

ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ
Автономная некоммерческая организация науки и образования
«Институт компьютеринга»

УТВЕРЖДАЮ

Директор АНО
«Институт компьютеринга»

_____ /А.И. Миков/

«__» _____ 2008 г.

М.П.

РЕКЛАМНО-ТЕХНИЧЕСКОЕ ОПИСАНИЕ

Программное ядро MDK Suite CASE-системы METAS

.31569113.00002-02 99 01

Листов 95

Разработчики:

_____ /Борисова Д.А./

_____ /Еремина М.Е./

_____ /Куделько Е.Ю./

_____ /Ланин В.В./

_____ /Лядова Л.Н./

_____ /Рыжков С.А./

_____ /Чичагова М.В./

_____ /Шаврин С.М./

28.07.2008

1. Функциональное назначение программы, область ее применения, ее ограничения

1.1. Назначение программы

Программный комплекс MDK Suite (MetaData Kernel Suite) METAS – это *программное ядро CASE-системы METAS*, предназначенной для разработки распределенных адаптируемых информационных систем (ИС) и поддержания их полного жизненного цикла. Комплекс включает инструментальные средства разработки ИС и runtime-компоненты, обеспечивающие ее функционирование.

CASE-технология, получившая название METAS (METAdata System – *система, основанная на метаданных*), удовлетворяет следующим требованиям:

1. Наличие высокоуровневого языка описания информационной системы и ее предметной области.
2. Поддержка объектов комплексной автоматизации: объектов предметной области, документов (документов, получаемых из внешних источников в различных форматах, первичных и результирующих (сводных) отчетов), бизнес-процессов.
3. Поддержка реинжиниринга на всех этапах разработки программного обеспечения при использовании спиральной модели жизненного цикла ИС.
4. Обеспечение гибкости программного обеспечения, полученного с помощью CASE-технологии, его адаптируемости к конкретным условиям (потребностям пользователей и бизнес-процессов) через настройку на различные источники данных, возможность реструктуризации данных и интеграции с программным обеспечением, созданным сторонними разработчиками для реализации «нестандартной» бизнес-логики.

Основой технологии являются *взаимосвязанные метаданные*, описывающие информационную систему, ее предметную область [2-4, 8-9, 12-17, 19, 22, 26, 34, 35].

Основное отличие CASE-технологии METAS от многих существующих, которые по некоторым спецификациям, описывающим предметную область, генерируют код на каком-либо языке программирования, в том, что данная система использует это описание *в режиме интерпретации*, во время своей работы, выполняя функции и отображая данные, описанные метаданными. Это дает возможность гибкой настройки приложения, реструктуризации описываемых объектов в процессе эксплуатации системы и создает хорошие предпосылки для создания интеллектуальной системы, которая может

настраиваться на пользователя, его потребности. Собирая определенную статистику о своей работе, система может кэшировать действия или переорганизовываться для достижения наилучшей производительности. При этом проблема реинжиниринга значительно ослабляется, т.к. метаданные взаимосвязаны, что позволяет проводить каскадные операции над метаданными при итеративной разработке, не нарушая целостность системы. Нестандартная логика приложения выносится в компоненты интеграции и не затрагивается при изменениях в метаданных системы.

Программный комплекс MDK Suite позволяет:

- С помощью *средств реструктуризации* строить описание предметной области информационной системы и при необходимости выполнять операции по его изменению с целью адаптации ИС к меняющимся условиям и требованиям.
- С помощью *средств генерации и настройки пользовательского интерфейса* создавать диалоговые окна и формы для поиска, ввода и редактирования данных, навигации по информационным объектам системы.
- С помощью *средств генерации SQL-запросов* строить и выполнять запросы к базе данных (БД) системы и обрабатывать их.
- С помощью *средств доступа к данным* настраиваться на использование различных СУБД.
- *Подключать программные компоненты сторонних разработчиков* для реализации нестандартной логики и интерфейса для нестандартных типов данных, используя стандартный программный интерфейс системы.
- *Обеспечить защиту объектов ИС от несанкционированного доступа* с помощью системы защиты.
- *Использовать средства интеграции с внешними системами*, основанные на технологии BizTalk Server.

Комплекс MDK включает как средства разработки ИС, так и run-time компоненты, обеспечивающие функционирование системы и ее адаптацию.

В качестве основного языка для описания ИС используется UML (Unified Modeling Language) – унифицированный язык моделирования. Процесс разработки и внедрения основан на RUP (Rational Unified Process) технологии.

1.2. Область применения программы

Основной задачей в области информационных технологий на сегодняшний день является создание ИС корпоративного уровня. В 1968 году на одной из конференций по проблемам разработки программного обеспечения был

предложен термин «Software Engineering». Так была названа новая научная дисциплина, объектом исследования которой являются проблемы создания больших компьютерных систем. А за последние два десятилетия «прижился» термин CASE-средства (CASE, Computer-Aided Software/System Engineering) – средства, которые помогают автоматизировать процесс проектирования и разработки программного обеспечения.

Определим CASE-систему как совокупность технологий, инструментальных средств, с помощью которых специалист в конкретной предметной области может построить свою ИС.

CASE-средства METAS *позволяют создавать распределенные информационные системы, обладающие уникальными возможностями для адаптации к меняющимся условиям эксплуатации и потребностям пользователей, гарантирующими их живучесть и эффективность вложений в их разработку и внедрение, для различных предметных областей.*

CASE-система прошла апробацию при создании *региональной информационной системы образования Пермской области* [18, 20, 21, 24].

1.3. Ограничения использования программы

CASE-система METAS позволяет настраиваться на различные программные платформы, работать под управлением различных операционных систем Microsoft, использовать для создания информационных систем различные реляционные СУБД и источники данных, для которых существуют драйверы ODBC.

Для функционирования runtime-компонентов необходимо установить .NET Framework (распространяется бесплатно), драйверы ODBC (при установке операционной системы) и СУБД (можно использовать, в частности, Microsoft SQL Server Express, которая распространяется бесплатно).

Возможности масштабирования системы определяются возможностями настройки на различные платформы.

Ограничения использования системы определяются только требованиями лицензионной чистоты и требованиями, предъявляемыми перечисленными выше программными средствами, применяемыми как для разработки ИС, так и для организации ее функционирования.

2. Техническое описание программы

Данная технология в большей степени предполагает интерпретирующий режим работы программного обеспечения ИС. Именно поэтому система является гибкой, в ней могут быть определены функции, которые ведут мониторинг системы и могут перестраивать метаданные для более правильного учета предметной области и повышения производительности системы.

Технология базируется на

- Языке моделирования UML.
- Технологиях разработки RUP, MSF, XP.
- Концепции распределенных многоуровневых приложений.
- Платформе .NET.
- Инфраструктуре BizTalk framework.

Составляющие технологии:

- Концепция метаданных, представляющих формализованное описание информационной системы.
- Интерпретатор метаданных.
- Архитектура конечной ИС.
- Методология разработки ИС METAS.

Технология METAS поддерживает спиральный, итеративный процесс разработки программного обеспечения, весь жизненный цикл ИС, начиная с ранних стадий проектирования и определений требований к ИС до вывода системы из эксплуатации.

2.1. Структура программного продукта

В укрупненном виде, структуру программного обеспечения системы METAS можно представить как набор из 5 основных блоков (рис. 1).

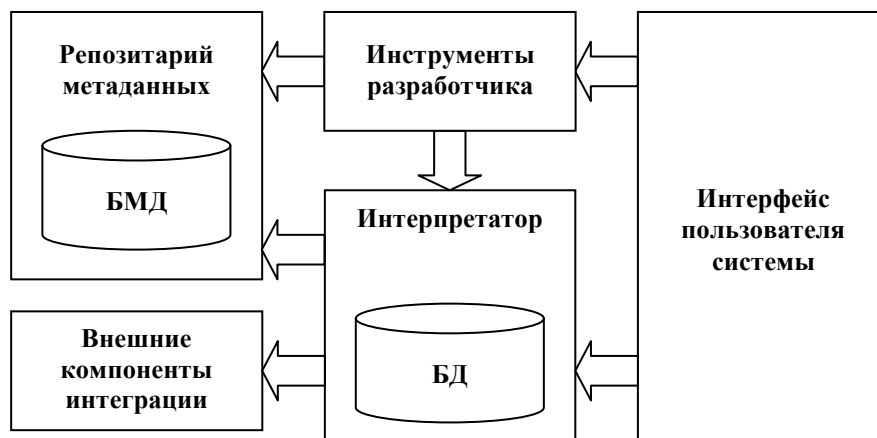


Рис. 1. Структура программного обеспечения системы METAS

Репозиторий метаданных служит для хранения метаданных системы и предоставления программного доступа к метаданным.

Внешние компоненты – блок управления компонентами интеграции, подключаемыми к системе для реализации нестандартной логики (нестандартная логика, также реализуется в метаданных системы за счет использования скриптов).

Интерпретатор, используя метаданные и внешние компоненты, выполняет создание БД ИС, настройку ее интерфейса и пр. при ее разработке и осуществляет управление ее функционированием во время работы, предоставляя доступ к объектам описываемым метаданными ИС.

Инструменты разработчика служат для наполнения репозитория (по большей части они являются визуальными).

Интерфейс пользователя системы – это компоненты для визуальной работы пользователя системы с объектами ИС.

Метаданные системы

Основа CASE-технологии METAS – метаданные, описывающие ИС. Метаданные должны представлять все основные объекты, функции и процессы ИС. Необходимо иметь описание ИС с различных точек зрения и, как следствие, возможно некоторое дублирование метаданных, но в любом случае метаданные должны быть связанными и непротиворечивыми, чтобы обеспечивать наиболее полное и целостное описание ИС.

С помощью метаданных описываются следующие аспекты ИС:

- объекты ИС,
- функции ИС,
- документы и отчеты ИС,
- бизнес-процессы и документооборот,
- пользовательский интерфейс,
- ролевую защиту.

Для хранения метаданных создается *централизованное хранилище* – *репозиторий* (база метаданных) и реализуется программный интерфейс доступа к ним. Это может быть набор низкоуровневых функций или объектно-ориентированная программная модель.

CASE-технология METAS позволяет работать с различными СУБД.

Для основного хранилища метаданных рекомендуется использовать СУБД серверного типа. Режим работы созданной ИС является интерпретирующим, при этом компонентам ИС необходим быстрый доступ к репозиторию метаданных, поэтому необходимо кэшировать метаданные в узлах ИС. Для кэшей

метаданных можно использовать СУБД настольного типа, либо хранить в XML, либо в двоичном виде, определяемом конкретной реализацией программного интерфейса доступа. Узлы ИС могут выполнять различные функции, для выполнения которых могут понадобиться не все метаданные ИС. С целью увеличения производительности необходимо кэшировать на этих узлах только те метаданные, которые необходимы этому узлу.

Помимо встроенных метаданных системы допускается использовать внешние источники метаданных, которые расширяют собственные метаданные. Например, такими метаданными могут служить метаданные СУБД (в MS SQL Server есть Meta Data Services), которые описывают физическое хранение объектов в реляционной БД. В любом случае, где бы реально ни хранились метаданные, доступ к ним должен осуществляться единообразно через программный интерфейс.

В представленной реализации метаданные хранятся в базе метаданных, управляемой той же СУБД, что и база данных ИС. Возможности настройки расширяются путем использования XML для представления дополнительной информации, которая может обрабатываться и анализироваться программными компонентами системы (в том числе и компонентами сторонних разработчиков).

