Business Studio 3.6
Система бизнес-моделирования

группа компаний
СОВРЕМЕННЫЕ ТЕХНОЛОГИИ УПРАВЛЕНИЯ

Руководство технического специалиста Business Studio

Версия 3.6

Информация, содержащаяся в этом документе, может быть изменена без предварительного уведомления, и Группа компаний «Современные технологии управления» не берет на себя на этот счет никаких обязательств.

Программное обеспечение, описываемое в этом документе, поставляется в соответствии с Лицензионным соглашением. Это программное обеспечение может быть использовано или скопировано лишь в строгом соответствии с условиями этого соглашения. Копирование этого программного обеспечения на какой-либо носитель информации, если на это нет специального разрешения в Лицензионном соглашении или в соглашении о нераспространении, является нарушением Закона Российской Федерации «О правовой охране программ для ЭВМ и баз данных» и норм международного права.

Никакая часть настоящего Руководства ни в каких целях не может быть воспроизведена или передана в какой бы то ни было форме и какими бы то ни было средствами, будь то электронные или механические, включая фотокопирование и запись на магнитные носители, если на то нет письменного разрешения Группы компаний «Современные технологии управления».

Редакция документа: 3.6.2

СОДЕРЖАНИЕ

ГЛАВА 1. УСТАНОВКА СЕРВЕРНОЙ ЧАСТИ BUSINESS STUDIO

Установка серверной части Business Studio на сервер должна осуществляться пользователем, имеющим права локального администратора. Перед началом установки рекомендуется закрыть все другие приложения.

Если на сервере была установлена Business Studio версии 3.5, необходимо удалить ее перед началом установки серверной части Business Studio. Для этого выберите меню «Пуск –> Все программы –> Business Studio –> Удаление Business Studio».

Ниже описаны варианты установки серверной части Business Studio:

- Автоматическая интерактивная установка.
- Ручная установка.

1.1 Автоматическая интерактивная установка

Автоматическая установка осуществляется с помощью setup.exe. При этом все компоненты, включая службу сервера лицензий, устанавливаются и регистрируются в системе автоматически. Подробное описание установки серверной части Business Studio дано в «Руководстве пользователя», п. 1.4 «Установка клиент-серверного варианта поставки Business Studio».

Варианты установки Business Studio:

- Полная серверная установка: аналогична персональной установке.
- Серверная установка: не включает в себя клиентскую часть.
- Клиентская установка: не включает в себя серверную часть.

Серверные варианты установки позволяют отказаться от установки SQL Server 2005 Express Edition.

1.2 Ручная установка

Ручная установка возможна двумя вариантами:

1. Установка с помощью командной строки.

   Выполнить строку вида

   «package.msi /q CFG_SERVER=1 CFG_ENT=1 TARGETDIR="<путь>"»,

   где <путь> – путь, по которому будут установлены серверные утилиты и сервис лицензий. При установке сервис лицензий будет зарегистрирован в системе, ярлыки утилит будут созданы в меню «Пуск». Подробное описание установки дано в п. 1.2.1 «Установка с помощью ключей командной строки».

2. Установка копированием.

   Скопировать файлы из установленной персональной версии Business Studio, кроме клиентских файлов (Business Studio.exe, *.xml, папка web).

   Зарегистрировать службу сервера лицензий, для этого выполнить строку:

   «%windir%\Microsoft.NET\Framework\v2.0.50727\InstallUtil.exe Ping.Service.exe».

Внимание: Путь к утилите InstallUtil в Вашей системе может быть другим.

После ручной установки необходимо выполнить следующие действия:

- произвести активацию сервиса лицензий с помощью Мастера активации (см. п. 1.2.2);
- установить (при необходимости) Microsoft SQL Server (см. п. 1.2.4);
- распаковать базы данных (см. п. 1.2.5).

1.2.1 Установка с помощью ключей командной строки

Установка осуществляется запуском файла package.msi с ключами. Командная строка:

«package.msi /q [CFG_SERVER=1] [CFG_CLIENT=1] [CFG_ENT=1] [CFG_PRO=1] [CFG_CPIT=1] [CFG_DOC=1] [TARGETDIR="<путь>"] [ALLUSERS=1]»

Здесь <путь> – путь, по которому будут установлены серверные утилиты и сервис лицензий.

Внимание: Должен быть задан хотя бы один из ключей CFG_SERVER или CFG_CLIENT.

Внимание: Должен быть задан хотя бы один из ключей CFG_PRO, CFG_ENT или CFG_CPIT.

Таблица 1.2.1 *Параметры командной строки*

Ключ	Описание
CFG_SERVER	Установка и регистрация серверных утилит и службы сервера лицензий.
CFG_CLIENT	Установка клиентской программы.
CFG_PRO	Установка версии Professional.
CFG_ENT	Установка версии Enterprise.
CFG_CPIT	Установка версии Cockpit.
CFG_DOC	Установка документации. Параметр игнорируется, если в пакете установки отсутствует папка «Документация».
TARGETDIR	Путь для установки.
ALLUSERS	Устанавливать для всех пользователей.

1.2.2 Удаление с помощью командной строки

Удаление Business Studio 3.5 или 3.6 с командной строки осуществляется строкой

«msiexec [/Q] /X{BDB217B2-6034-4579-A049-0A1CBFB9FEE4}»,

здесь Q – ключ запрета взаимодействия с пользователем: если ключ указан, то вопрос об удалении программы и ход удаления отображаться не будут.

1.2.3 Активация компьютера

Запуск Мастера активации осуществляется из папки установки программы, файл *Активация.exe*. Подробнее о Мастере активации см. Руководство пользователя, п. 1.6 «Активация программы».

1.2.4 Установка SQL Server

Если планируется использовать полную версию Microsoft SQL Server версии 2000 или выше, то установите Microsoft SQL Server перед установкой Business Studio (см. инструкцию по установке из комплекта Microsoft SQL Server). В противном случае, необходимо установить Microsoft SQL Server 2005 Express Edition, из комплекта установки Business Studio.

Внимание: В операционных системах Windows Vista, Windows 7, Windows 2008/R2 администратор компьютера не является по умолчанию администратором SQL Server.

Чтобы включить опцию администрирования необходимо (пример для Windows Vista):

- Выбрать меню «Пуск»  –> Все программы –> Microsoft SQL Server 2005 –> Средства настройки –> Настройка контактной зоны SQL Server».
- В открывшемся окне «Настройка контактной зоны SQL Server 2005» нажать гиперссылку «Добавление нового администратора». Откроется окно «Выделение ресурсов пользователям SQL Server под управлением ОС Vista - <Имя пользователя> on …».
- Перенести категорию «Член роли «Системный администратор SQL Server» on SQLEXPRESS» в список «Привилегии, предоставляемые пользователю».

Внимание: В SQL Server 2005 Express Edition по умолчанию разрешены только локальные соединения.

Чтобы разрешить удаленные соединения к SQL Server необходимо:

- Выбрать меню «Пуск»  –> Все программы –> Microsoft SQL Server 2005 –> Средства настройки –> Настройка контактной зоны SQL Server».
- В открывшемся окне «Настройка контактной зоны SQL Server 2005» нажать гиперссылку «Настройка контактной зоны для служб и соединений».
- В открывшемся окне выбрать компонент «Удаленные соединения». Выбрать опции «Локальные и удаленные соединения», «Использовать TCP/IP и именованные каналы».

1.2.5 Создание баз данных

Создание баз данных возможно несколькими способами:

1. с помощью утилиты DB Администратор (см. Руководство пользователя, п. 1.8 «Управление базами данных»);
2. с помощью SQL Server Management Studio, если установлена полная версия Microsoft SQL Server (см. п. «Создание баз данных в SQL Server Management Studio»).

Создание баз данных в SQL Server Management Studio

Создание базы данных Business Studio в SQL Server Management Studio является, по сути, восстановлением ее из резервной копии с последующем прописыванием имени сервера

лицензий в одной из таблиц. Резервные копии пустой (empty.db) и демонстрационной (demo.db) баз данных находятся в папке «Backup» в каталоге установки программы.

Чтобы восстановить базу данных из резервной копии, необходимо в списке баз сервера (раздел *Базы данных*) выбрать пункт контекстного меню «Восстановить базу данных…».

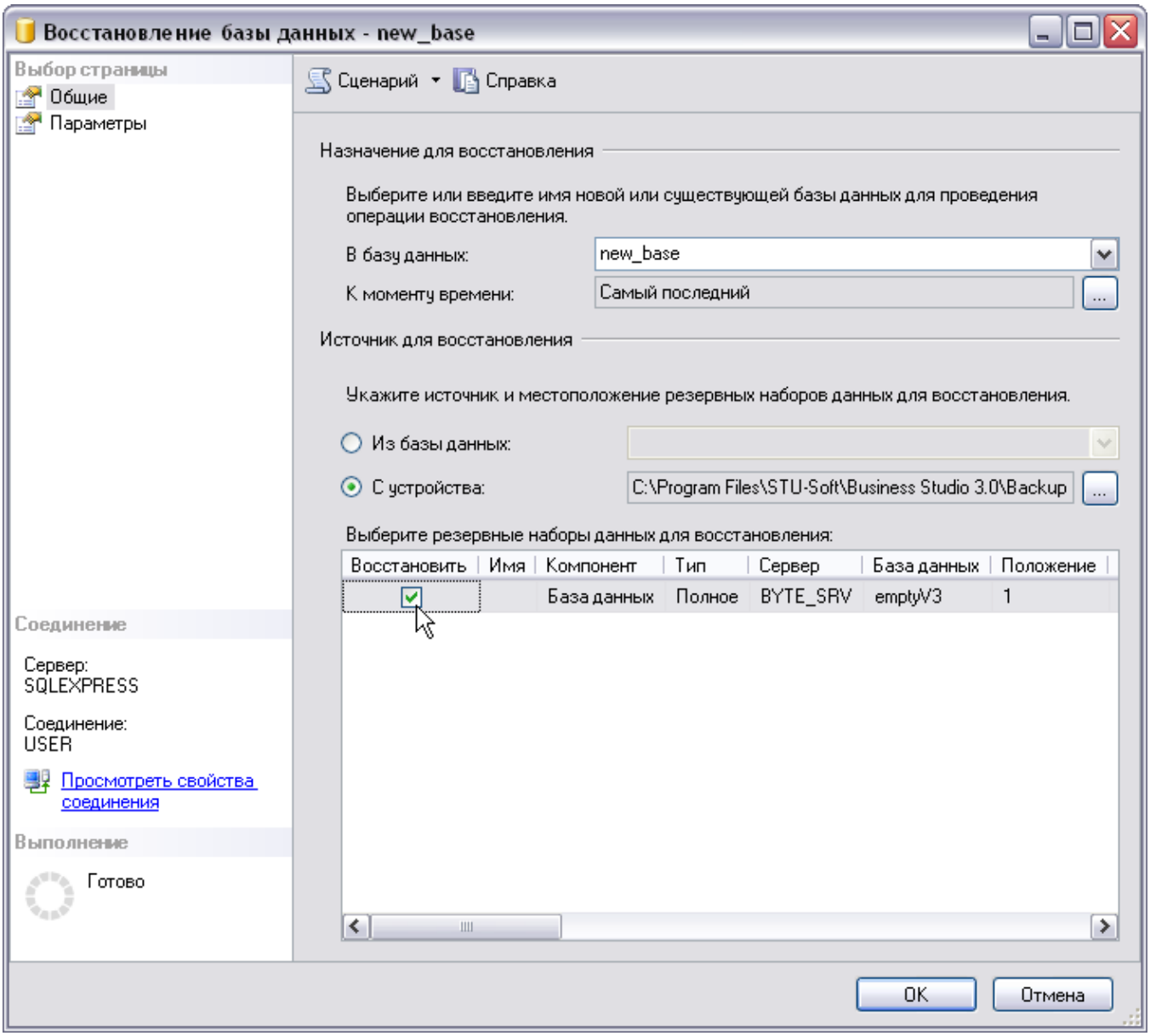

Рис. 1.2.1

В открывшемся окне «Восстановление базы данных» на странице ***Общие*** (Рис. 1.2.1) необходимо произвести следующие действия:

- в графе «В базу данных» ввести название новой базы (или выбрать название уже существующей);
- в качестве источника выбрать *С устройства* (восстановление из указанного источника).

Затем нажать на кнопку … для выбора резервной копии для восстановления базы данных.

В открывшемся окне «Указание резервной копии» в графе «Носитель резервной копии:» выбрать *Файл*. Нажать кнопку «Добавить», указать путь к файлу резервной копии базы данных, либо выбрать файл базы по кнопке ….

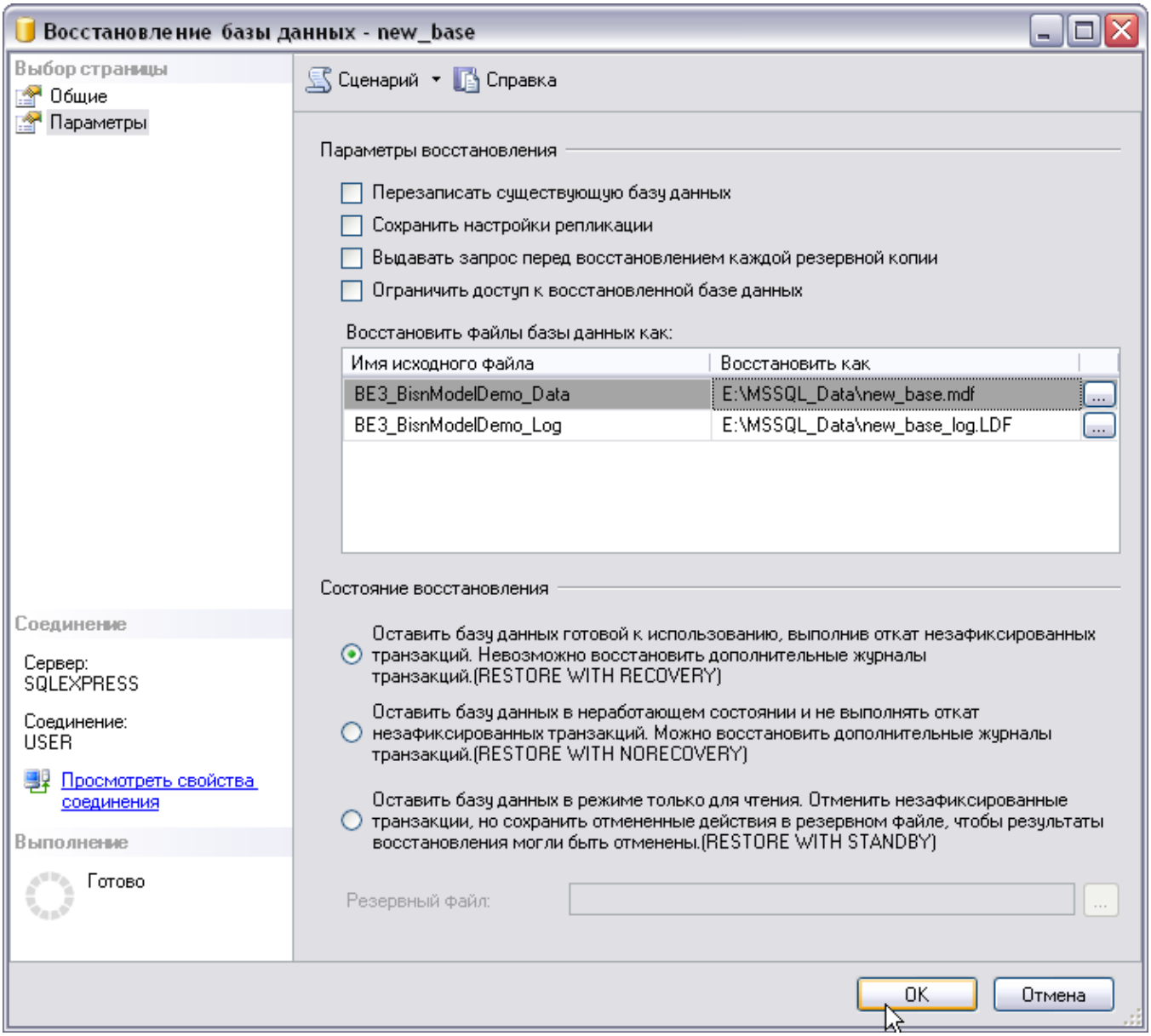

Рис. 1.2.2

При необходимости выбрать путь хранения новой базы данных. Для этого в окне «Восстановление базы данных» на странице ***Параметры*** (Рис. 1.2.2) редактируется колонка *Восстановить как*.

Внимание: Изменение физического местонахождения базы возможно только до ее создания! Перемещение уже созданной базы запрещено.

После подтверждения всех указанных настроек нажатием кнопки «ОК» дождаться завершения процесса создания базы данных и сообщения «Восстановление базы данных "<имя_базы>" успешно завершено!».

После этого созданная база данных появится в разделе *Базы данных* дерева «SQL Server Management Studio» и в списке доступных баз «Business Studio».

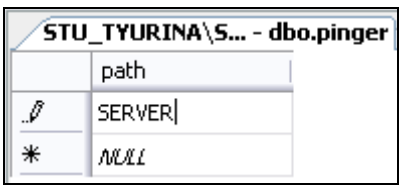

Рис. 1.2.3

Теперь необходимо указать имя сервера лицензий для созданной базы данных. Для этого в новой базе данных необходимо открыть таблицу *pinger* в списке базы *Таблицы* с помощью пункта контекстного меню таблицы «Открыть таблицу»; и ввести в первой строке таблицы путь к компьютеру, на котором установлен Сервис лицензий (Рис. 1.2.3). Путь к компьютеру может быть в виде:

- FQDN – полное доменное имя компьютера (например, server.mydomain.ru);
- имя NetBIOS – сетевое имя компьютера (например, SERVER);
- IP-адрес компьютера, в т.ч. внешний (например, 192.168.1.184).

В случае создания баз данных на том же компьютере, где установлен Сервис лицензий, в таблице *pinger* указывается путь к данному компьютеру.

Внимание: Все пользователи базы данных должны иметь доступ к серверу лицензий по пути, указанному в таблице *pinger*.

В частности, убедитесь, что никакое ПО не блокирует подключения к TCP-порту сервера лицензий (по умолчанию используется порт 5555). Таким ПО может быть как встроенный брандмауэр Windows, так и брандмауэры, и антивирусные средства сторонних производителей (см. п. 1.4).

Если путь указан не в виде IP-адреса, то также убедитесь, что клиентские компьютеры не имеют проблем с разрешением FQDN- или NetBIOS-имени сервера лицензий (например, командой «ping»).

Рекомендуется использовать автоматическое резервное копирование баз данных.

1.2.6 Список баз данных

Список баз данных хранится в реестре в ветке «HKEY_CURRENT_USER\Software\STU-Soft\Business Studio 3.6\DB», т.е. индивидуален для каждого пользователя. В списке содержится информация о каждой базе данных: имя сервера, имя базы, режим аутентификации, пользовательское наименование, комментарий. Чтобы скопировать список баз для других пользователей и/или на другие компьютеры, можно скопировать ветку реестра, для этого:

Под пользователем, имеющим образцовый список баз данных:

- запустить редактор реестра «regedit» из командной строки;
- открыть ветку «HKEY_CURRENT_USER\Software\STU-Soft\Business Studio 3.6\DB»;
- экспортировать ветку в файл с помощью пункта контекстного меню «Экспортировать».

Под другим пользователем или на другом компьютере:

- запустить редактор реестра «regedit» из командной строки;
- импортировать ветку реестра из созданного файла с помощью пункта главного меню «Файл –> Импорт…»;
- другой способ – запустить созданный файл и на вопрос о добавлении информации в реестр ответить «Да».

1.3 SQL аутентификация

SQL аутентификация применяется в одноранговых сетях. Для работы с Business Studio с использованием SQL аутентификации необходимо сделать следующее:

1) Создать пользователя SQL с помощью утилиты «DB администратор» (см. Руководство пользователя, п. 1.9 «Управление доступом к базам данных») или с использованием SQL Server Management Studio (см. п. 1.5.1).
2) Убедиться, что на сервере включена смешанная аутентификация с помощью SQL Server Management Studio (см. ниже) или через реестр, см. раздел «Определение и изменение метода проверки подлинности» в статье на сайте Microsoft: http://support.microsoft.com/kb/322336/ru.
3) Перевести базы данных на использование режима SQL аутентификации, для этого включить опцию *SQL аутентификация* в свойствах базы данных (подробнее см. Руководство пользователя, п. 1.8.2 «Свойства подключения к базе данных».

Изменение режима аутентификации сервера с помощью SQL Server Management Studio

Для изменения режима аутентификации необходимо выделить сервер в дереве объектов SQL Server Management Studio и в контекстном меню выбрать пункт «Свойства». В открывшемся окне «Свойства сервера» на странице ***Безопасность*** в разделе «Серверная проверка подлинности» установить опцию «Проверка подлинности SQL Server и Windows».

1.4 Доступ к службе сервера лицензий

Служба сервера лицензий – это служба с именем BS_PingHost.

Удаленные подключения к серверу лицензий осуществляются по адресу *http:||<Имя_сервера_лицензий>:<Номер_порта>.*

<Имя_сервера_лицензий> может быть в виде:

- имя NetBIOS – сетевое имя компьютера (например, SERVER);
- IP-адрес компьютера, в т.ч. внешний (например, 192.168.1.184);
- FQDN – полное доменное имя компьютера (например, server.mydomain.ru).

Внимание: некоторые варианты могут быть недоступны в зависимости от конфигурации компьютера или сети.

<Номер_порта> по умолчанию 5555. Убедитесь, что подключения к TCP-порту сервера лицензий не заблокированы. Блокировать подключения могут различны средства: встроенный брандмауэр Windows, брандмауэры и антивирусные средства сторонних производителей, а также активное сетевое оборудование. О том, как разрешить подключения на данный порт, смотрите документацию по этим средствам.

Если порт 5555 занят другим приложением, то можно заменить его на другой, например, порт 3333. Для этого необходимо выполнить следующие действия:

- На компьютере, на котором установлен сервис лицензий, открыть в блокноте файл "Ping.Service.config" из папки "PingService" в папке установки программы (по умолчанию, C:\Program Files\STU-Soft\Business Studio

3.6\PingService\Ping.Service.exe.config), заменить в нем строку *<channel ref="http" port="5555">* на *<channel ref="http" port="3333">)*.

