Deductor

Руководство администратора

Версия 5.2

В руководстве описана аналитическая платформа Deductor 5.2: технические особенности платформы, установка, настройка и обслуживание системы, построенной на ее основе, разрешение проблем и ответы на основные вопросы, возникающие в ходе интеграции Deductor в информационную систему предприятия. Книга предназначена для администраторов и IT-специалистов, которые обслуживают информационную систему. Специальных знаний в области анализа данных не требуется, но предполагается, что читатель знаком с основами работы платформы Deductor и может работать с операционной системой на уровне администратора.

Содержание

Введение

В процессе внедрения платформы Deductor в разных компаниях часто возникают вопросы, касающиеся не столько различных аспектов анализа данных, сколько чисто технической стороны работы с платформой. Например, каким образом настроить хранилище данных для сетевого доступа, как работать с аппаратным ключом защиты от копирования, как повысить производительность системы и устранить возникающие ошибки. Решение этих вопросов обычно возлагается на IT-специалиста – администратора системы и не касается аналитика и, тем более, конечного пользователя.

Это Руководство планировалось как сборник ответов часто возникающие вопросы по установке и обслуживанию системы, призванный помочь администратору в развертывании аналитической платформы Deductor на предприятии. Вопросы анализа данных, построения сценариев, генерации отчетов, работы с визуализаторами подробно освещаются в документе «Руководстве аналитика». Здесь же акцент сделан на техническую сторону интеграции платформы в информационную систему предприятия. Руководство предназначено для IT-специалиста, уже знакомого с платформой Deductor, достаточно хорошо понимающего ее архитектуру и основные принципы работы.
Руководство в основном рассчитано на пользователей программ Deductor Professional и Deductor Enterprise, т.к. Deductor Academic сам по себе не нуждается в администрировании. Лишь некоторые разделы применимы и к версии Academic. В дальнейшем информация относится ко всем версиям программы, если явно не говорится об обратном.

Комплект поставки

Существует три типа варианта поставки платформы Deductor: Academic, Professional, Enterprise. В зависимости от типа поставки набор доступных компонентов может различаться.

Состав компонентов платформы в зависимости от варианта поставки

Компоненты	Academic	Professional	Enterprise
Программные файлы			
Deductor Viewer	–	+	+
Deductor Studio	+	+	+
Deductor Client			+
Deductor Server			+
Guardant			
Guardant Drivers		+	+
Guardant Server		+	+
Guardant Tools		+	+
Документация			
Примеры	+	+	+
Deductor OLE – описание			+
Deductor Client – описание			+
SDK			
C			+
Delphi			+
MS SQL			+
Oracle			+
PHP			+

Описание назначения и применения компонентов платформы

Компоненты	Описание
Программные файлы	Общие программные файлы, необходимые для функционирования системы
Deductor Viewer	Рабочее место конечного пользователя
Deductor Studio	Рабочее место аналитика
Deductor Client	Клиент удаленного доступа к аналитическому серверу Deductor Server
Deductor Server	Аналитический сервер
Guardant	Файлы программ для работы с электронным ключом защиты от копирования
Guardant Drivers	Файлы драйверов электронного ключа. Для работы программы с локальным ключом эти драйвера обязательно должны быть установлены. Для работы с сетевым ключом они не требуются (см. следующий раздел «Установка ключа»)
Guardant Server	Сервер лицензий для сетевых электронных ключей, для работы требуются наличие драйвера.
Guardant Tools	Программы диагностики электронных ключей
Документация	Файлы документации по работе с пакетом Deductor
Примеры	Файлы демонстрационных примеров
Deductor OLE – описание	Описание интерфейса доступа к Deductor через механизм OLE Automation
Deductor Client – описание	Описание интерфейса доступа к Deductor Server через Deductor Client
SDK	Файлы примеров работы с механизмом Client/Server
C	Файлы примеров для C
Delphi	Файлы примеров для Delphi
MS SQL	Файлы примеров для MS SQL
Oracle	Файлы примеров для Oracle
PHP	Файлы примеров для PHP

Для программ серии Professional и Enterprise основной отличительной особенностью является необходимость установки драйверов Guardant, которые нужны для работы с электронным ключом защиты программы. В случае использования локального ключа защиты драйвер должен быть установлен обязательно. При работе с сетевым ключом инсталляция драйверов производится только на тот компьютер, на который непосредственно устанавливается электронный ключ, для остальных рабочих мест установка драйверов не требуется (см. следующий раздел «Установка ключа»).

По окончании копирования файлов программа установки выведет окно «Завершение Мастера установки Deductor». В случае работы программы с локальным ключом защиты от копирования при первой установке программы на компьютер здесь следует вскинуть флажок **Установить драйверы Guardant**, чтобы были установлены драйверы ключа защиты от копирования. При снятом флажке драйвера можно будет установить позже самостоятельно. При повторной установке этого не требуется, так же как и при работе программы с сетевым ключом.

Примечание

Для установки электронного ключа в операционную систему требуется наличие необходимых административных прав.

Системные требования

Системные требования определяются в основном объемами обрабатываемой информации и сложностью производимых вычислений.
Для работы ряда алгоритмов используется внутренний кэш данных, по этой причине при работе с большими объемами данных основным требованием является наличие достаточного количества свободной оперативной памяти. Кроме того, оперативная память необходима для хранения проекта Deductor в памяти, размер которого при сложных сценариях может превышать десятки мегабайт.
Во время своей работы платформа Deductor старается минимизировать объем используемой памяти, и везде, где возможно получить данные путем вычислений, используются вычисления, т.е. вторым основным узким местом может являться производительность процессора. Наиболее серьезные требования к производительности процессора предъявляются при построение моделей с использованием различных алгоритмов Data Mining: нейронные сети, деревья решений, ассоциативные правила и прочее. В большинстве случаев данные алгоритмы достаточно ресурсоемки. Правда, нужно иметь в виду, что Data Mining алгоритмы работают в 2-х режимах: построение модели и использование модели. Ресурсоемким является этап построения, т.к. для этого производится огромное количество расчетов, использование же уже построенной модели обычно не требует высокой производительности процессора.
Deductor построен таким образом, что можно одновременно просматривать результаты при помощи десятка визуализаторов, а кроме того, еще и просмотреть все промежуточные результаты. Поэтому еще одним критическим местом особенно для аналитика может являться используемое разрешение экрана, т.к. очень часто требуется представить большой объем графической информации: кубы, карты, графики, диаграммы и т.д.
Минимальные требования позволяют запустить программу и работать с относительно небольшими объемами данных. Для полноценной работы лучше использовать более производительные компьютеры.

Аппаратное обеспечение

Deductor Studio и **Viewer** работают в интерактивном режиме.

Системные требования к рабочей станции

Показатель	Минимум	Желательно
Процессор	Pentium II 450 МГц	Xeon II 1 ГГц
Оперативная память	1 Гб	2 Гб
Жесткий диск	200 Мб	1 Гб
Монитор	15"	19"
Видеокарта	800x600	1280x1024
Порты	USB 1.0	USB 2.0
Управление	Клавиатура, мышь	Клавиатура, мышь

Deductor Server функционирует в виде Windows службы. Он может быть установлен на рабочей станции, так же как и Studio, но желательно предоставить для его работы выделенный сервер.

Системные требования к аналитическому серверу

Показатель	Минимум	Желательно
Процессор	Xeon II 1 ГГц	Xeon II 3 ГГц
Оперативная память	1 Гб	4 Гб
Жесткий диск	IDE 1 Гб	RAID 10 Гб
Монитор	15"	15"
Видеокарта	800x600	800x600
Порты	USB 2.0	USB 2.0
Управление	Клавиатура, мышь	Клавиатура, мышь

Deductor Warehouse является специализированной надстройкой над реляционной базой данных, включающей семантический слой и реализующей механизмы импорта из СУБД.

В качестве платформы может выступать 3 базы данных: Firebird, MS SQL, Oracle. Желательно использовать для хранения и работы Warehouse выделенный сервер. Системные требования к серверу баз данных в значительной степени зависят от используемой СУБД.

Системные требования к серверу с СУБД

Показатель	Минимум	Желательно
Процессор	Xeon II 1 ГГц	Xeon II 3 ГГц
Оперативная память	1 Гб	4 Гб
Жесткий диск	RAID 10 Гб	RAID 100 Гб
Монитор	15"	17"
Видеокарта	800x600	1024x768

Программное обеспечение

Deductor может работать как в локальном режиме, так и в сетевом. Теоретически можно все модули системы запустить на одном компьютере, но все-таки желательно использовать несколько серверов. Работа на единственной рабочей станции может быть оправдана в случае обработки небольшого объема данных.

Системные требования к рабочей станции

Показатель	Минимум	Желательно
Операционная система	Windows XP	Windows XP, 2000, 2003
Необходимое ПО	MS Internet Explorer 6.0, ADO Drivers	MS Internet Explorer 6.0, ODBC Drivers, ADO Drivers, MS Office, Firebird 1.5, клиенты доступа к используемым базам данных

Системные требования к аналитическому серверу

Показатель	Минимум	Желательно

Операционная система	Windows 2000	Windows 2000, 2003
Необходимое ПО	MS Internet Explorer 6.0, ADO Drivers; драйверы Guardant	MS Internet Explorer 6.0, ODBC Drivers, ADO Drivers, MS Office, клиенты доступа к используемым базам данных; драйверы Guardant

Системные требования к **серверу баз данных**.

Требования к серверу баз данных для размещения хранилища данных в зависимости от используемой платформы сервера базы данных:

- СУБД **Firebird 1.5**, которое может быть использовано в качестве платформы Deductor Warehouse, является свободным ПО, основанным на открытых исходных текстах Borland Interbase 6.0. Firebird поддерживает платформы Linux i386, Windows (Win32), Solaris Sparc and i386, FreeBSD, MacOS X and HP-UX. Он совместим со стандартом SQL'92, не требует постоянного администрирования, прост в установке и сопровождении и, наконец, бесплатен и доступен в полных исходных текстах. Официальный сайт Firebird на английском языке http://www.firebirdsql.org. При работе с локальным хранилищем данных используется специальная версия Firebird 1.5 Embedded Server, которая входит в комплект поставки Deductor. Много полезной информации на русском языке по данной СУБД можно найти http://www.ibase.ru.
- СУБД **Microsoft SQL 2000, 2005**, которая может быть использована в качестве платформы Deductor Warehouse (поддерживается только в версии Deductor Enterprise), разработана компанией Microsoft. MS SQL может быть развернут только на платформе Windows. MS SQL поддерживает платформы Windows 2000, 2003, надмножество языка SQL (Transact-SQL), то есть полноценный SQL-92 со своими собственными расширениями. Существует урезанная версия MS SQL Server — MSDE (Microsoft SQL Server Desktop Engine), распространяемая с такими продуктами, как Visual Studio, Visual FoxPro, Microsoft Access и другими. MSDE имеет ряд ограничений: размер базы данных ограничен в 2Гб, отсутствуют графические инструменты администрирования. Важной особенностью MSDE является отсутствие необходимости приобретения лицензии конечным пользователем и возможность распространения вместе с использующим её ПО. Официальный сайт производителя – http://www.microsoft.com. Дополнительную информацию о данной СУБД можно получить http://www.microsoft.com/Rus/Sql/Default.mspx.
- СУБД **Oracle Database9i-10g** может использоваться в качестве платформы Deductor Warehouse (поддерживается только в версии Deductor Enterprise). СУБД поддерживает множество операционных систем: Windows, Linux, Solaris, HP-UX, AIX и другие. Oracle Database поставляется в нескольких редакциях, ориентированных на различные сценарии разработки и развертывания приложений. Все редакции содержат общий набор функций для разработки приложений, в том числе объектно-реляционные возможности SQL, программные интерфейсы PL/SQL и Java, предназначенные для написания хранимых процедур и триггеров. Приложения, написанные для любой из этих редакций, будут работать и с остальными редакциями. Все программные продукты Oracle Database созданы на базе единой надежной архитектуры ядра СУБД. Эта СУБД может быть эффективно развернута на любой платформе, начиная с небольших blade-серверов и заканчивая крупнейшими симметричными многопроцессорными серверами и многоузловыми кластерами любых размеров. Официальный сайт производителя – http://www.oracle.com.

Порядок установки

Для установки Deductor следует запустить программу установки и следовать в дальнейшем ее инструкциям.

На странице «Выбор компонентов» программы установки предоставляется выбор, какой набор компонентов пакета Deductor необходимо установить на компьютер. В выпадающем списке можно выбрать предопределенные конфигурации установки платформы, и программа установки сама предложит нужный набор компонентов. При желании необходимый состав компонентов можно указать самостоятельно.