Средства визуального проектирования

В качестве основного языка для описания ИС используется UML (Unified Modeling Language) – унифицированный язык моделирования.

Некоторые модели используют для описывания предметной области диаграммы языка UML, другие служат для управления функционированием системы либо являются прослойками между моделями первого типа и действительным (физическим) представлением объектов предметной области в памяти.

Современные системы проектирования немислимы без удобных визуальных средств, позволяющих проектировать и модифицировать ИС, адаптируя ее к меняющимся потребностям.

Обязательными *визуальными средствами редактирования* являются:

1. Средство визуального проектирования структуры объектов ИС, взаимосвязей и операций на основе диаграмм классов UML, представляющих описание статической структуры предметной области ИС.
2. Средство визуального проектирования пользовательского интерфейса, обеспечивающее возможность автоматической генерации и настройки пользовательского интерфейса, добавления стандартных и

специализированных элементов управления, автоматическую связь с данными объекта.

3. Конструктор отчетов и документов ИС.
4. Средство визуального описания бизнес-процессов.
5. Средства документирования:
 - генерация документации программиста,
 - генерация документации аналитика/проектировщика,
 - генерация документации пользователя.

Эти средства реализованы в системе (для их реализации могут быть использованы также внешние приложения визуализации, например, Microsoft Visio, средства генерации документации и пр.).

Средства визуального проектирования ускоряют процесс анализа предметной области, моделирования и разработки ИС. Проектировщик работает с общепринятыми языками моделирования типа UML, за счет чего сокращается время разработки.

Предусматривается также возможность расширения этих средств, в частности, через использование средств создания и интеграции в систему предметно-ориентированных языков (DSL – Domain Specific Languages), позволяющих создавать модели с использованием терминов конкретных предметных областей, для которых создается ИС. Применение языков DSL позволяет привлечь к анализу и проектированию ИС, ее настройке специалистов-непрограммистов, экспертов в своих предметных областях.

Интерпретатор метаданных

Одна из главных частей CASE-системы METAS – это *интерпретатор*. Интерпретатор – это runtime-компоненты CASE-технологии, представляющие программную реализацию ИС. Они реализуют различные алгоритмы, основанные на обходах графов (графы представляют модель взаимосвязанных метаданных системы).

В ходе функционирования на основе метаданных генерируется программный код на различных языках (SQL, XML и пр.). С целью повышения производительности функционирования системы применяется кэширование сгенерированного кода ИС. Это относится к любому коду, будь то SQL, XML или исполняемый код платформы, на которой функционирует ИС.

В метаданных, которые являются входным языком для интерпретатора, могут также содержаться скрипты, написанные на встроенном языке системы для манипулирования метаданными и объектами, которые описывают эти метаданные, или на языке средства разработки (SQL, Visual Basic).

Средства адаптации

Метаданные не могут учитывать все особенности разрабатываемых для различных предметных областей информационных систем. Поэтому очень важно реализовать поддержку открытости системы и реализации нестандартной (неформализованной в «стандартных» метаданных) логики.

Характерной чертой современного программного обеспечения является его способность к адаптации. Это и понятно, поскольку такое приложение обладает значительными конкурентными преимуществами. Во-первых, оно решает более широкий круг задач в интересах конечных пользователей. Во-вторых, вокруг адаптируемого приложения возникает сообщество людей (занимающихся адаптацией), которое заинтересовано в увеличении числа конечных пользователей и повсеместном распространении приложения.

Чтобы стать по-настоящему адаптируемым, приложение должно предоставить комфортные условия программистам, занимающимся его адаптацией. Это предполагает наличие хорошо документированных, развитых и удобных в использовании программных интерфейсов. Кроме того, необходимы специализированные инструментальные средства, призванные сделать процесс адаптации максимально простым и удобным.

Помимо этого необходимо обеспечить открытость системы, ее интеграцию с внешними ИС. Внешние приложения должны иметь возможность использования объектов системы аналогично тому, как это делает сама система.

В технологии METAS учет нестандартной логики поддерживается внутренним языком и внешними компонентами интеграции, подключаемыми к системе.

В качестве языка разработки дополнительных программных компонентов, реализующих нестандартную логику, лучше всего использовать Visual Basic, как стандарт встроенного языка приложений CASE-системы.

Помимо этого, предусмотрены средства создания и использования в процессе создания и функционирования ИС дополнительных языков более высокого уровня (языков DSL), позволяющих работать с объектами ИС, описывать бизнес-процессы и т.п. в терминах предметной области. Что позволяет привлекать к разработке экспертов в соответствующих предметных областях.

Открытость системы достигается за счет использования распространенных стандартов доступа к БД, компонентных технологий, распространенных протоколов связи и т.п.

Модели и программные компоненты METAS

Метаданные ИС разделены на слои (рис. 2), представляющие следующие основные модели информационной системы:

- *Физическая модель* (Physical Model) – метаданные, описывающие представление объектов ИС в БД (например, таблиц БД, в которой хранятся данные об объектах, и связей между ними). В процессе функционирования они служат основой логической модели. Модель автоматически генерируется по созданному на логическом уровне описанию системы.
- *Логическая модель* (Logical Model) – метаданные, описывающие сущности предметной области, для которой создается ИС, их поведение (через операции), а также общие операции ИС. Данная модель основывается на нотациях языка UML (используются диаграммы классов) и позволяет работать пользователям системы в терминах предметной области.

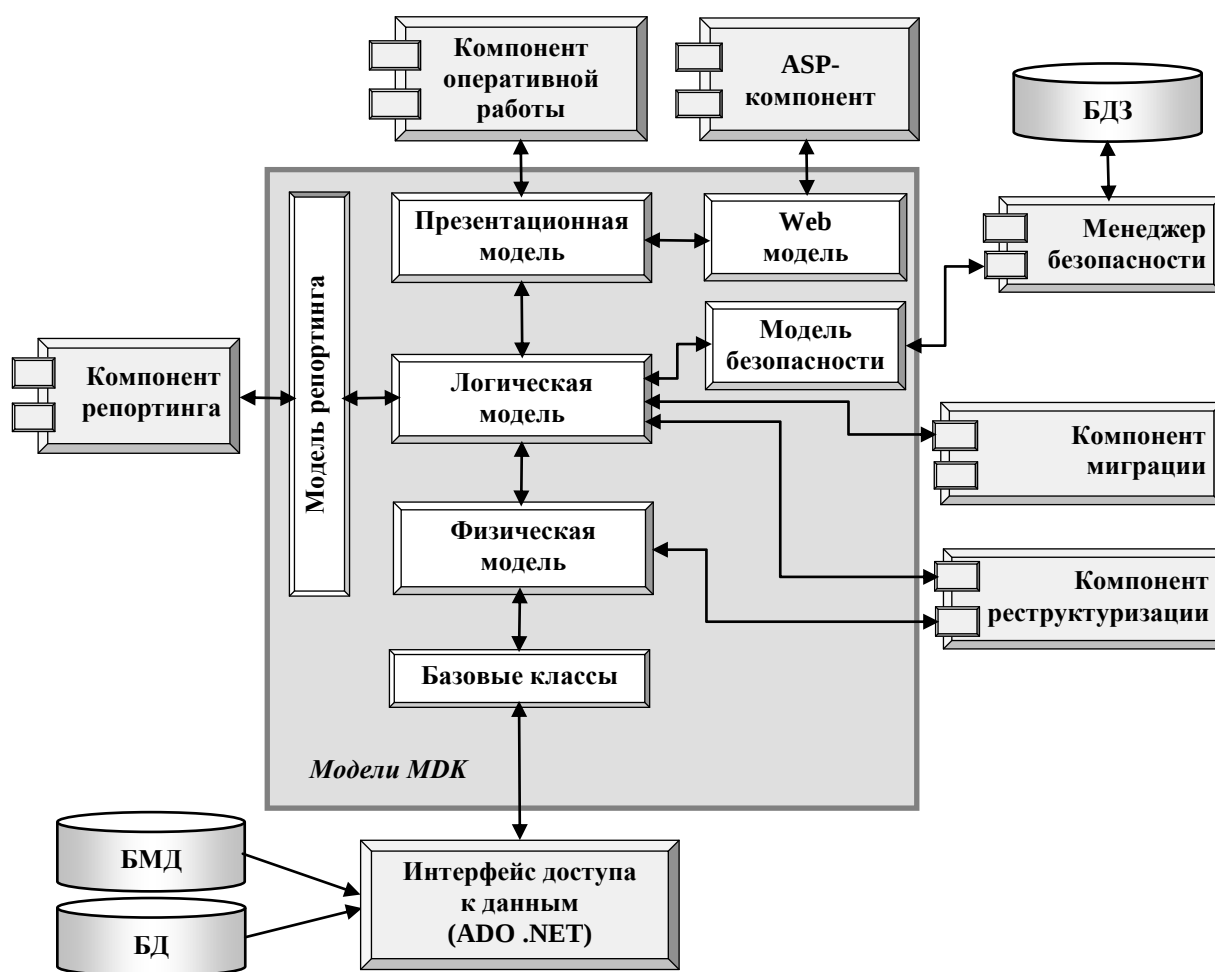


Рис. 2. Метаданные и программные компоненты METAS

- *Презентационная модель* (Presentation Model) – метаданные, описывающие визуальный интерфейс пользователя при работе с объектами ИС.

Набор характеристик, отражаемых в модели, может быть динамически расширен. Набор метаданных может также расширяться путем добавления новых моделей, описывающих новые стороны и свойства ИС или существующие, но с новых точек зрения. В частности, в систему включены следующие модели, опирающиеся на перечисленные выше основные:

- *модель репортинга* (Reporting Model) – метаданные, описывающие запросы, первичные документы и отчеты, формируемые в ходе выполнения бизнес-операций и бизнес-процессов, используемые для анализа данных;
- *модель бизнес-процессов* (Business-process Model) – метаданные, описывающие бизнес-операции и бизнес-процессы, поддерживаемые ИС;
- *Web-модель*, которая обеспечивает доступ к ресурсам ИС для удаленных пользователей через Web-интерфейс.

Модель безопасности (Security Model) позволяет контролировать полномочия пользователей, их права на выполнение операций над объектами ИС или на доступ к моделям метаданных. Подсистема защиты работает с собственной БД (SDB – Security Database).

Основные модели (уровни метаданных), их связь и программные компоненты, работающие с ними, показаны на рис. 2.

CASE-инструментарий позволяет описывать объекты и бизнес-процессы ИС, строить запросы и отчеты в терминах предметной области, настраивать стандартно сгенерированные формы ввода и отображения данных, размещенных в БД системы, а также экспортировать и импортировать модели и данные динамически.

Стандартная бизнес-логика может быть расширена путем определения новых типов и операций, специфичных для конкретной ИС.

Обеспечивается адаптация ИС без перепрограммирования ее компонентов и без участия разработчиков.

Средства интеграции ИС на основе технологии BizTalk Server реализованы как отдельное приложение.

Архитектура информационной системы

Создаваемые с помощью CASE-технологии METAS информационные системы можно использовать на предприятиях среднего и крупного масштаба. Такие предприятия могут иметь территориально распределенные подразделения. Связь между ними может быть организована через высокоскоростную внутреннюю сеть либо через Internet. Поэтому очень важно чтобы архитектура ИС была масштабируемой и высокопроизводительной.