- Открыть доступ к порту 3333 на этом компьютере.
- Перезапустить службу BS_PingHost. Это можно сделать из консоли "Службы" в разделе "Администрирование" Панели управления, либо перезапуском компьютера.
- На компьютерах, на которых установлено Business Studio, открыть в блокноте файл "Business Studio.exe.config" из папки установки программы (по умолчанию, C:\Program Files\STU-Soft\Business Studio 3.6\Business Studio.exe.config), заменить в данном файле строку *<add key="PingPort" value="5555"/>* на *<add key="PingPort" value="3333"/>)*.

1.4.1 Количество подключенных клиентов

Количество подключенных клиентов можно просмотреть в Business Studio в окне «О программе», которое вызывается из Главного меню «Помощь –> О программе». В п. 4.1 «Просмотр подключений к серверу лицензий» дан пример определения имен компьютеров, с которых установлено подключение.

1.5 Настройка прав доступа к базам данных

Управление доступом к базам данных можно производить несколькими способами:

1. с помощью утилиты DB Администратор, поставляемой в комплекте установки (см. Руководство пользователя п. 1.9 «Управление доступом к базам данных»);
2. с помощью SQL Server Management Studio (см. п. 1.5.1).

1.5.1 Создание пользователей и настройка прав доступа к базам данных в SQL Server Management Studio

Создание и редактирование свойств пользователей осуществляется из раздела *Безопасность* дерева объектов SQL Server Management Studio. Для добавления нового пользователя необходимо выделить раздел *Имена входа*, в контекстном меню выбрать пункт «Создать имя входа…». Откроется окно «Создание имени входа» (Рис. 1.5.1).

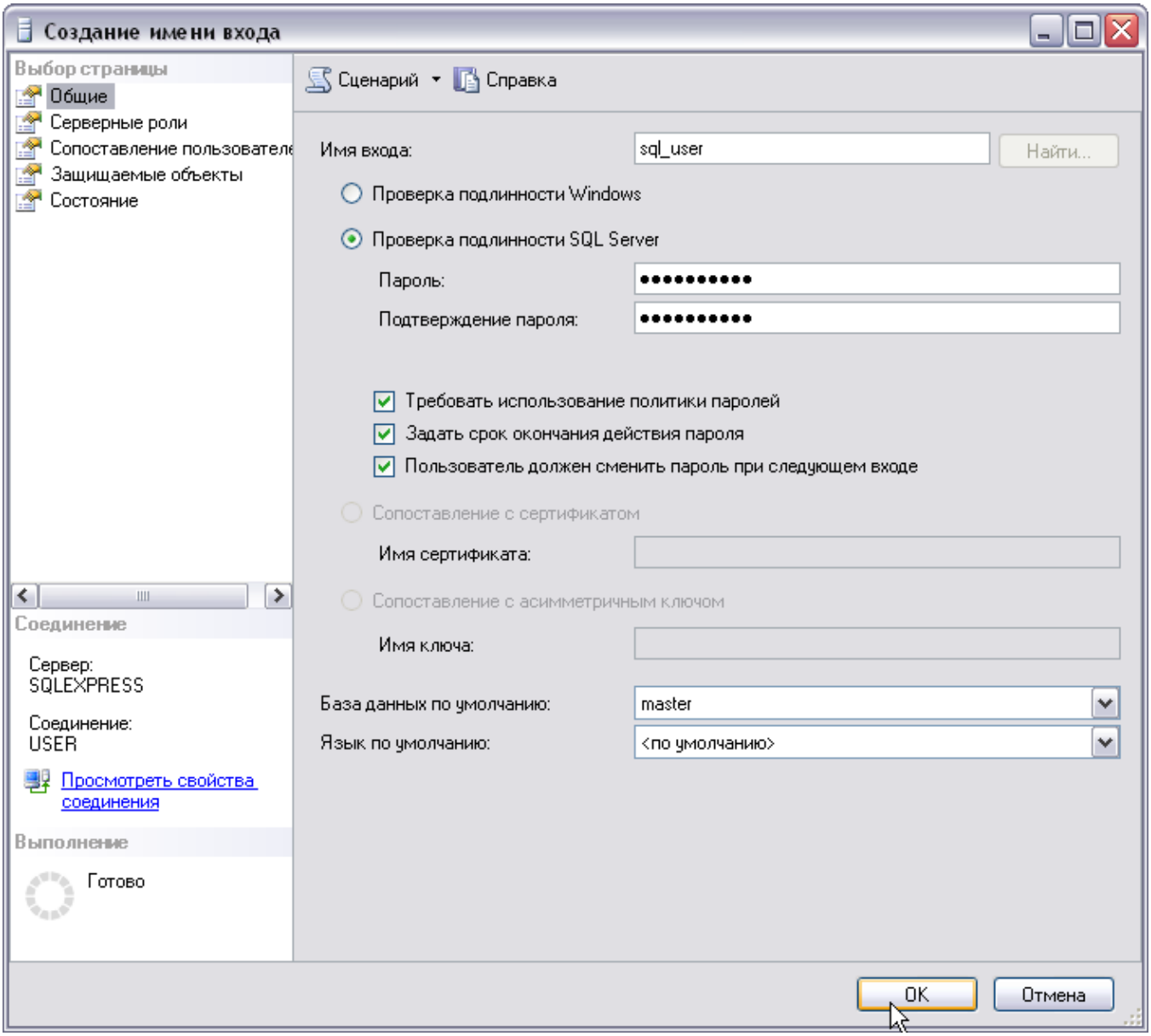

Рис. 1.5.1

Задать проверку подлинности пользователя – *Проверка подлинности Windows* или *Проверка подлинности SQL server*.

Для создания пользователя с аутентификацией Windows необходимо выбрать пользователя или группу пользователей по кнопке «Найти…».

Для создания пользователя с аутентификацией SQL необходимо ввести имя пользователя в графе *Имя пользователя*, указать пароль и подтверждение пароля. Пользователь с SQL аутентификацией создается в случае использования SQL аутентификации базы данных (подробнее см. Руководство пользователя, п. 1.8.2 «Свойства подключения к базе данных»).

На странице ***Сопоставление пользователей*** отмечаются галочками базы данных, для которых установлен доступ для данного пользователя и членство в роли в базе денных. Новый пользователь в базе данных создается с правами Пользователя (подробнее о возможностях групп пользователей см. Руководство пользователя, п. 1.9 «Управление доступом к базам данных»). Чтобы дать пользователю права Администратора базы данных, необходимо в списке *Членство в роли базы данных для: <имя_базы>* отметить галочкой роль ‘db_owner’.

Также возможно добавление существующего пользователя из раздела *Безопасность -> Пользователи* выделенной базы. Для этого необходимо выбрать пункт контекстного меню «Создать пользователя…», задать *Имя пользователя,* ввести вручную или выбрать по кнопке  *Имя входа* созданного ранее пользователя. Чтобы дать пользователю права Администратора базы данных, необходимо в списке *Членство в роли базы данных* отметить галочкой роль 'db_owner'.

1.6 Резервное копирование баз данных

Резервное копирование баз данных можно производить несколькими способами:

1. с помощью утилиты DB Администратор (см. Руководство пользователя, п. 1.8 «Управление базами данных»);
2. с помощью SQL Server Management Studio (см. п. 1.6.1).
3. Автоматическое резервное копирование (см. п. 1.6.2).

1.6.1 Резервное копирование баз данных в SQL Server Management Studio

Для создания резервной копии базы данных необходимо выделить ее в дереве объектов SQL Server Management Studio, в контекстном меню выбрать пункт «Задачи –>Создать резервную копию…».

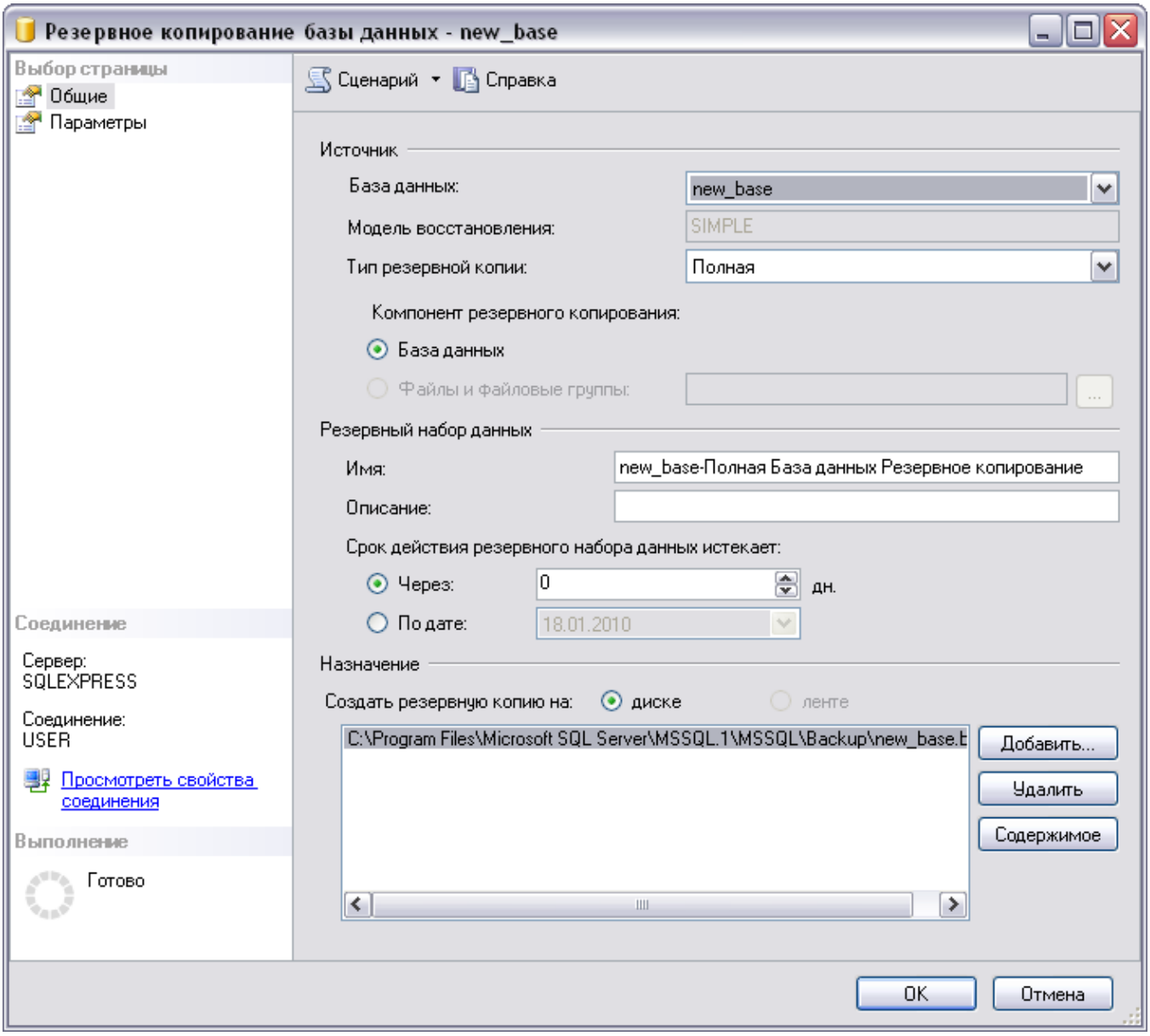

Рис. 1.6.1

В окне «Резервное копирование базы данных» (Рис. 1.6.1) на странице ***Общие*** в разделе «Источник» в поле «Тип резервной копии выбрать *Полная*; в разделе «Назначение» по кнопке «Добавить» указать файл резервной копии базы данных. Для проверки целостности копии базы данных на закладке ***Параметры*** в разделе «Надежность» установить флаг *Проверить резервную копию после завершения.*

Нажатием кнопки «OK» запустить создание резервной копии выбранной базы данных и дождаться сообщения «Резервное копирование базы данных "<имя_базы>" успешно завершено.».

1.6.2 Автоматическое резервное копирование

Настройка автоматического резервного копирования баз данных возможна разными способами. Здесь приведена схема работы скрипта, создающего резервные копии указанных баз данных в указанные папки.

Схема работы:

1. Скрипт запускается непосредственно на SQL Server'e, имя инстанции SQL Server указывается в скрипте. Для выполнения SQL-кода указывается путь к соответ-

ствующей утилите. Создается резервная копия базы данных с указанием даты в имени файла. Файл сохраняется локально по указанному пути. Создаются лог-файлы резервного копирования для каждой базы с указанием имени базы в названии файла и общий лог-файл.

2. Файл резервной копии запаковываются архиватором. В скрипте необходимо указать используемый архиватор.
3. Созданный архив копируется на указанные сетевые источника хранения архивов при необходимости.
4. Старые архивы удаляются. В скрипте указывается утилита для удаления файлов.

Пример скрипта приведен в п. 4.5.

1.7 Активация онлайн-лицензий для Business Studio

Онлайн-лицензия - лицензия, возможность использования которой контролируется через сеть "Интернет".

Онлайн-лицензии служат для:

- предоставления ограниченной по времени возможности использования Business Studio (временные лицензии),
- активации на компьютерах, созданных с использованием программного обеспечения, эмулирующего компьютер (виртуальные машины).

Внимание! Онлайн-лицензия предназначена для использования только на одном экземпляре виртуальной машины. При обнаружении одновременного использования онлайн-лицензии Группа компаний «Современные технологии управления» вправе приостановить её действие.

Активация онлайн-лицензии аналогична активации обычной лицензии и описана в пункте 1.6 «Активация программы» Руководства пользователя, которое находится в папке «Документация» в каталоге установки программы. При активации онлайн-лицензии будет предложено произвести настройку параметров подключения. Также можно настроить их позднее с помощью утилиты «Ping.Config.exe», см. п. 1.7.1.

Внимание: Компьютер, на котором установлена онлайн-лицензия, должен иметь доступ в Интернет по адресам http://ls.businessstudio.ru и http://ls2.businessstudio.ru, по одному из следующих портов: 80, 443 или 5550. Доступ в Интернет осуществляет служба сервера лицензий BS_PingHost (Ping.Service.exe).

1.7.1 Настройка параметров подключения

Настроить параметры подключения и проверить имеющиеся на данный момент лицензии можно с помощью утилиты «Ping.Config.exe», которая находится в папке «PingService» в каталоге установки программы. Также утилита может быть вызвана из Мастера активации после активации временной лицензии по кнопке «Параметры подключения».

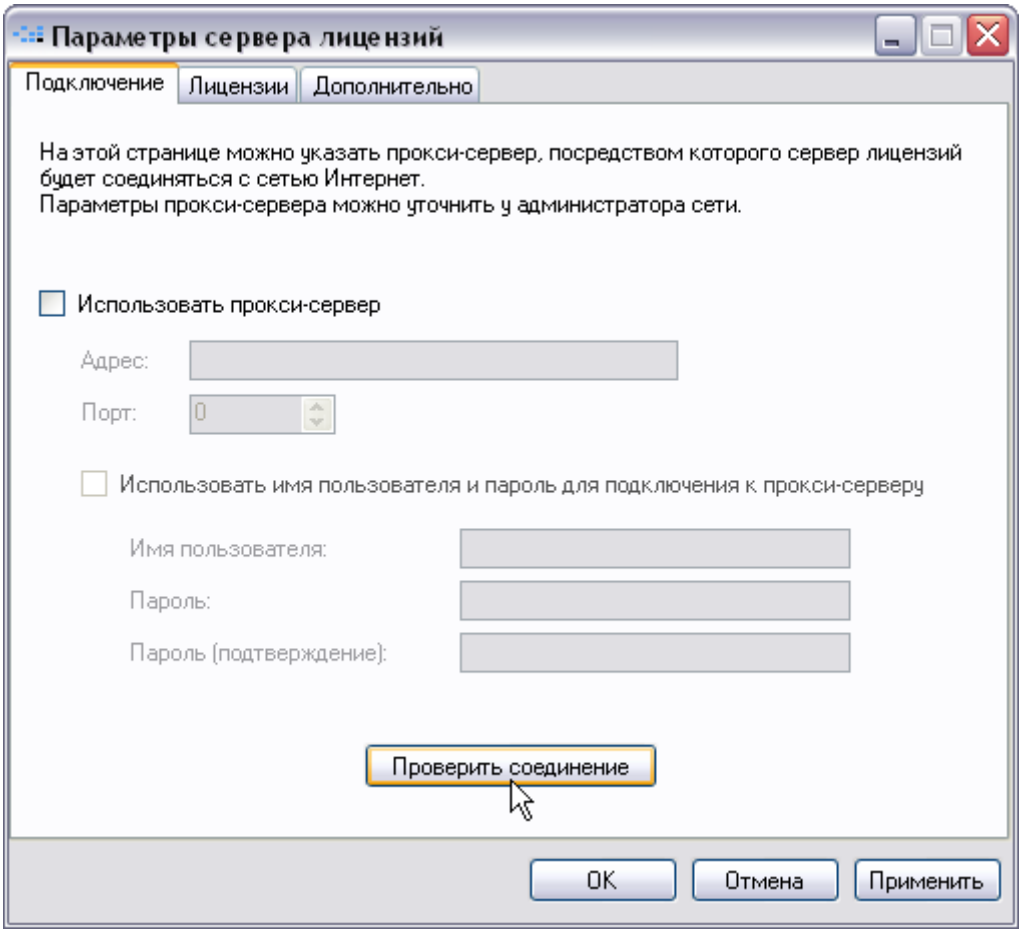

Рис. 1.7.1

На закладке «Подключение» (см. Рис. 1.7.1) можно задать настройки прокси-сервера, посредством которого сервер лицензий будет соединяться с сетью Интернет.

Опция «Использовать прокси-сервер» включает использование указанного прокси-сервера для подключения к сети Интернет. В полях «Адрес» и «Порт» необходимо задать параметры используемого прокси-сервера. Параметры прокси-сервера можно уточнить у администратора сети.

На этой же странице можно задать имя пользователя и пароль для доступа к прокси-серверу.

Внимание: Имя пользователя и пароль следует задавать только в случае необходимости и только по согласованию с администратором сети.

По кнопке «Проверить соединение» происходит проверка соединения с сервером контроля лицензий через Интернет. Для проверки соединения необходимо наличие хотя бы одной временной лицензии.

Варианты настройки подключения

Внимание: При выборе варианта настройки подключения к сети Интернет настоятельно рекомендуется проконсультироваться со своим сетевым администратором.

По умолчанию подключение осуществляется при следующих условиях: служба сервера лицензий BS_PingHost (Ping.Service.exe) работает под пользователем System, подключение к сети Интернет осуществляется без использования прокси-сервера.

Этот вариант обычно работает при непосредственном подключении компьютера к сети Интернет.

В сетях организаций может понадобиться встроенная (интегрированная) аутентификация при доступе в Интернет (как с использованием прокси-сервера, так и без). В этом случае следует настроить службу сервера лицензий так, чтобы она работала под соответствующей учетной записью. Для этого необходимо сделать следующее:

1. Зайти в Пуск –> Панель управления –> Администрирование –> Службы. При этом откроется окно «Службы» (см. Рис. 1.7.2)

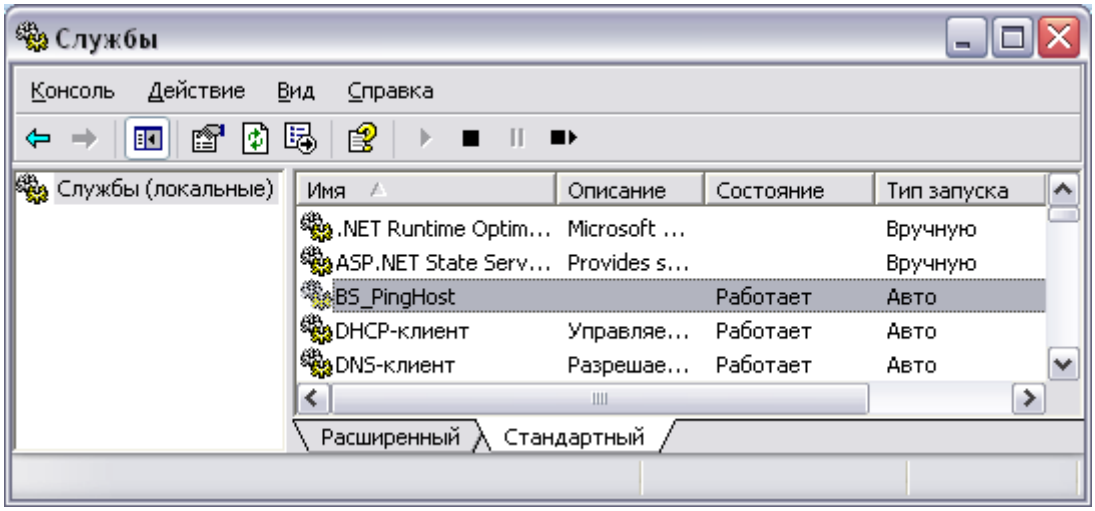

Рис. 1.7.2

2. Выбрать пункт «Свойства» контекстного меню службы «BS_PingHost».
3. Открыть закладку «Вход в систему» и выбрать тип входа в систему «С учетной записью» (см. Рис. 1.7.3).
4. Нажать кнопку «Обзор» и в окне «Выбор: Пользователь» ввести имя учетной записи, выданной администратором, после этого нажать «ОК».
5. Задать параметры выбранной учетной записи (Пароль, Подтверждение) и нажать кнопку «ОК».

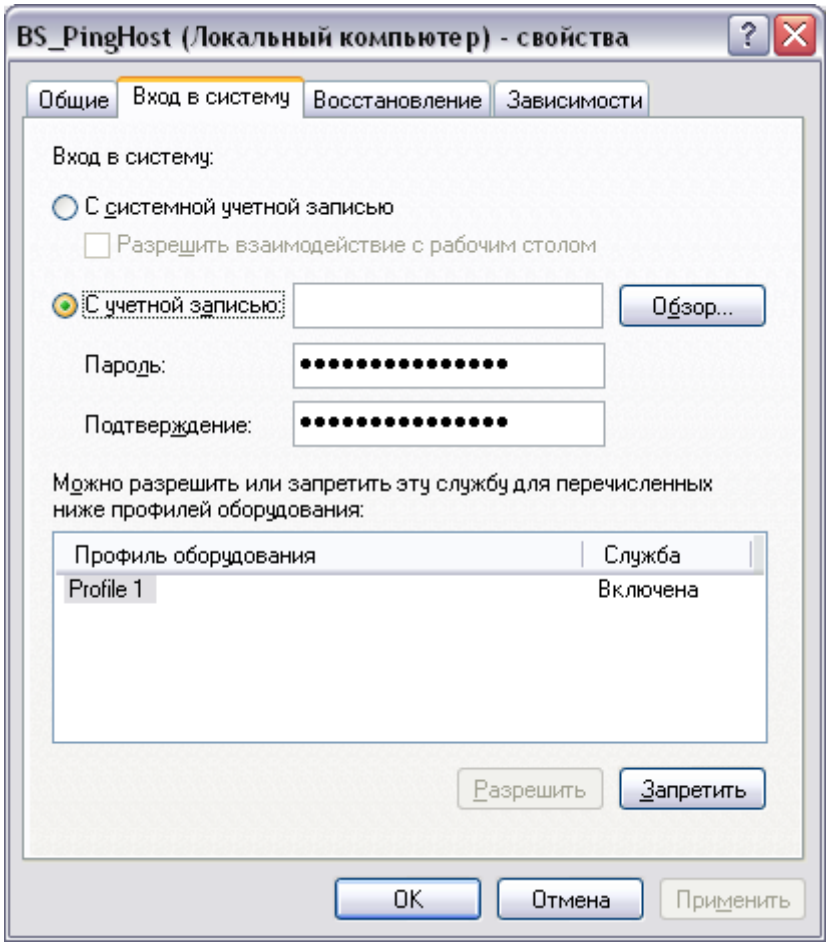

Рис. 1.7.3

Внимание: Не рекомендуется использовать для этой цели учетную запись пользователя компьютера. Обратитесь к администратору сети с просьбой выделить специальную учетную запись для этой цели.

