

40.1.3. Рекомендации по использованию механизма поиска компонент связности

Рекомендуются следующие сценарии использования механизма поиска компонент связности.

Использование поиска компонент слабой связности

Простейший алгоритм, разделяющий систему линейных уравнений на абсолютно несвязанные части. Каждая из этих частей может быть:

- Изменена без влияния на решения других частей СЛАУ.
- Решена независимо от других частей СЛАУ.
- Подвергнута каким-либо дополнительным обработкам без влияния на решения других частей СЛАУ.

Не рекомендуется производить отдельный независимый расчет частей СЛАУ, если эти части невелики (например, эффективнее решить 100 частей системы по 10 уравнений совместно, чем отдельно).

Поиск компонент слабой связности также рекомендуется использовать, если необходимо провести итеративное решение СЛАУ: решить, какую-то часть СЛАУ, подвергнуть дополнительным обработкам исходную систему, решить измененную систему. В этом случае рекомендуется поступать следующим образом:

- Выделить компоненты слабой связности.
- Найти те компоненты, в которых лежит изменяемая часть СЛАУ (эти компоненты будут называться «изменяемыми»). Оставшиеся компоненты будут называться «неизменными». Затем необходимо выполнить следующие действия:
  - Решить независимо изменяемые и неизменные компоненты.
  - Внести требуемые изменения в изменяемые компоненты.
  - Повторно решить измененные компоненты.
  - Объединить решение неизменных компонент и результирующее решение изменяемых компонент для получения общего решения исходной СЛАУ.

Использование поиска компонент сильной связности

Рекомендуется использовать механизм, когда использование механизма нахождения компонент слабой связности не принес ожидаемого результата: было выделено слишком мало компонент слабой связности (или вообще выделена одна компонента).

Компоненты сильной связности - более «плотные» подграфы исходного графа.

Рекомендуется попробовать построить компоненты сильной связности, в ручном или автоматическом режиме проанализировать, какие в исходном графе существуют связи между выделенными компонентами сильной связности, и попытаться их убрать одним из путей:

- Редактирование исходных данных.
- Изменение модели, по которой была сформирована данная СЛАУ.
- Временное удаление этих ребер по сценарию:
  - Временно удаляются ребра, связывающие компоненты сильной связности.

• Рассчитываются компоненты слабой связности. Если между двумя компонентами сильной связности были удалены все ребра, их связывающие, то при дальнейшем построении компонент слабой связности эти самые компоненты и сформируют слабые. Т.е. если удалить все пути между сильными компонентами - то в дальнейшем сильные и слабые компоненты будут выглядеть одинаково.

• Производится необходимый расчет по одному из сценариев, указанных в «Использование поиска компонент слабой связности».

• Временно удаленные ребра возвращаются в исходный граф.

• Производится дополнительная обработка по релаксации решения СЛАУ. Релаксация необходима, т. к. временно удаленные ребра могут изменить (и, наверняка, изменят) итоговое решение СЛАУ. Для более эффективной и простой релаксации рекомендуется также проверить устойчивость решений частей СЛАУ, выделенных с помощью поиска компонент слабой связности.

Использование поиска компонент из расчета СЛАУ

Данный механизм, в первую очередь, предназначен для эффективной отладки и более глубокого понимая работы механизма расчета систем линейных уравнений. Его рекомендуется использовать в случаях ручного анализа систем линейных уравнений. Компоненты, которые строит данный механизм (для любой вершины компоненты не существует пути в любую вершину любой другой компоненты), зависят от порядка нумерации вершин, но позволяют итеративно анализировать систему уравнений, начиная с компоненты с номером 0 и двигаясь далее по компонентам, т. к. компонента с номером 0 гарантированно не связана с последующими компонентами ориентированным ребром.

Данный способ рекомендуется использовать для следующих ситуаций:

1. Система линейных уравнений очень большая, но хочется понять основные связи - какие вершины оказывают наибольшее влияние на другие вершины. Для этого необходимо итеративно идти по компонентам связности, анализируя, какие вершины входят в уравнения с наибольшими коэффициентами. По результатам этого анализа можно проводить анализ на устойчивость решения: несильно меняя коэффициенты при ключевых вершинах, смотреть, насколько сильно изменится итоговое решение СЛАУ.