Deductor Studio – рабочее место аналитика

Предопределенная конфигурация модулей платформы для развертывания рабочего места аналитика. Устанавливается программа разработки сценариев Deductor Studio, регистрируется расширение *.ded как файлы сценариев Deductor Studio. При использовании локального электронного ключа необходима установка драйверов Gurdant. Порядок установки драйверов Guardant описан ниже в разделе «Драйвер электронного ключа».

Deductor Viewer – рабочее место конечного пользователя

Предопределенная конфигурация компонентов для развертывания рабочего места конечного пользователя. Устанавливается программа просмотра отчетов сценариев Deductor Viewer, регистрируется расширение *.ded как файлы сценариев Deductor Viewer. При использовании локального электронного ключа необходима установка драйверов Gurdant. Порядок установки драйверов Guardant описан ниже в разделе «Драйвер электронного ключа»

Deductor Warehouse – хранилище данных

Хранилище данных является неотъемлемой частью платформы Deductor, и все основные механизмы работы с ним встроены в платформу. Для создания нового локального хранилища данных или SQL-скрипта с текстом создания структуры хранилища данных на SQL сервере необходимо наличие файла BG_Warehouse.dll в папке с программой. Создать файл локального Хранилища Данных или SQL-скрипт с текстом создания структуры хранилища данных можно только в Deductor Studio. Этот файл устанавливается вместе с Deductor Strudio.

Deductor Server – аналитический сервер

Предопределенная конфигурация компонентов для развертывания аналитического сервера Deductor Server. Для его работы необходимо наличие управляющей программы DServer.exe и аналитического приложения DStudio. Deductor Server работает только при наличии установленного локального электронного ключа, поэтому так необходимо наличие установленных драйверов электронного ключа. Порядок установки драйверов Guardant описан ниже в разделе «Драйвер электронного ключа».

Структура каталогов установленного пакета Deductor

$(Deductor)	Корневой каталог установки
Bin	Исполняемые файлы программ
UDF	Файлы библиотек для локального хранилища данных
Guardant	Папка с файлами для работы с электронным ключом
Drivers	Файлы драйверов электронного ключа
Server	Сервер лицензий Guardant Net
Tools	Утилиты проверки электронного ключа
Manual	Файлы руководства по пакету Deductor
SDK	Набор инструментальных средств разработки
Samples	Файлы демонстрационных примеров по пакету Deductor

где $(Deductor) – каталог установки программы; по умолчанию – C:\Program Files\BaseGroup\Deductor);

Далее при указании путей **$(Deductor)** соответствует папке, в которую был установлен пакет Deductor.

Электронный ключ защиты программы

Программы Deductor серии Professional и Enterprise работают только с установленным в системе аппаратным USB-ключом Guardant. В отличие от них академическая версия Deductor Academic работает без использования средств защиты от копирования, но предназначена для некоммерческого использования, в ней присутствует только интерактивный режим построения сценариев и импорт из двух источников данных.

После установки программ серии Professional и Enterprise дополнительно потребуется настроить работу с электронным ключом защиты от копирования.
Существуют два вида ключей – локальный и сетевой. Локальный ключ устанавливается на том же компьютере, что и Deductor, и работать с ним можно только с этой рабочей станции. Сетевой ключ устанавливается на сервере, и к нему могут подключаться несколько пользователей одновременно (количество пользователей ограничивается типом приобретаемой лицензии).
При каждом запуске Deductor пытается найти доступный электронный ключ. Процесс поиска ключа состоит из следующих шагов. Сначала производится поиск локального ключа. Если локальный ключ отсутствует, начинается поиск сетевого ключа. Клиент Guardant на локальном компьютере отправляет широковещательный запрос по сети, пытаясь отыскать сервер Guardant. Сервер, получая запрос, отправляет свой ответ клиенту, и далее они совместно определяют параметры устанавливаемого соединения. Поиск сервера может занять немало времени. В случае, если ключ на сервере оказывается недоступным, клиент некоторое время ожидает ответа, и только после этого получает сообщение об отсутствии ключа.
Более подробно о порядке установки электронного ключа Guardant можно ознакомиться в документе, который поставляется с Deductor – «Электронные ключи Guardant. Инструкция по эксплуатации».

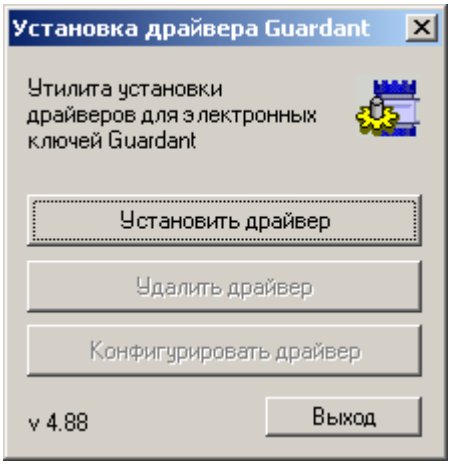
Драйвер электронного ключа

Для работы системы с электронным ключом необходимо наличие установленных драйверов электронного ключа в операционной системе. Для установки файлов драйверов в программе инсталляции должен быть отмечен пункт «Guardant Drivers» , для автоматической установки драйверов электронного ключа в систему после завершения процесса инсталляции на странице Мастера установки, появляющейся после окончания копирования файлов, необходимо установить флажок «Установить драйверы Guardant». После этого ключ следует установить в любой свободный порт USB, и программа полностью готова к работе. Драйвера для ключа будут загружаться автоматически каждый раз, когда ключ будет присоединяться к порту.

Для самостоятельной установки драйверов в систему необходимо запустить файл **INSTDRV.EXE**

$(Deductor)\Guardant\Drivers\instdrv.exe

и выбрать кнопку **Установить драйвер**.



Примечание

Для установки драйверов в операционную систему необходимо наличие административных прав.

Если по каким-то причинам драйверы отсутствуют, их последнюю версию можно загрузить с веб-сайта компании «Актив» – производителя электронных ключей Guardant – http://www.guardant.ru/hotline/download/drivers.

USB-ключ следует подсоединять к порту только после установки драйвера Guardant. Если ключ был подсоединен до установки драйвера и запустился стандартный Мастер установки USB-устройств Windows, то необходимо извлечь ключ из порта и отменить работу Мастера. В случае, если это не будет сделано, то USB-ключ появится в списке установленного оборудования, но не сможет нормально работать, т.к. операционная система не располагает драйверами для его корректной работы. В таком случае *наиболее простой вариант решения проблемы* следующий:

1. отсоединить ключ от компьютера;
2. удалить USB-ключ Guardant из списка оборудования Windows;
3. инсталлировать драйвер ключа;
4. вставить ключ в USB порт.

Windows автоматически запустит инсталляцию нового оборудования, т. к. драйвер уже присутствует, то установка будет проведена корректно.

Локальный ключ

Локальный ключ позволяет работать с пакетом программ Deductor только на рабочей станции, на которой он установлен. Электронный ключ позволяет запускать на этой станции произвольное количество копий программ Deductor, для которых есть лицензия. Для работы с локальным электронным ключом необходимо наличие установленных драйверов в операционной системе.

Примечание

Для установки электронного ключа в операционную систему требуется наличие необходимых административных прав.

Сетевой ключ

Для работы с сетевым ключом необходимо на одном из компьютеров сети с установленным сетевым электронным ключом запустить сервер лицензий Guardant Net. После этого рабочие станции получают доступ к сетевому ключу. Алгоритм работы программы с сетевым ключом несколько сложнее, чем с локальным. Для работы электронного ключа на станции, на которой запускается сервер лицензий Guardant Net, необходимо наличие установленных драйверов в

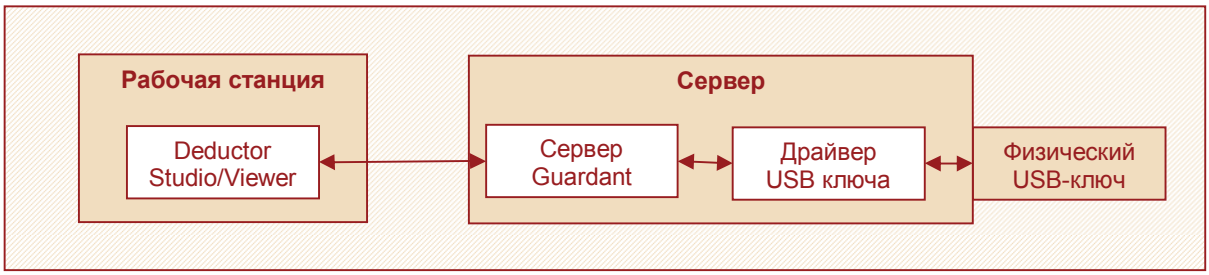
операционной системе. В следующем разделе особенности настройки сетевого ключа обсуждаются более подробно.

Сервер лицензий

Защищенное приложение обращается к сетевому электронному ключу через специальный программный сервер Guardant Net. Программа сервера находится в файле nnksrv32.exe дистрибутива сервера.

$(Deductor)\Guardant\Server\nnksrv32.exe.

Сервер лицензий Guardant Net должен быть загружен на том же компьютере, к которому подсоединен сетевой электронный ключ. В пределах локальной сети можно запускать несколько серверов Guardant Net. Они должны находиться на разных компьютерах и обладать уникальными NetBIOS-именами.



Сервер Guardant Net может работать в двух вариантах: как обычное оконное приложение и как сервис (служба) Windows NT/2000/XP.

Для того чтобы разрешить запуск сервера Guardant Net в качестве сервиса, нужно задать значение **On** в параметре **ServiceMode** файла **NNKSRV32.INI**. После этого сервер может быть запущен и как обычное приложение, и как сервис. В противном случае утилита **NNKSRV32.EXE** сможет работать только как обычное оконное приложение. Для установки сервиса нужно запустить сервер Guardant Net, файл **NNKSRV32.EXE** с опцией /i или /ii. (/i– установка службы Guardant Net без взаимодействия с рабочим столом; /ii– установка службы Guardant Net с возможностью взаимодействия с рабочим столом).

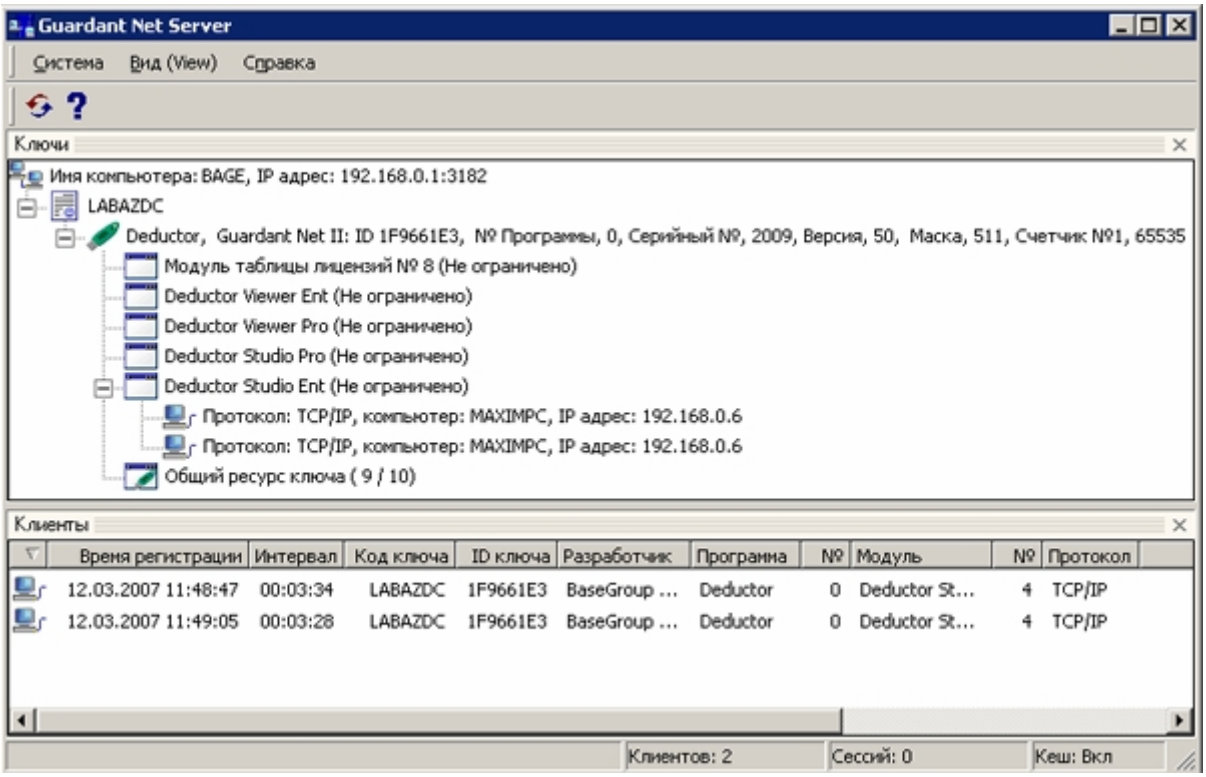
Установку сервиса (службы) Guardant Net нужно произвести только один раз. После того, как сервис Guardant Net будет успешно запущен, защищенные приложения получат доступ к ключам Guardant Net. Сервис будет запускаться автоматически при каждом старте Windows NT/2000/XP.