Если используемый прокси-сервер не поддерживает встроенную аутентификацию, можно применить обычную (basic). Для этого в настройках подключения следует указать имя пользователя и пароль.

Внимание: При обычной аутентификации пароль передается по сети в незащищенном виде, поэтому ее применение не рекомендуется с точки зрения безопасности. Если это неизбежно, то следует использовать специально выделенную для этой цели учетную запись. Собственную учетную запись следует использовать только в крайнем случае и только по согласованию с администратором сети.

Сведения об имеющихся лицензиях

На закладке «Лицензии» находится информация о лицензиях Business Studio, установленных на данном компьютере (см. Рис. 1.7.4).

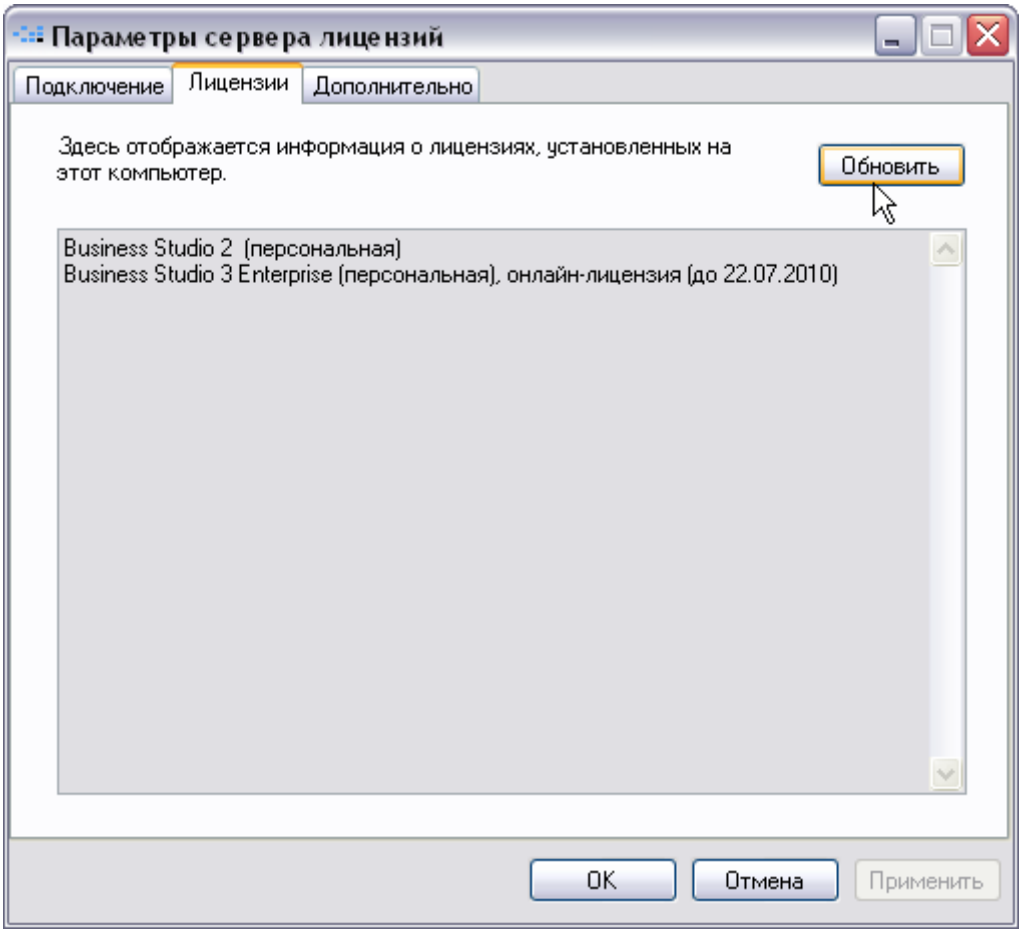

Рис. 1.7.4

По кнопке **«Обновить»** происходит обновление информации о действующих на данном компьютере лицензиях с предварительным обновлением всех имеющихся онлайн-лицензий через Интернет.

Дополнительные настройки

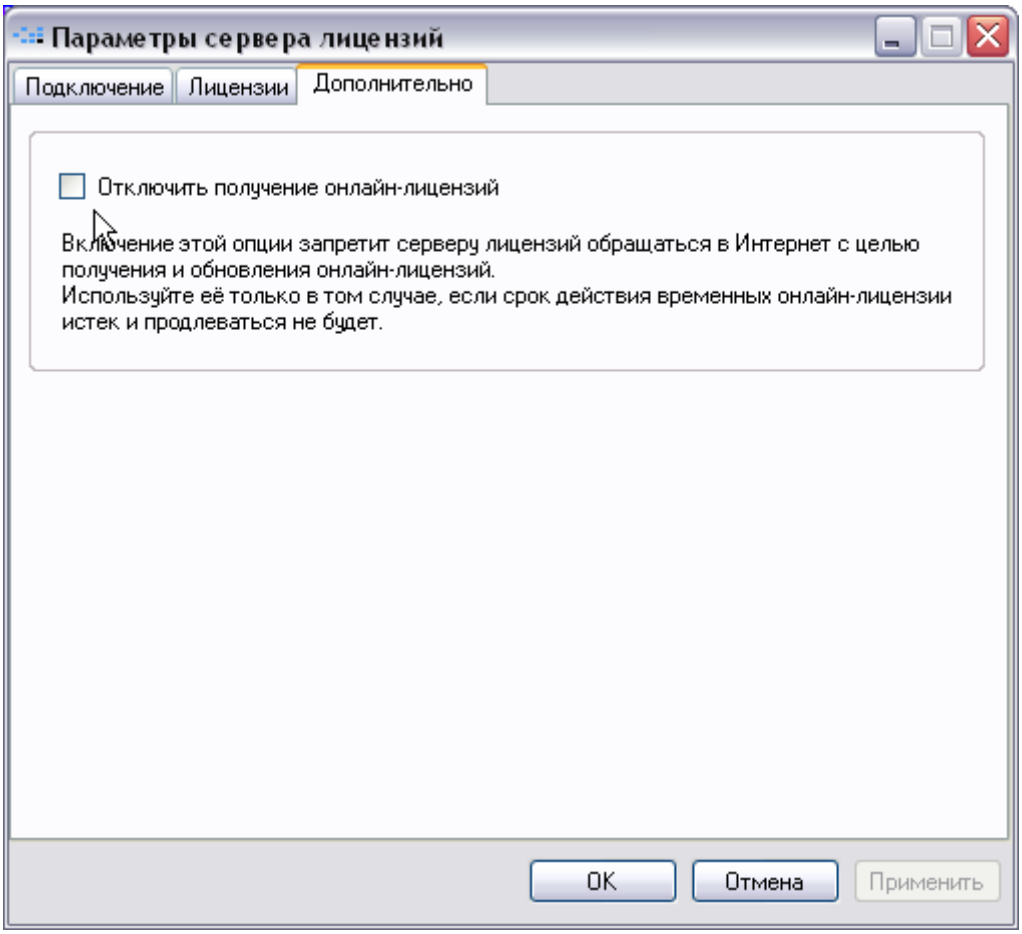

Рис. 1.7.5

При наличии онлайн-лицензий будет происходить периодическое обращение в Интернет для проверки их срока действия. Интервал обращения – раз в сутки (начиная от момента последней удачной проверки) при условии, что сервер лицензий не перезапускался. В связи с этим существует возможность работы с программой без постоянного соединения с сетью Интернет.

Проверку как персональных, так и серверных онлайн-лицензий осуществляет тот сервер лицензий, который указан в текущей базе данных. Таким образом, в ряде случаев (в частности, в клиент-серверном варианте установки) достаточно настроить для доступа в Интернет лишь один сервер лицензий в сети.

В случае, когда срок действия онлайн-лицензии истек и продлеваться не будет, можно включить опцию «Отключить получение онлайн-лицензий», чтобы запретить серверу лицензий обращаться в Интернет (см. Рис. 1.7.5).

ГЛАВА 2. РЕДАКТОР КЛАССОВ И ПАРАМЕТРОВ

Данный продукт предназначен для создания пользовательских параметров и справочников в структуре данных Business Studio. С его помощью изменения вносятся в структуру новых или существующих баз данных. Редактор классов и параметров запускается файлом MetaEdit.exe из папки установки программы.

Внимание: с помощью Редактора классов и параметров нельзя вносить изменения в свойства системных классов и параметров.

2.1 Термины и понятия

Метаданные – описательная информация о структуре и смысле данных.

Модуль – это отдельная, относительно независимая группа классов, в которую можно не только добавлять новые элементы, но также дополнять существующие.

Класс – множество всевозможных конечных объектов одного типа. Этот тип, в свою очередь, обычно является типом объектного параметра для объекта другого класса. Используются в тех случаях, когда необходимо исключить неоднозначный ввод информации.

Элементы списков – эти классы предназначены для реализации связи «один-ко-многим». В классах-потомках класса «Элементы списков» описываются модели объектов, выступающих в системе в роли «многих».

Перечисление – служит для определения перечислений в системе, не может пополняться в процессе работы с ним конечного пользователя.

Параметр класса – свойство объекта класса. Набор параметров характеризует объект класса.

Наследование – это свойство класса порождать своих потомков (наследников). Класс-наследник автоматически наследует от родителя все параметры, в нем также можно задавать новые параметры.

2.2 Загрузка и выгрузка метаданных

2.2.1 Загрузка метаданных из базы

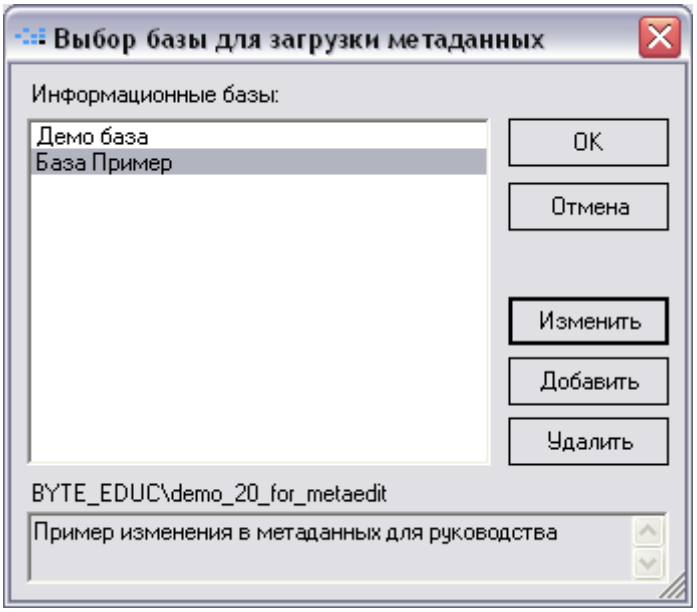

Рис. 2.2.1

Окно «Выбор базы для загрузки метаданных» (Рис. 2.2.1) открывается при запуске программы, либо из меню «Файл –> Загрузить из базы данных». В окне выбора базы можно создавать и удалять базы данных, подробнее см. Руководство пользователя, п. 16.1. «Резервное копирование и восстановление информационной базы данных».

Добавление, изменение и удаление баз данных происходит по кнопкам «Добавить», «Изменить» и «Удалить» соответственно. Подробнее см. Руководство пользователя, п. 1.8. «Управление базами данных».

Щелчок по кнопке «ОК» осуществит загрузку метаданных из выбранной базы для создания и редактирования пользовательских классов и параметров.

Щелчок по кнопке «Отмена» закроет окно выбора базы данных без загрузки. Потом можно будет загрузить метаданные из меню «Файл».

2.2.2 Загрузка метаданных из папки

Метаданные можно загружать также из файлов специального формата. Файлы данных представлены в формате *.mdm. В папке загрузки содержатся файлы по числу модулей в структуре, то есть каждый файл соответствует модулю в метаданных. Рекомендуется пользоваться процедурами загрузки и сохранения в папку на этапе разработки и редактирования структуры данных.

Оригинальные метаданные находятся в каталоге установки программы.

Осуществляется выбором пункта меню «Файл –> Загрузить из папки». Если необходимо при старте программы загрузить метаданные из папки, необходимо по кнопке «Отмена» закрыть окно выбора баз, открывающееся при запуске, а затем загрузить метаданные из папки с помощью пункта главного меню.

2.2.3 Применение к базе данных

Окно «Выбор баз для применения метаданных» (Рис. 2.2.2) открывается из меню «Файл –> Применить к базе данных».

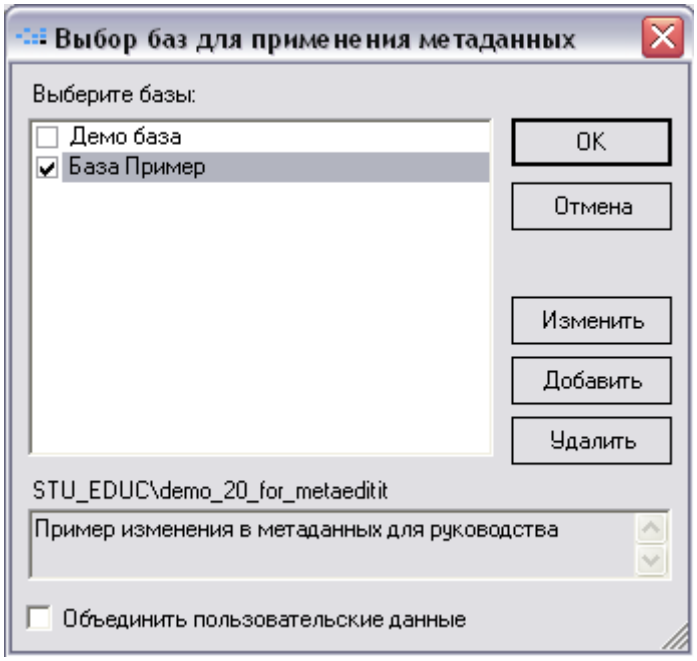

Рис. 2.2.2

Применение метаданных производится для баз данных, отмеченных галкой. Рекомендуется создавать резервную копию рабочей базы данных перед редактированием структуры, и производить тестирование произведенных изменений на копии базы данных.

Опция «Объединить пользовательские данные». Если опция отключена, то будут произведены изменения в полном соответствии с текущими метаданными. То есть структура базы данных станет полностью идентична структуре данных в редакторе. Если опция включена, то пользовательские данные, присутствующие в структуре базы данных, но отсутствующие в текущих метаданных, затронуты не будут. Эта опция будет полезна, например, при массовом применении к нескольким базам данных, когда базы уже имеют собственные (различные) классы или параметры, которые изменять не требуется.

Внимание: При отключенной опции «Объединить пользовательские данные» все пользовательские классы и параметры, не включенные в текущие метаданные, будут удалены из базы данных.

Внимание: В случае необходимости удалить класс или параметр в той же базе данных, из которой были загружены метаданные, опция «Объединить пользовательские данные» должна быть выключена! В противном случае операция объединения приведёт к тому, что этот класс или параметр фактически удален не будет.

При нажатии «ОК» будет открыто окно «Список необходимых изменений» (Рис. 2.2.1). Если изменение структуры информационной базы не требуется, то сразу будет произведено применение метаданных к выбранной базе данных.

На закладке «Классы» (Рис. 2.2.3) список изменений в классах, сгруппированный по категориям СозданиеКласса, УдалениеКласса.

Внимание: Все новые классы создаются в базе данных без прав доступа к ним. Для работы с новыми классами необходимо дать права доступа пользователям. Подробнее о раздаче прав см. Руководство пользователя, п. 16.9 «Права пользователя».

На закладке «Параметры» список изменений параметров существующих классов, сгруппированный по классам.

На закладке «Значения» список изменений значений параметров существующих классов, сгруппированный по классам.

На закладке «Ключи» список изменений ключей существующих классов, сгруппированный по классам.

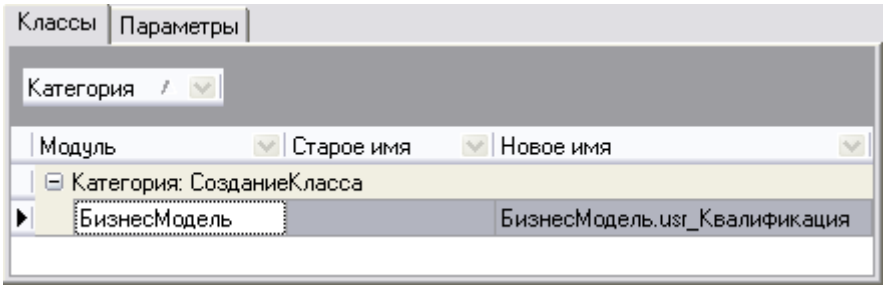

Рис. 2.2.3

При нажатии кнопки «ОК» будет произведено применение метаданных к отмеченным базам данных. По окончании применения будет выдано сообщение об успешном применении метаданных либо причина, по которой применение не произведено, в окне сообщений появится соответствующая строка.

Внимание: Применение метаданных возможно только к закрытым базам данных, которые не используются в данный момент другими пользователями.

Внимание: Перед применением метаданных рекомендуется создавать резервную копию базы данных (подробнее о создании резервной копии см. Руководство пользователя, п. 15.1 «Резервное копирование и восстановление информационной базы данных»).

2.2.4 Сохранение данных в папку

Назначение сохранения данных в папку приведено в п. 2.2.2 Загрузка метаданных из папки. Осуществляется выбором пункта меню «Файл –> Сохранить в папку». Откроется окно выбора папки для сохранения метаданных. Если указана не пустая папка будет выдано предупреждение с вопросом о перезаписи существующих файлов.

2.3 Редактирование метаданных

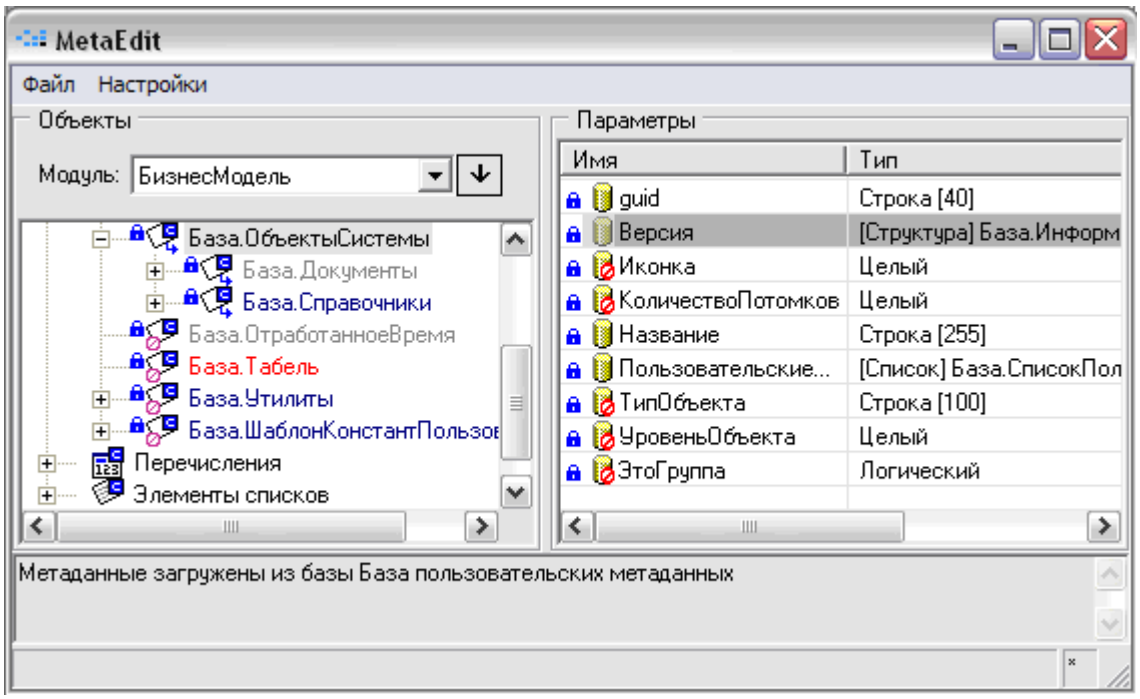

Рис. 2.3.1

В информационном окне отражаются сведения о загрузке и применении метаданных.

2.3.1 Список модулей

Вверху окна расположен раскрывающий список модулей и кнопка, вызывающая меню (Рис. 2.3.2). Раскрывающийся список содержит список модулей, включенных в метаданные.

Рис. 2.3.2

По кнопке  можно настроить отображающиеся в дереве классы: отображать классы только текущего модуля, показывать удаленные.

Отображать классы тек. модуля – Если опция меню отмечена, то при выборе модуля из раскрывающегося списка модулей, в дереве классов будут отображены только классы, принадлежащие выбранному модулю. Если опция не выбрана, то в дереве классов будут показаны все классы, независимо от их модульной принадлежности, но классы, не относящиеся к текущему модулю, будут неактивны, то есть будут выделены в дереве серым цветом. Если у класса, не принадлежащего текущему модулю, есть классы-наследники, принадлежащие текущему модулю, то класс-родитель выделяется в дереве классов синим цветом.

Показывать удаленные – При выборе этой опции меню в дереве классов будут отображены все классы, включая удаленные ранее классы. Удаленные классы выделяются в дереве красным цветом.