Для обеспечения *масштабируемости* (в идеале должна обеспечиваться независимость времени обработки запроса от количества работающих пользователей), разработана *многоуровневая архитектура*. Компоненты ИС распределяются как по логическим уровням, так и могут перераспределяться по вычислительным узлам для обеспечения максимального баланса их загрузки.

Каждый уровень несет определенную функциональность. Распределение компонентов ИС по уровням может производиться как по выполняемым функциям, так и по обрабатываемым данным. Например, сервер данных и Web-сервер разделены по выполняемым функциям, но сервер данных можно разделить и по обрабатываемым данным.

Начальная настройка ИС может быть оптимальной по производительности, но через некоторый промежуток эксплуатации системы может стать непроизводительной. Это может происходить по ряду причин: выросли объемы данных, выросло количество пользователей, не сбалансирована нагрузка на серверы. Поэтому очень важно иметь возможность перераспределения нагрузки или увеличения количества физических серверов ИС, на которых располагаются программные серверы.

Использование CASE-технологии METAS позволяет создать ИС, архитектура которой представляет собой клиент-серверное приложение (рис. 3), разбитое на *домены*.

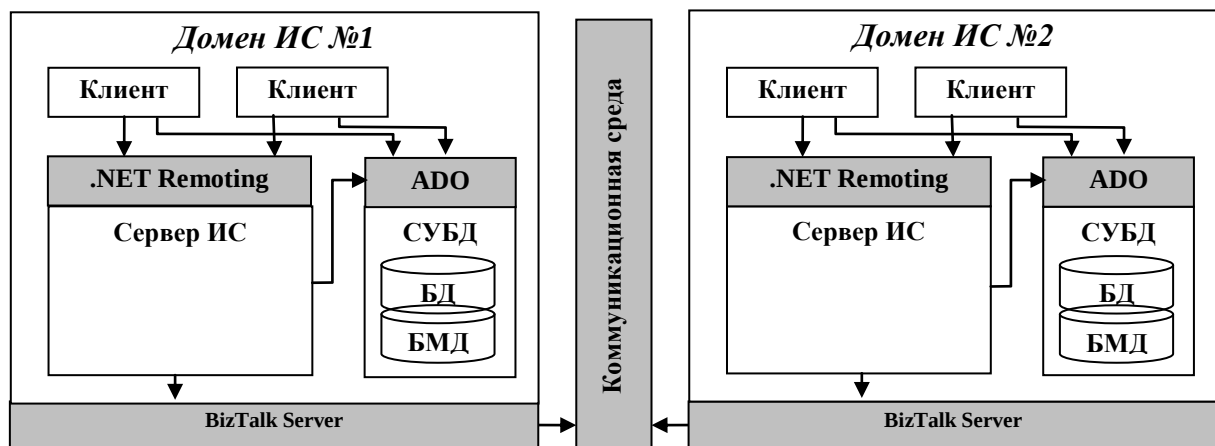


Рис. 3. Доменная архитектура ИС, созданной на основе METAS

Домен ИС – законченная распределенная ИС представляющая собой подсистему ИС, состоящую из серверов системы и клиентов соединенных высокоскоростными линиями связи, и обладающая возможностями по интеграции с другими доменами ИС (рис. 3).

Домен ИС состоит из учетной, аналитической подсистем, сервера интеграции, WEB-сервера и Windows-клиентов (рис. 4).

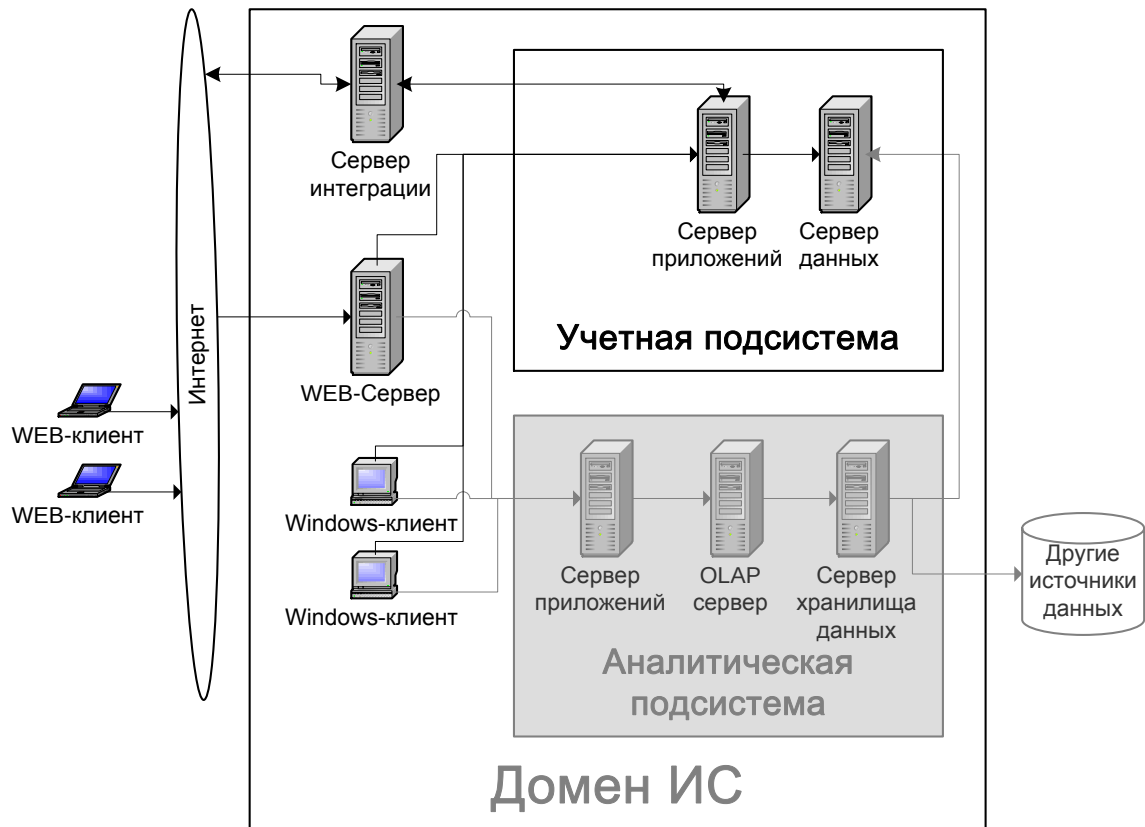


Рис. 4. Многоуровневая архитектура домена ИС

Наличие в домене аналитической системы не обязательно, она может находиться в других доменах.

Web-сервер устанавливается тогда, когда создается портал для обеспечения работы удаленных пользователей системы к ее ресурсам через Web-интерфейс.

Если ИС включает лишь одну подсистему, то и сервер интеграции может отсутствовать.

Все компоненты подсистемы могут выполняться как на одном компьютере для небольших систем, так и на различных компьютерах в локальной сети.

Учетная подсистема – это подсистема оперативной обработки данных – OLTP (On-Line Transaction Processing), которая служит для работы с оперативными данными. Она включает средства редактирования данных, выполнения типовых бизнес-операций и бизнес-процессов, а также средства репортинга (оперативной отчетности).

На *сервере данных* располагается оперативная база данных под управлением СУБД серверного типа.

Сервер приложений является слоем бизнес логики уровня оперативной обработки данных.

Аналитическая подсистема – это подсистема OLAP (On-Line Analytical Processing), которая предназначена для аналитической отчетности, анализа и извлечения нужных данных или закономерностей (Data Mining) и средств принятия решений (DSS – Decision Support System) на основе имеющихся данных. На *сервере хранилища данных* располагается БД с хранилищем и СУБД серверного типа с поддержкой функций хранилищ данных. Сервер хранилища данных может отсутствовать, если кроме базы оперативных данных других источников в этом домене нет. *Сервер OLAP* включает многомерную БД и OLAP-сервер. Источником данных для многомерной БД является хранилище (либо база оперативных данных). *Сервер приложений* является слоем бизнес-логики аналитической обработки данных.

На рис. 4 представлена типовая многоуровневая архитектура отдельной подсистемы ИС. Средства адаптации и интеграции обеспечивают гибкую настройку системы на конкретные условия.

Для объединения доменов ИС в единую распределенную ИС необходимо создать *Корневой домен*, выполняющий функции синхронизации оперативных данных в доменах. Функции корневого домена могут быть также и у доменов среднего слоя в иерархической схеме (рис. 5).

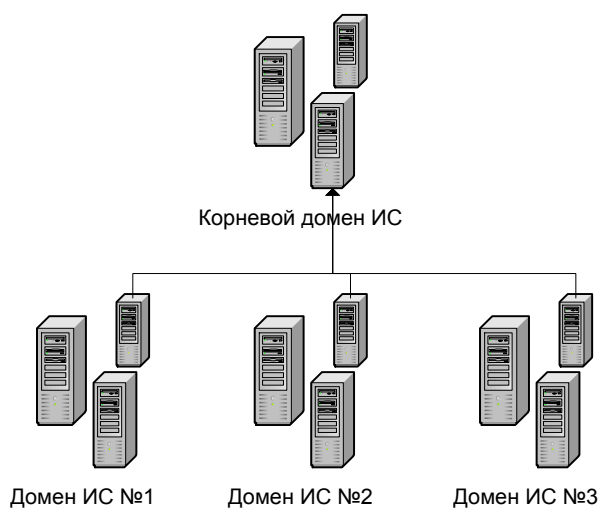


Рис. 5. Иерархическая схема распределенной ИС

Данная схема позволяет выполнять реплицирование данных и метаданных в распределенной ИС, обеспечивает интеграцию подсистем.

Методология и схема разработки ИС

Методология включает понятие этапов жизненного цикла ИС, потоков работ и модели команды. Процесс разработки и внедрения основан на гибридной технологии, которая базируется на спиральных, итерационных технологиях RUP (Rational Unified Process), MFS (Microsoft Solution Framework) и XP (eXtreme Programming) и включает несколько жизненных стадий проекта. Причем из XP взят только подход к тестированию и проектированию этапов, версий, итераций.

Основной упор делается на итеративный способ разработки программного обеспечения. Разработка ИС разделяется на версии, в каждой из которых реализуется определенный набор возможностей. При разработке версий каждый этап также делится на итерации.

Итеративный способ разработки позволяет получить хорошую обратную связь и вовремя реагировать как на изменения требований, так и на возможные проблемы проекта. При этом значительно уменьшаются риски, а в случае неудачного проекта можно намного раньше остановить проект.

Схема функционирования адаптируемой ИС с точки зрения пользователей системы показана на рис. 6.