2. Ручная проверка результатов расчета: анализ ситуаций, в которых возникло переполнение значений в решении СЛАУ, либо же в решении вершина отсутствует (к такому результату приводит, например, указание не всех коэффициентов СЛАУ или некорректное указание коэффициентов СЛАУ).

40.2. Интеграция с системой «1С:Аналитика»

40.2.1. Общая информация

Система аналитики - это отдельно поставляемая компонента платформы «1С:Предприятие», предоставляющий возможности по визуализации и интерактивному анализу данных, находящихся в информационной базе системы «1С:Предприятие». Система «1С:Предприятие» реализует «видимые» и «невидимые» (для пользователю) части, которые необходимы для того, чтобы пользователи могли использовать систему аналитики. К «видимым» частям можно отнести стандартную функцию, которая позволит настроить взаимодействие системы аналитики и платформы «1С:Предприятие», а также пункт меню клиентского приложения, с помощью которого можно открыть интерфейс системы «1С:Аналитика». Еще одной «видимой» частью можно назвать флажок настройки публикации информационной базы на веб-сервере. Под «невидимой» частью понимается программный интерфейс, с помощью которого «1С:Предприятие» взаимодействует с системой аналитики. Весь обмен данными выполняется по этому программному интерфейсу.

Для того чтобы пользователь получил возможность пользоваться системой аналитики, необходимо выполнить следующие действия:

• Опубликовать информационную базу системы «1С:Предприятие» на веб-сервере (см. здесь). При публикации следует обязательно установить флажок Публиковать систему аналитики.

• С помощью стандартной функции системы «1С:Предприятие» указывается адрес сервера «1С:Аналитики» в требуемой информационной базе.

• Нужным пользователям предоставляется право доступа `КлиентСистемыАналитики` (см. здесь).

• Для пользователей, которым предоставлено такое право, в интерфейсе клиентского приложения становится доступной команда запуска клиентской части системы аналитики.

Вопросы установки и настройки собственно «1С:Аналитики» в данной документации не рассматриваются. Кроме специальной команды меню клиентского приложения, к системе аналитики можно получить доступ по адресу, который формируется следующим образом: http://host/base/ans. Более подробно рассмотрим составные части адреса:

• http://host/base - обычный URL, который используется для доступа к информационной базе с помощью веб-клиента. При наличии разделителей, не поддерживается указание значений разделителей с помощью параметра `Z` командной строки запуска клиентского приложения.

• ans - признак того, что выполняется обращение к клиенту системы аналитики.

Для доступа системы аналитики к данным информационой базы будет использоваться URL http://host/base/ans/data. Оба указанных URL будут доступны только после публикации информационной базы с включенным флажком Публиковать систему аналитики.

Аутентификация пользователя осуществляется системой «1С:Предприятие». Если пользователь использует URL системы аналитики (вида http://host/base/ans), то используется диалог аутентификации веб-клиента. В случае запуска системы аналитики из клиентского приложения, данные аутентификации передаются из клиентского приложения «1С:Предприятия».

40.2.2. Программный интерфейс

Для взаимодействия с системой аналитики из встроенного языка предназначено свойство глобального контекста `СистемаАналитики`. В документации, для упрощения, будет опускаться префикс метода `СистемаАналитики.`. В примерах и реальных приложениях, очевидно, такие пропуски недопустимы.

Для использования из встроенного языка доступны следующие возможности:

• Установить и получить адрес сервера системы «1С:Аналитика». Для этого предназначены методы `УстановитьАдресСервераСистемыАналитики()`/`ПолучитьАдресСервераСистемыАналитики()`.

• Выполнить переход к объекту системы аналитики. Для этого предназначен метод `Открыть()`.

40.3. Сервисы интеграции

40.3.1. Общая информация

Часто встречаются ситуации, когда корпоративные информационные комплексы состоят из различных информационных систем. Скорее всего, это системы от различных производителей. В такой ситуации очевидным и естественным желанием является обеспечение эффективного взаимодействия всех этих систем. Можно выделить два подхода к интеграции: интеграция данных и функциональная интеграция. В первом случае обеспечивается обмен информацией между системами, во втором - обеспечивается возможность одной системы вызвать функцию другой системы с передачей данных и получением обратно результатов обработки. При решении задачи интеграции нескольких информационных систем необходимо обеспечивать:

• Достаточную оперативность - данные должны попадать потребителю как можно раньше и никак не позднее того, когда они должны потребителем использоваться.