2.3.2 Дерево классов

Классы, принадлежащие текущему модулю (модулю, выбранному из раскрывающегося списка), отображаются в дереве стандартным черным цветом. Классы, не принадлежащие текущему модулю, отображаются серым цветом. Однако, если у класса, не принадлежащего текущему модулю, хотя бы один потомок принадлежит текущему модулю, то класс-родитель отображается в дереве классов синим цветом. Такая подсветка классов позволяет легко ориентироваться в большой и многоуровневой иерархии классов. Для удобства чтения информации о классе, у каждого класса есть пиктограммы. Их значения описаны в Таблица 2.3.1 *Пиктограммы классов*. Все действия с классами производятся из контекстного меню дерева классов, описание пунктов контекстного меню приведено в Таблица 2.3.2 *Контекстное меню дерева классов*.

Таблица 2.3.1 *Пиктограммы классов*

Пиктограмма	Значение
	Класс – потомок класса «Классы».
	Перечисление – потомок типа «Перечисления».
	Класс – потомок класса «Элементы списков».
	Наличие этого элемента в пиктограмме обозначает, что класс допускает хранимые ссылки.
	Наличие этого элемента в пиктограмме обозначает, что класс нехранимый.
	Наличие этого элемента в пиктограмме обозначает, что класс доступен только для просмотра.
	Наличие этого элемента в пиктограмме обозначает, что класс пользовательский – есть возможность редактирования.

Таблица 2.3.2 *Контекстное меню дерева классов*

Пункт меню	Описание
Добавить	Добавляет класс на том же уровне, где находится текущий класс. Другими словами, новый класс будет иметь тот же класс-родитель, что и текущий класс, и принадлежать текущему модулю, который выбран в списке модулей.
Добавить от текущего	Добавляет класс уровнем ниже текущего класса. То есть текущий класс будет являться классом-родителем для нового класса, и новый класс будет принадлежать текущему модулю, который выбран в списке модулей.
Редактировать	Вызывает форму «Свойства класса» для редактирования названия класса, набора системных и дополнительных опций. Редактировать можно только пользовательские классы.
Просмотреть	Вызывает форму «Свойства класса» для просмотра.
Удалить	Удаляет текущий класс.
Найти	Выводит форму поиска для поиска класса в дереве по имени класса. При успешном поиске найденный класс становится текущим.
Найти далее	Продолжает поиск класса в дереве, по заданному ранее имени.

2.3.3 Свойства класса

Форма «Свойства класса» (Рис. 2.3.3) вызывается при добавлении нового класса, редактировании или просмотре существующего. Создание и редактирование возможно только

для пользовательских классов, к их названию добавляется префикс *usr_*. Вызов класса на редактирование осуществляется выбором пункта контекстного меню.

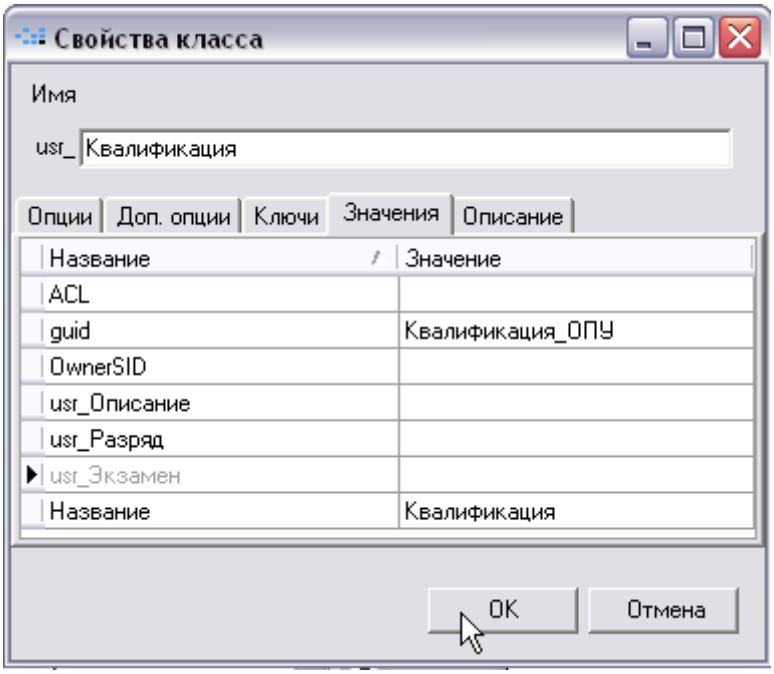

Рис. 2.3.3

Внимание: Все пользовательские справочники рекомендуется создавать в классе *База.Справочники*.

Закладка «Опции» недоступна для изменений. Пользовательские классы всегда хранимые, допускают хранимые ссылки; пользовательские элементы списков всегда хранимые, не допускают хранимых ссылок.

На закладке «Доп. опции» определяется набор дополнительных опций класса. Из контекстного меню можно вносить доп. опции по категориям. Удаление опции производится из контекстного меню выбором пункта «Удалить строку». Описание дополнительных опций класса приведено в Таблица 2.3.3 *Дополнительные опции класса*.

На закладке «Ключи» задаются ключи для класса – как простые (уникальность проверяется по одному параметру), так и составные (уникальность проверяется по сочетанию нескольких параметров). Удаление ключа производится из контекстного меню по пункту «Удалить строку».

На закладке «Значения» устанавливаются значения параметров по умолчанию во вновь создаваемом объекте класса.

Внимание: Для класса обязательно задание значений по умолчанию для параметров *guid* и *Название*.

На закладке «Описание» дается многострочное описание функционального назначения класса и его места в объектной иерархии.

Таблица 2.3.3 *Дополнительные опции класса*

Название	Тип	Назначение
Категория «ПоказКласса»		
Заголовок	Строка	Опция класса, содержит ***Заголовок*** класса, который будет показываться в стандартных формах в заголовке. Если значение не задано, берется ***ПолноеНаименование*** класса.

Название	Тип	Назначение
ТолькоЧтение	Логика	Объект класса нельзя модифицировать, независимо от настроек прав.
Видимый	Логика	Для показа по умолчанию в формах выбора справочника (**Нет** – класс будет показан только после включения опции «Показывать все»).
Иконка	Целое	Номер иконки класса для показа в формах выбора справочников.
ГлавныйКлюч		Имя главного ключа, по нему осуществляется автоматический импорт/экспорт и автогенерация объектов импорта.
Иерархический	Логика	Используется для импорта, для стандартных форм. Если значение опции **Да** – разрешается создавать группы в стандартных формах. При импорте объекты создаются группами, если этот момент не указан особо. **Нет** – Не разрешается создавать группы в стандартных формах. При импорте объекты создаются не группами, если этот момент не указан особо.
ЗависимыеКлассы	Строка	Значение – перечень зависимых классов, которые хотелось бы открыть из данного класса. Обычно в зависимых классах есть объектный параметр исходного класса.
ПолныйДоступ	Логика	**Да** – полный Доступ/Модификация объекта класса есть всегда, независимо от настроек прав.
Авторазмер	Логика	Если значение опции **Нет** – то авторазмер сетки снимается, и размер каждой колонки подбирается автоматически (появляется горизонтальная прокрутка). По умолчанию – **Да** (если ничего не задано).
Синхронизировать	Логика	Определяет, будет ли на форме включена кнопка синхронизации. В формах списков объектов и форме редактирования объекта синхронизация включена по умолчанию.

2.3.4 Параметры класса

Параметры выделенного класса отображаются в списке «Параметры». При создании параметров необходимо учитывать принцип наследования, описанный в п. 2.1 Термины и понятия. Не нужно создавать в классе параметры, дублирующие параметры класса-родителя.

Параметры также имеют пиктограммы, значения которых приведены в таблице ниже.

Таблица 2.3.4 *Пиктограммы параметров класса*

Пиктограмма	Значение
	Параметр класса.
	Наличие этого элемента в пиктограмме означает, что этот параметр не хранимый, то есть рассчитываемый, пользователь не сможет изменять его в программе.
	Наличие этого элемента в пиктограмме означает, что этот параметр только для чтения, то есть системный, редактирование такого параметра в структуре данных невозможно.

Все действия с параметрами классов производятся из контекстного меню списка параметров.

Таблица 2.3.5 *Контекстное меню списка параметров классов*

Пункт меню	Описание
Добавить	Добавляет параметр в текущем классе.
Удалить	Удаляет текущий параметр.
Редактировать параметр	Вызывает форму «Настройки параметра класса» для редактирования названия параметра, типа, набора дополнительных опций и описания параметра.
Просмотреть параметр	Вызывает форму «Настройки параметра класса» только для просмотра настроек текущего параметра.
Перейти по ссылке	Пункт меню доступен для не простых параметров (объектный, список, структура, перечисление). При выборе пункта осуществляется переход к классу, указанному в типе параметра. Таким образом, производится быстрая и наглядная навигация по дереву классов.

2.3.5 Настройки параметра класса

Форма «Настройки параметров класса» (Рис. 2.3.4) вызывается при добавлении нового параметра или редактировании существующего. Создание и редактирование возможно только для пользовательских параметров, к их названию добавляется префикс *usr_*. Вызов параметра на редактирование осуществляется либо выбором соответствующей опции всплывающего меню, либо двойным щелчком на параметре левой кнопкой мыши.

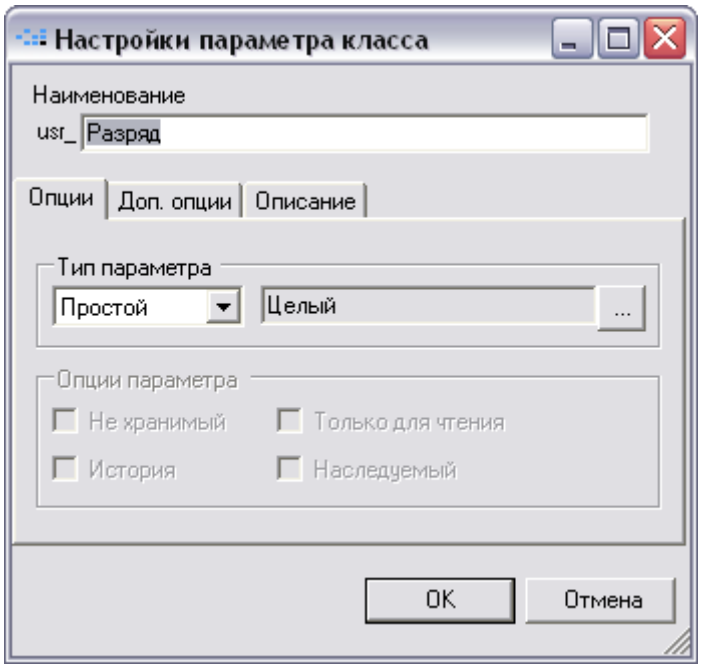

Рис. 2.3.4

На закладке «Опции» необходимо выбрать тип параметра: *простой, объектный, список, структура, перечисление*.

Если тип параметра *Простой*, то необходимо уточнить его тип: *логический, целый, строка, вещественный, датавремя, изображение, бинарный или текст.* Для параметра типа *Строка* необходимо указать длину, максимум 4000 символов. Для параметра типа *Веще-*

ственный необходимо указать длину и точность (количество знаков после запятой), максимум 28 символов. Если тип параметра *Объектный*, то необходимо указать класс, который будет являться типом объектов, на которые будет ссылаться этот параметр.

Если тип параметра *Список* или *Структура*, то необходимо указать элементы списка, которые будут являться типом объектов, на которые будет ссылаться этот параметр.

Если тип параметра *Перечисление*, то в качестве типа объектов указывается перечисление. Тип параметра выбирается по кнопке ....

Внимание: Общая длина (сумма длин всех параметров) класса, заданных на одном уровне в иерархии классов, не должна превышать 8000 Байт. Таблица длин параметров приведена в п. 4.4 Длина в байтах для различных типов параметров.

Опции параметра недоступны для изменений. Пользовательские параметры всегда хранимые, редактируемые, без истории, не наследуемые.

На закладке «Доп. опции» определяется набор дополнительных опций параметра класса. Они позволяют указывать заголовки, управлять порядком и видимостью по умолчанию, настраивать действия. Из контекстного меню можно вносить доп. опции по категориям. Удаление опции производится из контекстного меню выбором пункта «Удалить строку». Описание дополнительных опций параметра приведено в Таблица 2.3.6 *Дополнительные опции параметров классов*.

На закладке «Описание» можно дать многострочное описание функционального назначения параметра класса и его места в объектной иерархии.

Таблица 2.3.6 *Дополнительные опции параметров классов*

Название	Тип	Назначение
Категория «Показ»		
Заголовок	Строка	Содержит ***Заголовок*** параметра, который будет показываться в стандартных формах. Если значение не задано, берется ***Наименование*** параметра.
Видимый	Логика	Если значение опции **Да** – то параметр показывается в форме списка. Если опция не заполнена – **Да**.
ВидимыйОб	Логика	Если значение опции **Да** – то параметр показывается в форме единичного показа. Если опция не заполнена, то используется значение опции «Видимый».
НаВкладке	Логика	Указывается для параметров-списков. Если значение опции **Да**, то список отображается на вкладке в форме единичного показа. Если опция не заполнена – **Нет**.
Редактирование	Логика	Если значение опции **Да** – то параметр редактируется в форме списка. Если опция не заполнена – **Нет**.
РедактированиеОб	Логика	Если значение опции **Да** – то параметр редактируется в форме единичного показа. Если опция не заполнена – **Нет**.
Обязательный	Логика	**Да/Нет.** Служит для выделения параметров, обязательных для ввода.

Название	Тип	Назначение
Номер	Целое	Показывает порядковый номер расположения параметра на форме списка объектов. **Внимание!** В системе есть возможность добавлять/убирать колонки, используя специальную форму. Если вы хотите, чтобы параметр не показывался по умолчанию, но пользователь в дальнейшем может захотеть его показать, нужно проставить Показ.Номер = -1. Тем самым параметр появится в форме кустомизации колонок и можно его перетащить на основную форму. Также если используется опция Показ.Превью то нужно чтобы поле присутствовало в сетке, т.е. нужно ставить Показ.Номер = -1. Следует отметить, что в отличие от формы настройки колонок скрытые колонки загружаются, что снижает быстродействие.
НомерОб	Целое	Задает порядковый номер расположения параметра на форме единичного показа объекта. Для параметров-списков – порядковый номер вкладки. **Внимание!** Если опция не заполнена, то порядковый номер заполнится от Показ.Номер.
МинРазмер	Целое	Минимальный размер колонки, по умолчанию 0. Необходимо, например, при авторазмере, когда нужно, чтобы название обязательно было полностью видимым.
Размер	Целое	Задает размер колонки в сетке в форме списка объектов.
Сортировка	Строка	Возможные значения: **Возрастание, Убывание**. Используется для сортировки по параметру в форме списка.
Формат	Строка	Значение – формат вывода значения параметра в формате C#. Выводит значение параметра в заданном формате в форме списка.
ФорматОб	Строка	Значение – формат вывода значения параметра в формате C#. Выводит значение параметра в заданном формате в форме редактирования объекта.
Превью	Логика	Параметр, у которого стоит значение **Да**, появится в виде поля превью под строкой (если заполнен), например, комментарий. **Внимание!** Чтобы этот параметр показывался, нужно поставить Показ.Номер = -1. Также, если хотите, чтобы это поле показывалось в форме показа объекта - проставьте Показ.НомерОб = 100 (например). Следовательно, данную опцию нужно заполнять, когда порядок в единичной форме показа отличается от формы списка, например Комментарий, который не показывается в форме списка (Показ.Номер=-1), но показывается в форме единичного показа Показ.НомерОб = 100.

Название	Тип	Назначение
Категория «Редактирование»		
Эдитор	Строка	Задает эдитор на экранных формах списков, отличный от стандартного (по умолчанию) для формы списка объектов. Например, для параметра типа Текст эдитор может быть MemoEdit (с просмотром содержимого) или RichEdit (текст в формате RTF), по умолчанию привязан эдитор MemoEditEx – без просмотра содержимого в сетке. **Примечание:** Эдитор MemoEdit может быть привязан и к параметру типа строка, если нужно автоматически увеличивать высоту ячейки, если строка не помещается. Пользоваться этой возможностью нужно осторожно – нельзя позволять редактировать строку этим эдитором (только просмотр) – так как пользователь может вставить перевод строки. Возможные значения: – MemoEdit, RichEdit (вместо MemoEditEx по умолчанию); – PictureEdit (вместо ImageEdit по умолчанию); – ProgressBar (вместо SpinEdit); – ButtonEdit (для привязки эдитора для выбора, при нажатии на кнопку которого должен выполниться метод, описанный в опции **ДействиеПоКнопке**).
ЭдиторОб	Строка	Задает эдитор на экранных формах объекта, отличный от стандартного (по умолчанию) для формы редактирования объекта.
Действие-ПоКнопке	Строка	В качестве значения используется имя метода, который будет запускаться на исполнение при нажатии кнопки **«...»** эдитора (переопределенного в опции **Эдитор**, либо **ЭдиторОб** в значение ButtonEdit).
Действие-ПоКнопкеУдалить	Строка	В качестве значения используется имя метода, который будет запускаться на исполнение при нажатии кнопки с крестом эдитора (переопределенного в опции **Эдитор**, либо **ЭдиторОб** в значение ButtonEdit).
Категория «Связи»		
Синхронизировать	Логика	Если значение опции **Да**, то по этому объектному полю будет синхронизироваться форма редактирования объекта данного класса. Если значение **Нет** – синхронизация не используется.

2.3.6 Свойства перечисления

Форма «Свойства перечисления» (Рис. 2.3.5) вызывается при добавлении нового перечисления или редактировании существующего. Создание и редактирование возможно только для пользовательских перечислений, к их названию добавляется префикс *usr_*. Вызов перечисления на редактирование осуществляется выбором соответствующей опции контекстного меню.

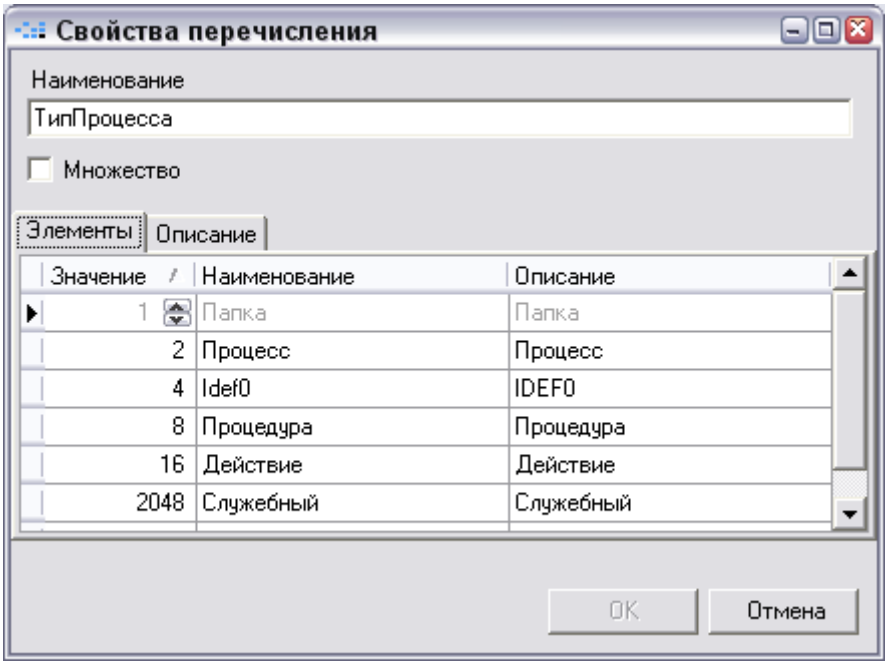

Рис. 2.3.5

На закладке «Элементы» задается список элементов перечисления. Параметры перечисления: *значение, наименование, описание.* Значение – подставляемое значение, название – системное название элемента, описание – подставляемое название элемента. Удаление элемента перечисления осуществляется из контекстного меню.

Параметр *Множество* позволяет задавать параметру объекта несколько значений из перечисления. При изменении параметра *Множество* во включенное состояние для уже созданных объектов, информация в базе не теряется. При отключении параметра *Множество* информация в базе по параметрам объектов этого перечисления утратится.

Внимание: Для независимых элементов перечисления-множества, необходимо значениям элементов присваивать степени двойки: 1,2,4,8,…

На закладке «Описание» дается многострочное описание перечисления.

ГЛАВА 3. РАБОТА С BUSINESS STUDIO ЧЕРЕЗ OLE

В крупных компаниях, где существует большой объем разнообразной информации, зачастую создаваемый в различных приложениях, возникает задача автоматического обмена данными с другими приложениями. Причем желательно, чтобы импорт и экспорт данных происходил в реальном времени. Технология OLE предназначена для интеграции приложений.

Внимание: Модификация данных в базе может осуществляться только с помощью Business Studio, либо посредством OLE, либо интерактивно. Прямая модификация таблиц недопустима и, как правило, ведет к нарушению логической целостности данных и ошибкам в работе. Это связано с тем, что для хранения информации применяется технология ORM (объектно-реляционное отображение), использующая сложную и неочевидную структуру БД и требующая особого обращения с данными.

Использование OLE становится доступным после регистрации на компьютере необходимой библиотеки и соответствующего разрешения на работу с конкретной базой:

1) Для обеспечения возможности работы с приложением Business Studio через OLE необходимо зарегистрировать библиотеку «Система.Клиент.dll», которая находится в папке установки программы. Регистрация осуществляется путем запуска файла RegisterOleServer.bat, находящегося в той же папке. Для запуска файла необходимы права администратора.

   Внимание: Перед удалением Business Studio рекомендуется отменить регистрацию библиотеки Система.Клиент.dll, выполнив файл UnregisterOleServer.bat, находящийся в папке установки программы. Для запуска файла необходимы права администратора.

2) Возможность использования OLE в конкретной базе определяется параметром «Разрешено использование OLE». По умолчанию такая возможность отключена.

   Чтобы включить данную возможность нужно через пункт меню «Справочники» - «Все справочники», нажав кнопку «  » «Показывать всё», открыть справочник «Системные настройки пользователя» и установить для пользователя опцию «Разрешено использование OLE».