Примечание

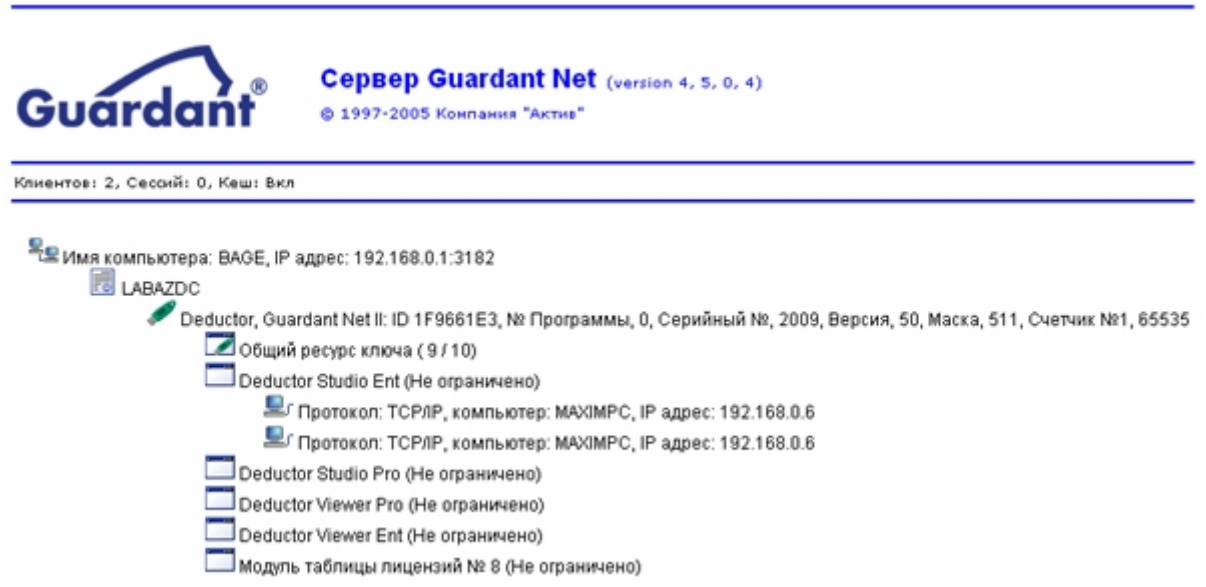
Для регистрации и запуска Guardant Net в виде службы требуется наличие необходимых прав на выполнение данных операций.

Наблюдать за состоянием сервера Guardant Net и использованием лицензий можно двумя способами.

1. Если сервер запущен как обычное приложение с оконным интерфейсом или как интерактивный сервис (служба), то информацию о сервере можно просмотреть в окне.



2 Если сервер запущен в качестве сервиса (службы) Windows NT/2000/XP/2003 без оконного интерфейса или если информацию о сервере нужно получить с удаленного компьютера, то можно воспользоваться web-интерфейсом, открыв в браузере URL http://[IP/hostname сервера]:[порт сервера]. Если в файле **NNKSRV32.INI** http-порт установлен по умолчанию, то номер порта 3185.



Более подробно о параметрах сервера Guardant Net можно ознакомиться в «Руководстве администратора Guardant Net».

Настройка доступа к сетевому ключу

В большинстве случаев для доступа к сетевому ключу не требуется проводить дополнительную настройку. Основным требованием для использования сетевого ключа является доступность

рабочей станции с электронным ключом и запущенным сервером лицензий для остальных станций сети, а также открытость необходимых портов при использование различных брандмауэров (по умолчанию используются порты 3182-3184).

Оптимизация

Для того чтобы ускорить процесс поиска сетевого ключа, необходимо немного изменить настройки клиента Guardant, используемые по умолчанию. Для этого в текстовом редакторе следует открыть файл $(Deductor)\Bin\Gnclient.ini из каталога установки Deductor.

В этом файле можно изменить следующие параметры:

- [PROTOCOLS] – используемый протокол для взаимодействия с сервером Guardant: в разделе настраиваются доступные протоколы (TCP/IP и NetBIOS). Приоритетом обладает протокол с меньшим номером. По умолчанию первым используется протокол TCP/IP, лучше по возможности использовать только его, а протокол NetBIOS отключить.
- [TIMEOUT] – в разделе находятся тайм-ауты для различных событий, связанных с взаимодействием клиента и сервера: тайм-аут запроса, тайм-аут ответа и тайм-аут широковещательного запроса. Изменять их следует на свой страх и риск, так как при загруженной сети пакеты в пути между клиентом и сервером могут находиться достаточно долго.
- [SERVER] - в разделе находится информация о сервере Guardant, на основании которой клиент пытается установить соединение. Для взаимодействия клиенту и серверу требуются один порт TCP и два порта UDP. По умолчанию TCP_PORT=3182; UDP_PORT_CLIENT=3183; UDP_PORT_SERVER=3184, причем номера соответствующих портов у клиента и сервера должны совпадать. Изменять значения по умолчанию имеет смысл, только если во внутренней сети эти порты уже используются другими службами. В этом же разделе находятся поля для задания IP-адреса сервера Guardant и широковещательного адреса. Здесь же указывается NetBIOS-имя сервера и включается или отключается широковещательный поиск.
- Чтобы отказаться от использования широковещательного запроса при поиске сетевого ключа, достаточно указать IP-адрес сервера и установить параметр SEARCH в значение OFF (SEARCH=OFF). Теперь клиент Guardant будет искать ключ только на указанном сервере. Аналогично можно поступить при использовании протокола NetBIOS, а именно параметру NB_NAME присвоить NetBIOS имя сервера и отключить широковещательный поиск. Нужно отметить, что в сети не должно быть двух серверов Guardant с одинаковыми именами NetBIOS, иначе работать совместно они не смогут.

Кроме того, можно заменить широковещательный адрес другим, например, групповым. В этом случае запрос при поиске сервера будет отправлен только на некоторые узлы сети.

Необходимо отметить, что часто проблемы при настройке сетевых ключей возникают из-за установленного брандмауэра (файрвола). В этом случае Deductor не будет загружаться, выдавая через некоторое время после попытки запуска сообщение об отсутствии ключа. Кроме того, могут появляться окна брандмауэра с сообщениями о попытках доступа к портам, используемым Guardant, или появляться соответствующие записи в логе. В любом случае при установленном брандмауэре нужно проследить за тем, чтобы порты взаимодействия между клиентом и сервером были открыты на обоих компьютерах. В настройках брандмауэра можно указать приложения, которым разрешен доступ в сеть по указанным портам. На компьютере-клиенте это должен быть Deductor Studio или Viewer (**DStudio.exe** и **DViewer.exe**), на сервере – сервер Guardant Net. (**NNKSRV32.EXE**)

Еще одна особенность сетевых ключей состоит в том, что они могут поддерживать ограниченное число сессий (максимальное число определяется типом лицензии). Бывают ситуации, когда в результате некорректного завершения работы клиента на рабочей станции, сессии на сервере не закрываются, т. е. сервер считает, что к нему присоединен какой-то клиент, хотя в реальности это соединение уже не существует. В результате может наступить момент, когда все сессии ключа окажутся исчерпанными и соединение с ним установить не удастся.

Если число текущих сессий равно максимально возможному, то открыть новое подключение к серверу Guardant будет невозможно, т.е. невозможно будет запустить приложения Deductor.

Чтобы не дожидаться, пока сервер сам автоматически отключит неиспользуемые сессии, в такой ситуации можно перезапустить службу сервера. Для этого следует щелчком мыши на значке сервера Guardant в системном трее вызвать всплывающее меню и выбрать пункт «Перезапустить сервер». После этого все сессии ключа будут завершены.

Замечание

При перезапуске сервера все реальные открытые сессии будут принудительно закрыты, так что предварительно следует убедиться, не работают ли в это время с Deductor другие пользователи.

Обновление прошивки электронного ключа

Для изменения количества и состава лицензий или для перехода на новую версию платформы Deductor необходимо произвести обновление прошивки электронного ключа. Подробно механизм обновления прошивки электронного ключа описан в документе «Порядок обновления электронного ключа».

Замечание

При использовании сетевого электронного ключа после процедуры обновления прошивки необходимо перезапустить программу или сервис сервера лицензий Guardant Net.

Deductor Studio – рабочее место аналитика

Аналитическое приложение для разработки и выполнения сценариев обработки данных. Основным его пользователем является аналитик. Сценарии, подготовленные в Deductor Studio, используются в последствии в Deductor Viewer и Deductor Server.

Конфигурационные файлы

Deductor Studio кроме основного используемого файла сценариев (***.ded**) имеет несколько конфигурационных файлов, хранящаяся в них информация влияет на процесс обработки.

- **Connections.sys** – файл с параметрами подключений. Настройка подключений происходит на панели «Подключения». Эта информация не привязана к конкретному сценарию, а относится ко всем проектам сразу.
- **Enveroment.sys** – файл с переменными приложения и настройками используемых механизмов обработки. Переменные приложения не относятся к конкретному проекту, а к приложению в целом. Их настройка производится в окне **Переменные**, закладка **Приложение**, которое вызывается из пункта меню **Сервис ► Переменные…**

По умолчанию эти файлы располагаются в папке с программой ($(Deductor)\Bin).

Можно использовать общие файлы настроек, расположив их на общих доступных на сетевых дисках. Расположение и название файлов можно изменить, указав новый путь в окне настроек программы, пункт меню **Сервис ►Настройка**. После этого путь к файлу сохраняется в реестре в разделе HKEY_CURRENT_USER\Software\BaseGroup Labs\Deductor\Common.

При переносе сценариев на другую рабочую станцию необходимо учитывать, что для корректной работы сценариев могут потребоваться и эти файлы.

Пакетная обработка

Созданные аналитиком сценарии анализа данных могут требоваться не только конечному пользователю для просмотра отчетов, но и другим программам, например, для промежуточной обработки данных. Этот способ использования Deductor является на практике очень важным, и он становится доступным благодаря механизму пакетной обработки, т.е. запуску Deductor Studio в автономном режиме.

Deductor Studio может принимать ряд параметров, переданных с использованием командной строки:

$(Deductor)\Bin\DStudio [<файл-проекта>] [параметры].

Здесь:

- $(Deductor) – каталог установки программы (по умолчанию «C:\Program Files\BaseGroup\Deductor»);
- [<файл проекта>] – имя файла проекта с полным путем;
- [параметры] – дополнительные параметры запуска, начинающиеся с символа «/».

Расшифровка дополнительных параметров в таблице ниже.

Параметр	Назначение
/HELP или /?	Выдать справку по синтаксису командной строки
/RUN	Выполнить сценарии проекта в пакетном режиме
/TEACH	Переобучить модели проекта в пакетном режиме
/SYSFILE=<sys-файл>	Загрузить сведения о подключениях из файла <sys-файл>
/LOG	Подробно фиксировать в журнале регистрации (лог-файле) выполнение каждого узла
/LOGFILE=<log-файл>	Указать имя файла для журнала регистрации (лог-файла)
/LOGMODE[=OVERWRITE]	Указать режим работы журнала регистрации (добавления/перезаписи), если указано OVERWRITE – перезаписывать файл, иначе добавлять в конец
/VARFILE =<var-файл>	Загрузить значения переменных из файла <var-файл>

Примеры.

"C:\Program Files\BaseGroup\Deductor\Bin\DStudio.exe" "D:\Мои документы\demo.ded" /RUN /SYSFILE=c:\connections.sys

Выполнить сценарии проекта <D:\Мои документы\demo.ded> в пакетном режиме. Параметры приложения взять из файла <c:\connections.sys>.