• Достаточную надежность - данные должны дойти до потребителя даже в случае временного отказа инфраструктуры или временной недоступности самого потребителя.

• Минимально возможную связанность систем - системы должны продолжать работать независимо друг от друга, обновление одной системы не должно требовать обновления всех остальных и т. д.

Для организации взаимодействия различных информационных систем, как правило, можно использовать следующие способы:

• Обмен файлами.

• Использование общей базы данных.

• Вызов (в том числе удаленный) методов программного интерфейса одной системы из другой системы.

• Обмен сообщениями систем класса сервисная шина предприятия (Enterprise Service Bus, сокращенно ESB или СШП). Под термином «сервисная шина предприятия» понимается программное обеспечение, обеспечивающее централизованный и унифицированный событийно-ориентированный обмен сообщениями между различными информационными системами на принципах сервисно-ориентированной архитектуры.

Каждый из указанных выше способов интеграции имеет свои особенности:

• Файловый обмен. Системы-участники по расписанию генерируют файлы, содержащие данные для передачи другим участникам, и по расписанию же ищут подготовленные для них файлы. Оперативность, с которой данные оказываются в системе-потребителе, зависит от расписания выгрузки и загрузки. Можно увеличивать частоту выгрузки данных и добиваться большей оперативности передачи данных, но интенсивная работа с большим количеством файлов может привести потере производительности. Также производительность теряется, если требуется организовать файловый обмен с использованием низкоскоростных каналов передачи данных. Таким образом, файловый обмен позволяет выполнить интеграцию данных, но не позволит выполнить функциональную интеграцию. Файловый обмен не требует сильного связывания, но не обеспечивает высокой скорости обмена.

  Для реализации такого обмена в системе «1С:Предприятие» существуют объекты для работы с файлами и различными файловыми форматами. В качестве примеров можно рассмотреть работу с форматами XML/XDTO (см. здесь), JSON (см. здесь), двоичные данные (см. здесь), DBF (объект `XBase`) и текст (объекты `ТекстовыйДокумент`, `ЧтениеТекста`, `ЗаписьТекста`).

• Прямое обращение к базе данных другой системы плохо тем, что схема данных системы часто не является ее документированным контрактом, который эта система обязуется выполнять. Соответственно, при смене версии часто теряется работоспособность интеграции из-за изменений в схеме или изменений способа работы системы со своими данными. В некоторых случаях могут возникать сложности с лицензиями СУБД, которые могут явно запрещать ее использование сторонними системами. Для проприетарных СУБД важно также увеличение числа используемых подключений, которое может быть ограничено. Таким образом, обмен через общую базу данных позволяет выполнить интеграцию данных, но не позволит выполнить функциональную интеграцию. Обмен через общую базу данных требует сильного связывания, но позволяет обеспечить высокую скорость обмена.

  Для реализации такого обмена можно воспользоваться механизмом внешних источников данных (см. здесь).

• Вызов методов программного интерфейса одной системы из другой системы обеспечивает функциональную интеграцию, но приводит к еще более сильной связанности взаимодействующих систем.

  Система «1С:Предприятие» предоставляет возможность использовать программный интерфейс других систем с помощью интерфейсов Web-сервисов/SOAP (см. здесь), REST (см. здесь) или COM-интерфейс (только при работе под управлением ОС Windows).

• Обмен данными через сообщения. Обмен сообщениями сохраняет слабую связанность взаимодействующих систем аналогично файловому обмену. Обмен сообщениями дает возможность частого и быстрого взаимодействия небольшими сообщениями. При помощи обмена сообщениями можно интегрировать функциональность систем, используя паттерн «запрос-ответ». Передаваемые сообщения, при необходимости, могут быть трансформированы без участия и уведомления отправителя и получателей. При обмене сообщениями предоставляется возможность доставить одно сообщение множеству потребителей в оригинальном или измененном виде без изменения логики работы отправителя. Работа через сообщения меняет логику взаимодействия систем, обмен становится асинхронным. В этой схеме отправитель не должен ожидать ответа на свое сообщение, а должен быть готовым в любой момент времени принять какое-либо сообщение и адекватно на него отреагировать.