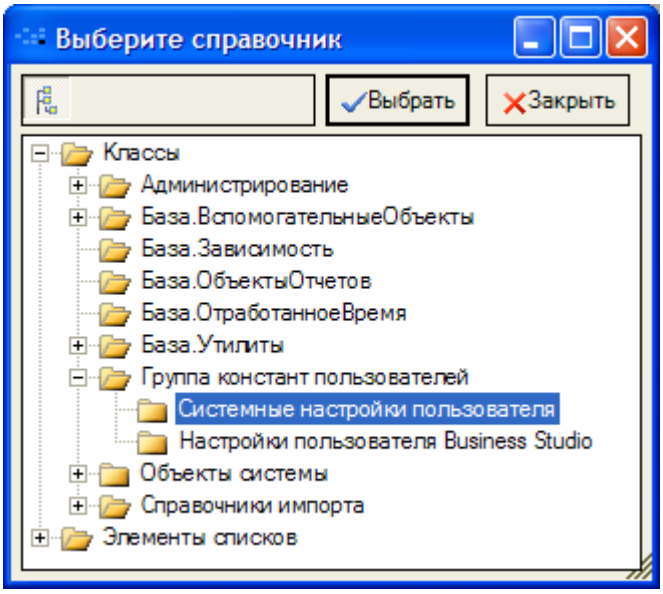

Рис. 3.1 Открытие справочника «Системные настройки пользователя»

Для предотвращения несанкционированного изменения опции «Разрешено использование OLE» и получения, тем самым, пользователем доступа к данным через OLE, рекомендуется давать доступ на изменение данной опции только администратору системы Business Studio.

Установление запрета на изменение определенных параметров, осуществляется назначением пользователю дополнительной категории прав. Подробно о работе с категориями прав описано в Руководстве пользователя, п. 16.9.2.

Все окна, которые вызываются средствами OLE, являются модальными.

Ниже приведенные примеры кода по использованию методов и свойств классов формируются с использованием:

- среды Visual Basic for Application;
- демонстрационной базы, которая устанавливается в процессе установки Business Studio. Демонстрационную базу данных можно загрузить самостоятельно: база находится в папке «Backup» в каталоге установки программы (по умолчанию, «C:\Program Files\STU-Soft\Business Studio 3.6\»). Загрузка базы данных описана в Руководстве пользователя (п.1.8.5).

3.1 Запуск приложения

3.1.1 Метод CreateObject

Название метода: CreateObject("ByteEnterprise.OleApplication")

Возвращаемый результат: Система.OleApplication (см. п.3.2)

Мотед получения объекта приложения.

Пример кода. См. примеры по методам ниже.

3.2 Класс «Система.OleApplication»

3.2.1 Метод ЗапуститьКлиентскоеПриложение

Синтаксис: ЗапуститьКлиентскоеПриложение(string "<Сервер>", string "<База>", string "<Версия>")

Возвращаемый результат: Система.КлиентскоеПриложение.Приложение (см. п.3.3).

Метод позволяет запустить клиентское приложение или получить уже запущенное. Используется для запуска элементов пользовательского интерфейса. В качестве параметров передаются три строки: имя сервера БД, название базы, версия продукта – Enterprise, Professional, Cockpit. Версия продукта должна соответствовать имеющейся лицензии.

Замечание: в качестве имени сервера и/или базы данных могут быть переданы пустые строки. В этом случае при запуске появится стандартное окно выбора базы данных. При каждом запуске без параметров "<Сервер>" или "<База>" будет создан новый экземпляр приложения, который необходимо будет завершить явно, используя метод класса Система.OleApplication ЗавершитьПриложение().

Пример кода.

Задача: запустить приложение Business Studio.

```
Sub ПримерOLE_ЗапускПриложения()

  'Получение объекта приложения
  Set oleapp = CreateObject("ByteEnterprise.OleApplication")

  'Запустить Business Studio Enterprise с базой под именем "demo_ole" на сервере 'U6S\SQLEXPRESS2005.
  'В Диспетчере задач появится Business Studio. В панели задач приложения не будет видно.

  Set client_app = oleapp.ЗапуститьКлиентскоеПриложение("U6S\SQLEXPRESS2005", "demo_ole", "Enterprise")

End Sub
```

3.2.2 Метод ПолучитьКорневуюГруппуКласса

Синтаксис: ПолучитьКорневуюГруппуКласса(string "<ИмяКласса>")

Возвращаемый результат: Система.МетаКласс (см. п.3.4)

Метод возвращает корневую группу класса по имени класса.

Пример кода. См. метод «ОткрытьФайл» (п.3.2.10).

3.2.3 Метод ПолучитьОбъекты

Синтаксис: ПолучитьОбъекты(string "<ИмяКласса>", string "<ИмяПараметра>", object "<ЗначениеПараметра>")

Возвращаемый результат: Система.Список (см. п.3.6).

Возвращает список объектов класса, указанный параметр которых равен заданному значению.

Пример кода. См. методы РедактироватьОбъект (п.3.2.5) и СоздатьОбъект (п.3.2.8).

3.2.4 Метод ВыбратьОбъект

Синтаксис: ВыбратьОбъект(object "<Объект>")

Возвращаемый результат: Система.МетаКласс (см. п.3.4).

Метод для выбора объекта с использованием окна выбора.

Пример кода. См. метод «ОткрытьФайл» (п.3.2.10).

3.2.5 Метод РедактироватьОбъект

Синтаксис: РедактироватьОбъект(object "<Объект>")

Возвращаемый результат: не возвращает.

Метод для редактирования объекта с использованием окна редактирования. Окно редактирования при этом является модальным.

Пример кода.

Задача: открыть на редактирование окно свойств заданного показателя.

```
Sub ПримерOLE_РедактированиеОбъекта()
  'БД и редакция Business Studio, с которыми будем работать
  СерверБД = "U6S\SQLEXPRESS2005"
  База = "demo_ole"
  Версия = "Enterprise"

  'Получение объекта приложения
  Set oleapp = CreateObject("ByteEnterprise.OleApplication")

  'Запустить Business Studio в редакции и базой на сервере, указанными ранее.
  'В Диспетчере задач появится Business Studio. В панели задач приложения не будет видно.
  Set client_app = oleapp.ЗапуститьКлиентскоеПриложение(СерверБД, База, Версия)
  'В панели задач появится приложение
  'Данное действие приводится для наглядности и не является обязательным
  oleapp.ПоказатьКлиентскоеПриложение

  'Получить список всех показателей с заданным названием
  НазваниеПоказателя = "Процент запасов, запланированных к выдаче"
  Set СписокПоказателей = oleapp.ПолучитьОбъекты("БизнесМодель.ПоказателиBSC", "Название",
НазваниеПоказателя)

  'Если в списке полученных показателей всего один элемент
  If (СписокПоказателей.КоличествоЭлементов = 1) Then
     'Тогда взять первый элемент списка
     Set НужныйПоказатель = СписокПоказателей.Item(0)
```

```
        'И открыть на редактирование окно свойств выбранного показателя
        oleapp.РедактироватьОбъект (НужныйПоказатель)
    Else
        'Иначе вывести сообщение, что таких показателей несколько
        MsgBox "Существует несколько показателей '" + НазваниеПоказателя + "'."
    End If
End Sub
```

3.2.6 Метод ВыполнитьКоманду

Синтаксис: ВыполнитьКоманду(string "<Команда>", object "<Аргументы>")

Возвращаемый результат: object (в зависимости от выполняемой команды)

Метод запускает команду на исполнение. Примеры команд см. Руководство пользователя п.16.8 «Командная строка». Если требуется выполнить команду, в которой нет аргументов, то на месте пустого аргумента следует вставлять значение с пустым значением - Null, Nothing или другие значения, предопределенные конкретным языком программирования.

Пример кода.

Задача: открыть окно объектной модели.

```
Sub ПримерOLE_ВыполнитьКоманду()
    'Получение объекта приложения
    Set oleapp = CreateObject("ByteEnterprise.OleApplication")
    'Запустить Business Studio в редакции и базой на сервере, указанными ранее.
    'В Диспетчере задач появится Business Studio. В панели задач приложения не будет видно.
    Set client_app = oleapp.ЗапуститьКлиентскоеПриложение("U6S\SQLEXPRESS2005", " demo_ole", "Enterprise")
    'В панели задач появится приложение
    oleapp.ПоказатьКлиентскоеПриложение
    'Открыть окно объектной модели (Справочники –> Объектная модель)
    object_model = oleapp.ВыполнитьКоманду("База.КлиентскиеМетоды.ВыполнитьФорму", "База.ФормаОписаниеОбъектнойМодели")
End Sub
```

3.2.7 Метод ВыбратьКласс

Синтаксис: ВыбратьКласс()

Возвращаемый результат: string

Метод для выбора класса с использованием окна выбора. Возвращает строкой системное название выбранного класса.

Пример кода.

Задача: открыть окно выбора справочников.

```
Sub ПримерOLE_ВыбратьКласс()
```

```
  'Получение объекта приложения
  Set oleapp = CreateObject("ByteEnterprise.OleApplication")
  'Запустить Business Studio в редакции и базой на сервере, указанными ранее.
  'В Диспетчере задач появится Business Studio. В панели задач приложения не будет видно.
  Set client_app = oleapp.ЗапуститьКлиентскоеПриложение("U6S\SQLEXPRESS2005", " demo_ole", "Enterprise")
  'В панели задач появится приложение
  oleapp.ПоказатьКлиентскоеПриложение
  'Открыть окно для выбора справочников (Справочники -> Все справочники)
  class_select = oleapp.ВыбратьКласс()
End Sub
```

3.2.8 Метод СоздатьОбъект

Синтаксис: СоздатьОбъект(object "<Группа>")

Возвращаемый результат: Система.МетаКласс (см. п.3.4).

Метод создает объект – потомок по иерархии в группе. Для добавления элементов в списках нужно использовать метод «Добавить» класса «Система.Список» (см.п.3.6.2).

Структуры объекта (например, «Параметры должности», «Параметры подразделения», «Параметры СМК») создаются только вместе с объектом. Обращаться к ним надо через объект.

Пример кода.

Задача: в папке «Объекты деятельности\Документы\Бумажный документ\Документы СМК» создать документ и заполнить некоторые его свойства, в том числе и параметры СМК.

```
Sub ПримерOLE_СоздатьОбъект()
  'Получение объекта приложения
  Set oleapp = CreateObject("ByteEnterprise.OleApplication")
  'Запустить Business Studio в редакции и базой на сервере, указанными ранее.
  'В Диспетчере задач появится Business Studio. В панели задач приложения не будет видно.
  Set client_app = oleapp.ЗапуститьКлиентскоеПриложение("U6S\SQLEXPRESS2005", " demo_ole", "Enterprise")
  'В панели задач появится приложение
  oleapp.ПоказатьКлиентскоеПриложение

  'Указываем папку (не корневую), в которой будет создаваться объект
  'В данном примере требуемая папка задается через guid, соответствующий папке "Документы СМК" в демо-базе
  Set СписокПапкаБумДока = oleapp.ПолучитьОбъекты("БизнесМодель.БумажныйДокумент", "guid", "fb366191-6c0b-4e06-8ccf-ebe219dd2bfd")
  Set ПапкаНужная = СписокПапкаБумДока.ПолучитьЭлемент(0)
```

```
'Создать новый бумажный документ в заданной папке
Set НовБумДокумент = oleapp.СоздатьОбъект(ПапкаНужная)

  'Заполняем название документа и его параметры
  НовБумДокумент.Название = "Документ через OLE"
  НовБумДокумент.КодДокумента = "OLE-1"
  НовБумДокумент.Комментарий = "Документ создан средствами OLE"

    'Заполняем объектный параметр "Тип документа"
    ТребуемыйТипДока = "Запись"
    Set ТипыДоковСТребуемымТипом = _
     oleapp.ПолучитьОбъекты("БизнесМодель.ТипыДокумента", "Название", ТребуемыйТипДока)
    НовБумДокумент.ТипДокумента = ТипыДоковСТребуемымТипом.ПолучитьЭлемент(0)

    'Заполняем параметры СМК, которые являются структурой
    Set ПараметрыСМК = НовБумДокумент.ПараметрыСМК
    ПараметрыСМК.ВерсияДокумента = "1.0"
    ПараметрыСМК.ОбластьРаспространенияДокумента = 0 ' Внутренний
    ПараметрыСМК.СтатусДокумента = 1 'Действующий
    ПараметрыСМК.НеобходимостьАктуализации = True 'Логический параметр: True, False

    'Задаем параметры типа дата в Параметрах СМК
    Set ДатаВВедения = ПараметрыСМК.НайтиПараметр("ДатаВведенияВДействие")
    ДатаВВедения.Значение = CDate(Now) ' Сегодня

    Set ПересмотрПлан = ПараметрыСМК.НайтиПараметр("ДатаПересмотраПлан")
    ПересмотрПлан.Значение = CDate("07.12.2012")

'Сохранить созданный бумажный документ, иначе его не будет видно
НовБумДокумент.Сохранить

End Sub
```

3.2.9 Метод СоздатьГруппу

Синтаксис: СоздатьГруппу(object "<Группа>")

Возвращаемый результат: Система.МетаКласс (см. п.3.4)

Метод создает группу – потомок по иерархии в группе. Если объект является группой – он может содержать другие объекты ниже по иерархии.

Пример кода.

Задача: в бумажных документах создать группу (папку). Примечание: в других классах понятия «Папка» и «Группа» могут не совпадать.

```
Sub ПримерOLE_СоздатьГруппу()
  'Получение объекта приложения
  Set oleapp = CreateObject("ByteEnterprise.OleApplication")
  'Запустить Business Studio в редакции и базой на сервере, указанными ранее.
  'В Диспетчере задач появится Business Studio. В панели задач приложения не будет видно.
  Set client_app = oleapp.ЗапуститьКлиентскоеПриложение("U6S\SQLEXPRESS2005", " demo_ole", "Enterprise")
  'В панели задач появится приложение
  oleapp.ПоказатьКлиентскоеПриложение

  'Дальше обращение будет с бумажными документами
  Set БумДокументы = _
  oleapp.ПолучитьКорневуюГруппуКласса("БизнесМодель.БумажныйДокумент")

  'Создать группу (папку) в бумажных документах
  Set НоваяГруппа = oleapp.СоздатьГруппу(БумДокументы)
    'Заполняем название папки
    НоваяГруппа.Название = "Папка через OLE"

  'Сохранить созданную папку, иначе его не будет видно
  НоваяГруппа.Сохранить

End Sub
```

3.2.10 Метод ОкрытьФайл

Синтаксис: ОкрытьФайл(object "<Объект>")

Возвращаемый результат: не возвращает.

Метод открытия файла закрепленного за объектом, если объект его содержит. В случае, если файла нет, то будет выдано соответствующее сообщение.

Пример кода.

Задача 1: открыть справочник бумажных документов и после выбора пользователем одного из документов открыть файл, закрепленный за ним.

```
Sub ПримерOLE_ОткрытьФайлСУчастиемПользователя()
  'Получение объекта приложения
  Set oleapp = CreateObject("ByteEnterprise.OleApplication")
  'Запустить Business Studio в редакции и базой на сервере, указанными ранее.
  'В Диспетчере задач появится Business Studio. В панели задач приложения не будет видно.
```

```
    Set client_app = oleapp.ЗапуститьКлиентскоеПриложение("U6S\SQLEXPRESS2005", " demo_ole", "Enterprise")
    'В панели задач появится приложение
    oleapp.ПоказатьКлиентскоеПриложение

    'Дальнейшее обращение будет с бумажными документами
    Set БумДокументы = _
    oleapp.ПолучитьКорневуюГруппуКласса("БизнесМодель.БумажныйДокумент")

    'Открываем окно выбора бумажного документа и ждем выбор пользователя
     Set Документ = oleapp.ВыбратьОбъект(БумДокументы)
      'BS открывает файл выбранного объекта
      'или выдает сообщение, что к объекту файл не прикреплен
      oleapp.ОткрытьФайл (Документ)

End Sub
```

Задача 2: открыть файл бумажного документа «Акт выполненных работ».

```
Sub ПримерOLE_ОткрытьФайлБезУчастияПользователя()
    'Получение объекта приложения
    Set oleapp = CreateObject("ByteEnterprise.OleApplication")
    'Запустить Business Studio в редакции и базой на сервере, указанными ранее.
    'В Диспетчере задач появится Business Studio. В панели задач приложения не будет видно.
    Set client_app = oleapp.ЗапуститьКлиентскоеПриложение("U6S\SQLEXPRESS2005", " demo_ole", "Enterprise")
    'В панели задач появится приложение
    oleapp.ПоказатьКлиентскоеПриложение

    'Получить список всех объектов, у которых guid = myguid
    'myguid соответствует документу «Акт выполненных работ»
    myguid = "6bc61f07-2181-4e26-94c0-8cc9d2b805a3"
    Set СписокБумДокументы = oleapp.ПолучитьОбъекты("БизнесМодель.БумажныйДокумент", "guid", myguid)

    'Если в списке полученных объектов всего один элемент
    If (СписокБумДокументы.КоличествоЭлементов = 1) Then
        'Тогда взять первый элемент списка бумажных документов
        Set НужныйБумДокумент = СписокБумДокументы.Item(0)

        'И открыть файл выбранного объекта
```

```
        'или выдает сообщение, что к объекту файл не прикреплен
        oleapp.ОткрытьФайл (НужныйБумДокумент)

      Else

        'Иначе выбрать вывести сообщение, что таких объектов несколько
        MsgBox "Существует несколько объектов с guid = '" + myguid + "'."
    End If

End Sub
```

3.2.11 Метод ПоказатьКлиентскоеПриложение

Синтаксис: ПоказатьКлиентскоеПриложение()

Возвращаемый результат: не возвращает.

Метод показывает текущий экземпляр Business Studio и отображает его в панели задач).

Пример кода. См. код во всех методах данного класса.

3.2.12 Метод ЗавершитьПриложение

Синтаксис: ЗавершитьПриложение()

Возвращаемый результат: не возвращает.

Метод выгружает экземпляр Business Studio из памяти.

Пример кода.

Задача: открыть и закрыть Business Studio.

```
Sub ПримерOLE_ЗавершитьКлиентскоеПриложение()
  'Получение объекта приложения
  Set oleapp = CreateObject("ByteEnterprise.OleApplication")
  'Запустить Business Studio в редакции и базой на сервере, указанными ранее.
  'В Диспетчере задач появится Business Studio. В панели задач приложения не будет видно.
  Set client_app = oleapp.ЗапуститьКлиентскоеПриложение("U6S\SQLEXPRESS2005", " demo_ole",
"Enterprise")
  'В панели задач появится приложение
  oleapp.ПоказатьКлиентскоеПриложение
  'Закрыть окно Business Studio
  app_exit = oleapp.ЗавершитьПриложение()
End Sub
```

3.3 Класс «Система.КлиентскоеПриложение.Приложение»

3.3.1 Свойство СервПриложение

Синтаксис: СервПриложение

Тип параметра: Система.Приложение

Предоставляет ссылку на серверное приложение. Свойство предоставляется для обеспечения совместимости с программными кодами, разработанными с использованием более ранних версий Business Studio.

3.4 Класс «Система.МетаКласс»

К классу «Система.МетаКласс» относится любой объект системы.

3.4.1 Свойство ИмяПараметра

Свойство: <ИмяПараметра>

Тип параметра: object (соответствует типу параметра в Объектной модели).

Обращение к значениям (параметрам) объектов осуществляется по имени параметра из объектной модели Business Studio.

Пример кода.

Задача: вывести сообщение о заданном показателе, а так же некоторые его параметры.

```
Sub ПримерOLE_РаботаСПараметрами()


  'Получение объекта приложения
  Set oleapp = CreateObject("ByteEnterprise.OleApplication")
  'Запустить Business Studio в редакции и базой на сервере, указанными ранее.
  'В Диспетчере задач появится Business Studio. В панели задач приложения не будет видно.
  Set client_app = oleapp.ЗапуститьКлиентскоеПриложение("U6S\SQLEXPRESS2005", " demo_ole", "Enterprise")
  'В панели задач появится приложение
  oleapp.ПоказатьКлиентскоеПриложение


  'Обращаемся к показателю с заданным названием
  ИмяПоказателя = "Издержки на запасы (% от общих издержек)"
  Set СписокПоказателей = oleapp.ПолучитьОбъекты("БизнесМодель.ПоказателиBSC", "Название", ИмяПоказателя)
  Set МойПоказатель = СписокПоказателей.ПолучитьЭлемент(0)


  'Считываем параметрам показателя
  ЦельДата = МойПоказатель.ЦелеваяДата
```

```
    ЦельЗначение = CStr(МойПоказатель.ЦелевоеЗначение) 'Вещественное значение сразу преобразовываем в строку
    Измерение = МойПоказатель.ЕдиницаИзмерения

      'Определение периодичности измерения показателя. Периодичность - это перечисление
      ПериодичностьТекст = _
      МойПоказатель.Параметры.ПолучитьЭлемент("Периодичность").ЗначениеВСтроку

    'Выводим сообщение с данными показателя
    MsgBox "Показатель: " + ИмяПоказателя + Chr(13) + _
    "Периодичность измерения: " + ПериодичностьТекст + Chr(13) + _
    "Единица измерения: " + Измерение + Chr(13) + _
    "Целевое значение и дата: " + ЦельЗначение + " на " + ЦельДата

End Sub
```

3.4.2 Метод НайтиПараметр

Синтаксис: НайтиПараметр(string "<ИмяПараметра>")

Возвращаемый результат: Система.Параметр (см. п.3.5)

Метод для получения параметра. Используется, если нет возможности обратиться к параметру напрямую, например, для обращения к параметрам типа "ДатаВремя".

Пример кода. См. код в методе «СоздатьОбъект» (п.3.2.8).

3.4.3 Метод СоздатьФильтр

Синтаксис: СоздатьФильтр()

Возвращаемый результат: Система.Фильтр (см. п.3.7).

Создает Фильтр по объекту-группе.

Внимание: Рекомендуется использовать для корневых групп класса (ОПУ).

Пример кода.

Задача: вывести сообщение, показывающее количество должностей в бизнес-модели.

```
Sub ПримерOLE_СоздатьФильтр()
    'Получение объекта приложения
    Set oleapp = CreateObject("ByteEnterprise.OleApplication")
    'Запустить Business Studio в редакции и базой на сервере, указанными ранее.
    'В Диспетчере задач появится Business Studio. В панели задач приложения не будет видно.
    Set client_app = oleapp.ЗапуститьКлиентскоеПриложение("U6S\SQLEXPRESS2005", "demo_ole", "Enterprise")
    'В панели задач появится приложение
    oleapp.ПоказатьКлиентскоеПриложение
```

```
'Дальнейшая работа будет с Субъектами
Set ВсеСубъекты = oleapp.ПолучитьКорневуюГруппуКласса("БизнесМодель.Субъекты")

'Создать фильтр, отбирающий только должности
Set ФильтрДолжности = ВсеСубъекты.СоздатьФильтр
ФильтрДолжности.Условия.Параметры.ТипСубъекта.Значение = "Должность"
 'строка выше может быть и такой:
 'ФильтрДолжности.Условия.Параметры.ТипСубъекта.Значение = 2 ' 2 - Должность

'Выполняем фильтр и определяем количество записей
Set РезультатФильтрДолжности = ФильтрДолжности.Выполнить
КолвоДолжностей = РезультатФильтрДолжности.КоличествоЭлементов

'Выводим сообщение
MsgBox "Количество должностей: " + CStr(КолвоДолжностей)

End Sub
```

3.4.4 Метод Сохранить

Синтаксис: Сохранить()

Возвращаемый результат: не возвращает.