DStudio c:\demo.ded /TEACH /LOG /LOGFILE=c:\demo.log /LOGMODE

Переобучить модели проекта <c:\demo.ded> в пакетном режиме. Подробно фиксировать в журнале регистрации выполнение каждого узла. Для ведения журнала регистрации использовать файл <c:\demo.log>. Добавлять новые записи в конец журнала регистрации.

При использовании пакетной обработки Deductor Studio может работать в двух режимах: пакетного выполнения (*/run*) и пакетного обучения (*/teach*).

Пакетное выполнение

Под пакетным выполнением понимается запуск Deductor с помощью параметра командной строки */run*. В этом случае в автоматическом режиме будет выполнен указанный сценарий, и все результаты работы будут выгружены во внешние приемники данных. Целью такого режима работы является получение результатов анализа во внешнем источнике данных. Пакетное выполнение широко применяется при автоматической инкрементной загрузке данных в хранилище, генерации отчетов во внешних программах (например, публикация в Web), промежуточной обработке данных.

Пакетное выполнение сценария запускается с помощью следующей командной строки:

$(Deductor)\Bin\DStudio.exe <файл проекта> /run.

В результате Studio выполнит загрузку, обработку всех данных и выгрузит в узлах экспорта результаты в настроенные приемники данных, а затем завершит работу.

Пакетное выполнение может быть настроено на запуск по расписанию с помощью планировщика заданий, например, стандартного Windows Scheduler. Такая возможность удобна для автоматического запуска процесса загрузки в хранилище данных из учетной системы в ночное время. Для этого создается ярлык для файла **DStudio.exe**, для которого в строке «Объект» вводится командная строка запуска Deductor Studio в пакетном режиме. Затем в Windows Scheduler настраивается время запуска этого задания.

Deductor Studio может быть использован для проведения промежуточных расчетов, помещая результаты их выполнения в согласованный приемник данных. В этом случае из внешней

программы, управляющей взаимодействием, запускается в пакетном режиме Deductor с нужным сценарием обработки. По окончании выполнения сценария Deductor Studio возвращает в случае успеха вызывающей программе нулевой код завершения работы, и она может использовать результаты обработки по своему усмотрению. Не нулевое возвращаемое значение говорит об ошибке, возникшей в процессе обработки. Дополнительную информацию о возникших проблемах можно узнать из лог-файла (см. раздел «Диагностика пакетной обработки»).

В Deductor существует возможность задавать для выполнения в пакетном режиме лишь часть узлов. Регулируется это при разработке сценария с помощью флага **Выполнить** пункта всплывающего меню **Статус пакетной обработки**. Если флаг **Выполнить** сброшен, узел и все его потомки исключаются из пакетного выполнения. Это удобно в тех случаях, когда в проекте находится несколько веток, часть из которых используется в ходе разработки и настройки сценария и которые могут пригодиться в будущем. При установленном флаге **Выполнить** узел будет выполнен только в том случае, если у всех его родителей этот флаг также установлен. По умолчанию при добавлении нового узла в сценарий флаг **Выполнить** включен.

Пакетное обучение

В ситуациях, когда модель перестала быть адекватной задаче, существует возможность избежать построения нового сценария с нуля за счет того, что в некоторых обработчиках Deductor используются самообучающиеся алгоритмы – нейронные сети, самоорганизующиеся карты, деревья решений и прочие. Потребность в переобучении может возникнуть еще, если модель адекватно отражает предметную область, но исходные данные просто выходят за пределы выборки, на которой обучалась нейронная сеть или другой алгоритм. Чтобы модель учла новую информацию и могла работать с изменившимися данными, ее нужно переобучить. Сделать это можно двумя способами: вручную, путем перенастройки узлов сценария, и автоматически, путем пакетного обучения. Преимущество ручного переобучения состоит в возможности контроля параметров переобучения модели и просмотра полученных результатов. В то же время автоматический способ гораздо быстрее и хорошо подходит при незначительных изменениях в исходных данных.

Пакетное обучение сценария осуществляется запуском Deductor Studio с параметром */teach:*

$(Deductor)\Bin\DStudio.exe <файл проекта> /teach.

В результате Deductor выполнит загрузку и предобработку всех данных, как это предусмотрено сценарием, переобучение моделей, встречающихся в узлах проекта, сохранит модифицированный проект, а затем завершит свою работу. При следующем выполнении сценария будет работать обновленная модель.

В Deductor существует возможность обучать лишь часть узлов при переобучении. Регулируется это при разработке сценария с помощью флага **Переобучить** пункта всплывающего меню **Статус пакетной обработки**. Если флаг у узла сброшен, он не будет переобучаться при запуске сценария с параметром */teach* . Такая возможность может потребоваться, когда некоторые узлы проекта должны оставаться неизменными или требуют исключительно ручного переобучения. По умолчанию при добавлении нового узла в сценарий флаг **Переобучить** включен.

Оптимизация пакетной обработки

Оптимизировать пакетное выполнение можно различными способами:

- Изменение логики работы сценария, например, кэширование данных, расчет промежуточных результатов и прочее.
- Использование «быстрых» источников данных. Наиболее производительными являются промышленные СУБД, из которых информация получается при помощи драйверов прямого доступа. Можно использовать предварительно подготовленные данные в таблицах и материализованных представлениях.
- Выполнение проекта специально сохраненного для пакетного выполнения. Для этого нужно вызвать в Deductor Studio пункт меню **Файл ► Сохранить для пакетной**

обработки... В такой проект будет сохранена только информация об обработчиках, и не будет сведений о визуализаторах, которые не используются для пакетной обработки.

- Применять Deductor Server для пакетной обработки и обучения. В Deductor Server реализованы различные механизмы, позволяющие увеличить производительность обработки.
- В сценарии Deductor использовать узел «Сценарий Deductor» для многократного вызова других сценариев через механизм OLE Automation. Этим можно сэкономить время, затрачиваемое каждый раз на загрузку приложения Deductor и сценария в оперативную память.

Диагностика пакетной обработки

Пакетная обработка не требует вмешательства человека. В этом кроме достоинств есть и недостаток, состоящий в невозможности непосредственно контролировать ход обработки. Для того чтобы можно было диагностировать ошибки и контролировать процесс выполнения в режиме пакетной обработки, используются лог-файлы – журналы регистрации операций.

При каждом выполнении Deductor создает лог-файл с информацией о ходе работы. В интерактивном режиме использование лог-файла имеет смысл обычно только для отправки разработчикам сообщений об ошибках приложения (см. раздел «Что делать при возникновении ошибок»). В пакетном же режиме это незаменимый инструмент для выявления ошибок в данных, сценариях, оценки производительности и поиска «узких» мест. По умолчанию в лог-файл пишется только информация о начале и завершении пакетной обработки, а также о возникших в ее процессе ошибках. Однако существует набор параметров командной строки, позволяющих изменить режим создания лога:

- /LOG – включить подробный режим лог-файла, в логе будет сохраняться информация о времени начала и окончания обработки каждого узла;
- /LOGFILE=<Имя файла> – указывает имя лог-файла, по умолчанию оно генерируется автоматически;
- /LOGMODE=OVERWRITE – изменяет режим работы с логом на перезапись, по умолчанию используется режим добавления записей в лог.

Если эти параметры управления логом не указаны, по умолчанию используются настройки, сделанные в окне **Сервис ► Настройка** приложения Deductor Studio.

Параметр /LOG позволяет проследить последовательность выполнения узлов (например, для локализации ошибки) и оценить время выполнения каждого узла. Оценка производительности необходима для оптимизации сценария обработки. Сценарий обработки может готовиться на некотором небольшом подмножестве рабочего набора данных, а в последствии обрабатывать большие объемы данных. В этом случае встает вопрос оценки эффективности работы сценария на рабочей выборке. Для данных большого объема, которые обрабатываются длительное время, оценивать вручную время выполнения каждого узла слишком трудоемко. В этом случае можно использовать подробный режим лога и по нему судить о критичных для производительности узлах и принимать решения по оптимизации сценария.

С помощью параметров /LOGFILE и /LOGMODE можно изменять режим создания лог-файлов: создавать при каждом выполнении новый файл, дописывать информацию в один файл или перезаписывать его.

Интеграция

Deductor практически всегда используется не как отдельное приложение, а как часть единой информационной системы предприятия. Поэтому одна из задач, которые должны решаться при установке комплекса Deductor, заключается в обеспечении интеграции с другими системами и программами.

Интеграция со сторонними системам предполагает возможность, как получить из них данные, так и передать туда обработанную информацию. В Deductor не предусмотрено средств ввода данных –

платформа ориентирована исключительно на аналитическую обработку. Первичные данные обычно хранятся в различных СУБД, учетных системах, офисных документах, поэтому особое внимание при разработке платформы было уделено механизмам обмена данными. Для использования информации, хранящейся в разнородных системах, предусмотрены гибкие механизмы импорта-экспорта.

Для импорта данных нужно всего лишь указать источник и предоставить данные из любой системы в виде обычной таблицы или подготовить SQL запрос. Все остальные операции будут проведены автоматически. Платформа, откуда они были получены, значения не имеет, программа работает со всеми данными единообразно вне зависимости от их природы. Это позволяет легко расширять список поддерживаемых платформ.

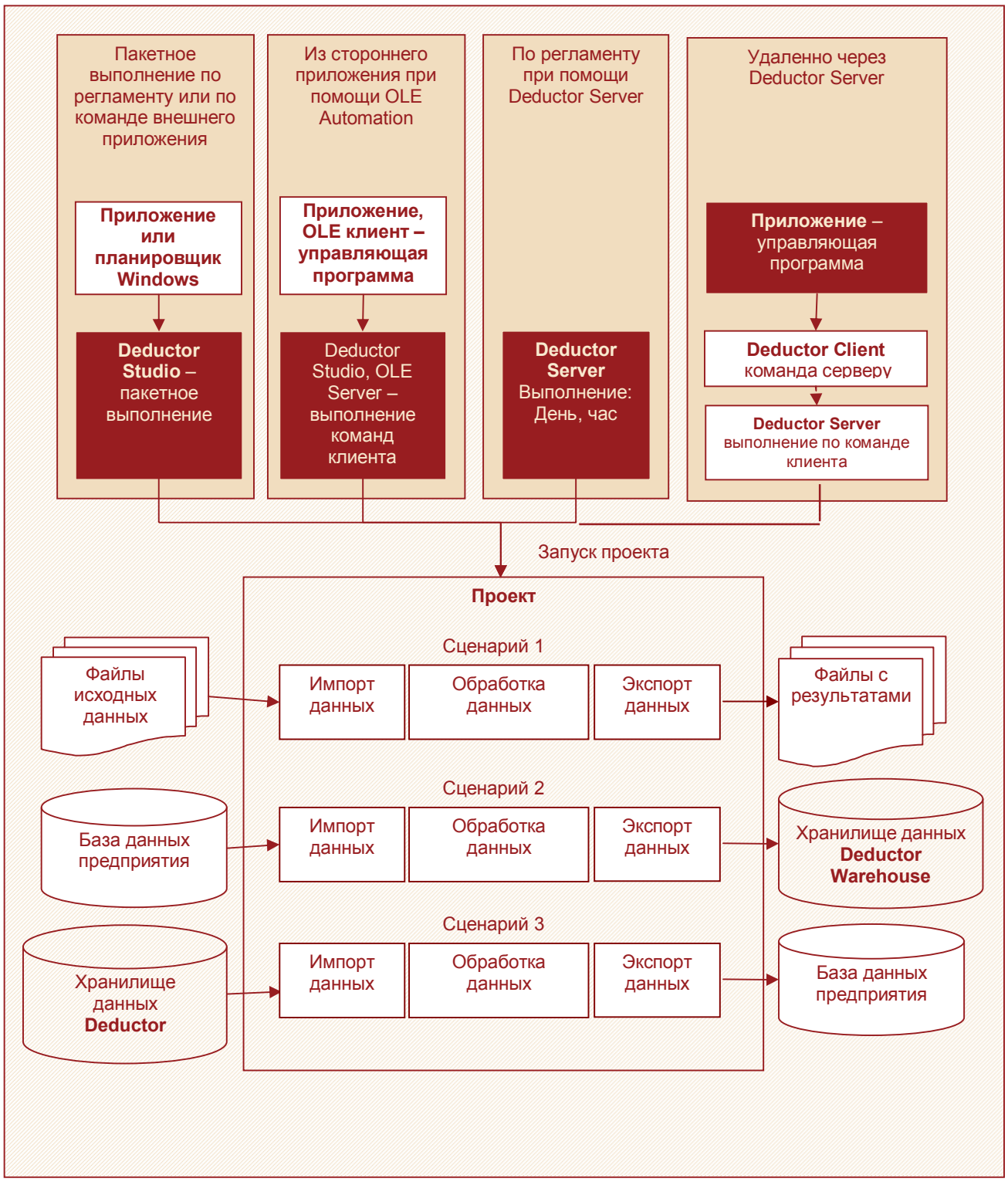
Обратное действие – перенос результатов обработки из Deductor Studio в другие системы производится при помощи механизмов экспорта. Все необходимые настройки производятся при помощи Мастера экспорта. Данную операцию можно произвести на любом шаге анализа данных.