Рассмотрению вопросов обмена сообщениями посвящена данная глава.

Для поддержки обмена сообщениями в платформе «1С:Предприятие» предназначен механизм сервисов интеграции. При рассмотрении механизма будут использоваться следующие термины:

• **Внешний сервис интеграции** - продукт класса ESB, к которому выполняется подключение системы «1С:Предприятие». «1С:Предприятие» подключается к внешнему сервису интеграции с помощью протокола AMQP (https://www.amqp.org/). Адаптером для внешнего сервиса интеграции в объектной модели «1С:Предприятие» является объект Сервис интеграции.

• **Канал** - логический адрес в системе обмена сообщениями. Канал предназначен для передачи сообщений определенного типа. Канал может быть:

  • Входящим - используется для получения сообщений от внешнего сервиса интеграции.

  • Исходящим - используется для отправки сообщений внешнему сервису интеграции.

  • Канал не может быть двунаправленным (одновременно и входящим, и исходящим).

• **Сообщение** - это минимальный и неделимый элемент обмена. Сообщение включает в себя собственно данные (в любом поддерживаемом формате) и заголовок сообщения, который используется внешним сервисом интеграции. Сообщение имеет определенное время жизни, которое используется сервисом интеграции во время доставки сообщения.

При работе с сообщениями следует помнить о следующих особенностях:

• Сообщения помещаются в канал отправителем последовательно.

• Сообщения получаются из канала получателем последовательно. Последовательность получения сообщений из канала совпадает с последовательностью помещения сообщений в канал.

• Сообщения разных каналов обрабатываются и доставляются параллельно.

• Любые два сообщения, полученные из разных каналов в определенной последовательности, не обязательно будут обработаны в этой же последовательности, так как обработка сообщений из разных каналов может идти с разной скоростью.

Механизм сервисов интеграции «1С:Предприятия» не является альтернативной механизмам планов обмена, так как отвечает только за транспортировку сообщений, а не за формирование исходящих и интерпретацию входящих сообщений. В тоже время, механизм планов обмена может выступать в качестве источника данных для сообщений, отправляемых с помощью сервиса интеграции.

Взаимодействие с внешним сервисом интеграции выполняется с гарантированной доставкой сообщения, что означает:

• Отправляемое сообщение сохраняется в информационной базе до тех пор, пока от внешнего сервиса интеграции не будет получено подтверждение того, что сообщение им получено.

• Система «1С:Предприятие» будет выполнять попытки доставить сообщения внешнему сервису интеграции до тех пор, пока не будет получено подтверждение получения сообщения или сообщение не устареет.

• При получении сообщения от внешнего сервиса интеграции это сообщение сохраняется в информационной базе, и только после этого внешний сервис интеграции получает подтверждение о получении сообщения.

40.3.2. Редактирование сервиса интеграции

Сервисы интеграции расположены в дереве метаданных конфигурации в ветки Общие. Создание и редактирование объекта выполняется стандартным образом. Свойство объекта Адрес внешнего сервиса интеграции позволяет указать адрес используемого внешнего сервиса интеграции. После указания адреса внешнего сервиса интеграции необходимо указать каналы сервиса интеграции, которые будут использоваться из системы «1С:Предприятие». Для указания каналов доступны два способа:

1. Указать каналы вручную. Для этого предназначена закладка Каналы окна редактирования сервиса интеграции. При создании канала необходимо указать:

   • Имя - это идентификатор канала в рамках системы «1С:Предприятие».

   • Имя канала внешнего сервиса интеграции - это идентификатор канала в рамках внешнего сервиса интеграции. С помощью этого идентификатора будет выполняться однозначное сопоставление объекта в системе «1С:Предприятие» и объекта во внешнем сервисе интеграции.

   • Направление сообщения - указывает, что канал используется для приема или отправки сообщений. Если канал используется для приема сообщений, то становится возможным указать обработчик для обработки получаемых сообщений (свойство Обработчик получения сообщения). Обработчик будет размещаться в модуле сервиса интеграции. Как и все модули объектов, данный модуль исполняется на сервере системы «1С:Предприятие».