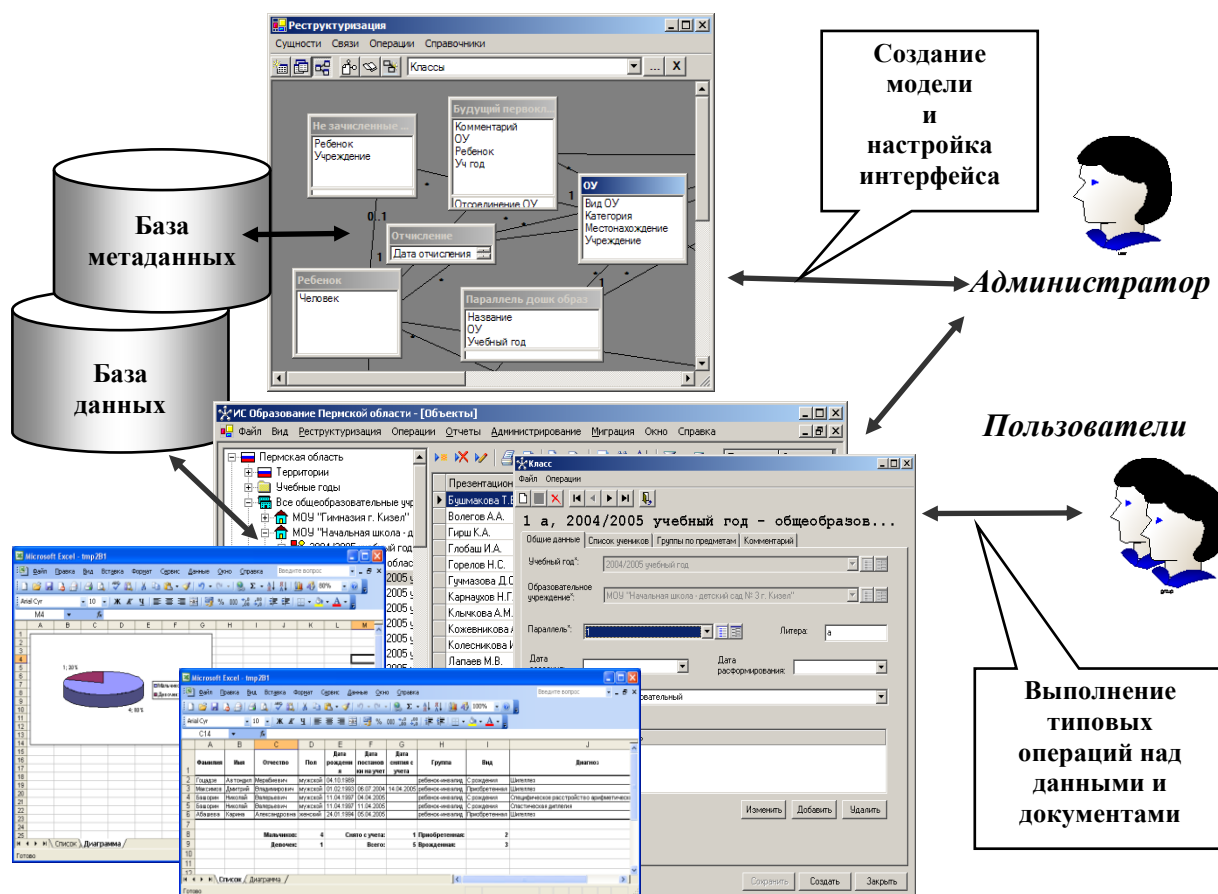


Рис. 6. Схема функционирования динамически настраиваемой системы, управляемой метаданными

Основные средства создания и настройки ИС базируются на использовании компонентов реструктуризации, генерации и настройки интерфейса, генерации SQL-операторов и репортинга.

Средства реструктуризации

С точки зрения пользователя любой объект информационной системы представляется *набором характеристик* (атрибутов). Эти атрибуты могут физически храниться в разных таблицах БД. Поэтому для удобства работы используется обобщающая логическая конструкция – *сущность* – представляющая собой *совокупность этих атрибутов*. Например, можно рассмотреть сущность «Школа» с атрибутами «Номер», «Название», «Адрес» и т.д.

Сущность обеспечивает работу в терминах предметной области без относительно того, как информация представлена на физическом уровне. В программе реструктуризации пользователь оперирует объектами предметной области в терминах сущностей и атрибутов.

Программа предоставляет возможность выполнения следующих основных операций:

- создание сущности, ее атрибутов, операций, индексов и связей между сущностями,
- редактирование свойств сущности, свойств атрибутов, операций, индексов и связей,
- удаление сущности, атрибута, операции, индекса, связи.

Главное окно компонента реструктуризации содержит область, где отображаются сущности (рис.7).

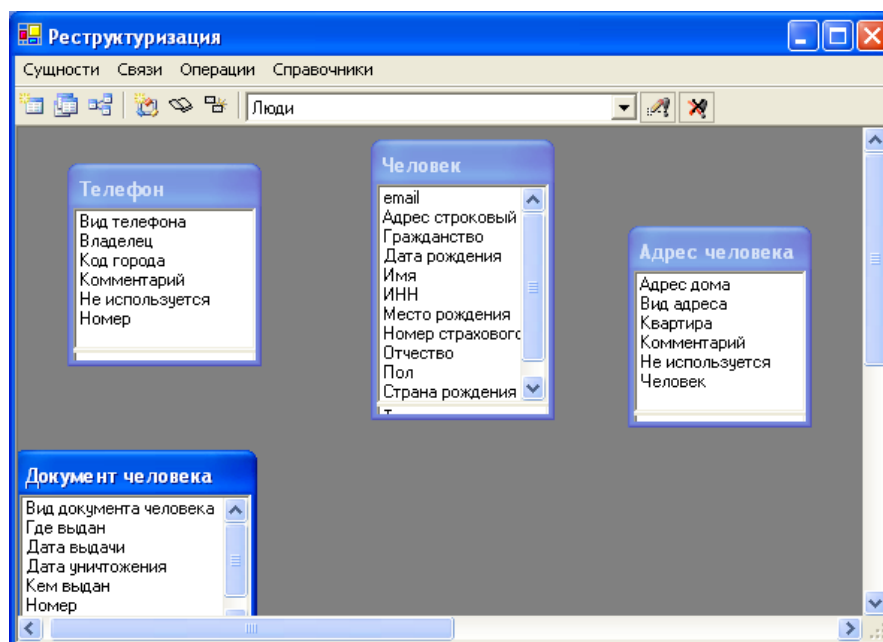


Рис. 7. Общий вид главного окна реструктуризации

В области отображения сущностей каждая сущность представлена своей формой. Эта форма имеет два списка: *список атрибутов* (верхний) и *список операций* (нижний). Размер этих списков можно изменять, перетаскивая разделитель между ними.

Непосредственно под строкой заголовка располагаются *строка меню* и *панель инструментов*. На панели инструментов находятся *кнопки*, отвечающие за отображение сущностей и связей, отображение операций, работу со справочниками, работу с подсхемами, создание сущностей. Пункты меню дублируют действия кнопок на панели инструментов.

Подсхемы

Все сущности предметной области отображаются на подсхемах. Одна и та же сущность может использоваться на нескольких подсхемах (соответствующих подмоделям). Расположение сущности на подсхеме запоминается для каждой конкретной подсхемы. Данная возможность обеспечивает удобство работы с большой предметной областью. Каждая подсхема содержит некоторую часть предметной области. Разделение на подсхемы рекомендуется проводить в зависимости от семантики предметной области.

Для *создания новой подсхемы* нужно выбрать запись «Создать подсхему...» в выпадающем списке, расположенном на панели инструментов в главном окне «Реструктуризация» (в левой части окна) (рис. 8).

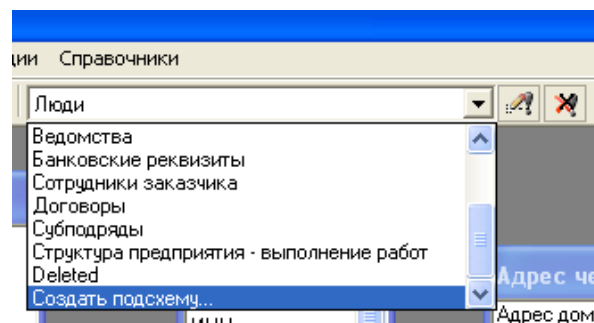


Рис. 8. Выпадающее меню со списком подсхем

Далее в открывшемся окне «Подсхема» (рис. 9) необходимо ввести ее название.

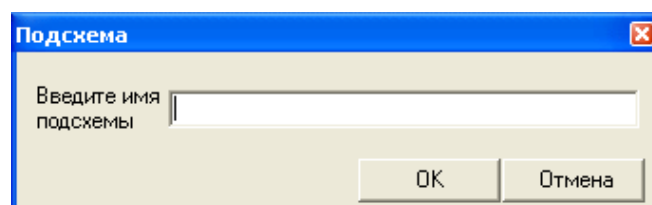


Рис. 9. Форма для изменения названия подсхемы

Если подсхема создана, то в выпадающем списке отображается ее название, область отображения сущностей очищается. При создании подсхемы никакие сущности автоматически в нее не добавляются.

Чтобы *изменить название текущей подсхемы*, следует вызвать форму для его редактирования кнопкой .

Чтобы *удалить текущую подсхему*, следует нажать кнопку  рядом с выпадающим списком. После этого будет выдан запрос на подтверждение удаления.

Можно *изменить или просмотреть набор сущностей*, входящих в подсхему. Это можно сделать с помощью контекстного меню, вызываемого щелчком мыши в области отображения сущностей, или с помощью команд главного меню окна компонента реструктуризации (рис. 10).

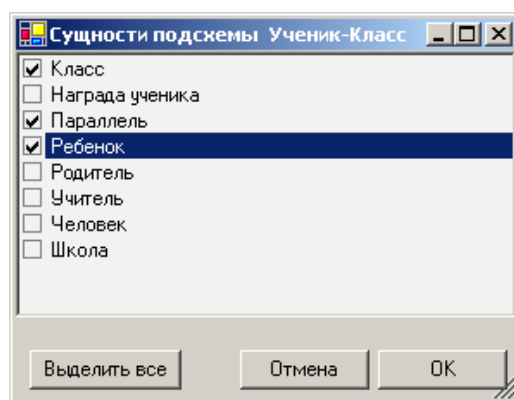


Рис. 10. Список сущностей подсхемы

В случае, когда на подсхему добавляется сущность, имеющаяся связи с другими сущностями, уже отображенными на подсхеме, ее связи также отобразятся на подсхеме.

Удаление сущности с подсхемы (то есть снятие установленного флажка) не ведет к удалению сущности из предметной области и не влияет на отображение этой сущности на других подсхемах.

Сущности

Все операции, которые можно выполнить для сущности, доступны через *контекстное меню*, которое вызывается щелчком правой кнопки мыши на форме сущности, или через команды главного меню. Новая сущность включается в модель с помощью операции *создания сущности*.

Далее в окне «Свойства сущности» (рис. 11) необходимо заполнить предлагаемые поля. *Изменить свойства сущности* можно с помощью соответствующей команды меню. Поле «Имя сущности» является *обязательным* для заполнения (его наименование помечено звездочкой). В случае если оно не заполнено, справа от поля появляется восклицательный знак, сигнализирующий об ошибке. Наведение на него указателя мыши позволяет увидеть текст сообщения об ошибке.

Рис. 11. Вид окна Свойства сущности

Важно, чтобы имя сущности было информативным (соответствовало понятию предметной области, ее объекту), так как оно используется в приложении для ввода первичных данных, а также при построении отчетов (рис. 12).

Рис. 12. Форма для ввода данных об ученике

В поле «Описание» заносится *дополнительная информация о сущности*. Эта информация представляет собой комментарий для разработчика, аналитика: он может указать, зачем создана сущность, с чем она связана, какие-то данные о ней из предметной области. В поле «SQLWhere» заносятся дополнительные ограничения, накладываемые на сущность.