Реализованные в Deductor механизмы позволяют эффективно использовать уже сделанные вложения в информационные системы, добавлять новые источники и адаптироваться к меняющимся условиям. Deductor имеет встроенную поддержку многих популярных платформ, поэтому техническим специалистам не придется устанавливать дополнительное программное обеспечение на рабочие станции.

В Deductor Studio предусмотрены средства, позволяющие выполнять все действия автоматически в пакетном режиме. Благодаря этому большую часть аналитической обработки можно проводить по регламенту, например, ночью, когда загрузка системы минимальна.

Сам проект подготавливается аналитиком при помощи Deductor Studio. Проект состоит из нескольких сценариев. Любой сценарий всегда начинается с узла импорта данных, но узел экспорта может присутствовать не всегда. В пакетном режиме обрабатываются только те фрагменты сценария, которые заканчиваются экспортом данных; т.к. вся обработка проходит без присутствия сотрудника, то очевидно, что если результат не будет куда-то выгружаться, то отсутствует смысл проводить его пакетную обработку.

Подготовленные в Studio проекты можно обрабатывать еще и при помощи Deductor Server. Технически реализовать интеграцию Deductor со сторонними системами можно несколькими способами.



Пакетный режим

Подготовленный заранее сценарий автоматически «прогоняется» на новых данных с экспортом результатов обработки в сторонний приемник. Таковым может выступать таблица в любой СУБД, текстовый файл с разделителями, dbf-файл, RTF, HTML, XML и прочее. Для этого используется приложение Deductor Studio, вызываемое с опцией /run. Результаты выполнения протоколируются в лог-файл. Приложению могут передаваться различные параметры в командной строке, определяющие выполнение тех или иных фрагментов сценария обработки.

При помощи любого планировщика можно настроить выполнение данного сценария по заданному регламенту, но, в принципе, вызвать пакетную обработку можно из любой программы, позволяющей выполнить команду операционной системы.

Для пакетного режима лучше использовать специальным образом подготовленные файлы сценария, не содержащие визуализаторов, что значительно сократит время на их загрузку. Получить такой файл из обычного сценария можно командой главного меню **Файл ► Сохранить для пакетной обработки...**.

Замечание

> *Команда **Сохранить для пакетной обработки...** удалит всю «лишнюю» визуальную часть из сценария, восстановить которую невозможно. Поэтому применять команду следует осторожно, выбирая другое, отличное от оригинального сценария, имя файла.*

OLE сервер

Deductor Studio версии Enterprise может работать в режиме OLE-Automation сервера. Механизм OLE-Automation позволяет реализовать обращение к Deductor Studio из внешних приложений, написанных на других языках. Внешнее приложение может указывать параметры и посылать команды OLE серверу.

Интеграция с использованием OLE предполагает написание кода на языках программирования, поддерживающих этот механизм, а это в значительной степени касается разработчиков программного обеспечения. Более подробно механизмы взаимодействия с Deductor Studio из внешних приложений описаны в API OLE сервера Deductor Studio, входящего в состав поставки – Deductor Library.chm.

Для регистрации Deductor Studio в качестве OLE Automation сервера необходимо единожды запустить Deductor Studio с параметром /RegServer:

$(Deductor)\Bin\DStudio.exe /RegServer.

Deductor Server

Deductor Server регистрируется в Windows как служба, обрабатывающая данные заданного порта по протоколу TCP/IP. Доступ к серверу обеспечивается удаленно при помощи специальной бесплатно распространяемой библиотеки **DClient.dll**. Данный механизм наиболее оптимальный при корпоративном использовании Deductor, особенно при удаленной работе с системой через Интернет.

Кроме того, в Deductor сервер встроен планировщик, позволяющий не только задать время запуска проекта, но и передать параметры.

Deductor Server автоматически оптимизирует скорость обработки данных за счет повторного использования загруженных ранее проектов. Имеется специальное приложение для просмотра статуса загруженных проектов и их отключения. Поддерживается многопроцессорная многопоточная обработка данных.

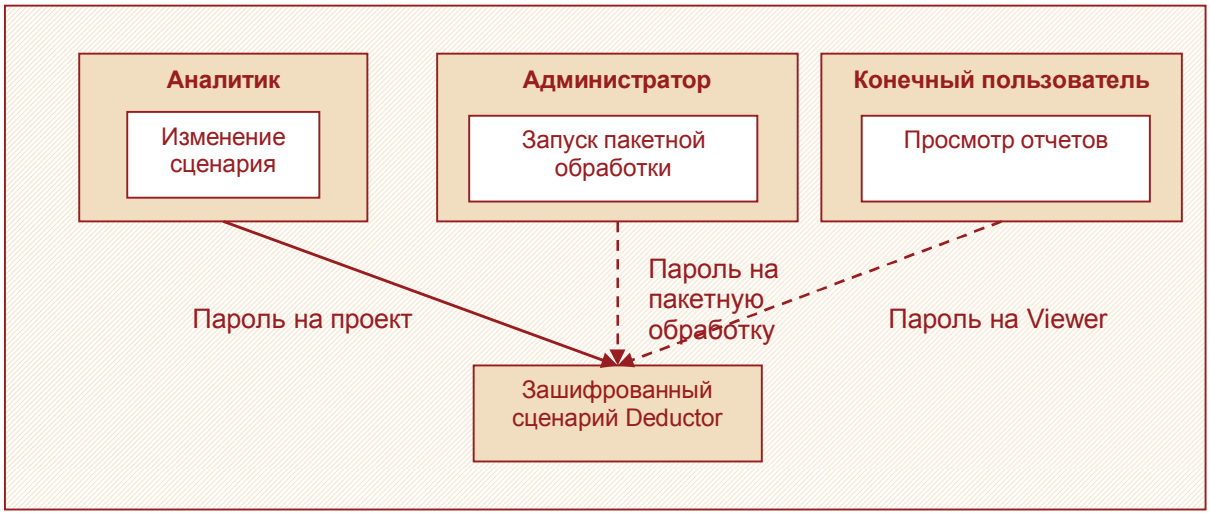
Более подробно механизмы взаимодействия с Deductor Server описаны в API сервера Deductor Client, входящего в состав поставки, файл Deductor Client API for Delphi.chm.

Защита сценария паролем

При создании и эксплуатации аналитического решения могут принимать участие много специалистов: аналитики, эксперты, системные администраторы и администраторы баз данных, конечные пользователи. Неизбежно возникнет задача, связанная с конфиденциальностью моделей обработки данных, несанкционированным вмешательством в сценарии. Для решения этих и других проблем в Deductor имеется возможность шифрования (**Файл ► Свойства проекта ► Защита**). Имеется три уровня защиты:

- Защита проекта Deductor *(обязательный уровень)*.
- Установка пароля на просмотр проекта через Deductor Viewer *(необязательный уровень)*. Изменить и просмотреть сценарий возможности нет.
- Установка пароля для пакетной обработки *(необязательный уровень)*. Изменить и просмотреть сценарий возможности нет.

В случае, когда пароли на просмотр Viewer и(или) на пакетную обработку не установлены, при инициировании этих событий будет запрашиваться пароль на проект.



Шифрование осуществляется при помощи алгоритма 3DES. При утере пароля восстановить зашифрованный сценарий невозможно.

Замечание

*Установить пароль можно только на сценарий, имеющий упакованный формат файла (флаг **Использовать упакованный формат файла** меню **Файл ► Свойства проекта**).*

Deductor Viewer – рабочее место конечного пользователя

Одна из особенностей платформы Deductor состоит в том, что модели, получаемые с ее помощью, могут в дальнейшем использоваться человеком, незнакомый с особенностями математического аппарата и методов, применяемых при анализе. Ему предоставляется набор отчетов, на основании которых он может принимать конкретное решение в терминах предметной области, а не формальных моделей. Достигаться это может двумя путями: использованием системы отчетов Deductor Studio или специального приложения для просмотра отчетов Deductor Viewer.

Deductor Viewer является облегченной версией Studio, не включающей в свой состав средств для разработки сценариев и проведения анализа. Тем не менее, он имеет точно такие же механизмы исполнения сценариев и визуализации, как и Deductor Studio. В результате получается, что человеку, который будет использовать только готовые сценарии и не станет заниматься их разработкой, предоставляется простой инструмент просмотра подготовленных отчетов. Нужно отметить, что дополнительно он сможет настраивать вид отчетов, то есть он сможет увидеть нужную ему информацию не в виде, скажем, графика, как предполагал аналитик, разрабатывавший сценарий, а в виде OLAP-куба или таблицы.

Принцип работы Deductor Viewer состоит в следующем. В программу загружается файл проекта, подготовленного в Deductor Studio. На единственной панели «Отчеты» отображается дерево подготовленных в Studio отчетов. Пользователь может посмотреть любой из них, дважды кликнув на иконке отчета, и настроить вид с помощью кнопки «Мастер визуализации» на панели инструментов. В случае, если пользователю понадобится дополнительная информация, отсутствующая в подготовленных отчетах, он может обратиться к аналитику, который уже в Deductor Studio внесет нужные изменения. Пользователю не требуется владеть формальными методами проведения анализа, ему достаточно уметь на основе полученных знаний принимать решения. Распространение знаний от аналитика к конечным пользователям информации получило название тиражирование знаний. Благодаря такому разделению не требуется, с одной стороны, обучать пользователей методике анализа данных и с другой стороны, загружать аналитика рутинной работой, связанной с принятием повседневных решений.

Конфигурационные файлы

Deductor Viewer использует те же конфигурационные файлы, что и Deductor Studio, однако в отличие от Deductor Studio в программа Deductor Viewer может только использовать их для чтения, поэтому эти файлы должны быть предварительно созданы с использованием Deductor Studio. Существует несколько вариантов подключения конфигурационных файлов:

- Копирование файлов в папку с программой (по умолчанию $(Deductor)\Bin).
- Указание пути к файлам в программе меню **Сервис ► Настройка**.
- Указание пути к файлам с использованием параметров командной строки.