2. Выбрать каналы, которые будут использоваться в системе «1С:Предприятие» из списка доступных каналов указанного внешнего сервиса интеграции. Для этого предназначен диалог Загрузка каналов внешнего сервиса интеграции, доступ к которому предоставляется после выполнения команды Загрузить каналы контекстного меню конкретного сервиса интеграции. В данном диалоге выводятся все каналы выбранного внешнего сервиса интеграции.

40.3.3. Схема использования сервиса интеграции

Для использования внешнего сервиса интеграции необходимо:

1. Предварительные требования:

   • Узнать адрес внешнего сервиса интеграции (одного или нескольких), который планируется использовать. Указать адреса внешних сервисов интеграции можно при внедрении прикладного решения.

   • Узнать имена каналов (в каждом сервисе интеграции), которые будут использоваться в системе «1С:Предприятие».

2. Порядок действий при разработке:

   • Создать в конфигурации нужное количество объектов Сервис интеграции. Один объект соответствует одному внешнему сервису интеграции.

   • Для каждого созданного сервиса интеграции создать список каналов, который будет использоваться конфигурацией. Корректно указать имена каналов и направления их (каналов) использования.

   • Для исходящих каналов реализовать программный код, который позволит отправлять сообщения в нужный канал требуемого внешнего сервиса интеграции. В результате прикладное решение сможет отправлять данные в СШП.

   • Для каждого входящего канала реализовать обработчик, который будет срабатывать при получении события в конкретном канале конкретного сервиса интеграции. В результате прикладное решение сможет получать данные из СШП.

   • Для отладки созданного программного интерфейса, разработчику прикладного решения необходимо предоставить доступ к развернутому тестовому внешнему сервису интеграции. Тестовый внешний сервис интеграции должен предоставлять точно такие же возможности, что и реальная СШП, которая будет использоваться в производственной системе.

3. Порядок действий при внедрении:

   • С помощью стандартной функции управления сервисами интеграции указать адреса используемых внешних сервисов интеграции.

   • С помощью стандартной функции управления сервисами интеграции указать признаки использования сервисов интеграции (при необходимости).

40.3.4. Использование сервиса интеграции

Чтобы отправить сообщение, необходимо создать объект типа `СообщениеСервисаИнтеграции`. Для этого предназначен метод `СоздатьСообщение()` менеджера соответствующего сервиса интеграции. Объект типа `СообщениеСервисаИнтеграции` используется как при отправке, так и при получении сообщений. В связи с этим некоторые свойства данного объекта доступны только на чтение, а некоторым свойствам могут быть присвоены новые значения. Рассмотрим свойства объекта более подробно:

• `ДатаОтправки` - устанавливается в момент отправки сообщения. Недоступно для изменения из встроенного языка.

• `ДатаУстаревания` - позволяет установить момент времени, после наступления которого считается, что сообщение устарело и не должно обрабатываться. Сообщение считается устаревшим, если `ДатаУстаревания` сообщения меньше текущей универсальной даты компьютера, который выполняет обработку сообщений. Значение свойства можно изменить только для сообщения, которое еще не отправлено в канал.

Удаление устаревших сообщений выполняется из очереди во время выполнения фоновых заданий отправки и получения сообщений.

Хранится в формате UTC. При установке/получении значения выполняется преобразование с учетом часового пояса текущего сеанса.

• `Идентификатор` - уникальный идентификатор, который используется для однозначной идентификации сообщения. Данный идентификатор может использоваться при формировании цепочек «запрос-ответ». В этом случае идентификатор входящего сообщения следует установить в качестве значения свойства `ИдентификаторСообщенияЗапроса`.

• `ИдентификаторСообщенияЗапроса` - уникальный идентификатор, который позволит принимающей стороне определить, ответом на какое сообщение выступает данное сообщение. Можно изменить только для сообщения, которое еще не отправлено в канал.