Поле «Выражение для презентационного поля» позволяет задать *ключевую информацию о сущности*. Однако при создании сущности заполнить данное поле невозможно, т.к. оно может быть заполнено пользователем, если у сущности уже имеются атрибуты (при создании сущности в качестве выражения для презентационного поля автоматически указывается ее идентификатор; это значение можно изменить позднее, редактируя свойства сущности).

Поле «Тип темпоральности» позволяет указать, следует ли отслеживать изменение сущности во времени.

В разделе «Атрибуты защиты» задаются *атрибуты защиты сущности*. Они ограничивают полномочия всех пользователей на работу с сущностью, определяя операции, которые можно для нее выполнять. Эти атрибуты могут отменять права, назначенные пользователям в подсистеме защиты. Атрибуты защиты могут использоваться, в частности, для того, чтобы запретить добавление, изменение и удаление значений из справочников и классификаторов, используемых в системе. Наличие такой возможности необходимо, так как требуется соблюдать единство представления и кодирования данных, их сопоставимость, поэтому нельзя давать возможность всем пользователям вносить в них изменения — они меняются только централизованно.

Дата модификации проставляется автоматически.

Для *удаления сущности* следует выделить ее щелчком правой кнопки мыши и выбрать пункт «Удалить сущность» в появившемся контекстном меню. Удаление сущности означает ее удаление из предметной области, т.е. данная операция удалит сущность из всех подсхем, в которые она включалась, и уничтожится любая информация об этой сущности. Для упрощения *одновременного удаления нескольких сущностей* служит операция «Удалить сущности», которая доступна в пункте «Сущности» окна реструктуризации.

Для *просмотра имеющихся экземпляров* некоторой сущности можно воспользоваться пунктом «Список экземпляров» контекстного меню, вызываемого щелчком правой кнопки мыши на форме интересующей сущности. При этом появляется форма приложения для первичного ввода данных (рис. 13).

Чтобы *настроить внешний вид формы* для первичного ввода данных, соответствующей сущности, следует выбрать пункт «Форма редактирования» контекстного меню, вызываемого щелчком правой кнопки мыши на форме сущности. При этом откроется форма, которая будет находиться в режиме настройки (рис. 14).

В этом режиме можно менять размеры и положение элементов управления на форме, переносить элементы управления с одной страницы формы на другую, добавлять новые страницы, отображать или скрывать связанные дочерние сущности, менять заголовки.

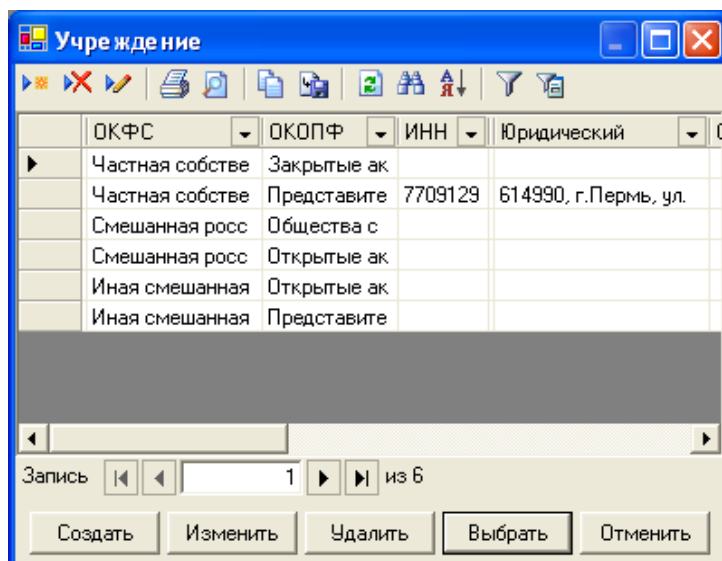


Рис. 13. Форма со списком экземпляров сущности

Рис. 14. Режим настройки формы редактирования сущности

Для любой сущности, отображаемой на некоторой подсхеме компонента реструктуризации, доступна дополнительная информация, позволяющая понять место сущности в предметной области. Для ее получения надо выбрать пункт «Информация о сущности» контекстного меню формы сущности. При этом отобразится форма, показанная на рис. 15.

В центральной части окна графически отображаются связи данной сущности с другими сущностями системы. Следует отметить, что связи остальных сущностей между собой в данном окне не показываются.

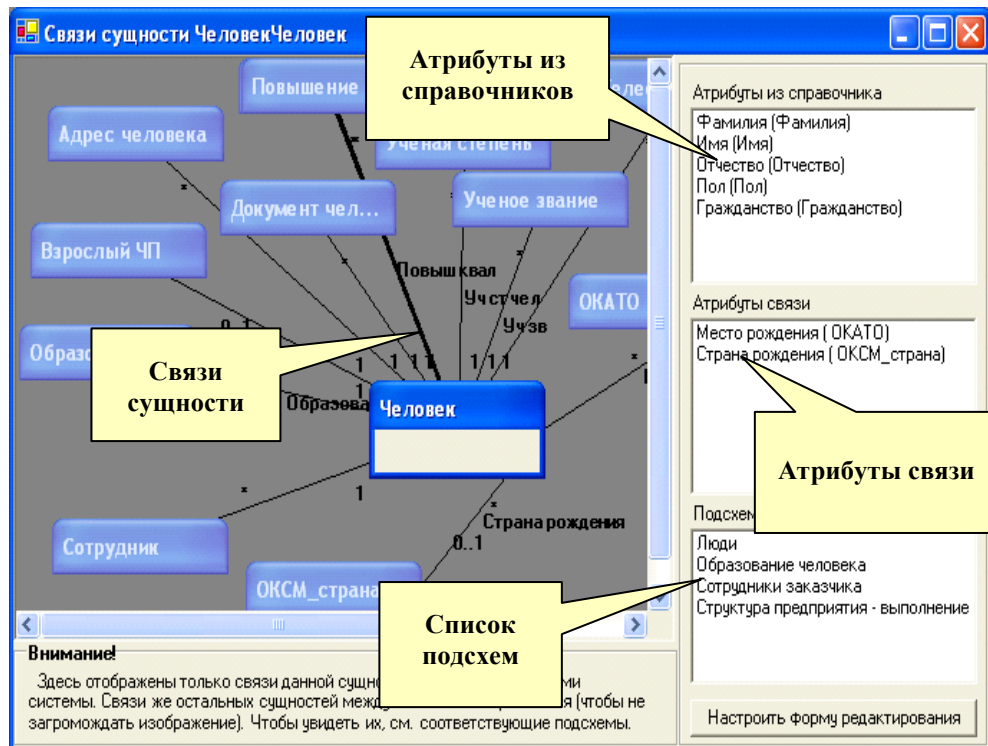


Рис. 15. Информация о сущности

Справа в верхнем списке перечислены атрибуты данной сущности, значения которых выбираются из справочников (подробнее о типах атрибутов см. следующий раздел настоящего руководства). В среднем списке указаны атрибуты, по которым устанавливается связь рассматриваемой сущности с остальными сущностями предметной области. Нижний список содержит названия всех подсхем, на которых отображается сущность.

Созданная сущность является «пустой». Для представления информации об объекте в БД необходимо определить его атрибуты. Атрибуты выражают свойства сущности, либо определяют ее текущее состояние.

При *работе с атрибутами* в пользовательском приложении для ввода и редактирования первичных данных в диалоговых окнах, представляющих собой формы для заполнения и отображения информации о сущностях при выполнении пользователями типовых операций, значения атрибутов экземпляра сущности могут как вводиться с клавиатуры, так и выбираться из справочника, либо выбираться из списка других сущностей. Поэтому условно все атрибуты можно разделить на три группы:

- собственные атрибуты сущности,
- атрибуты, выбираемые из справочника,

- внешние атрибуты, или атрибуты связи (для работы с другими сущностями).

Работа с атрибутами третьего типа в программе реструктуризации отличается от работы с атрибутами первых двух типов. А именно: такой атрибут нельзя создать непосредственно – он создается автоматически при создании связи, также его нельзя удалить – он удаляется только при удалении связи. Однако, поскольку данный атрибут используется в приложении для ввода первичных данных, то программа реструктуризации предоставляет возможность изменения некоторых характеристик этих атрибутов (см. ниже).

Для работы с атрибутами сущности предназначена форма «Конструктор сущности», общий вид которого показан на рис. 16. Чтобы вызвать эту форму можно воспользоваться командой меню. С ее помощью можно задать новые атрибуты, изменить существующие. Конструктор позволяет задать имена и типы атрибутов, ограничения для их значений, способ ввода/вычисления значений, а также атрибуты защиты.

Название атрибута	Тип атрибута	Обязательность	Темпоральность	Просмотр	Изменение	Описание
Фамилия	Текстовый(255)	☒	☐	☒	☒	
Имя	Текстовый(255)	☒	☐	☒	☒	
Отчество	Текстовый(255)	☐	☐	☒	☒	
Дата рождения	Дата/Время	☒	☐	☒	☒	
email	Текстовый(50)	☐	☐	☒	☒	
Адрес строковый	Текстовый(100)	☐	☐	☒	☒	Неструктурированный адрес
Место рождения	Двойное с плавающей точкой	☐	☐	☒	☒	Ссылка на OKATO
Пол	Текстовый(50)	☒	☐	☒	☒	
Страна рождения	Целое	☐	☐	☒	☒	Ссылка на OKCM_страна
Гражданство	Текстовый(100)	☐	☐	☒	☒	
ИНН	Текстовый(50)	☐	☐	☒	☒	
Номер страхового свиде	Текстовый(50)	☐	☐	☒	☒	

* Свойства атрибута

☐ допускаются изменения значения для группы экземпляров сущности Значение:

Значение по умолчанию: Маска: Выражение для вычисления:

Индексированное поле: Вводится при условии:

☒ значения выбираются из справочника

☐ из нового ☒ из существующего

Рис. 16. Вид конструктора сущности

Операции сущности – это операции, которые производятся данной сущностью или над данной сущностью. Для *добавления операции сущности* нужно вызвать пункт «Добавить операцию» контекстного меню для нижнего списка (списка операций) на форме сущности. Появится форма для работы с операциями (рис. 17).

Рис. 17. Вид формы для редактирования операции сущности

Операция сущности может быть реализована сторонними разработчиками и интегрирована в систему с помощью показанной на рис. 17 формы, на которой указывается:

- «Путь к файлу» – надо указать путь к файлу .dll, в котором находится объект, поддерживающий интерфейс IEntityOperation. Этот объект будет выполнять добавляемую операцию.
- «Тип объекта» – надо выбрать нужный объект из выпадающего списка. Этот список содержит все объекты в выше указанном файле .dll.
- «Заголовок операции» – нужно ввести заголовок операции, который будет отображаться в меню приложения для ввода первичных данных.
- «Описание операции» – здесь вводится описание операции, то есть что она делает, какую информацию использует и т.д. Эта информация будет использоваться в качестве подсказки к пункту меню, соответствующему данной операции.
- «Дополнительные свойства» – заполняются в том случае, если для выполнения операции необходима дополнительная информация.

Построитель выражений

«Построитель выражений» используется для задания SQL-подобного выражения, составленного из атрибутов некоторой сущности и операций, выполняемых над ними. Вид окна «Построителя выражений» показан на рис. 18.