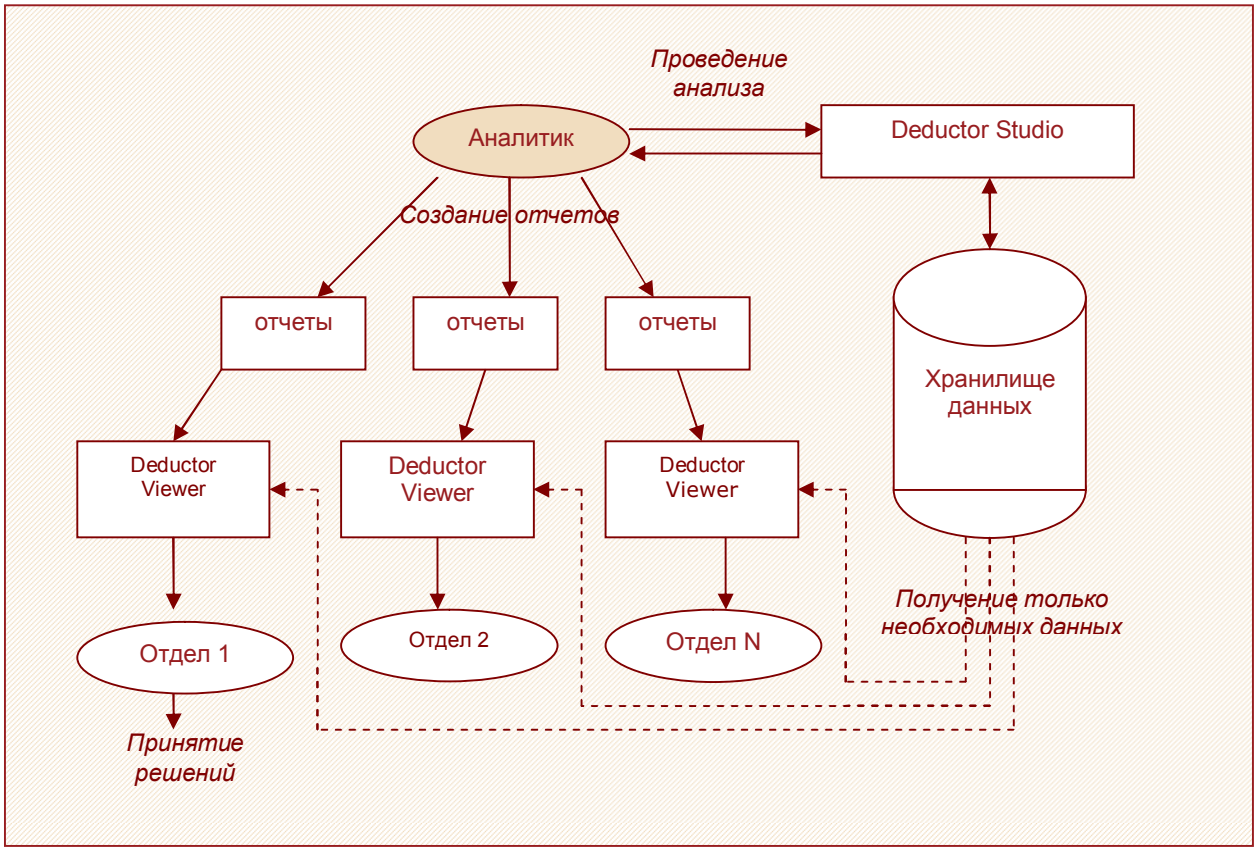
Разграничение прав

При использовании Deductor Viewer может быть решен такой важный вопрос, как разграничение прав пользователей на доступ к информации. Для того чтобы обеспечить аналитическую поддержку всех подразделений предприятия, в хранилище данных нужно загружать большое количество разнообразной информации, например, о финансовом положении фирмы, отношениях с поставщиками и клиентами, товарной, ценовой и конкурентной политике и т.д. В результате человек, имеющий доступ к хранилищу, получает в свое распоряжение большую часть информации о деятельности компании, и неизвестно, каким образом он захочет ею распорядиться.

Если каждому пользователю аналитической информации установить Deductor Studio, он сможет извлечь из хранилища не только те данные, которые ему нужны для работы, но и другую

конфиденциальную информацию, доступа к которой он иметь не должен. Например, сотруднику склада, которому нужны отчеты по остаткам товаров и срокам годности, совершенно нет необходимости знать о финансовых результатах деятельности компании за прошедший год. Deductor Viewer позволяет ограничить доступ различных категорий пользователей к информации из хранилища или других источников данных.

Процесс разграничения прав состоит из следующих этапов. Сначала аналитик в Deductor Studio готовит отчеты для каждой группы пользователей информации. Например, финансовые отчеты для отдела планирования, отчеты о состоянии рынка для отдела маркетинга, товарные отчеты для отдела логистики и т.д. После этого отчеты распределяются между пользователями, которые смогут работать с ними только посредством Deductor Viewer. Так как в Deductor Viewer отсутствуют самостоятельные средства извлечения и анализа данных, то они физически не смогут получить доступ к «лишней» информации и нарушить политику безопасности предприятия.



Использование Deductor Studio, Deductor Viewer и их место в компании схематично показано на следующем рисунке.

Deductor Warehouse – хранилище данных

После подготовки таблиц с исходными данными можно перейти к следующему этапу – созданию хранилища данных. В стандартном варианте решения Deductor получает данные для анализа из хранилища, но существуют ситуации, в которых создавать его не требуется.

Оценка потребности в использовании

Хранилище данных предназначено для консолидации разрозненных данных, обеспечения прозрачного для аналитика и быстрого доступа к ним. Загрузка данных в хранилище производится гораздо (на порядки) медленнее, чем извлечение данных. Объясняется это тем, что при загрузке выполняется множество вспомогательных операций, позволяющих в последствии быстрее получить данные их хранилища. Загрузка данных в хранилище может производиться в ночное время или выходные без отрыва пользователей от работы. Извлечение же должно осуществляться с минимальными задержками, чтобы обеспечить возможность экспресс-анализа и снизить задержки для пользователей.

Deductor Studio или Viewer может работать с Deductor Warehouse совершенно так же, как с любым другим источником данных. Тем не менее, существует несколько причин создания хранилища.

Основным достоинством хранилища с точки зрения аналитика является удобное представление данных. Оно обеспечивает семантическую прослойку между реляционной базой данных и предметной областью. В результате работа с данными из хранилища осуществляется в терминах предметной области (в бизнес-терминах). Удобство такого представления для анализа данных трудно переоценить, и это является наиболее частой причиной создания хранилища.

Вторая причина, по которой может потребоваться создание хранилища, появляется в тех случаях, когда данные для анализа находятся в разных источниках. Например, на предприятии есть две разные учетные системы для бухгалтерии и управления складом, а в анализе и при построении отчетов используются данные из двух сразу. Хранение данных в одном месте ускоряет доступ к данным, позволяет еще на этапе консолидации избежать появления дублирующихся или противоречивых данных, значительно упрощает разработку сценариев. Кроме того, гораздо проще становится внесение изменений в аналитическую систему, например, при переходе на другую учетную систему не потребуется перенастройка всех существующих сценариев, достаточно будет изменить сценарий загрузки данных в хранилище.

Третья причина – используется медленный источник данных. Например, данные для анализа забираются из ADO-источника или конфигурации 1С:Предприятия. В этом случае каждый раз при выполнении сценария потребуется импортировать данные из медленного источника, что приведет к заметным задержкам в работе системы. Применяя хранилище данных в этом случае, можно, всего один раз загрузив в него данные, в дальнейшем быстро забирать их оттуда по мере необходимости.

Хранилище данных не требуется, когда исходные данные для анализа находятся в быстрой базе данных предприятия уже в нужном виде и их дублирование в новом источнике не имеет смысла. Кроме того, оно может стать излишним, когда Deductor используется для некоторой промежуточной обработки данных, результаты которой будут сохраняться в БД, и при работе с данными малого объема (например, при «прогоне» нескольких строк данных через построенную модель). В таком случае наиболее оптимальным решением является применение Виртуального Хранилища Данных (Virtual Warehouse), которое может быть использовано после настройки семантического слоя как обычное хранилище данных (Deductor Warehouse), но при этом получая данные напрямую из базы данных предприятия. Более подробную информацию о виртуальном Хранилище Данных можно найти в документе «Настройка виртуального хранилища данных».

Решение о необходимости создания хранилища должно приниматься индивидуально в каждом случае в зависимости от особенностей существующей информационной системы предприятия и задач анализа, стоящих перед платформой.

Выбор платформы

Версия Professional поддерживает Deductor Warehouse только на платформе Firebird (версии 1.5 и выше).

Версия Enterprise поддерживает размещение Deductor Warehouse на платформах трех популярных СУБД: Firebird (версии 1.5 и выше), Microsoft SQL (версий 2000/2005), Oracle (начиная с версии 9i).

Поддержка нескольких СУБД в качестве платформы для Deductor Warehouse позволяет в каждом конкретном случае использовать наиболее подходящую базу данных.

Небольшим предприятиям вполне достаточно будет Deductor Warehouse на платформе СУБД Firebird, но при этом необходимо иметь ввиду, что даже такая небольшая СУБД для нормального функционирования требует обслуживание. Для средних вполне подойдет СУБД Microsoft SQL. Для крупных рекомендуется СУБД ORACLE.

Выбор СУБД в значительной степени определяется объемом данных, загружаемых в хранилище. При объеме данных до 4-5 Гб достаточно Firebird, при объеме в 5-15 можно воспользоваться MS SQL, при больших размерах лучше использовать Oracle.

Создание хранилища

Хранилище Deductor Warehouse может располагаться как локально на компьютере аналитика, так и удаленно, на выделенном сервере. Первый способ имеет смысл использовать, когда анализ данных производится на одном отдельном компьютере. Таким образом, можно разгрузить сеть от достаточно больших объемов передаваемых данных. В том случае, если требуется разделять информацию из хранилища между компьютерами сети, потребуется размещение хранилища на сервере.

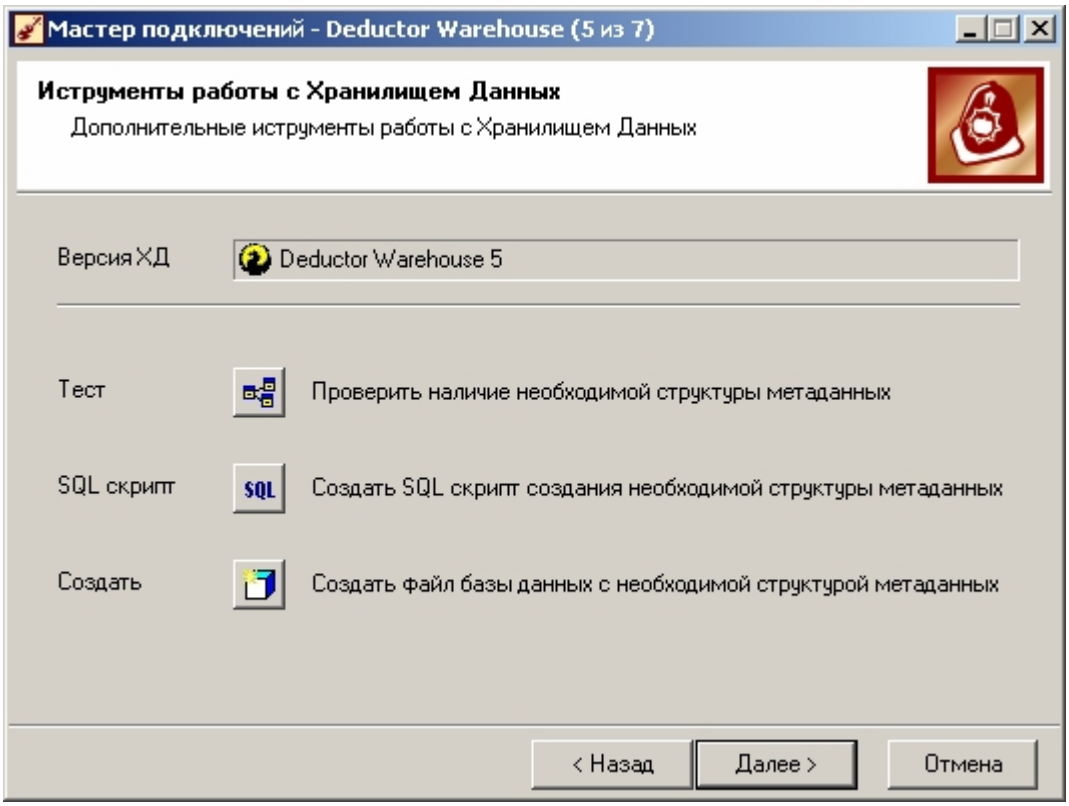
Во время первой загрузки в хранилище обычно перекачивается очень большой объем накопленных в учетных системах данных, что занимает много времени. При этом могут возникнуть проблемы с производительностью сети. Поэтому вопрос с первоначальной загрузкой данных в хранилище должен решаться особо, например, можно все таблицы перенести на сервер, где располагается хранилище данных и произвести загрузку локально. В дальнейшем при добавлении в хранилище новых порций данных таких проблем возникать не будет, так как их объем скорее всего будет на порядки меньше того, который загружается первоначально.

Создание локального хранилища и подключение к нему подробно описаны в документе «Руководство по импорту и экспорту данных».

Рассмотрим вопрос создания хранилища на выделенном сервере.

Для сервера Firebird существует два варианта создания хранилища данных: первый – это создание локального хранилища данных и перенос его в последствии на выделенный сервер, второй вариант – создание необходимой структуры базы данных с использованием SQL-скрипта. Перемещение хранилища с одного компьютера на другой в случае использования СУБД Firebird осуществляется обычным копированием файла хранилища. После этого следует в Deductor настроить параметры доступа к хранилищу (подключение к хранилищу).

Для СУБД Microsoft SQL и Oracle существует только один способ создания хранилища – генерация структуры с использованием SQL-скрипта. Текст SQL-скрипта можно получить с помощью Мастера подключений на закладке «Инструменты работы с хранилищем данных», нажав на кнопку **SQL скрипт** можно создать файл с текстом SQL-скрипта для выбранной платформы СУБД. В последствии этот SQL-скрипт необходимо выполнить на сервере выбранной СУБД, используя инструментарий, прилагающийся к SQL серверу базы данных.