• `КодОтправителя` и `КодПолучателя` - строковые идентификаторы, которые позволяют определить отправителя и получателя сообщения. Идентификаторы соответствуют кодам из справочника участников интеграции выбранного сервиса интеграции. При необходимости, можно указать одновременно несколько получателей сообщения. Для этого необходимо в свойство `КодПолучателя` записать строку, где коды получателей разделены символом «,». Значения свойств можно изменить только для сообщения, которое еще не отправлено в канал.

• `Параметры` - с помощью данного свойства можно установить для нового сообщения специфический набор параметров, который будет использоваться получателем для обработки сообщения. Имя параметра и его значение - значения строкового типа.

Для того чтобы установить в сообщение какие-либо данные (или получить какие-либо данные из сообщения), предназначен метод `ПолучитьТелоКакПоток()`. В результате работы метода возвращается значение типа `Поток` (подробнее см. здесь). При отправке сообщения в этот поток можно записывать произвольные данные, которые будут переданы в сообщении, а при чтении сообщения из этого потока можно считывать данные сообщения.

После того, как сообщение сформировано, его необходимо отправить получателю. Для этого необходимо использовать метод менеджера канала сервиса интеграции `ОтправитьСообщение()`. Для вызова метода следует использовать канал того сервиса интеграции, для которого было создано сообщение. Не поддерживается возможность создать сообщение для одного сервиса интеграции, а отправить его в канал другого сервиса интеграции. Сообщение попадает в очередь сообщений, которая расположена в базе данных, с которой работает система «1С:Предприятие». Для работы с очередью сообщений предназначены методы менеджера сервисов интеграции `ВыполнитьОбработку()` и `ОстановитьОбработку()`. Метод `ВыполнитьОбработку()` предназначен для запуска системных фоновых заданий, которые занимаются отправкой и получением сообщений из внешних сервисов интеграции. Если необходимо прервать работу с сообщениями, то это можно сделать двумя способами: прекратить выполнение регламентного задания или вызвав метод менеджера сервисов интеграции `ОстановитьОбработку()`.

40.3.5. Настройка и активность

Каждый сервис интеграции имеет настройки, которые можно устанавливать или из встроенного языка, или с помощью стандартной обработки. Описание стандартной обработки см. здесь. Данный раздел описывает программное изменение настроек сервиса интеграции.

Получение и установка настроек сервиса регистрации выполняются с помощью пары методов менеджера сервиса интеграции: `ПолучитьНастройки()` и `УстановитьНастройки()`. В обоих случаях используется объект `НастройкиСервисаИнтеграции`. Рассмотрим свойства этого объекта более подробно:

- `АдресВнешнегоСервисаИнтеграции` - позволяет указать адрес внешнего сервиса интеграции.

- `ИмяПользователяВнешнегоСервисаИнтеграции` и `ПарольПользователяВнешнегоСервисаИнтеграции` позволяют указать параметры аутентификации, используя которые платформа «1С:Предприятие» будет подключаться к внешнему сервису интеграции. Другими словами, это параметры пользователя внешнего сервиса интеграции. Задаются в том случае, если внешний сервис интеграции обладает механизмом авторизации.

- `ИмяПользователя` - позволяет указать пользователя системы «1С:Предприятие», от имени которого будет выполняться системное фоновое задание обработки полученных сообщений. Указанный пользователь будет определять права доступа. Если пользователь не указан, то используемый пользователь зависит от варианта работы системы «1С:Предприятие», а права доступа определяются свойством конфигурации `ОсновныеРоли`. Если это свойство не заполнено - контроль прав доступа не выполняется.

40.3.6. Организация работы сервисов интеграции

40.3.6.1. Получение сообщений

40.3.6.1.1. Общая информация

В общем случае, процесс получения (и обработки) сообщения из внешнего сервиса интеграции выглядит следующим образом:

- При выполнении метода менеджера сервисов интеграции `ВыполнитьОбработку()` платформа создает сетевое соединение с внешним сервисом интеграции. Данное соединение удерживается в течение 2 минут. В момент каждого вызова метода `ВыполнитьОбработку()` таймаут начинает отсчитываться от момента вызова метода. Если вызов метод не выполнялся в течение 2 минут - сетевое соединение закрывается, и взаимодействие с внешним сервисом интеграции прерывается. Данное поведение не зависит от того, в каком варианте работы информационной базы (файловом или клиент-серверном) выполняется работа с сервисами интеграции. В связи с этим, рекомендуется вызывать метод `ВыполнитьОбработку()` с помощью регламентного задания, которое должно срабатывать 1 раз в минуту.