Сохраняет изменения объекта в базу данных.

Пример кода. См. код в методах «СоздатьОбъект» (п.3.2.8) и «СоздатьГруппу» (п.3.2.9).

3.4.5 Метод Обновить

Синтаксис: Обновить()

Возвращаемый результат: не возвращает.

Обновляет текущий объект, зачитывает свойства объекта из базы данных, при этом все произведенные изменения будут утеряны. Актуализирует состояние класса при наличии сохраненных изменений в объекте, внесенных в другом экземпляре Business Studio. **Пример кода**.

```
…
Set Субъекты = oleapp.ПолучитьОбъекты("БизнесМодель.Субъекты", "Название", "Директор")
Set НужныйСубъект = Субъекты.ПолучитьЭлемент(0)
НужныйСубъект.Обновить
…
```

3.4.6 Метод Удалить

Синтаксис: Удалить()

Возвращаемый результат: не возвращает.

Помечает объект на удаление. Выполнение метода соответствует удалению объекта вручную, при котором не происходит очистки ссылок на удаляемый объект.

Пример кода.

Задача: из организационной структуры удалить юриста.

```
Sub ПримерOLE_Удалить()
  'Получение объекта приложения
  Set oleapp = CreateObject("ByteEnterprise.OleApplication")
  'Запустить Business Studio в редакции и базой на сервере, указанными ранее.
  'В Диспетчере задач появится Business Studio. В панели задач приложения не будет видно.
  Set client_app = oleapp.ЗапуститьКлиентскоеПриложение("U6S\SQLEXPRESS2005", " demo_ole", "Enterprise")
  'В панели задач появится приложение
  oleapp.ПоказатьКлиентскоеПриложение

  'Определяем субъект для удаления
  GuidСубъекта = "23e3bc1b-1b6b-46b6-80c1-3508d9d06b6f" 'Соответствует 'Юрист" в демо-базе
  Set Субъект = oleapp.ПолучитьОбъекты("БизнесМодель.Субъекты", "guid", GuidСубъекта)
  Set СубъектЮрист = Субъект.ПолучитьЭлемент(0)

  'Удаляем субъект
  СубъектЮрист.Удалить

End Sub
```

3.5 Класс «Система.Параметр»

3.5.1 Свойство Значение

Свойство: Значение

Тип параметра: Соответствует типу параметра в Объектной модели. Для параметров типа "ДатаВремя" поле Значение типа "DateTime".

Свойство содержит значение параметра объекта.

Значения для перечислений можно присваивать только в числовом виде. Например:

```
Set НовыйСубъект = oleapp.СоздатьОбъект(ВыбранныйСубъект)
    НовыйСубъект.ТипСубъекта = 1 '1 - Подразделение
```

Определить числовое значение, которое соответствует элементу перечисления нужно следующим образом:

- открыть объектную модель (Справочники -> Объектная модель);
- в объектной модели в ветке «Перечисления» найти необходимое перечисление;
- для конкретного значения, выбрать пункт контекстного меню «Открыть свойства строки»;
- в окне свойства строки отобразить скрытый параметр (Действия –> Настройка колонок) с названием «НомерПараметра». Его значение и необходимо использовать.

Исключением является использование значений перечислений при поиске (фильтрах), где используются как числовые значения, так и строки, содержащие системные названия значений перечисления. Например:

```
Set СМКДокаСписок = oleapp.ПолучитьОбъекты("БизнесМодель.ПараметрыСМК", "СтатусДокумента", "Проект")
```

Можно с тем же результатом можно записать как:

```
Set СМКДокаСписок = oleapp.ПолучитьОбъекты("БизнесМодель.ПараметрыСМК", "СтатусДокумента",0)
```

Пример кода. См. код в методе «СоздатьФильтр» (п.3.4.3).

3.6 Класс «Система.Список»

Данный класс используется для работы со списками элементов.

3.6.1 Свойство КоличествоЭлементов

Свойство: КоличествоЭлементов

Тип параметра: Int (целое число).

Свойство содержит число, показывающее количество элементов списка.

Пример кода.

Задача: определить количество в справочнике физических лиц, определить ФИО первого и третьего элемента, удалить и добавить физлицо.

```
Sub ПримерOLE_РаботаСоСписком()

    'БД и редакция Business Studio, с которыми будем работать
    СерверБД = "U6S\SQLEXPRESS2005"
    База = "demo_ole"
    РедакцияBS = "Enterprise"

    'Получение объекта приложения
    Set oleapp = CreateObject("ByteEnterprise.OleApplication")

    'Запустить Business Studio в редакции и базой на сервере, указанными ранее.
    'В Диспетчере задач появится Business Studio. В панели задач приложения не будет видно.
```

```
Set client_app = oleapp.ЗапуститьКлиентскоеПриложение(СерверБД, База, РедакцияBS)
oleapp.ПоказатьКлиентскоеПриложение 'В панели задач появится приложение

'*****************************************************************
'Начало работы с данными, сортировка списка и удаление по индексу
'*****************************************************************

'Получаем список всех физ.лиц
Set ФизЛицаОПУ = oleapp.ПолучитьКорневуюГруппуКласса("БизнесМодель.ФизЛица")
Set ФизЛицаФильтр = ФизЛицаОПУ.СоздатьФильтр
Set ФизЛицаСписок = ФизЛицаФильтр.Выполнить

'Количество физ.лиц в списке
КолвоФизЛицДоУдаления = ФизЛицаСписок.КоличествоЭлементов

'Запоминаем первое физ.лицо полученного списка
ПервоеФизЛицоДоСортировки = ФизЛицаСписок.ПолучитьЭлемент(0)

'Сортируем список физ.лиц по дате рождения
ФизЛицаСписок.Сортировать ("ДатаРождения")

'Запоминаем 1 и 3-е физ.лицо отсортированного списка
ПервоеФизЛицоПослеСортировки = ФизЛицаСписок.ПолучитьЭлемент(0)
ТретьеФизЛицоПослеСортировки = ФизЛицаСписок.ПолучитьЭлемент(2)

'Удаляем третье физ.лицо списка. При этом в справочнике физ.лиц оно не будет удалено.
ФизЛицаСписок.УдалитьИзСпискаПоИндексу (2)

'Количество физ.лиц в списке
КолвоФизЛицПослеУдаленияПоИдексу = ФизЛицаСписок.КоличествоЭлементов

'*****************************************************************
'Добавление физ.лиц, удаление по объекту
'*****************************************************************

'Добавляем новое физ.лицо с фамилией "Пушкин"
Set НовоеФизЛицо1 = ФизЛицаСписок.Добавить
    НовоеФизЛицо1.Фамилия = "Пушкин"
    НовоеФизЛицо1.Имя = "Александр"
    НовоеФизЛицо1.Отчество = "Сергеевич"
НовоеФизЛицо1.Сохранить 'Физлицом сохраниться в справочнике физ.лиц

'Добавляем новое физ.лицо с фамилией "Дантес".
Set НовоеФизЛицо2 = ФизЛицаСписок.Добавить
```

```
    НовоеФизЛицо2.Фамилия = "Дантес"
    НовоеФизЛицо2.Имя = "Жорж"
    НовоеФизЛицо2.Отчество = "Шарль"
НовоеФизЛицо2.Сохранить 'Физлицом сохраниться в справочнике физ.лиц

'Количество физ.лиц в списке после добавления
КолвоФизЛицПослеДобавления = ФизЛицаСписок.КоличествоЭлементов

'Последнее физ.лицо в списке после добавления
ПослФизЛицоСпискаПослеДобавления = _
ФизЛицаСписок.ПолучитьЭлемент(КолвоФизЛицПослеДобавления - 1)

'Удаляем второе добавленное физ.лицо. При этом в справочнике физ.лиц оно не будет удалено.
Set ФизЛицоУдаления = НовоеФизЛицо2
ФизЛицаСписок.УдалитьИзСпискаОбъект (ФизЛицоУдаления)

'Количество физ.лиц в списке после удаления
КолвоФизЛицПослеУдаленияПоОбъекту = ФизЛицаСписок.КоличествоЭлементов

'Последнее физ.лицо в списке после удаления
ПослФизЛицоСпискаПослеУдалПоОбъекту = _
ФизЛицаСписок.ПолучитьЭлемент(КолвоФизЛицПослеУдаленияПоОбъекту - 1)

'Выводим сообщение о проделанной работе
MsgBox "Физические лица" + Chr(13) + Chr(13) + _
"Исходный список физ.лиц" + Chr(13) + _
"   Кол-во физ.лиц =" + CStr(КолвоФизЛицДоУдаления) + Chr(13) + _
"   Первое физ.лицо: " + ПервоеФизЛицоДоСортировки + Chr(13) + Chr(13) + _
"Список после сортировки по Дате рождения" + Chr(13) + _
"   Первое физ.лицо: " + ПервоеФизЛицоПослеСортировки + Chr(13) + Chr(13) + _
"Список после удаления 3-го физ.лица (по индексу)" + Chr(13) + _
"   Кол-во физ.лиц = " + CStr(КолвоФизЛицПослеУдаленияПоИдексу) + Chr(13) + _
"   Третье физ.лицо '" + ТретьеФизЛицоПослеСортировки + "' было удалено" + Chr(13) + _
Chr(13) + _
"Список после добавления физ.лиц" + Chr(13) + _
"   Кол-во физ.лиц = " + CStr(КолвоФизЛицПослеДобавления) + Chr(13) + _
"   Добавлены: " + НовоеФизЛицо1.Фамилия + " и " + НовоеФизЛицо2.Фамилия + Chr(13) + _
"   Последнее Физ.лицо списка: " + ПослФизЛицоСпискаПослеДобавления + Chr(13) + Chr(13) + _
"Список после удаления физ.лица" + Chr(13) + _
"   Кол-во физ.лиц = " + CStr(КолвоФизЛицПослеУдаленияПоОбъекту) + Chr(13) + _
"   Удалено физ.лицо: " + ФизЛицоУдаления + Chr(13) + _
```

```
"   Последнее Физ.лицо списка: " + ПослФизЛицоСпискаПослеУдалПоОбъекту

End Sub
```

3.6.2 Метод Добавить

Синтаксис: Добавить()

Возвращаемый результат: Система.МетаКласс (см. п.3.4)

Метод создает объект и добавляет его в качестве элемента в конец списка.

Чтобы добавленный элемент появился в справочнике модели, необходимо выполнить сохранение объекта-владельца списка. (см. метод «Сохранить» в классе «Система.Метакласс», п.3.4.4).

Пример кода. См. код в свойстве «КоличествоЭлементов» (п.3.6.1).

3.6.3 Метод ДобавитьОбъект

Синтаксис: ДобавитьОбъект(object "<Объект>")

Возвращаемый результат: не возвращает.

Метод добавляет к списку элемент на основе уже существующего в базе объекта, в отличие от метода «Добавить» (п.3.6.2), который добавляет полностью новый объект. Может использоваться для формирования вспомогательных временных списков в памяти, не имеющих владельца.

Пример кода.

Задача: получить список, состоящий из всех ролей организации и подразделения-организации (субъект «ИнТехПроект»).

```
Sub ПримерOLE_РаботаСоСписком()

 ' БД и редакция Business Studio, с которыми будем работать
 СерверБД = "U6S\SQLEXPRESS2005"
 База = "demo364211fullfarsh"
 РедакцияBS = "Enterprise"

 'Получение объекта приложения
 Set oleapp = CreateObject("ByteEnterprise.OleApplication")

 'Запустить Business Studio в редакции и базой на сервере, указанными ранее.
 'В Диспетчере задач появится Business Studio. В панели задач приложения не будет видно.
 Set client_app = oleapp.ЗапуститьКлиентскоеПриложение(СерверБД, База, РедакцияBS)
 oleapp.ПоказатьКлиентскоеПриложение 'В панели задач появится приложение

 'Сначала создаем список, состоящий из Ролей
 Set ПолныйСписок = oleapp.ПолучитьОбъекты("БизнесМодель.Субъекты", "ТипСубъекта", "Роль")
 КолвоТолькоРолей = ПолныйСписок.КоличествоЭлементов
```

```
    'Добавляем в список объект подразделение-организация
      'Определяем конкретный объект

      Set СписокОрганизация = oleapp.ПолучитьОбъекты("БизнесМодель.Субъекты", "Название",
"ИнТехПроект")

      Set НужныйСубъект = СписокОрганизация.ПолучитьЭлемент(0) ' Первый элемент из списка


      'Добавляем объект к списку
      'Данная операция возможна потому, что класс добавляемого в список элемента тот же,
      'что и у прочих элементов списка. Если требуется добавить в список объекты разных классов,
      'то тип для элементов списка нужно выбирать базовый для всех классов
      'добавляемых объектов. См. объектную модель Business Studio.

      ПолныйСписок.ДобавитьОбъект (НужныйСубъект)


    КолвоЭлементовСписка = ПолныйСписок.КоличествоЭлементов


    'Выводим сообщение о проделанной работе

MsgBox "Список из Ролей. Кол-во: " + CStr(КолвоТолькоРолей) + Chr(13) + _

        "Полный список. Кол-во: " + CStr(КолвоЭлементовСписка)


End Sub
```

3.6.4 Метод ПолучитьЭлемент

Синтаксис: ПолучитьЭлемент(int <Номер>)

Возвращаемый результат: Система.МетаКласс (см. п.3.4)

Метод позволяет получить элемент списка по указанному номеру. *Первый элемент списка имеет номер 0. В VBA можно использовать вместо ПолучитьЭлемент Item и выбирать элемент не только по номеру, но и по имени.*

Пример кода. См. код в свойстве «КоличествоЭлементов» (п.3.6.1).

3.6.5 Метод УдалитьИзСпискаПоИндексу

Синтаксис: УдалитьИзСпискаПоИндексу(int <Номер>)

Возвращаемый результат: не возвращает.

Метод удаляет элемент списка по указанному номеру.

Пример кода. См. код в свойстве «КоличествоЭлементов» (п.3.6.1).

3.6.6 Метод УдалитьИзСпискаОбъект

Синтаксис: УдалитьИзСпискаОбъект(object "<Объект>")

Возвращаемый результат: не возвращает.

Метод удаляет списка указанный объект.

Пример кода. См. код в свойстве «КоличествоЭлементов» (п.3.6.1).

3.6.7 Метод Сортировать

Синтаксис: Сортировать(string "<НаборПолей>")

Возвращаемый результат: не возвращает.

Сортирует список по указанному набору полей.

Например: "Поле1,-Поле2", где префикс '-' задает сортировку по убыванию.

Пример кода. См. код в свойстве «КоличествоЭлементов» (п.3.6.1).

3.7 Класс «Система.Фильтр»

3.7.1 Метод Выполнить

Синтаксис: Выполнить()

Возвращаемый результат: Система.Список (см. п.3.6)

Метод запускает фильтр на выполнение, в результате возвращает список элементов класса.

Пример кода. См. код в методе «СоздатьФильтр» (п.3.4.3).

3.7.2 Свойство Условия.Параметры.

Свойство: Условия.Параметры.<ИмяПараметра>

Тип параметра: Система.ПараметрФильтра (см. п.3.8)

Обращение к параметру фильтра осуществляется по имени параметра из Объектной модели Business Studio.

Внимание: Рекомендуется задавать условия только по хранимым параметрам класса. Возможность построения фильтра по нехранимым параметрам можно проверить, открыв окно фильтра по соответствующему классу в Business Studio.

Пример кода. См. код в методе «СоздатьФильтр» (п.3.4.3).

3.8 Класс «Система.ПараметрФильтра»

3.8.1 Свойство Значение

Свойство: Значение

Тип параметра: Соответствует типу параметра в объектной модели.

Свойство содержит значение параметра фильтра.

Пример кода. См. код в свойствах данного класса.

3.8.2 Свойство ОтрицаниеУсловия

Свойство: ОтрицаниеУсловия

Тип параметра: Bool (True, False).

Свойство задает отрицание заданного условия фильтра.

Пример кода.

Задача: показать количество субъектов, у которых название не равно «директор».

```
Sub ПримерOLE_ФильтрСубъектовБезДиректора()
  'Получение объекта приложения
  Set oleapp = CreateObject("ByteEnterprise.OleApplication")
  'Запустить Business Studio в редакции и базой на сервере, указанными ранее.
  'В Диспетчере задач появится Business Studio. В панели задач приложения не будет видно.
  Set client_app = oleapp.ЗапуститьКлиентскоеПриложение("U6S\SQLEXPRESS2005", " demo_ole", "Enterprise")
  'В панели задач появится приложение
  oleapp.ПоказатьКлиентскоеПриложение

  'Указываем, что дальнейшая работа будет с Субъектами
  Set ВсеСубъекты = oleapp.ПолучитьКорневуюГруппуКласса("БизнесМодель.Субъекты")

  'Создать фильтр, отбирающий только должности
  Set ФильтрНеДиректор = ВсеСубъекты.СоздатьФильтр
    ФильтрНеДиректор.Условия.Параметры.Название.Значение = "Директор"
    ФильтрНеДиректор.Условия.Параметры.Название.ОтрицаниеУсловия = True

  'Выполняем фильтр и определяем количество записей
  Set РезультатФильтрНеДиректор = ФильтрНеДиректор.Выполнить
  КолвоСубъектов = РезультатФильтрНеДиректор.КоличествоЭлементов

  'Выводим сообщение
  MsgBox "Количество субъектов без Директора: " + CStr(КолвоСубъектов)

End Sub
```

3.8.3 Свойство ТипФильтрации

Свойство: ТипФильтрации

Результат: Int (целое число от 0 до 4)

Свойство задает тип фильтрации. При задании условия по умолчанию устанавливается тип фильтрации "Значение".

Допустимые варианты:

0 - Значение,

1 - Диапазон,

3 - Подфильтр,

4 - Нет.

Задание типа фильтрации отличного от умолчания (Значение) может осуществляться по номеру.

Пример кода.

Задача: определить список показателей, к целевому значению которых будет стремиться компания в указанных год.

```
Sub ПримерOLE_РаботаСФильтромДиапазон()
  ' Получение объекта приложения
  Set oleapp = CreateObject("ByteEnterprise.OleApplication")
  'Запустить Business Studio Enterprise с базой под именем demo_ole на сервере U6S\SQLEXPRESS2005. В Диспетчере задач появится Business Studio. В панели задач приложения не будет видно.
  Set client_app = oleapp.ЗапуститьКлиентскоеПриложение("U6S\SQLEXPRESS2005", "demo_ole", "Enterprise")
  'В панели задач появится приложение
  oleapp.ПоказатьКлиентскоеПриложение

  'Указываем, что дальнейшая работа будет с Показателями
  Set ПоказателиОПУ = oleapp.ПолучитьКорневуюГруппуКласса("БизнесМодель.ПоказателиBSC")

  'Создать фильтр, показывающий количество показателей, к достижению которых стремимся в 2014 г.
  Set ФильтрПоказатели = ПоказателиОПУ.СоздатьФильтр
    'Задание условий
    ФильтрПоказатели.Условия.Параметры.ЦелеваяДата.ТипФильтрации = 1 'Диапазон
    ФильтрПоказатели.Условия.Параметры.ЦелеваяДата.Мин = CDate("01.01.2014")
    ФильтрПоказатели.Условия.Параметры.ЦелеваяДата.ВключатьМин = True
    ФильтрПоказатели.Условия.Параметры.ЦелеваяДата.Макс = CDate("31.12.2014")
    ФильтрПоказатели.Условия.Параметры.ЦелеваяДата.ВключатьМакс = True
  'Выполняем фильтр
  Set Результат = ФильтрПоказатели.Выполнить

  КолвоПоказатель = Результат.КоличествоЭлементов

  MsgBox "Количество показателей, к достижению которых стремимся в 2014: " + CStr(КолвоПоказатель)

End Sub
```

3.8.4 Свойство ОператорСравнения

Свойство: ОператорСравнения

Результат: Int (целое число от 0 до 5)

Оператор сравнения для типа фильтрации "Значение" (см.п.3.8.3). По умолчанию "Равно".

Возможные варианты:

0 - Больше,

1 - БольшеИлиРавно,

2 - Меньше,

3 - МеньшеИлиРавно,

4 - Равно,

5 - Подобный (только для строковых параметров).

Задание оператора сравнения отличного от умолчания (Равно) может осуществляться по номеру.

Пример кода.

Задача: показать количество субъектов, у которых в названии есть слово «директор».

```
Sub ПримерOLE_СоздатьФильтрОператорСравнения()
  'Получение объекта приложения
  Set oleapp = CreateObject("ByteEnterprise.OleApplication")
  'Запустить Business Studio в редакции и базой на сервере, указанными ранее.
  'В Диспетчере задач появится Business Studio. В панели задач приложения не будет видно.
  Set client_app = oleapp.ЗапуститьКлиентскоеПриложение("U6S\SQLEXPRESS2005", " demo_ole", "Enterprise")
  'В панели задач появится приложение
  oleapp.ПоказатьКлиентскоеПриложение

  'Указываем, что дальнейшая работа будет с Субъектами
  Set ВсеСубъекты = oleapp.ПолучитьКорневуюГруппуКласса("БизнесМодель.Субъекты")

  'Создать фильтр, отбирающий только должности
  Set Фильтр = ВсеСубъекты.СоздатьФильтр

    'Условие - в названии субъекта содержится слово "директор"
     Фильтр.Условия.Параметры.Название.ОператорСравнения = 5 'строковый параметр, соответствует «~»
     Фильтр.Условия.Параметры.Название.Значение = "%директор%"

  'Выполняем фильтр и определяем количество записей
  Set РезультатФильтр = Фильтр.Выполнить
```

```
КолвоДолжностей = РезультатФильтр.КоличествоЭлементов


'Выводим сообщение
MsgBox "Количество субъектов, у которых в названии есть слово 'директор' : " + CStr(КолвоДолжностей)


End Sub
```

3.8.5 Свойства Мин и Макс

Свойство: Мин

Свойство: Макс

Тип параметра: Соответствует типу параметра в объектной модели.

Минимальное и максимальное значения параметра фильтра для типа фильтрации "Диапазон" (см.п.3.8.3).

Пример кода. См. код в свойстве «ТипФильтрации» (п.3.8.3).