При вызове «Построителя выражений» в списке «Атрибуты сущности» отображаются все доступные атрибуты сущности, как собственные, так и ссылочные, используемые для организации связи с другими сущностями. Каждая строка этого списка содержит наименование атрибута, используемое в предметной области, в скобках указывается внутреннее имя атрибута, которым оперирует система. Например, Фамилия(Afamiliya).

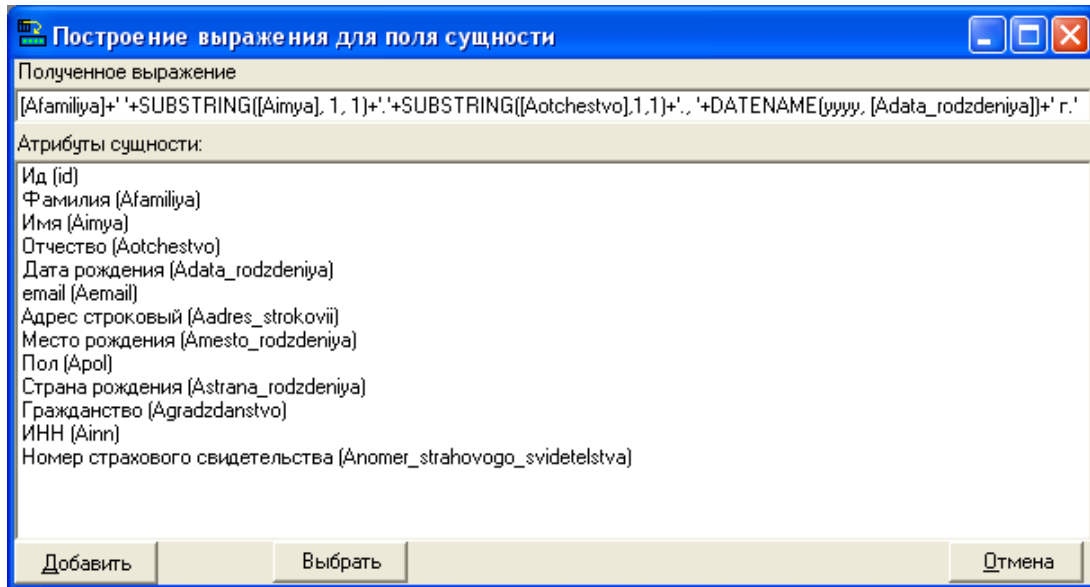


Рис. 18. Окно построителя выражений

Если формируемое выражение должно содержать несколько атрибутов, они должны быть не просто перечислены в строке, а связаны в выражение, которое удовлетворяет синтаксису SQL. Например, если для сущности «Ученик» необходимо сформировать выражение из его фамилии, имени и отчества, то соответствующие атрибуты необходимо представить в виде одной строки. В такой ситуации нужно воспользоваться *операцией конкатенации строк*, задаваемой с помощью знака «+», т.е. итоговая строка будет иметь вид

[Afamiliya]+[Aimya]+[Aotchestvo]

Кроме склеивания строк в формируемых выражениях допускается использовать все *стандартные функции языка SQL*. Их использование может понадобиться, например, для выделения подстрок, замены пустых значений, форматирования и т.п. Так выражение для фамилии и инициалов имеет вид

[Afamiliya]+' '+SUBSTRING([Aimya], 1, 1)+' '+SUBSTRING([Aotchestvo],1,1)+' '

где SUBSTRING – функция выделения подстроки.

Связи между сущностями

Между сущностями могут *существовать некоторые отношения, отражающие отношения между объектами реального мира*, представленными этими сущностями. Они обуславливают наличие связей.

Важной характеристикой связи является множественность, которая определяет, сколько экземпляров одной сущности можно связать с экземплярами другой сущности. Таким образом, множественность ограничивает число связанных объектов.

В системе реализована поддержка следующих видов связей:

– **0..1:1** – связь ОДИН-К-ОДНОМУ. Означает, что каждому экземпляру

Сущности2 соответствует 1 или 0 экземпляров Сущности1, и для каждого экземпляра Сущности1 найдется единственный экземпляр Сущности2 (рис. 19).



Рис. 19. Связь типа 0..1 : 1

- **1:M** – связь ОДИН-КО-МНОГИМ. Означает, что одному экземпляру Сущности1 соответствуют 0, 1 или несколько экземпляров Сущности2, и для каждого экземпляра Сущности2 находится соответствующий экземпляр Сущности1, причем только один (рис. 20).

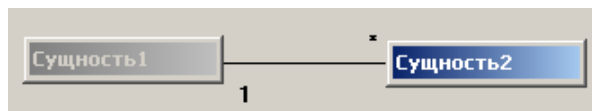


Рис. 20. Связь типа 1:M

- **0..1:M** – связь НОЛЬ-ОДИН-МНОГИЕ. Означает, что одному экземпляру Сущности1 соответствуют 0, 1 или несколько экземпляров Сущности2, а для экземпляра Сущности2 может найтись соответствующий экземпляр Сущности1, но не всегда (рис. 21).

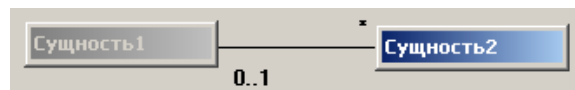


Рис. 21. Связь типа 0..1:M

- **M:M** – связь МНОГИЕ-КО-МНОГИМ. Означает, что каждому экземпляру Сущности1 соответствуют 0, 1 или несколько экземпляров Сущности2, и наоборот (рис. 22).



Рис. 22. Связь типа M:M

- Связь произвольной множественности (рис. 23).

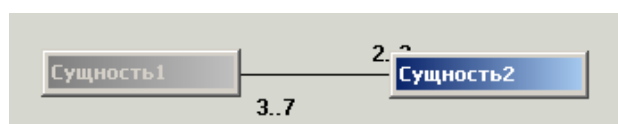


Рис. 23. Связь произвольной множественности

Поскольку прорисовка связей требует больших затрат ресурсов, то предусмотрено два режима: с отображением связей и без отображения. Для переключения между ними можно воспользоваться соответствующим пунктом контекстного меню главного окна или кнопкой на панели инструментов . Если она находится в «нажатом» состоянии, то связи отображаются (рис. 24).

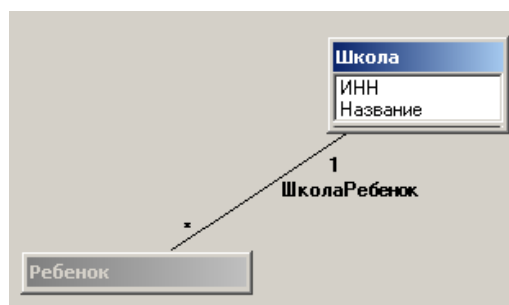


Рис. 24. Отображение связей между сущностями

Для создания связи следует вызвать форму «Связь между сущностями» (рис. 25) с помощью контекстного меню, меню окна реструктуризации или с помощью кнопки панели инструментов.

Связь между сущностями

Сущность: Параллель | Сущность: Класс

Название: включает

Описание: Каждый класс относится к некоторой параллели

☐ каскадное удаление связанных полей

Тип связи*

☐ 1:0..1 ☐ другой левый конец (сущность) 0 .. 0

☒ 1:M

☐ 0..1:M правый конец (сущность) 0 .. 0

☐ M:M

☐ Допустимость группового назначения значений ссылки ->

☐ Допустимость группового назначения значений ссылки <-

OK Отмена

Рис. 25. Вид формы для создания связи

Для *редактирования связи* между сущностями надо при включенном режиме отображения связей выделить нужную связь щелчком мыши по этой связи. Линия, отображающая связь, станет толстой (рис. 26). Далее следует вызвать пункт «Свойства связи» контекстного меню.



Рис. 26. Выделение связи

Для *удаления связи* между сущностями надо при включенном режиме отображения связей выделить нужную связь щелчком мыши по этой связи. Далее следует вызвать пункт «Удалить связь» контекстного меню, подтвердить удаление нажатием кнопки «ОК» в появившемся окне.

Иногда в процессе уточнения предметной области может оказаться, что ранее созданную связь типа «М:М» удобнее представить в виде сущности, которая имеет связи типа «1:М» с ранее связанными сущностями. Такая ситуация возникает в следующих случаях:

- сама связь имеет некоторые характеристики,
- связь участвует в ассоциации с другими сущностями,
- над связью могут выполняться операции.

Для преобразования связи типа «М:М» в сущность нужно выделить эту связь щелчком мыши по изображающей ее линии и воспользоваться пунктом «Преобразовать в сущность» контекстного меню для связи.

После выполнения данной операции:

- создается новая сущность, имя которой представляет собой конкатенацию имен двух сущностей, связь между которыми преобразуется;
- создаются связи «1:М» с каждой из ранее связанных сущностей.

Если сущность имеет связь типа «0..1:М» с несколькими другими сущностями, участвуя со стороны «М», то возможно, что экземпляр сущности может ссылаться только на экземпляр одной из этих сущностей. В этом случае мы говорим, что существует одна связь *ИЛИ* другая. Будем называть связи, которые не могут существовать у экземпляра сущности одновременно *взаимоисключающими* или *альтернативными*. Отметим, что может существовать более двух альтернативных связей, т.е. такие связи образуют группу альтернативных связей с произвольным числом членов. На практике это, как правило, две связи.

Индексы

Основное назначение индексов состоит в *обеспечении эффективного доступа к данным*. Индексы позволяют «упорядочить» информацию об объектах, отсортировать ее в зависимости от значений заданных атрибутов. Упорядочение информации ускоряет поиск нужных данных при выполнении пользователями операций. Однако следует учитывать, что создание индекса на большой заполненной таблице (когда информационная система функционирует) требует большого количества времени, для хранения индексов дополнительно резервируется дисковая память. Поэтому при их создании необходимо определить, какие запросы будут выполняться наиболее часто.

Для каждой сущности может быть создано несколько индексов, определяющих порядок сортировки и поиска информации по различным атрибутам. В индекс может входить один или несколько атрибутов. Некоторый атрибут может включаться в состав нескольких индексов, причем с разным порядком сортировки.

Для просмотра индексов сущности и работы с ними используется форма «Индексы сущности», которая вызывается с помощью пункта «Индексы» контекстного меню, вызываемого щелчком правой кнопки мыши на форме сущности. В списке, расположенном слева на этой форме, отображаются все индексы (рис. 27). Для просмотра свойств конкретного индекса его следует выбрать в списке.

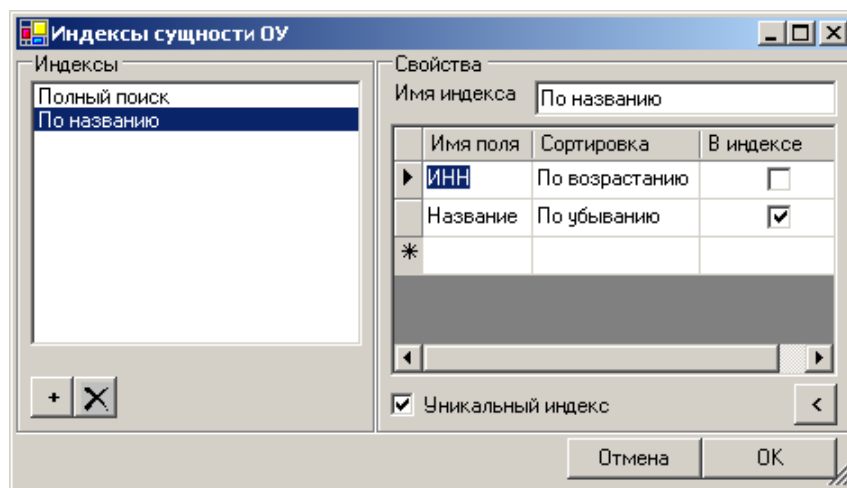


Рис. 27. Форма для работы с индексами

Ограничения сущности

Для сущности можно задать ограничения, которые определяют допустимые значения для атрибутов этой сущности. Эти ограничения задаются с помощью формы «Ограничения на сущность» (рис. 28). Для вызова этой формы следует воспользоваться пунктом «Ограничения» контекстного меню формы сущности.