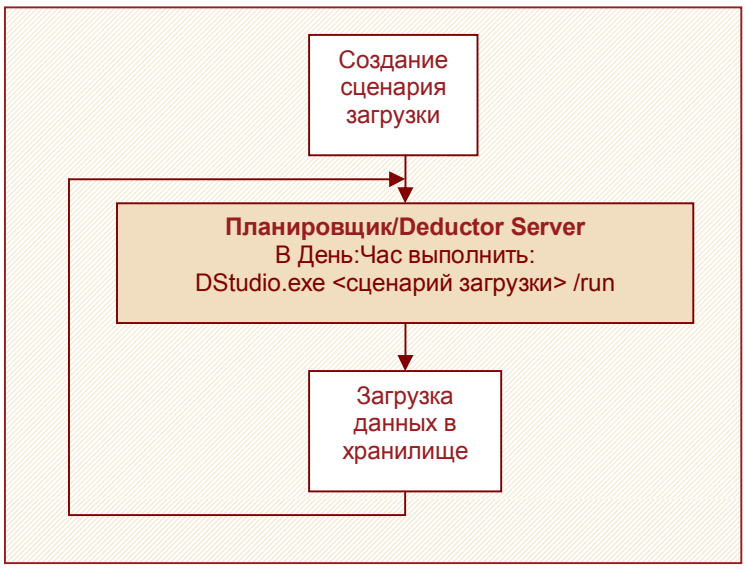
Создание хранилища данных фактически сводится к созданию в базе данных нужных объектов, в которых будут храниться метаданные, реализующие семантический слой. После создания метаданных все операции по работе с хранилищем (проектирование, импорт, экспорт) можно выполнить средствами, имеющимися в Deductor Studio, в частности, используя «Редактор метаданных».

Автоматизация загрузки данных

После создания структуры хранилища нужно заполнить его данными. Данные будут выбираться из исходных таблиц, сформированных на этапе «Подготовка данных», и загружаться в созданные на этом шаге объекты хранилища.

Существуют два типа загрузки данных в хранилище - первоначальная и инкрементная. Первоначальная загрузка осуществляется при первом после установки системы заполнении хранилища, она помещает в хранилище большой объем данных, накопленных в учетной системе. Такая загрузка обычно осуществляется только один раз при настройке системы. Из-за большого объема загружаемых данных она может занимать длительное время. Инкрементная загрузка выполняется регулярно, она добавляет в хранилище вновь появившиеся или измененные с момента последней загрузки хранилища данные. Объем загружаемых данных здесь сравнительно невелик, поэтому времени такая загрузка занимает гораздо меньше первоначальной. Дополнительно ускорить инкрементную загрузку можно с помощью очистки процесса по измерению.

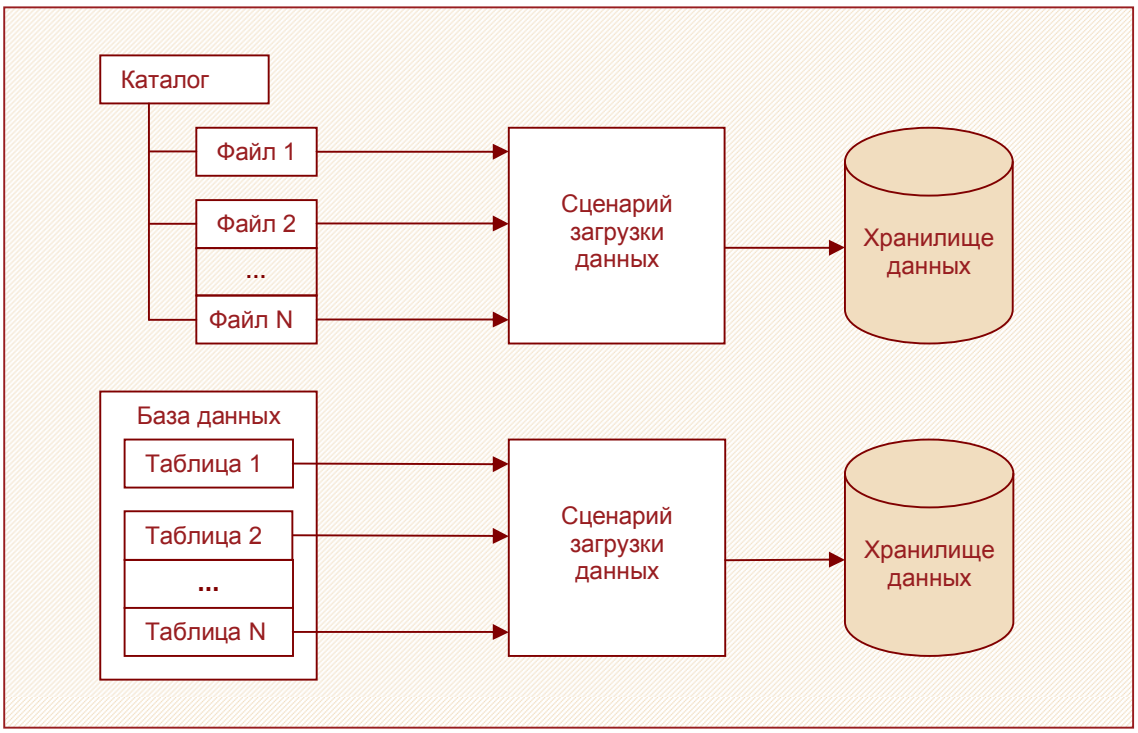
Инкрементная загрузка обычно производится в автоматическом режиме по расписанию. Для этого создается отдельный сценарий загрузки хранилища, и с помощью какой-либо программы-планировщика (например, Windows Scheduler) планируется запуск Deductor в пакетном режиме для его выполнения. В качестве планировщика можно использовать также Deductor Server, который позволяет производить плановые запуски Deductor Studio со всеми необходимыми параметрами. Загрузку данных удобно осуществлять в ночное время или выходные дни, когда свободны сетевые ресурсы и хранилище не используется для работы. Схематично процесс загрузки изображен на рисунке.



Сценарий загрузки должен выполнять следующие функции:

1. Импорт данных в Deductor Studio из базы данных, учетной системы или предопределенных файлов;
2. Опциональная предобработка данных, например, очистка или преобразование формата;
3. Загрузка данных в измерения и процессы хранилища Deductor Warehouse.

Перед импортом данных в Deductor они могут быть обработаны любой другой программой и сохранены в согласованном со сценарием загрузки формате, например, в предопределенные таблицы базы данных или в определенные файлы заданного каталога. В таком случае сценарий загрузки данных настраивается на использование в качестве источников данных этих временных таблиц или файлов.



Для первоначальной загрузки может использоваться сценарий инкрементной загрузки или создаваться специальный сценарий.

Оптимизация хранилища данных

Может оказаться, что созданное и загруженное хранилище данных работает очень медленно. Объясняется это тем, что для повышения быстродействия хранилища требуется выполнить несколько дополнительных операций.

Первое – проверить, существует ли для «медленного» процесса, из которого происходит получение данных, «временная таблица». Временная таблица создается в момент экспорта данных в хранилище. Этот параметр определяется аналитиком при проектировании сценария в Deductor Studio.

Второе – после загрузки большого объема данных необходимо запустить сбор статистики по индексам для используемой платформы. Порядок запуска сбора статистики для используемой платформы описан в документации к базе данных.

Оптимизация хранилища данных на платформе Firebird

Хранилище данных Deductor Warehouse основано на свободно распространяемой реляционной СУБД Firebird ver.1.5. В пакет установки Deductor включена динамическая библиотека для доступа к базам данных Firebird. С помощью этой библиотеки возможен как прямой доступ к базе данных, так и подключение к серверу СУБД.

Без установки сервера Firebird возможна работа только с локальным хранилищем данных, причем Deductor будет подключаться к нему в монопольном режиме, то есть работать с хранилищем в каждый момент времени сможет только одно приложение. Установка сервера Firebird на компьютер, на котором расположено хранилище, позволяет открыть доступ к удаленному хранилищу. СУБД Firebird входит в комплект поставки Deductor, последнюю версию сервера Firebird и документацию к нему можно найти на сайте разработчиков http://www.firebirdsql.org .

Первое, что следует сделать после установки сервера, это скопировать в подкаталог UDF каталога установки Firebird (при установке по умолчанию в каталог C:\Program Files\Firebird\UDF) динамическую библиотеку для работы со строками rfunc.dll. Эта библиотека находится в каталоге

$(Deductor)\Bin\Udf (при установке по умолчанию в каталоге C:\Program Files\BaseGroup\Deductor\Bin\Udf). Использование этой библиотеки увеличивает скорость получения строковой информации из базы данных хранилища в несколько раз.

Далее нужно изменить некоторые параметры в файле настроек сервера. Файл настроек Firebird находится в каталоге установки (по умолчанию C:\Program Files\Firebird) в файле **firebird.conf**. В нем следует сделать следующие настройки. Во-первых, необходимо установить количество кэшируемых в памяти страниц базы данных. Для этого параметру **DefaultDbCachePages** следует присвоить нужное значение. Выбирать его следует, исходя из доступного объема оперативной памяти. Объем памяти, занимаемый кэшем базы данных, равен значению **DefaultDbCachePages**, умноженному на размер страницы базы данных (по умолчанию 4 кб). Это значение не должно превышать размер свободной памяти, остающейся после загрузки операционной системы и приложений. В противном случае операционная система начнет выгружать кэш базы данных на жесткий диск и лишь еще больше замедлит обращения к базе.

Firebird активно использует временные файлы для хранения промежуточных данных, например, при сортировке и группировке. Серьезно поднять скорость работы при выполнении подобных операций можно, если использовать для создания этих временных файлов раздел в оперативной памяти (так называемый memory disk). Существуют различные инструменты для создания таких разделов, например, RamDisk. После создания диска в памяти следует создать на этом диске папку и указать Firebird, чтобы тот использовал указанную папку в ОЗУ для хранения временных файлов. Для этого параметру TempDirectories из файла настроек следует присвоить имя созданного диска, после чего будет происходить кэширование временных файлов в памяти.

Остальные параметры сервера Firebird не оказывают заметного влияния на производительность системы. Однако, для того, чтобы работа хранилища не замедлялась с течением времени, дополнительно потребуется производить текущее обслуживание базы данных хранилища.

В процессе добавления и удаления данных из базы данных хранилища возможна ее фрагментация. Сильная фрагментация базы может значительно замедлить ее работу, поэтому рекомендуется проводить периодическое обслуживание хранилища. Для этого следует проводить резервирование и восстановление базы данных с помощью дополнительных инструментов. Рекомендуется проводить подобную операцию каждый раз при обновлении или удалении больших объемов данных в хранилище. Резервирование базы может проводиться не только для дефрагментации, но и для сохранения копии на случай сбоя сервера. В этом случае после восстановления потребуется загрузить в хранилище только информацию, обновленную с момента последнего резервирования.

В состав сервера Firebird входят инструменты для обслуживания баз данных Firebird. Для проверки целостности и восстановления базы данных предназначена утилита gfix. Для дефрагментации – утилита gbak. Управление ими производится из командной строки, список ключей можно получить с помощью команд «gfix -?» и «gbak -?». Работа со стандартными утилитами может показаться не очень удобной, но зато позволяет выполнять требуемые операции значительно быстрее утилит сторонних разработчиков.

Одной из утилит для обслуживания баз данных Firebird, обладающей графическим интерфейсом, является свободно распространяемая программа IBExpert от HK Software, доступная для скачивания на сайте http://www.ibexpert.com.

Оптимизация хранилища данных на платформе MS SQL

Для оптимизации хранилища данных на данной платформе в принципе подходят все механизмы, рекомендуемые при оптимизации другой базы данных.

Приведем краткий список, на что стоит обратить внимание:

- Сбор статистики по данным для работы оптимизатора SQL запросов;
- Выполнение процедуры сжатия после операции загрузки данных.

Оптимизация хранилища данных на платформе Oracle

Для оптимизации хранилища данных на данной платформе в принципе подходят все механизмы, рекомендуемые при оптимизации другой базы данных.

Приведем краткий список, на что стоит обратить внимание:

- Сбор статистики по данным для работы оптимизатора SQL запросов;
- Разделение таблиц с данными и индексов в различные табличные пространства, которые физически расположены на разных носителя;
- Разделение таблиц по различным табличным пространствам.

Deductor Server – аналитический сервер

Аналитический сервер, позволяет удаленно управлять выполнением сценариев Deductor Studio, а также обрабатывать проекты Deductor в автоматическом режиме. В автоматическом режиме может производиться как выполнение сценариев, так и переобучение моделей.