3.8.6 Свойства ВключатьМин и ВключатьМакс

Свойство: ВключатьМин

Свойство: ВключатьМакс

Тип параметра: Bool (True, False)

Свойство логики, управляющей включением минимального и максимального значения в результат фильтрации при типе фильтрации "Диапазон" (см.п.3.8.3). Если логика включена, граничное значение войдет в результат фильтрации. По умолчанию логика включена.

Пример кода. См. код в свойстве «ТипФильтрации» (п.3.8.3).

3.8.7 Свойство Подфильтр

Свойство: Подфильтр

Тип параметра: Система.Фильтр (см. п.3.7).

Свойство задает вложенный фильтр для данного параметра фильтра. Используется с типом фильтрации "Подфильтр" (см.п.3.8.3). Только для объектных и списковых параметров. Подфильтр строится по тому классу, которому принадлежит параметр.

Пример кода.

Задача:

```
Sub ПримерOLE_СоздатьФильтрСПодфильтром()
  'Получение объекта приложения
  Set oleapp = CreateObject("ByteEnterprise.OleApplication")
  'Запустить Business Studio в редакции и базой на сервере, указанными ранее.
  'В Диспетчере задач появится Business Studio. В панели задач приложения не будет видно.
```

```
Set client_app = oleapp.ЗапуститьКлиентскоеПриложение("U6S\SQLEXPRESS2005", " demo_ole", "Enterprise")
'В панели задач появится приложение
oleapp.ПоказатьКлиентскоеПриложение

'Указываем, что дальнейшая работа будет с Показателями
Set ПоказателиОПУ = oleapp.ПолучитьКорневуюГруппуКласса("БизнесМодель.ПоказателиBSC")

'Создаем фильтр по процессам
Set ФильтрПоказатели = ПоказателиОПУ.СоздатьФильтр
    ФильтрПоказатели.Условия.Параметры.ЕдиницаИзмерения.ТипФильтрации = 3 'Подфильтр

  'задаем условия подфильтра по единице измерения показателя
  'подфильтр будет по Единицам измерения
   Set ЕдиницыОПУ = _
   oleapp.ПолучитьКорневуюГруппуКласса("БизнесМодель.ЕдиницыИзмеренияЗатрат")

    'подфильтр отбирает единицы с сокращением "руб."
    Set ФильтрЕдиницы = ЕдиницыОПУ.СоздатьФильтр
    ФильтрЕдиницы.Условия.Параметры.Сокращение.Значение = "руб."

  'непосредственное задание условия фильтра Показателя по подфильтру
  ФильтрПоказатели.Условия.Параметры.ЕдиницаИзмерения.Подфильтр = ФильтрЕдиницы


'Выполняем фильтр по Показателю и определяем количество записей
Set РезультатФильтрПоказатели = ФильтрПоказатели.Выполнить
КолвоПоказателей = РезультатФильтрПоказатели.КоличествоЭлементов

'Выводим сообщение
MsgBox "Количество показателей, которые измеряются в рублях: " + CStr(КолвоПоказателей)

End Sub
```

3.8.8 Свойство ВключатьПодгруппы

Свойство: ВключатьПодгруппы

Тип параметра: Bool (True, False).

Свойство указывает, будут ли в результат выполнения фильтра включены подгруппы класса, по которому выполняется фильтр. Соответствует выбору кнопки «Смотреть в подгруппах» на панели инструментов фильтра.

Значение по умолчанию True.

Пример кода. См. код в свойстве «РезультатВключает» (п.3.8.10).

3.8.9 Свойство ВключатьГруппуФильтра

Свойство: ВключатьГруппуФильтра

Тип параметра: Bool (True, False).

Свойство указывает, будет ли в результат выполнения фильтра включена группа, по которой выполняется фильтр. Соответствует выбору кнопки «Включать группу фильтра» на панели инструментов окна фильтра.

Значение по умолчанию False.

Пример кода. См. код в свойстве «РезультатВключает» (п.3.8.10).

3.8.10 Свойство РезультатВключает

Свойство: РезультатВключает

Тип параметра: Int (целое число)

Свойство задает условия вхождения полученных данных в результат выполнения фильтра. Соответствует выбору кнопок панели инструментов окна фильтра: «Включать подгруппы», «Включать конечные объекты», «Показать удаленные», «Показать неудаленные».

Свойство может принимать следующие значения:

1 – Группы,

2 – Конечные объекты,

4 – Удаленные,

8 – НеУдаленные.

Для выбора нескольких значений необходимо брать их сумму. Например, 11 - это 1 + 2 + + 8, т.е. выбрать все неудаленные, 7 – это 1 + 2 + 4, т.е. включаем удаленные группы и конечные объекты.

Значение по умолчанию – 11.

Пример кода.

Задача: определить, есть ли удаленные объекты в базе.

```
Sub ПримерOLE_НаличиеУдаленныхОбъектов()
'============================================================
' Задание необходимых параметров перед запуском
   'БД и редакция Business Studio, с которыми будем работать
   СерверБД = "U6S\SQLEXPRESS2005"
   База = "demo_ole"
   РедакцияBS = "Enterprise"
'============================================================
   'Запуск Business Studio
```

```
'Получение объекта приложения
Set oleapp = CreateObject("ByteEnterprise.OleApplication")

'Запустить Business Studio в редакции и базой на сервере, указанными ранее.
'В Диспетчере задач появится Business Studio. В панели задач приложения не будет видно.
Set client_app = oleapp.ЗапуститьКлиентскоеПриложение(СерверБД, База, РедакцияBS)
'В панели задач появится приложение
oleapp.ПоказатьКлиентскоеПриложение

'Создаем фильтр по всем Справочникам
Set ВсеСправочники = oleapp.ПолучитьКорневуюГруппуКласса("База.Справочники")

  'Определяем количество удаленных объектов
Set ФильтрУдаленных = ВсеСправочники.СоздатьФильтр
    ФильтрУдаленных.ВключатьПодгруппы = True
    ФильтрУдаленных.ВключатьГруппуФильтра = True
    ФильтрУдаленных. РезультатВключает = 7 'Включаем удаленные группы и конечные объекты
Set СписокУдаленных = ФильтрУдаленных.Выполнить
КолвоУдаленных = СписокУдаленных.КоличествоЭлементов

'Готовимся к выводу результатов
If КолвоУдаленных = 0 Then
     РезультатУдаленных = "В базе нет удаленных объектов."
  Else
     РезультатУдаленных = "В базе есть удаленные объекты."
End If

 'Вывод сообщения о результатах
 MsgBox "Проверка удаленных: " + РезультатУдаленных

End Sub
```

3.9 Сценарии работы с Business Studio через OLE

3.9.1 Пример синхронизации справочника сотрудников с внешним хранилищем

Список сотрудников будет находиться в файле MS Excel (Таблица 3.9.1). Пример приведен на языке программирования VBA.

Таблица 3.9.1 Структура файла Excel со списком сотрудников

Фамилия	Имя	Отчество	Дата рождения	р.тел	e-mail
Иванов	Иван	Иванович	29.09.1969	202-19-00	name@firma.ru
Петров	Петр	Петрович	04.05.1978	202-19-01	name2@firma.ru

```
Sub СинхронизацияФизЛиц()
    Set appEx = Application
    Dim i As Integer
    Dim Фамилия, Имя, Отчество, ДатаРождения, РабочийТелефон As String

    ' Получение объекта приложения
    Set oleapp = CreateObject("ByteEnterprise.OleApplication")
    ' Инициализации приложения, в качестве параметров передаются имя сервера, название базы, версия продукта
    Set client_app = oleapp.ЗапуститьКлиентскоеПриложение("<Имя_сервера_БД>", "<Имя_базы>", "<Версия_продукта>")

    ' Получение корневой группы класса Физлиц
    Set ОПУ_ФЛ = oleapp.ПолучитьКорневуюГруппуКласса("БизнесМодель.ФизЛица")
    ' Получение корневой группы класса Контактов физлиц
    Set ОПУ_Контакты = oleapp.ПолучитьКорневуюГруппуКласса("БизнесМодель.КонтактыФизЛиц")

    ' Получение корневой группы класса Типов контактов
    Set списокТипов = oleapp.ПолучитьОбъекты("БизнесМодель.ТипыКонтактов", "Название", "Рабочий телефон")
    Set типРабочийТелефон = списокТипов.ПолучитьЭлемент(0)

    i = 2
    While appEx.Cells(i, 1) <> ""
        Фамилия = CStr(appEx.Cells(i, 1))
        Имя = CStr(appEx.Cells(i, 2))
        Отчество = CStr(appEx.Cells(i, 3))
        ДатаРождения = CStr(appEx.Cells(i, 4))
        РабочийТелефон = CStr(appEx.Cells(i, 5))
        ЭлектроннаяПочта = CStr(appEx.Cells(i, 6))

        ' Создание фильтра по классу Физлиц
        Set фильтрФЛ = ОПУ_ФЛ.СоздатьФильтр
        ' настраиваем условия фильтра для поиска по Фамилии, Имени, Отчеству, Дате рождения
        фильтрФЛ.Условия.Параметры.Фамилия.Значение = Фамилия
```

```
фильтрФЛ.Условия.Параметры.Имя.Значение = Имя
фильтрФЛ.Условия.Параметры.Отчество.Значение = Отчество
фильтрФЛ.Условия.Параметры.ДатаРождения.Значение = CDate(ДатаРождения)
' выполняем фильтр - получаем список физлиц с заданными Фамилией, Именем, Отчеством, Датой рождения
Set списокФЛ = фильтрФЛ.Выполнить
' получаем элемент Физлицо
If списокФЛ.КоличествоЭлементов = 0 Then
    ' если физлицо не найдено - создаем новое
    Set фл = oleapp.СоздатьОбъект(ОПУ_ФЛ)
    фл.Фамилия = Фамилия
    фл.Имя = Имя
    фл.Отчество = Отчество
    Set пар = фл.НайтиПараметр("ДатаРождения")
    пар.Значение = CDate(ДатаРождения)
Else
    ' если физлицо найдено - будем обновлять его данные
    Set фл = списокФЛ.ПолучитьЭлемент(0)
End If
' обновляем рабочий телефон физлица - элемент списка контаков
If РабочийТелефон <> "" Then
    ' Создание фильтра по классу контактов Физлиц
    Set ФильтрКонтакты = ОПУ_Контакты.СоздатьФильтр
    ' настраиваем условия фильтра для поиска контакта физлица
    ФильтрКонтакты.Условия.Параметры.ТипКонтакта.Значение = типРабочийТелефон
    ' настраиваем подфильтр для задания условия по физлицу - владельцу контакта
    ФильтрКонтакты.Условия.Параметры.Владелец.ТипФильтрации = 3 ' подфильтр
    ФильтрКонтакты.Условия.Параметры.Владелец.Подфильтр = фильтрФЛ
    ' выполняем фильтр - получаем список контактов с заданным условием
    Set списокКонтактов = ФильтрКонтакты.Выполнить
    If списокКонтактов.КоличествоЭлементов = 0 Then
        ' если контакт не найден - создаем новый элемент в списке контактов физлица
        Set конт = фл.Контакты.Добавить
        конт.ТипКонтакта = типРабочийТелефон
    Else
        ' если контакт найден - будем обновлять его данные
        Set конт = списокКонтактов.ПолучитьЭлемент(0)
    End If
    ' обновляем данные контакта
```

```
            конт.Контакт = РабочийТелефон
        End If
        ' сохраняем изменения физлица
        фл.Сохранить
        i = i + 1
    Wend
End Sub
```

3.9.2 Пример создание субъекта и работа с ним

```
Sub ДругиеПримеры()
    Set oleapp = CreateObject("ByteEnterprise.OleApplication")
    ' Инициализации клиентского приложения, в качестве параметров передаются имя сервера, название базы,  версия продукта
    Set Client_app = oleapp.ЗапуститьКлиентскоеПриложение("<Имя_сервера_БД>", "<Имя_базы>", "<Версия_продукта>")

    ' Пример создания Субъекта и работы с ним
    Set СубъектКорень = oleapp.ПолучитьКорневуюГруппуКласса("БизнесМодель.Субъекты")
    Set ВыбранныйСубъект = oleapp.ВыбратьОбъект(СубъектКорень)
    Set НовыйСубъект = oleapp.СоздатьОбъект(ВыбранныйСубъект)
    ' Параметры «Название» и «ТипСубъекта» являются обязательными для субъектов
    НовыйСубъект.Название = "НазваниеНовогоСубъекта"
    НовыйСубъект.ТипСубъекта = 1    '1 – Подразделение; 2 – Должность (см. Объектную модель)
    НовыйСубъект.Сохранить

   `Если названия субъектов повторяются – используем для идентификации guid
    a = НовыйСубъект.GUID
    Set ФильтрСубъектПоGUID = СубъектКорень.СоздатьФильтр
    ФильтрСубъектПоGUID.Условия.Параметры.GUID.Значение = a
    ФильтрСубъектПоGUID.ВключатьПодгруппы = True
    Set РезультатФильтра = ФильтрСубъектПоGUID.Выполнить
    If (РезультатФильтра.КоличествоЭлементов > 0) Then
        Set СубъектПоGUID = РезультатФильтра.ПолучитьЭлемент(0)
    Else
     MsgBox "Субъект с guid='" + a + "' не найден."
    End If

'Пример получения константы
    Set конст = oleapp.ПолучитьКорневуюГруппуКласса("БизнесМодель.Константы")
```

```
Руководитель = конст.РуководительОрганизации

'Добавить Субъект в группу; для этого создать группу и связь
    Set ГруппаПУ = oleapp.ПолучитьКорневуюГруппуКласса ("БизнесМодель.Группы")
    Set Группа = oleapp.СоздатьОбъект(ГруппаПУ)
    Группа.Название = "Группа_1"
    Группа.Сохранить
   Set СвязиГруппыАнализаПУ = oleapp.ПолучитьКорневуюГруппуКласса("БизнесМодель.СвязиГруппАнализа")
    Set ОбъектСвязь = oleapp.СоздатьОбъект(СвязиГруппыАнализаПУ)
    ОбъектСвязь.ГруппаАнализа = Группа
    ОбъектСвязь.Справочник = ВыбранныйСубъект
    ОбъектСвязь.Сохранить

'Обращение к пользовательскому параметру. Допустим пользовательский справочник "БизнесМодель.usr_Класс" содержит несколько записей.
    Set Объект = oleapp.ПолучитьКорневуюГруппуКласса("БизнесМодель.usr_Класс")
    Set Фильтр = Объект.СоздатьФильтр
    Set Объекты = Фильтр.Выполнить   ' Получаю весь справочник
    col = Объекты.Count   ' - количество элементов в этом справочнике
    Set Элемент = Объекты.Item(0) 'Или так: Set Элемент = Объекты.ПолучитьЭлемент(0)
  ' Класс содержит стандартный параметр "Название"
    'Парам1 = Элемент.Название - так можно только для системых параметров
    Парам1 = Элемент.Параметры.Item("Название").Значение
    'Или так: Парам1 = Элемент.Параметры.ПолучитьЭлемент("Название").Значение
  ' Класс содержит пользовательский параметр "usr_полеКласса" простого типа (вещественный)
    Парам2 = Элемент.Параметры.Item("usr_полеКласса").Значение
    'Или так: Парам2 = Элемент.Параметры.ПолучитьЭлемент("usr_полеКласса").Значение
  ' Класс содержит параметр - список "usr_полеСпск" - для списка используем Set
    Set Парам3 = Элемент.Параметры.Item("usr_полеСпск").Значение
    'Или так: Set Парам3 = Элемент.Параметры.ПолучитьЭлемент("usr_полеСпск").Значение
  ' Элементы списка объектного типа, чтобы получить элемент используем Set
    Set ЭлСписка = Парам3.Item(0)
  ' Элемент списка содержит параметр "usr_поле1" простого типа (строка)
    ПЭС1 = ЭлСписка.Параметры.Item("usr_поле1").Значение

'Перебрать все параметры:
    Параметры = "Параметры: "
     For Each парам In Объект.Параметры
        Параметры = Параметры + парам.Наименование + ", "
```

```
    Next

'Назначить физическое лицо субъекту:
    Set ФизЛица = oleapp.ПолучитьКорневуюГруппуКласса("БизнесМодель.ФизЛица")
    Set ФизЛицаФильтр = ФизЛица.СоздатьФильтр
    ФизЛицаФильтр.Условия.Параметры.Фамилия.Значение = "Гаврилова"
    ФизЛицаФильтр.Условия.Параметры.Имя.Значение = "Анна"
    ФизЛицаФильтр.Условия.Параметры.Отчество.Значение = "Петровна"
    Set СписокФЛ = ФизЛицаФильтр.Выполнить
  If (СписокФЛ.КоличествоЭлементов > 0) Then
    Set ФЛ = СписокФЛ.ПолучитьЭлемент(0)
    Set ФизлицаСубъектов =
oleapp.ПолучитьКорневуюГруппуКласса("БизнесМодель.Физлица_Субъектов")
    Set ФизЛицоСубъекта = oleapp.СоздатьОбъект(ФизлицаСубъектов)
    ФизЛицоСубъекта.Субъект = ВыбранныйСубъект
    ФизЛицоСубъекта.Физлицо = ФЛ
    ФизЛицоСубъекта.Сохранить
  Else
    MsgBox "Запрашиваемое лицо не найдено"
  End If

'Удалить физлицо из субъекта:
    Set ФильтрФизлицСубъектов = ФизлицаСубъектов.СоздатьФильтр
    ФильтрФизлицСубъектов.Условия.Параметры.Физлицо.Значение = ФЛ
    ФильтрФизлицСубъектов.Условия.Параметры.Субъект.Значение = ВыбранныйСубъект
    Set СписокФизлицСубъектов = ФильтрФизлицСубъектов.Выполнить
     For Each ФизЛицоСубъекта In СписокФизлицСубъектов
        ФизЛицоСубъекта.Удалить
     Next

'Получить правильное название класса, воспользовавшись окном выбора:
    ИмяКласса = oleapp.ВыбратьКласс


End Sub
```

Здесь <Имя_сервера_БД> – имя сервера базы данных, <Имя_базы> – название базы данных, <Версия_продукта> – Enterprise, Professional или Cockpit. Версия продукта должна соответствовать имеющейся лицензии, иначе запуск приложения не удастся.

3.10 Возможные сообщения об ошибках

3.10.1 Невозможность соединения с удаленным сервером

Одна из возможных причин, по которой может появиться сообщение вида «*Ошибка при обращении к службе брокера: Невозможно соединиться с удаленным сервером*», это отсутствие запущенной службы BS_PingHost на сервере лицензий Business Studio (дополнительно см.п.1.4).

ГЛАВА 4. ПРИЛОЖЕНИЕ

4.1 Просмотр подключений к серверу лицензий

Можно определить имена компьютеров, с которых установлено подключение, запустив на сервере лицензий какую-либо утилиту, показывающую статистику соединений TCP/IP. Например, на базе встроенной команды netstat можно сделать bat-файл такого вида:

```
@echo off
set port=5555
echo Connections to BS_PingHost (port %port%):
netstat|findstr /R /c:"TCP.*:%port%.*:.*"|findstr /R /N /C:".*"
pause
```

Запустив этот файл, можно увидеть все подключения к службе BS_PingHost.

4.2 Пример создания пользовательского класса с помощью Metaedit

В качестве примера возьмем задачу создать объектное поле *Квалификация* для субъектов-должностей, у этого поля в свою очередь должны быть параметры *Разряд* (целое), *Описание* (строка) и *Экзамен* (текст).

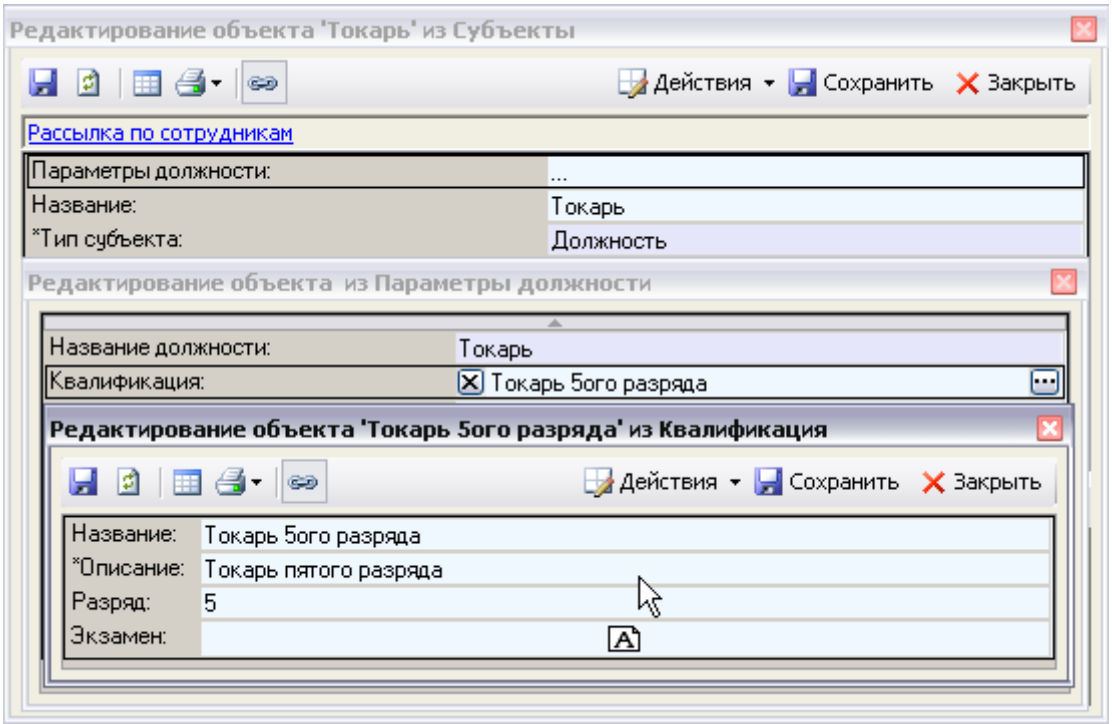

Рис. 4.2.1

Загрузить метаданные из базы данных (см. п. 2.2.1). Для этого выбрать пункт меню «Загрузить из базы данных», выделить в списке баз данных нужную, нажать кнопку «ОК».

Поскольку для каждого субъекта-должности необходимо будет создавать однозначное поле *Квалификация*, то необходимо добавить класс-справочник *Квалификация*.

Добавить пользовательский справочник как потомок класса *База.Справочники*. Для этого выбрать модуль *БизнесМодель*, выделить в дереве класс *Классы| База.ОбъектыСистемы База.Справочники*, в контекстном меню выбрать пункт «Добавить от текущего».