В столбце таблицы «Сообщение об ошибке» указывается текст сообщения, которое выдается в пользовательском приложении, если для сущности, указанные значения атрибутов противоречат заданным ограничениям.

Для добавления ограничения в таблицу на форме добавляется строка. В поле «Код ограничения» заносится текст ограничения. В системе принято два вида ограничений:

- ограничения на встроенном скриптовом языке MDKScript,
- SQL-подобные ограничения.

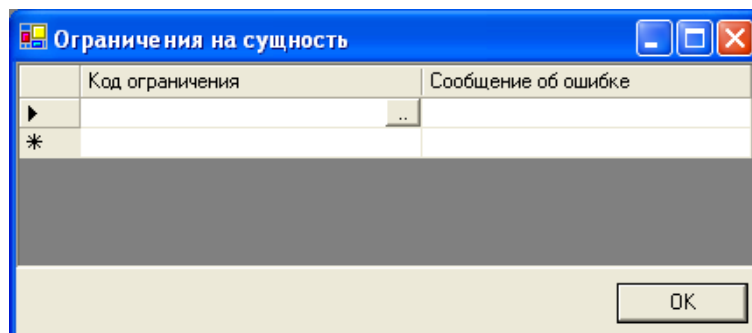


Рис. 28. Форма для задания ограничений сущности

Если выбрано ограничение в SQL-выражении, то будет вызван «Построитель выражений» (см. выше). Если выбрано ограничение на MDKScript, будет вызван «Редактор скриптов». Вид его окна показан на рис. 29.

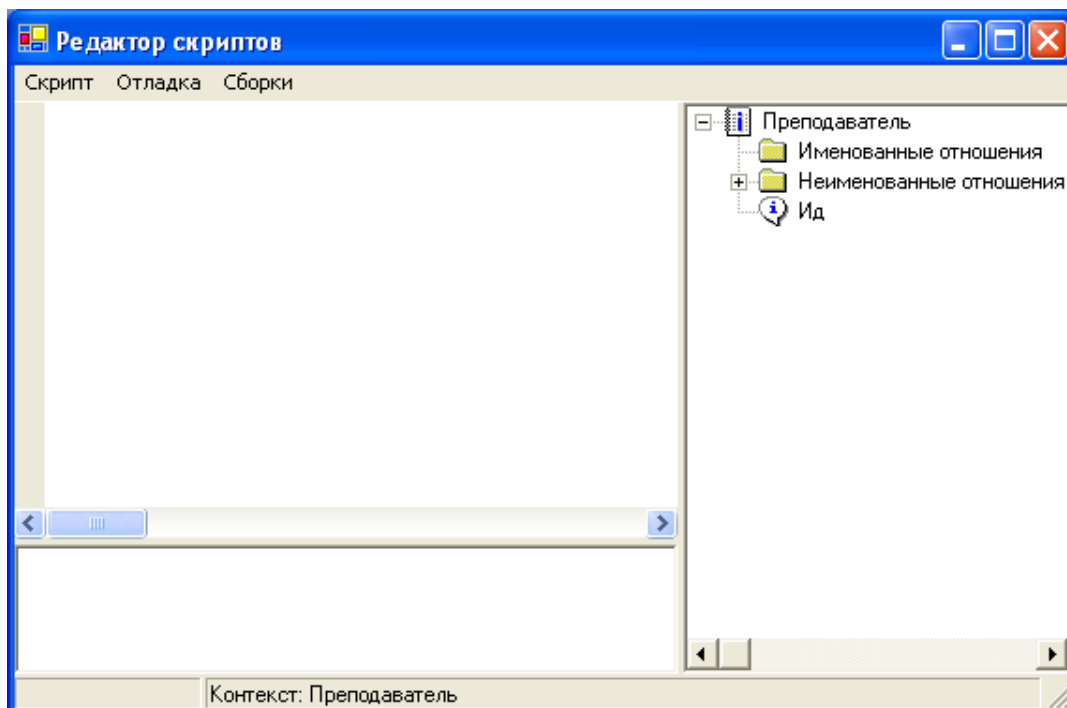


Рис. 29. Вид окна редактора скриптов

После того, как скрипт откомпилируется в данном редакторе, и редактор закроется, текст скрипта попадает в поле «Код ограничения», а исполнимый файл в виде dll, который будет запускаться на выполнение при проверке ограничения в ходе работы приложения, сохраняется в каталоге системы.

Для того чтобы удалить некоторое ограничение, достаточно удалить в таблице на форме соответствующую ему строчку.

Справочники и классификаторы

В данной версии программы предусматривается работа только с одноколоначатыми справочниками. Это означает, что справочник содержит только одно информационное поле, доступное для пользователя. Кроме него также имеется поле-идентификатор, но доступа к нему пользователь не получает.

Работа со справочниками производится с помощью формы «Справочники» командой меню или с помощью кнопки на панели. Форма «Справочники» имеет две вкладки: используемые справочники и неиспользуемые справочники. Ее общий вид показан на рис. 30.

Значения	
id	vidobucheniya
1	очное
2	очно-заочно
3	заочное
4	экстернат
5	семейное о
6	самообразо

Рис. 30. Вид формы для работы со справочниками

Для создания справочника следует нажать кнопку «Создать справочник». При этом появляется окно «Создание справочника», показанное на рис. 31.

В открытом окне можно ввести имя справочника, отражающее его назначение, тип информационного поля (как и для атрибутов сущности), а также

уровень защиты: есть справочники, которые после создания *не может редактировать никто* (например, «Пол»), есть справочники, значения которых можно менять только администратору (разработчику) на основании нормативных актов, а есть справочники, которые используются только для снижения трудоемкости работы пользователя, экономии (имена, например).

Рис. 31. Вид формы для создания справочника

На вкладке «Используемые справочники» можно лишь просмотреть, содержащиеся в нем значения. Чтобы *внести изменения в значения*, необходимо нажать кнопку «Значения». При этом появится окно, показанное на рис. 32.

	id	vidadresa
▶	1	По прописке (р
	2	Фактический
	3	Другой
	4	23
*		

Рис. 32. Значения справочника

Чтобы добавить значение, надо добавить новую строку. Удалить ненужное значение можно удалением соответствующей строки.

Изменять состав значений можно как для используемых, так и для неиспользуемых справочников.

Удалять разрешается лишь те справочники, на которые не ссылается ни один атрибут. Поэтому кнопка «Удалить справочник» располагается на вкладке «Неиспользуемые справочники». Если нужно удалить используемый справочник, то сначала следует изменить свойства всех атрибутов, которые на него ссылаются.

Документирование структуры БД

В системе имеется возможность генерировать описание структуры БД – ее подмоделей (подсхем). Сгенерированные описания сохраняются в файлах MS Word.

Тиражирование подмоделей

Средства системы позволяют тиражировать изменения, внесенные в структуру данных (расширение модели), передавать их в другие подсистемы. Это дает возможность расширить функциональность в ходе эксплуатации системы.

Операции создания копии подсхемы (новой подмодели, расширяющей существующую модель) и импорта ее в базу метаданных другой системы находятся в меню «Подсхемы».

Средства настройки интерфейса пользователя

Основными понятиями логического уровня являются сущности, атрибуты, связи между сущностями, а также экземпляры сущностей или объекты данных.

Метаданные презентационного уровня описывают элементы Windows-интерфейса: экранные формы, страницы, группы данных, поля ввода информации различного типа, меню, навигационную систему приложения. Идея автоматического создания форм ввода и редактирования данных основана на построении соответствия между логическим уровнем метаданных и их визуальным представлением на экране пользователя.

Пользовательский интерфейс можно разделить на две части:

- *проводник* или *дерево объектов предметной области*, представляющий объекты предметной области в виде иерархии и позволяющий быстро просматривать объекты;
- *формы ввода и редактирования данных* для работы с детальной информацией об объектах

Общая схема работы компонентов управления интерфейсом представлена на рис. 33.

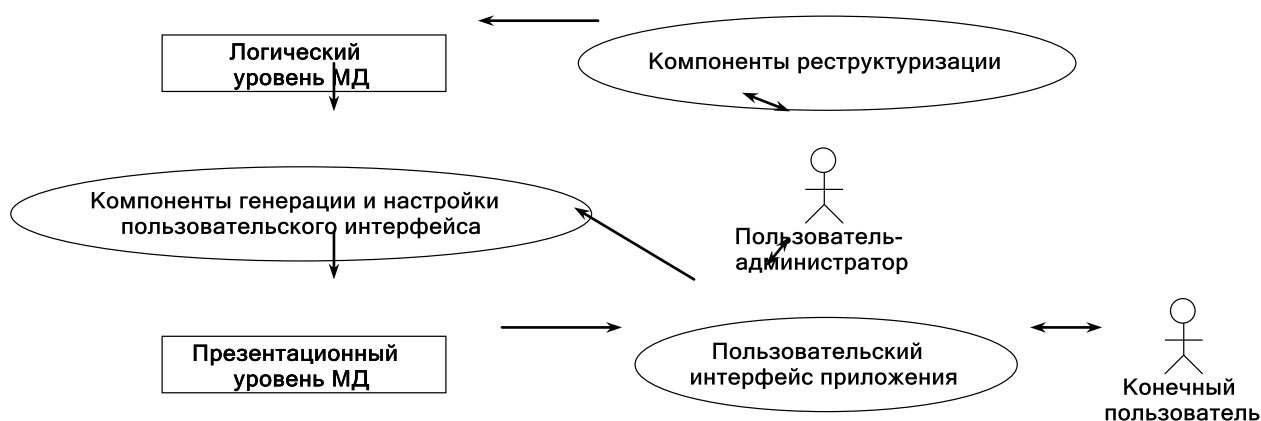


Рис. 33. Схема работы компонентов управления интерфейсом

Компоненты генерации и настройки пользовательского интерфейса автоматически по описанию логического уровня строят элементы интерфейса приложения, касающиеся форм ввода данных. Пользователем-администратором настраивается иерархия объектов предметной области. В результате получается прототип интерфейса приложения, обладающий полным набором функций по вводу и редактированию данных. После этого приложение может быть настроено пользователем-администратором, т.е. на формы могут быть внесены родительские и дочерние объекты, а также дописана нестандартная функциональность путем подключения дополнительных элементов управления, обработки событий загрузки форм и сохранения данных. Изменения, происходящие в логической модели посредством компонентов реструктуризации, изменяют и презентационную модель.

Настройка элементов интерфейса пользователем-администратором происходит с помощью тех же форм и дерева объектов предметной области, которые использует для просмотра и редактирования данных конечный пользователь. Администратор работает в привилегированном режиме настройки, внося изменения во внешний вид приложения. Сделанные им изменения через компоненты генерации и настройки интерфейса вносятся на презентационный уровень метаданных.