Установка и запуск

Deductor Server может работать в двух режимах:

- Приложение.
- Служба Windows.

Для регистрации Deductor Server в виде службы необходимо запустить Deductor Server с параметром /Install :

$(Deductor)\Bin\DServer.exe /Install

где $(Deductor) – каталог установки программы (по умолчанию – "C:\Program Files\BaseGroup\Deductor").

Для работы сервера требуется наличие установленного Deductor Studio и его регистрация в качестве OLE Automation сервера.

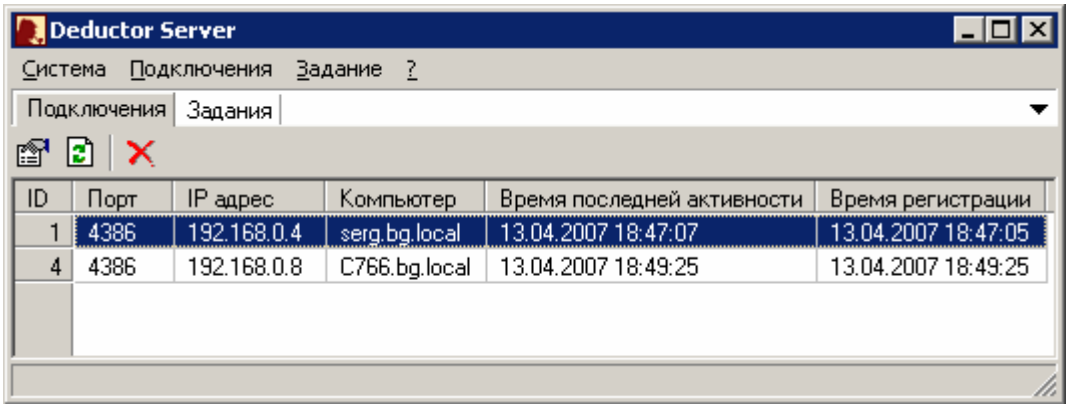
Для регистрации Deductor Studio в качестве OLE Automation сервера необходимо единожды запустить Deductor Studio с параметром /RegServer:

$(Deductor)\DStudio.exe /RegServer.

Для упрощения процедуры регистрации в процессе инсталляции создаются необходимые ярлыки в папке программы меню **Пуск**.

Конфигурирование сервера

После запуска сервера в системном трее появляется иконка  , используя ее контекстное меню, можно вызвать основное окно приложения настройки и мониторинга сервера.



Сервер имеет несколько параметров, определяющих режим функционирования. Окно настройки параметров вызывается через пункт меню **Подключения ► Настройки**.

- **Порт TCP/IP** – это целое число в диапазоне от 1 до 32767 (по умолчанию 4386), являющееся идентификатором соединения, по которому сервер принимает запросы от

клиентов. Этот номер должен быть уникальным в пределах одного компьютера. Если порт по умолчанию (4386) занят, то его можно изменить на любой другой. Лучше выбирать большие номера портов во избежание конфликтов с другими сервисами и приложениями. Обязательно нужно убедиться, что данный порт не блокируется брандмауэром.

- **Отключать неактивные подключения, мин.** – это параметр определяет количество минут, через которое будет принудительно отключено (удалено) неактивное подключение. Если флажок не включен, это означает, что принудительное отключение (удаление) по времени не будет производиться никогда ни для каких подключений.
- **Размер кэша подключений** – это очень важный параметр, значение которого нужно выбирать очень внимательно. Размер кэша подключений – это целое число больше или равное 0 (по умолчанию 3), которое определяет максимальное количество закрытых подключений, которые могут быть использованы повторно. При появлении нового подключения требуется определенная инициализация, которая может занимать достаточный промежуток времени. Также очень много времени тратиться на загрузку сценария. Если же необходимо сократить время на инициализацию нового подключения, а также время на загрузку сценария, это можно сделать, повторно использовав ресурсы, которые не были удалены при закрытии/завершении подключения. Значение равное 0 отключает повторное использование открытых настроек подключений и открытых и подготовленных сценариев, что значительно снижает производительность работы. В любом случае значение этого параметра не имеет смысла делать больше, чем кол-во предполагаемых одновременных подключений к Deductor Server.
- **Размер кэша потоков** – это целое число больше или равное 0 (по умолчанию 10), определяющее максимальное количество неактивных потоков, которые могут быть использованы повторно. При появлении нового подключения вся работа с ним ведется в контексте отдельного потока, т.е. все подключения обрабатываются параллельно друг другу. Для создания потока требуется определенные системные ресурсы, такие как процессорное время и память. Если же необходимо сократить время на инициализацию нового подключения, это можно сделать, повторно использовав поток, который не был удален при закрытие/завершение подключения. Увеличение данного параметра уменьшает скорость инициализации нового подключения, однако увеличивает требования к системе, хотя не в такой степени, как увеличение параметра "Размер кэша подключений". В любом случае значение этого параметра не имеет смысла делать больше, чем кол-во предполагаемых одновременных подключений к Deductor Server.
- **Максимальное количество подключений** – значение максимального количества подключений определяет порог, превысив который новые подключения становятся в очередь на обработку. Если флажок *Очередь подключений* не включен, это означает, что порог не ограничен. При помощи параметров, входящих в группу Очередь подключений, можно, например, организовать схему работы, когда новые подключения гарантировано будут обработаны даже с ограниченным количеством оперативной памяти, но только после того, как предыдущие подключения закроются. Значения параметров *Размер кэша подключений* и *Размер кэша потоков* не имеет смысла устанавливать больше, чем значение этого параметра.
- **Сразу отклонять новое подключение, если лимит уже достигнут** – если этот флажок включен, то при появлении нового подключения к Deductor Server при достигнутом лимите количества подключений, это новое подключение будет отклонено. Если флажок выключен, то при достигнутом лимите количества подключений новое подключение будет поставлено в очередь.
- **Время ожидания в очереди нового подключения, мин.** – данный параметр определяет количество минут, прошедших с начала постановки в очередь нового подключения, через которое это новое подключение будет отклонено. Если этот флажок не включен, то новое подключение в очереди не будет отклонено по тайм-ауту.
- **Приостановить работу сервера** – если этот флажок включен, то все новые подключения будут всегда отклоняться. Фактически, если этот флажок включен, это означает временное отключение Deductor Server.

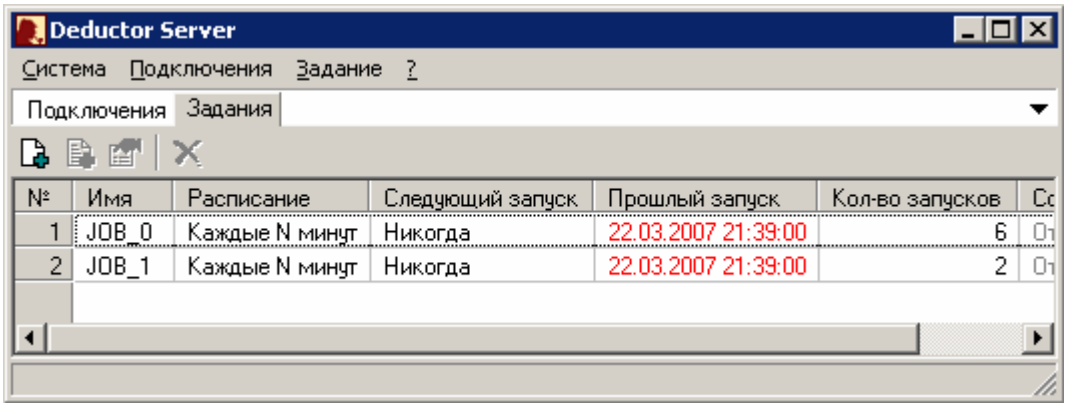
Параметры сервера сохраняются в файле $(Deductor)\Bin\DServer.sys.

Подключение к серверу

Подключение к серверу из сторонних приложений осуществляется с использованием функций библиотеки DClient.dll к Deductor Server по порту, указанному в параметре *Порт TCP/IP* сервера (по умолчанию 4386). Для установления соединения необходимо, чтоб этот порт был доступен, и в случае использования брандмауэров нужно внести необходимые изменения в их настройку.

Планировщик

Deductor Server имеет встроенный планировщик заданий, позволяющий с необходимой периодичностью и в нужный момент времени производить запуск Deductor Studio с необходимыми параметрами запуска. Список заданий можно увидеть на закладке «Задания». Управлять списком заданий можно либо через пункт главного меню «Задание», либо используя контекстное меню списка заданий.



Диагностика ошибок

Существует несколько видов ошибок:

- Ошибка при запуске программы из-за недоступности ключа;
- Ошибки выполнения сценариев при неверных настройках подключений;
- Ошибки работы программы.

Остановимся более подробно на ошибках по типам.

Ошибки отсутствия ключа

Такие ошибки обычно возникают при запуске программы и изредка при проблемах в сети во время работы:

- «Guardant dongle not found or printer is OFF !».
- «Invalid Guardant dongle found !».
- «No more executions left !».
- «System error !».

Если происходит работа с локальным ключом, то необходимо проверить следующее:

- установлены драйверы ключа;
- установлен ключ в USB порт, на нем горит светодиод;
- версия ключа соответствует версии программы.

При работе с сетевым ключом:

- имеется доступ к станции, на которой установлен сетевой ключ (PING);
- проверить состояние сервера лицензий;
- сервер лицензий «определил ключ», если ключ не определяется, см. пункты решения проблем при локальном ключе, а также проверить, что ключ сетевой;
- проверить соответствие установок в файле GNCLIENT.INI: протоколы IP/Имя сервера
- версия ключа соответствует версии программы.
- есть свободные лицензии.
- нет «зависших» сессий.

Ошибки подключения к источникам данных

Данные ошибки возникают в процессе выполнения программой узлов файла сценариев, а именно узлов импорта и экспорта. Для определения проблем необходимо проверить следующее:

- Если в импорте/экспорте используется указание пути к файлам/папкам, то для импорта необходимо проверить существование данных файлов/папок, а для импорта/экспорта еще права на доступ к ним;
- Если в импорте/экспорте указано подключение к серверу БД, необходимо наличие требуемых подключений и корректность параметров.
- Если проверка подключения к серверу БД неудачна, проверить возможность подключения к этому серверу и БД с этой станции другими программами (используя инструментарий поставляемы с БД)

- При корректных подключениях к БД и ошибках импорте/экспорте проверить доступность необходимых таблиц и прочих нужных объектов БД (представления, функции и процедуры).

Ошибки работы программы

Данные ошибки делятся на два класса: ошибки при отсутствии необходимой свободной памяти и ошибки в коде программы.

При ошибках недостаточности свободной памяти необходимо:

- Закрыть и снять активность для всех неиспользуемых узлов сценария;
- Проверить наличие свободного места на диске, на котором находится программа, и на диске, на котором находится папка для временных файлов /TEMP, а также на диске, где установлена операционная система;
- Увеличить размер файла подкачки.

При ошибках в коде программы сообщить максимум информации разработчикам пакета по адресу deductor@basegroup.ru. Желательно сообщить следующую информацию.

- Текст ошибки, который выдала программа в окне ошибки (если в окне есть кнопка **Подробнее**, добавить информацию и из этого окна).
- Номер версию и релиз программы (можно получить из пункта меню **? ► О программе**).
- Описать последовательность действий, выполняемых перед получением сообщения об ошибке.

Заключение

Deductor является достаточно сложной системой. Он плотно связан со многими различными технологиями, начиная от СУБД и заканчивая динамическим обменом данных. В связи с этим на практике часто возникают вопросы технического характера по установке, настройке и сопровождению системы.

В этом Руководстве даны ответы на наиболее часто возникающие у администраторов вопросы при эксплуатации Deductor. Тем не менее, на практике могут возникнуть ситуации, которые не описаны здесь. В этом случае нужно обратиться к разработчикам платформы, подробно описав интересующий вопрос или возникшую проблему, по электронной почте deductor@basegroup.ru, или задав вопрос на специализированном форуме (http://www.basegroup.ru/forum/deductor). Вы получите ответ в ближайшее время, и, возможно, в будущем он будет включен в один из разделов этого Руководства.

Контакты

Адрес:

Россия, 390046, г.Рязань, ул.Введенская 115, оф. 447.

Телефон: +7 (4912) 24-09-77

24-06-99, 25-83-97

Факс: +7 (4912) 24-09-77

e-mail:

info@basegroup.ru – общая информация

sale@basegroup.ru – служба продаж

deductor@basegroup.ru – служба поддержки Deductor

education@basegroup.ru – дистанционное обучение