- За обработку сообщений, поступающих от внешнего сервиса интеграции, отвечают несколько фоновых заданий. Специфика их работы зависит от используемого варианта информационной базы (файловый или клиент-серверный). Эта специфика будет рассмотрена подробнее в следующих разделах. В общем описании работы специфика файлового варианта не будет отдельно рассматриваться.

- Как только вешний сервис интеграции оповещает платформу о появлении нового сообщения, начинают свою работу 3 фоновых задания:

  - Фоновое задание получения сообщений выполняется как реакция на уведомление от внешнего сервиса интеграции. Фоновое задание получает сообщение, помещает его в очередь сообщений информационной базы, уведомляет внешний сервис интеграции о получении сообщения и завершает свою работу.

  - На следующем этапе запускается фоновое задание обработки очереди сообщений. Данное фоновое задание анализирует очередь сообщений и запускает фоновые задания обработки полученных сообщений. При запуске фоновым заданиям передается идентификатор обрабатываемого сообщения, а также указывается, от имени какого пользователя, и в какой области данных должно работать фоновое задание. После обработки всей очереди фоновое задание завершается.

  - Последним этапом является выполнение фонового задания обработки полученного сообщения. Это фоновое задание выполняет обработку конкретного сообщения, при этом фоновое задание выполняется от имени конкретного пользователя и в конкретной области данных. Именно из этого фонового задания происходит вызов обработчика получения сообщения внешнего сервиса интеграции, которые размещен в модуле объекта сервиса интеграции. Вызов обработчика полученного сообщения всегда выполняется в транзакции.

Наличие 3 фоновых заданий призвано решить несколько задач:

- Обеспечить подобные схемы работы в файловом и клиент-серверном вариантах работы.

- Реализовать минимальные задержки в получении сообщений от внешних сервисов интеграции.

- Обеспечить исполнение фонового задания обработки конкретного сообщения в конкретной области данных и от имени определенного пользователя, и сделать это с минимальными накладными расходами.

- Снизить накладные расходы при получении сообщений от внешних сервисов интеграции.

40.3.6.1.2. Файловый вариант

Каждое фоновое задание обработки сообщений всегда обрабатывает одно сообщение и завершает работу. Если сообщения для обработки нет, то ожидание сообщений не выполняется. Следует помнить, что в файловом варианте фоновые задания всегда выполняются последовательно и это выполнение обеспечивается одним экземпляром клиентского приложения. Вся последовательность действий, описанная ниже, будет выполняться для каждого (определенного в конфигурации) сервиса интеграции.

- В момент получения нового сообщения (через установленное сетевое соединение), платформа запускает фоновое задание получения сообщений. Данное фоновое задание запускается с максимальным приоритетом. Другими словами, вне зависимости от того, сколько фоновых заданий ожидают своего выполнения в момент поступления сообщения от внешнего сервиса интеграции, фоновое задание получения сообщений будет всегда запущено после завершения выполнения текущего фонового задания (если таковое есть).

- Фоновое задание получения сообщений выполняет получение одного сообщения. Сообщение записывается в очередь сообщений базы данных. Запускается (а фактически - планируется запуск) фоновое задание обработки получения сообщений. Внешний сервис интеграции уведомляется об успешном получении сообщения. Фоновое задание получения сообщений завершается.

- Фоновое задание обработки очереди сообщений получает из очереди одно сообщение. Определяются параметры обработки этого сообщения: имя пользователя и пароль, от имени которых будет обрабатываться сообщение, параметры области данных, в рамках которой будет выполняться обработка сообщения. Затем планируется запуск фонового задания обработки полученных сообщений, которому передается полученное сообщение. Фоновое задание запускается с ранее определенными параметрами. Затем фоновое задание обработки очереди сообщений анализирует, есть в локальной очереди необработанные сообщения или нет. Если сообщения есть - фоновое задание обработки очереди сообщения планирует свой повторный запуск. После завершения всех вышеописанных действий фоновое задание обработки очереди завершает свою работу.