В открывшемся окне «Свойства класса» (см. п. 2.3.3) ввести название справочника – *Квалификация*. Теперь в списке классов добавился пользовательский класс *БизнесМодель.usr_Квалификация*. Чтобы изменить отображение названия в программе, необходимо открыть закладку «Доп. опции», ввести опцию *ПоказКласса.Заголовок* – значение *Квалификация*. На закладке «Описание» дать описание справочника *Необходимая квалификация должности*.

В классе *БизнесМодель.usr_Квалификация* добавить параметры *Разряд, Описание, Экзамен*. Для этого выделить в дереве класс *БизнесМодель.usr_Квалификация*, в контекстном меню списка параметров выбрать пункт «Добавить».

В открывшемся окне «Настройки параметра класса» (см. п. 2.3.5) ввести название *usr_Описание*, на закладке «Опции» выбрать тип параметра *Простой*, нажать кнопку  , выбрать тип свойства *Строка*, длина *100*. Теперь у класса *БизнесМодель.usr_Квалификация* добавился параметр *usr_Описание* типа *Строка*. Чтобы изменить отображение названия параметра в программе, необходимо на закладке «Доп. опции» ввести опцию *Показ.Заголовок* – значение *Описание*. Сделать параметр *Описание* обязательным для заполнения, для этого в свойствах параметра на закладе «Доп. опции» добавить опцию *Показ.Обязательный* – значение *Да*. На закладке «Описание» ввести *Описание квалификации*.

Добавить параметр *usr_Разряд*, тип параметра *Простой*, тип свойства *Целый.* Доп. опция *Показ.Заголовок* – значение *Разряд*. Описание *Необходимый разряд для квалификации*.

Добавить параметр *usr_Экзамен* тип параметра *Простой*, тип свойства *Текст*. Доп. опция *Показ.Заголовок* – значение *Экзамен*. Описание *Многострочное описание квалификационного экзамена, необходимого для получения разряда*.

Поскольку поле *Квалификация* необходимо только для субъектов типа «Должность», то добавлять его будем не в справочник «Субъекты», а в элементы списков «Параметры должности».

В класс *БизнесМодель.ПараметрыДолжности* необходимо добавить параметр *usr_Квалификация*. Выбрать тип параметра *Объектный*, нажать кнопку  , выбрать класс *БизнесМодель.usr_Квалификация*. На закладке «Доп. опции» задать *Показ.Заголовок* – значение *Квалификация*.

Редактирование метаданных закончено, теперь необходимо применить их к базе данных (см. п. 2.2.3). Для этого выбрать пункт меню «Файл\ Применить к базе данных». В открывшемся списке баз отметить галочками необходимые базы данных, нажать кнопку «ОК». Откроется окно «Список необходимых изменений».

На закладке «Классы» в категории «СозданиеКласса» новый класс *БизнесМодель.usr_Квалификация* (Рис. 4.2.2).

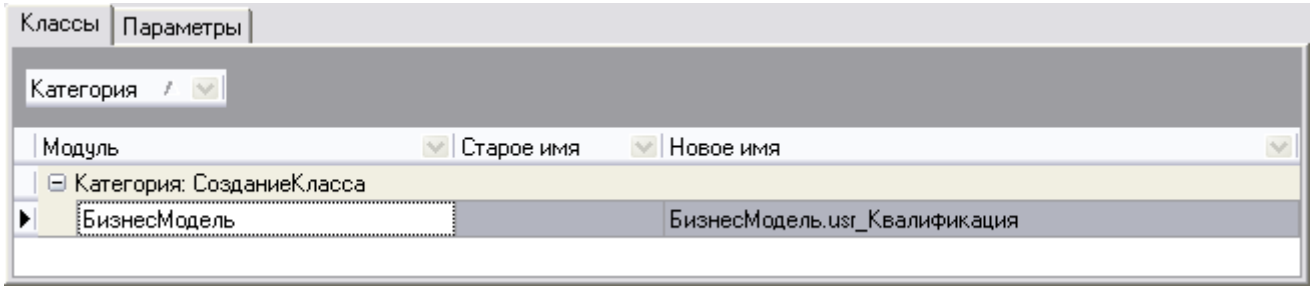

Рис. 4.2.2

На закладке «Параметры» в классе «ПараметрыДолжности» новый параметр *usr_Квалификация* (Рис. 4.2.3).

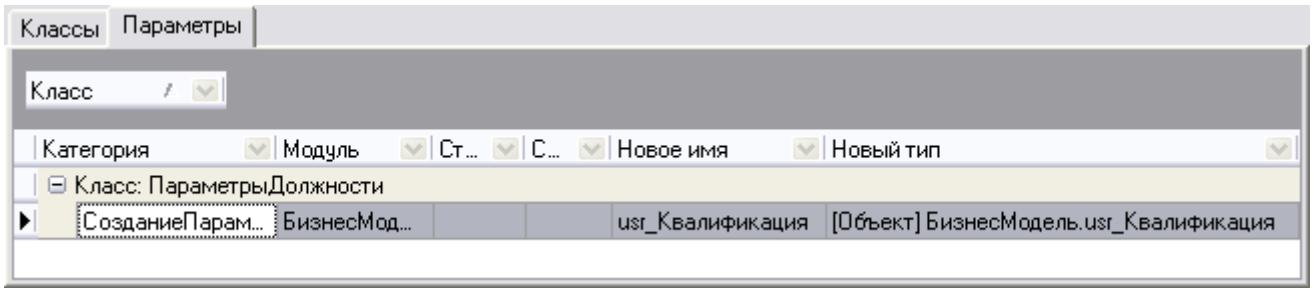

Рис. 4.2.3

Закладки «Ключи» и «Значения» отсутствуют, так как в примере не изменялись ключи существующих классов и значения по умолчанию для параметров существующих классов.

По нажатию кнопки «ОК» запустится процесс применения метаданных к выбранной базе данных. При успешном применении метаданных в базе данных появится новый класс *Квалификация* и новый параметр *Квалификация* в списке *Параметры должности*.

Необходимо дать пользователю права на доступ к новому классу. Для этого переключиться в интерфейс администратора системы (см. главу 16 Администрирование системы в «Руководстве пользователя»), выбрать пункт главного меню «Администрирование –> Справочники администрирования». Открыть справочник «Категории прав», выделить нужную категорию прав, нажать гиперссылку *Редактирование прав*. В открывшемся окне «Права» выделить в ветке *Классы*| *Объекты системы*| *Справочники* справочник «Квалификация» и установить опции «Разрешить» для прав *Доступ, Изменение, Создание* и *Удаление* (Рис. 4.2.4).

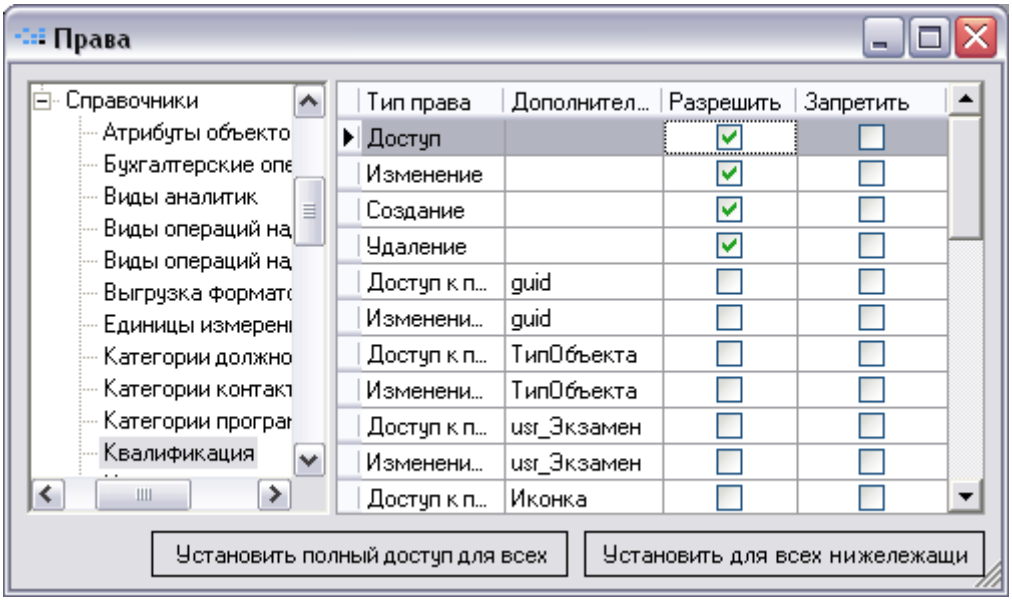

Рис. 4.2.4

4.3 Пример создания пользовательского списка с помощью Metaedit

В качестве примера создадим список терминов, используемых в процессе, которые можно было бы выбирать из некоего общего справочника. Для каждого термина должна быть возможность дать развернутое текстовое описание.

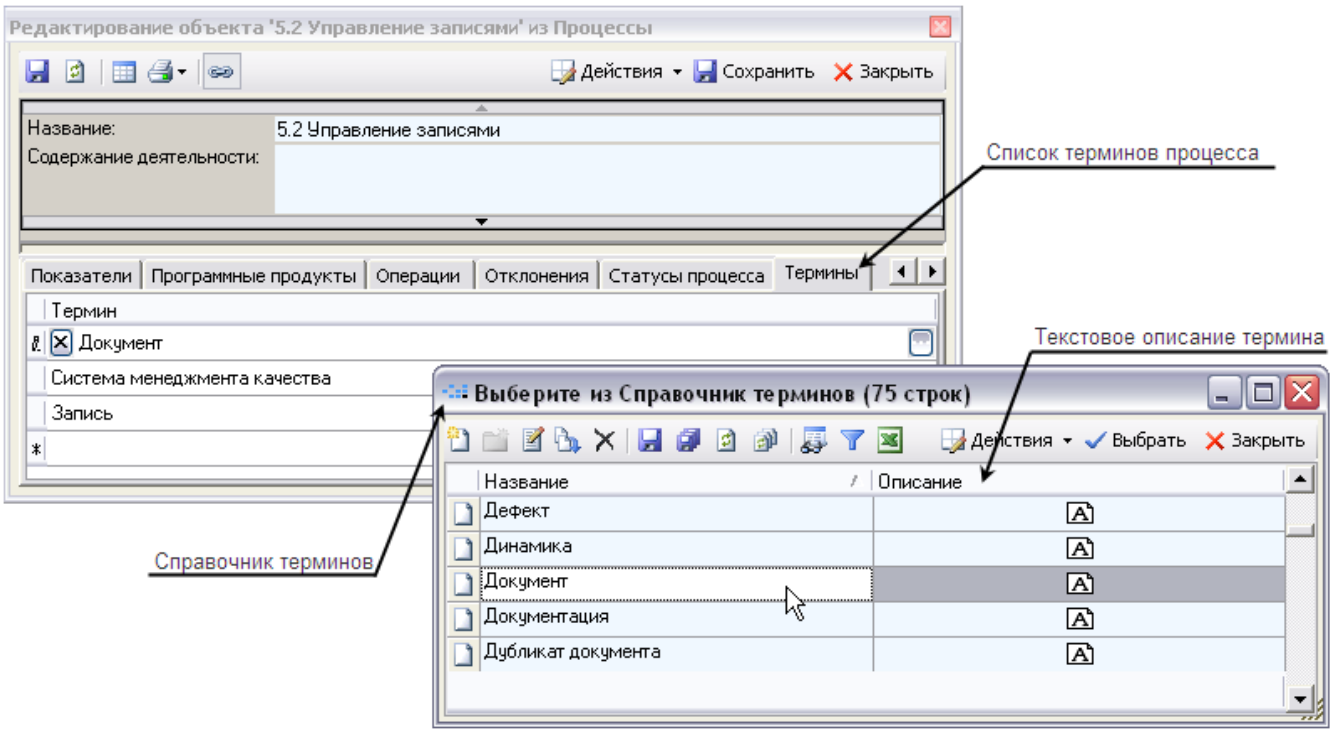

Рис. 4.3.1

Загрузить метаданные из базы данных (см. п. 2.2.1). Для этого выбрать пункт меню «Загрузить из базы данных», выделить в списке баз данных нужную, нажать кнопку «ОК».

Поскольку в различных процессах могут использоваться общие термины, необходимо добавить класс-справочник *Термины*, из которого будет впоследствии заполняться список для конкретного процесса. Для этого выбрать модуль *БизнесМодель.* Добавить пользовательский справочник *БизнесМодель.usr_Термины* как потомок класса *Классы*| *Ба-*

за.ОбъектыСистемы| *База.Справочники*. Чтобы изменить отображение названия класса в программе на закладке «Доп. опции» свойств класса создать опцию *Показ-Класса.Заголовок* – значение *Справочник терминов*. На закладке «Описание» дать описание справочника *Справочник терминов, использующихся в процессах*.

В классе *БизнесМодель.usr_Термины* добавить параметр *usr_Описание.* На закладке «Опции» выбрать тип параметра *Простой*, выбрать тип свойства *Текст*. На закладке «Доп. опции» ввести опцию *Показ.Заголовок* – значение *Описание*. Изменить отображение параметра в окне свойств таким образом, чтобы он был показан в виде многострочного текстового поля с просмотром содержимого, для этого в свойствах параметра на закладе «Доп. опции» добавить опцию *Редактирование.ЭдиторОб* – значение *MemoEdit*. На закладке «Описание» ввести *Текстовое описание термина*.

Теперь необходимо создать класс списка, в котором буду храниться собственно экземпляры списка *Термины* процессов.

Для этого выделить в дереве *Элементы списков*| *БизнесМодель.СпискиБизнесМоделей*, добавить от него класс *БизнесМодель.usr_СписокТерминов*. На закладке «Описание» дать описание списка *Список терминов, использующихся в процессе*. Добавить параметр *usr_Термин*, на закладке «Опции» выбрать тип параметра *Объектный*, указать созданный класс *БизнесМодель.usr_Термины*. На закладке «Доп. опции» добавить опцию *Показ.Заголовок* – значение *Термин*. На закладке «Описание» ввести описание *Термин, используемый в процессе*.

Осталось создать собственно параметр-список, который будет отображаться на закладке в окне свойств процесса.

В класс *БизнесМодель.Процессы* добавить параметр *usr_ТерминыПроцесса*. Выбрать тип параметра *Список*, указать созданный список *БизнесМодель.usr_СписокТерминов*. На закладке «Доп. опции» задать опцию *Показ.Заголовок* – значение *Термины*. Чтобы вновь созданный список по умолчанию отображался на вкладке в окне свойств процесса на закладке «Доп. Опции» добавить опцию *Показ.НаВкладке* – значение *Да*.

Редактирование метаданных закончено, теперь необходимо применить их к базе данных (см. п. 2.2.3).

В базе данных дать пользователю права на доступ к новому классу *Классы*| *Объекты системы*| *Термины* и классу списка *Элементы списков*| *БизнесМодель.СпискиБизнесМоделей*| *usr_СписокТерминов*. Для этого переключиться в интерфейс администратора системы (см. главу 16 Администрирование системы в «Руководстве пользователя»), выбрать пункт главного меню «Администрирование –> Справочники администрирования». Открыть справочник «Категории прав», выделить нужную категорию прав, нажать гиперссылку *Редактирование прав*. В открывшемся окне «Права» для новых классов установить опции «Разрешить» для прав *Доступ, Изменение, Создание* и *Удаление*.

4.4 Длина в байтах для различных типов параметров

Тип параметра	Длина в байтах
Логический	1
Целый	4
Строка [Длина]	Длина

Тип параметра	Длина в байтах	
Вещественный [Длина, Точность]	**Длина**	**Длина в байтах**
	1 - 9	5
	10-19	9
	20-28	13
ДатаВремя	8	
Изображение	16	
Бинарный	16	
Текст	16	
Объектный	31	
Список	0	
Структура	0	
Перечисление	4	

4.5 Пример скрипта создания резервных копий баз данных

Запуск скрипта с необходимой периодичностью осуществляется на сервере баз данных с помощью «Назначенных заданий» Windows. Учетная запись, под которой запускается скрипт, должна обладать всеми необходимыми разрешениями для выполнения операции резервного копирования на сервере SQL, а также для записи файлов в указанные папки. Учетная запись, под которой запускается скрипт, должна обладать всеми необходимыми разрешениями для выполнения операции резервного копирования на сервере SQL, а также для записи файлов в указанные папки.

Для работы скрипта необходимы:

- архиватор, в примере используется «WinRar»;
- утилита для удаления файлов, в примере используется «Forfiles.exe» из Windows Resource Kit;
- файл «Bases.txt» со списком имен баз данных. Имя каждой базы записывается с новой строки.

Скрипт запускается непосредственно на SQL Server'e, имя инстанции SQL Server указывается в переменной %SQLSERVER%. Код SQL запускается с помощью утилиты «SQL Query Tool», путь к утилите указывается в переменной %ISQL%, например «C:\Program Files\Microsoft SQL Server\80\Tools\Binn\OSQL.EXE».

Скрипт работает по следующей схеме:

5. Создается резервная копия, имя файла при этом формируется следующим образом: «Имя_базы_ДД_ММ_ГГГГ.db». База сохраняется локально, путь к базе данных указывается в переменной %BACKUP%. Создаются лог-файлы резервного копирования для каждой базы в виде «log_autobackupsql_<Имя_базы>.txt» и общий лог-файл «backup_log.txt», путь к папке для хранения лог-файлов указывается в переменной %LOG%.
6. Созданный файл запаковывается архиватором. Расположение архиватора указывается переменной %RAR%.

7. Созданный архив копируется на два указанных сетевых источника хранения архивов: переменные %PATH01% и %PATH02%. Если путь не указан, копирование архива не производится.
8. Архивы, созданные ранее определенного количества дней, удаляются. Утилита для удаления указывается в переменной %FORFILES%. Количество дней указывается в переменной %DAYS%.

Пример

```
@echo off
set SQLSERVER=<Имя_инстанции_SQL_Server>
set ISQL=<Имя_OSQL>
set BACKUP=<Путь_к_локальному_каталогу_резервных_копий>
set LOG=<Путь_к_папке_лог_файлов>
set RAR=%CD%\Rar.exe
set FORFILES=%CD%\Forfiles.exe
set DAYS=<Количество_дней>
Set PATH01=<Путь_сетевого_ресурса_1>
Set PATH02=<Путь_сетевого_ресурса_2>

rem Директория в которой будет создан SQL скрипт
set temp=%cd%
rem Считывание названий баз из файла Bases.txt
for /F "eol=; tokens=1 delims=," %%i in (Bases.txt) do (
set Database=%%i
call :BackUp)
goto :EOF

:BackUp
rem Проверки
IF NOT EXIST "%ISQL%" (echo Microsoft SQL Server Command Line Tool "%ISQL%" не найден!
echo Поправьте значение переменной ISQL в командном файле!
goto :EOF)
IF NOT EXIST "%RAR%" (echo Архиватор "%RAR%" не найден!
echo Поправьте значение переменной RAR в командном файле!
goto :EOF)
IF NOT EXIST "%FORFILES%" (echo Утилита "%FORFILES%" не найдена!
echo Поправьте значение переменной FORFILES в командном файле!
goto :EOF)
IF NOT EXIST "%BACKUP%" (echo Каталог бэкапов "%BACKUP%" не найден!
echo Поправьте значение переменной BACKUP в командном файле!
```

```
goto :EOF)
if NOT '%PATH01%'=='' (IF NOT EXIST "%PATH01%" (echo Каталог бэкапов "%PATH01%" не найден!
  echo Поправьте значение переменной PATH01 в командном файле!
  echo %date% %time% Поправьте значение переменной PATH01 в командном файле! >> %LOG%\backup_log.txt
  set PATH01=)
)
if NOT '%PATH02%'=='' (IF NOT EXIST "%PATH02%" (echo Каталог бэкапов "%PATH02%" не найден!
  echo Поправьте значение переменной PATH02 в командном файле!
  echo %date% %time% Поправьте значение переменной PATH02 в командном файле! >> %LOG%\backup_log.txt
  set PATH02=)
)

for /f "tokens=1-4 delims=. " %%i in ('date /t') do set longdate=%%i%%j%%k

set FileName=%longdate:~0,2%_%longdate:~2,2%_%longdate:~4,4%
rem дата в формате 2003_04_12
echo Дата: %FileName%
echo SQL Server: %SQLServer%
echo База данных: %Database%
echo.

echo 1. Создание резервной копии...
set ArcSQL=%temp%\arcsrv.sql
echo USE master > %ArcSQL%
echo EXEC sp_addumpdevice 'disk', '%Database%_Backup', '%BACKUP%\%Database%_%FileName%.db' >> %ArcSQL%
echo BACKUP DATABASE %DataBase% TO %Database%_Backup >> %ArcSQL%
echo exec sp_dropdevice '%Database%_Backup' >> %ArcSQL%
"%ISQL%"  -E -S %SQLServer% -d master -i %ArcSQL% -n -o %log%\log_autobackupsql_%Database%.txt
echo.

echo 2. Архивирование резервной копии...
cd "%BACKUP%"
"%RAR%" a "%BACKUP%\%Database%_%FileName%" "%BACKUP%\*%FileName%.db" -rr -c -ep -av -idp -id -idc
echo.

echo 3. Копирование резервных копий в сети...
if not '%PATH01%'=='' Call :CopyProc "%PATH01%"
if not '%PATH02%'=='' Call :CopyProc "%PATH02%"
echo.
```

```
echo 4. Лог результата копирования
set i=0
IF EXIST "%BACKUP%\%Database%_%FileName%.rar" (set /a i=%i%+1)
IF EXIST "%PATH01%\%Database%_%FileName%.rar" (set /a i=%i%+1)
IF EXIST "%PATH02%\%Database%_%FileName%.rar" (set /a i=%i%+1)
echo %Date% %time% - Backup %Database% завершен! Готово архивов: %i% из 3 >> %log%\backup_log.txt
echo.

echo Резервное копирование завершено

if /i %i% NEQ 0 (Call :Delete)
goto :EOF

:Delete
echo Удаление…
"%forfiles%"  -p"%Backup%"  -s -m*.rar -d-%DAYS% -c"CMD /C del @PATH\@FILE"
if not '%PATH01%'==" ("%forfiles%"  -p"%PATH01%"  -s -m*.rar -d-90 -c"CMD /C del @PATH\@FILE")
if not '%PATH02%'==" ("%forfiles%"  -p"%PATH02%"  -s -m*.rar -d-90 -c"CMD /C del @PATH\@FILE")
del "%BACKUP%\*.db"
goto :EOF

:CopyProc
echo Копирование... %1
copy "%BACKUP%\%Database%_%FileName%.rar" %1
goto :EOF

:EOF
```