• Фоновое задание обработки полученных сообщений вызывает обработчика получения сообщения внешнего сервиса интеграции, который размещен в модуле объекта сервиса интеграции. Если сообщение получено успешно, то сообщение в очереди помечается как обработанное. Если код на встроенном языке «отказался» от получения сообщения (произошла ошибка или обработчик завершен с параметром `Отказ`, установленным в значение `Истина`), то выполняется отмена транзакции. В этом случае попытка обработки сообщения будет предпринята повторно. После выполнения этих действий фоновое задание завершает свою работу.

40.3.6.1.3. Клиент-серверный вариант

При работе клиент-серверного варианта для каждого сервиса интеграции, определенного в конфигурации, запускается все три фоновых задания работы с сообщениями. Однако фоновое задание обработки полученных сообщений после завершения своей работы не завершает свою работу, а «засыпает» на время 20 секунд. Если в течение 20 секунд потребуется обработать новое сообщение от внешнего сервиса интеграции, то обработка будет выполнена достаточно быстро, т. к. кластеру серверов не потребуется тратить время на создание нового сеанса фонового задания. Если следующее сообщение поступило более чем через 20 секунд после поступления предыдущего сообщения, то фоновое задание будет запущено заново.

• В момент получения нового сообщения (через установленное сетевое соединение), платформа передает управление фоновому заданию получения сообщений. Это фоновое задание получает все сообщения по «своему» каналу, записывает их в очередь сообщений базы данных и уведомляет фоновое задание обработки полученных сообщений. Затем внешний сервис интеграции уведомляется об успешном получении сообщения.

• Фоновое задание обработки очереди сообщений получает управление после получения очередного сообщения. Для каждого из полученных сообщений определяются параметры запуска фонового задания обработки полученных сообщений. Затем фоновому заданию обработки полученных сообщений передается для обработки идентификатор сообщения, которое требуется загрузить.

• Фоновое задание обработки полученных сообщений вызывает обработчика получения сообщения внешнего сервиса интеграции, который размещен в модуле объекта сервиса интеграции. Если сообщение получено успешно, то сообщение в очереди помечается как обработанное. Если код на встроенном языке «отказался» от получения сообщения (произошла ошибка или обработчик завершен с параметром `Отказ`, установленным в значение `Истина`), то выполняется отмена транзакции. В этом случае попытка обработки сообщения будет предпринята повторно. После выполнения этих действий фоновое задание «завершает» свою работу (с учетом особенностей, описанных в начале раздела).

Таким образом, ключевое отличие между файловым и клиент-серверным вариантом заключается (фактически) в устройстве механизма работы с фоновыми заданиями. В файловом варианте в каждый момент времени работает только одно фоновое задание и фоновые задания выполняются последовательно. В клиент-серверном варианте фоновые задания могут работать параллельно.

40.3.6.2. Отправка сообщений

Перед непосредственно отправкой, сообщение записывается в очередь сообщений, которая располагается в базе данных информационной базы. Запись в очередь сообщений выполняется при вызове метода `ОтправитьСообщение()`. Если вызов метода выполняется в транзакции, то запись в очередь выполняется при успешном завершении транзакции. При вызове метода `ВыполнитьОбработку()` запускается системное фоновое задание обработки очереди сообщений, которое занимается физической отправкой сообщений внешнему сервису интеграции. Если сообщение отправлено удачно - оно удаляется из очереди сообщений. Если отправка не удалась - сообщение остается в очереди до тех пор, пока не будет отправлено или не устареет, при этом оправка остальных сообщения в канал не выполняется, пока не будет выполнена отправка первого сообщения.

Сообщения будут отправляться только в случае, если зафиксирована транзакция в информационной базе, в рамках которой созданы сообщения. Очередность отправки сообщений в канал сервиса интеграции определяется по следующим правилам:

• Сообщения отправляются по порядку фиксации транзакций в информационной базе.

• Если в рамках одной транзакции создано несколько сообщений - они отправляются в порядке создания.

Не определена очередность отправки между различными каналами и сервисами интеграции. Не определена очередность отправки между различными областями данных одного канала.

40.3.7. Сервисы интеграции и разделение данных

Описание особенностей работы сервисов интеграции в режиме разделения данных см. здесь.