Доступ к БД осуществляется через логическую модель метаданных, описывающую объекты предметной области. Логическая модель в свою очередь транслирует запросы к БД на выборку и модификацию данных через физический уровень, оперирующий понятиями БД. Запрос в терминах физического уровня передается на выполнение посредством ADO.Net. Выбранные данные отображаются в формах ввода данных, в дереве объектов.

Интерфейс пользователя представляет собой набор последовательно раскрывающихся форм и списков объектов определенного типа. Работа пользователя начинается с дерева объектов, реализуемого специальным программным компонентом. Последовательно раскрывая вершины дерева объектов, пользователь переходит к нужной ему вершине, над которой может производить как стандартные операции (просмотр, вставка, удаление, редактирование), так и бизнес-операции (через вызов компонентов бизнес-логики).

Из дерева объектов открываются формы ввода/редактирования данных и списковые формы. Формы могут вызывать другие формы в соответствии со связями в графе презентационной модели и параметрами, сохраненными в метаданных. Логика работы форм ввода/редактирования данных реализована отдельным компонентом. Для генерации и настройки форм используется специальный программный компонент генерации и настройки экранных форм.

Настройка форм

Программное обеспечение системы допускает возможность динамической настройки баз данных на потребности пользователей. В частности, имеется возможность расширения данных, представленных в БД. При этом приложение автоматически настраивается на работу с новыми данными, размещая элементы управления для отображения этих данных на формах.

Если форма не была настроена вручную, то она содержит одну вкладку, на которой расположены все элементы управления, необходимые для ввода информации об объекте предметной области. При этом элементы управления расположены в том порядке, в котором они описаны в базе метаданных, т.е. в порядке следования атрибутов объекта (рис. 34).

Рис. 34. Автоматически сгенерированная форма ввода/редактирования

Для *настройки формы* пользователь-администратор должен перейти в режим настройки, воспользовавшись для этого контекстным меню формы (пункт «Режим работы ▸ Настройка»), вызываемым щелчком правой кнопки мыши по форме. Аналогичным образом происходит возврат в режим ввода/редактирования данных (пункт «Режим работы ▸ Редактирование» или «Режим работы ▸ Просмотр»).

В режиме настройки пользователь может располагать элементы управления по своему усмотрению, изменять их размер, заголовки, создавать вкладки, группы элементов управления, перемещать элементы управления между вкладками и группами, а группы элементов управления – между вкладками. Все операции доступны пользователю-администратору через контекстные меню

элементов формы. Далее подробно рассматриваются операции, производимые над формой редактирования.

Для создания вкладки (страницы) необходимо:

- Вызвать контекстное меню формы.
- Выбрать пункт «Добавить страницу».
- В появившейся форме ввести заголовок созданной страницы.
- Нажать кнопку «Сохранить».

Пример получившейся формы показан на рис. 35.

Рис. 35. Форма редактирования после создания новой вкладки

Для *удаления вкладки* нужно выделить нужную вкладку; вызвать контекстное меню щелчком правой кнопки по вкладке; выбрать в контекстном меню пункт «Удалить страницу».

Вкладку можно удалить, если она не содержит элементов управления. Если вкладка не пуста, то надо предварительно перенести все поля ввода на другие вкладки или группы формы (см. операцию перемещения элементов управления между группами и вкладками формы, описанную ниже).

Создание группы происходит следующим образом:

- Выбрать вкладку для размещения группы.
- Выбрать пункт контекстного меню «Добавить группу».
- В появившейся форме ввести название группы.
- Нажать кнопку «Сохранить».

Результат может быть таким, как показано на рис. 36.

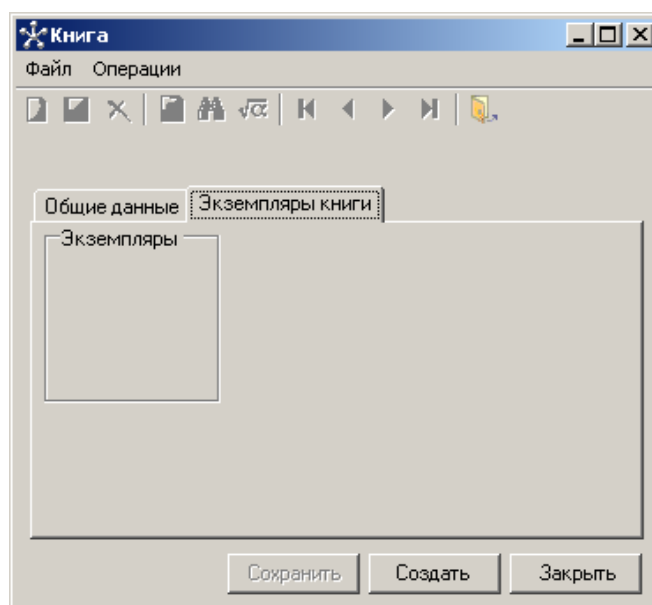


Рис. 36. Форма с созданной группой

Для удаления группы достаточно выбрать группу и вызвать пункт контекстного меню «Удалить группу». Группа будет удалена, если она не содержит полей ввода. Если группа не пуста, то надо предварительно перенести все элементы управления в другие группы и/или вкладки формы.

Для переименования форм, страниц, групп, полей нужно (рис. 37):

- Выбрать объект для изменения заголовка. Если надо изменить заголовок элемента управления, то можно выбрать относящуюся к нему подпись.
- В контекстном меню выбрать пункт «Переименовать».
- Ввести новый заголовок в текстовое поле появившейся на экране формы.
- Нажать «Сохранить» для сохранения изменений.

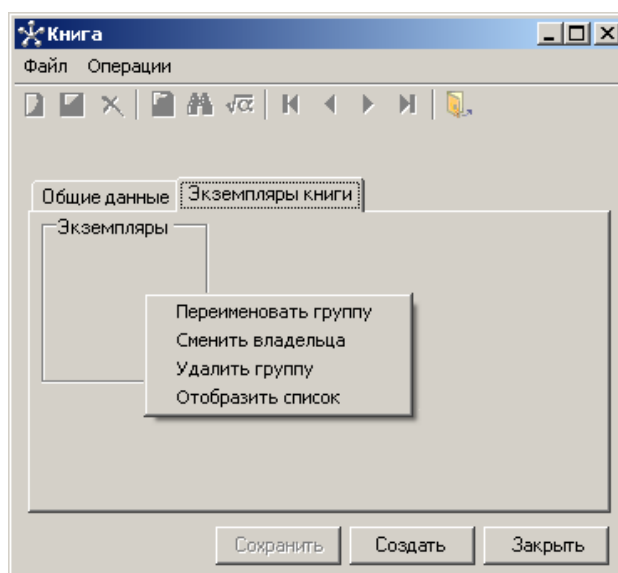


Рис. 37. Контекстное меню группы

В любой момент до нажатия кнопки «ОК» возможна отмена изменений.

Перемещение и изменение размера элемента управления происходит с помощью мыши.

Для *изменения местоположения* элемента управления необходимо:

- Подвести указатель мыши к элементу управления. При этом курсор изменит свой вид: .
- Нажав левую кнопку мыши и удерживая ее нажатой, переместить указатель таким образом, чтобы элемент управления переместился в нужное место формы.
- Отпустить кнопку мыши.

При изменении местоположения элемента управления изменяет свое расположение и относящийся к данному элементу заголовок.

Для *изменения размера* необходимо:

- Подвести указатель мыши к нижнему или правому краю элемента управления. При этом курсор изменит свой вид (,  или ).
- Нажав левую кнопку мыши и удерживая ее нажатой, переместить указатель таким образом, чтобы элемент управления приобрел нужный размер.
- Отпустить кнопку мыши.

Размер заголовка элемента управления при изменении размера самого элемента управления не изменяется.

Заголовок элемента управления может быть перемещен отдельно от элемента управления, которому он принадлежит. Также возможно *изменение размера* заголовка. Для выполнения указанных операций с заголовком необходимо произвести те же действия, что и при перемещении или изменении размера самого элемента управления.

Перемещение и изменение размера группы атрибутов происходит аналогичным образом.

Для *перемещения элементов управления между группами и вкладками формы* нужно:

- Выбрать элемент для перемещения, выделив его щелчком мыши.
- Выбрать пункт контекстного меню «Сменить владельца» (рис. 38).
- В появившемся диалоговом окне (рис. 39) выбрать нового владельца (вкладку, группу) элемента управления.
- Сохранить изменения, нажав на кнопку «Сохранить».

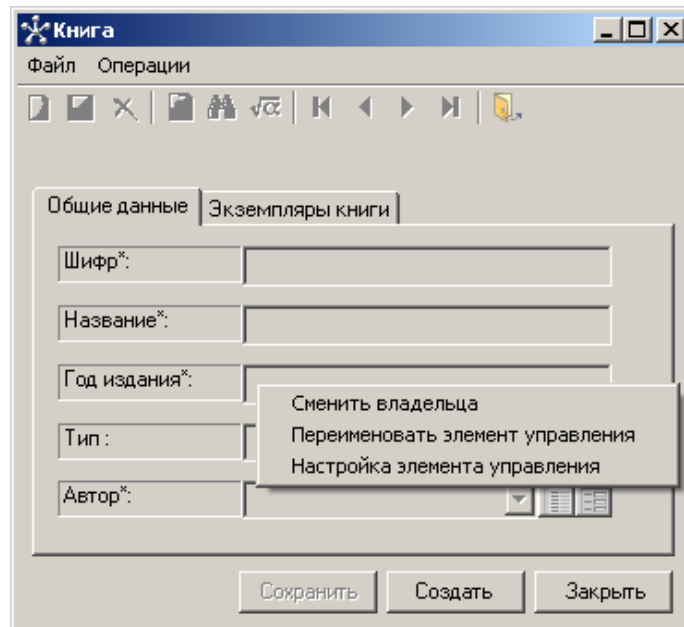


Рис. 38. Контекстное меню элемента управления

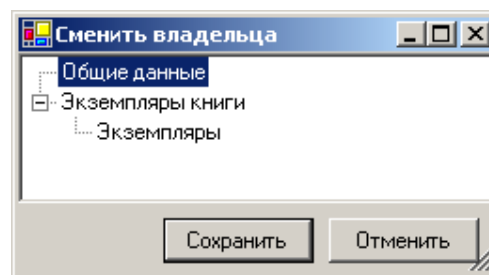


Рис. 39. Форма выбора вкладок и групп

До нажатия кнопки «Сохранить» возможна отмена изменений с помощью кнопки «Отмена».

Для *перемещения групп между вкладками формы* нужно:

- Выбрать группу для перемещения.
- Выбрать пункт контекстного меню «Сменить владельца».
- В появившемся диалоговом окне выбрать нового владельца.
- Сохранить изменения, нажав на кнопку «Сохранить».

Для *изменения порядка следования вкладок* нужно:

- Вызвать контекстное меню формы и выбрать в нем пункт «Изменить порядок страниц».
- В появившемся диалоговом окне (рис. 40) выбрать вкладку для перемещения.
- С помощью кнопок  (вверх) и  (вниз) поместить вкладку в нужной позиции.

- Повторить последние два шага для всех вкладок, которые необходимо переместить.
- Нажать кнопку «Сохранить» для сохранения внесенных изменений.

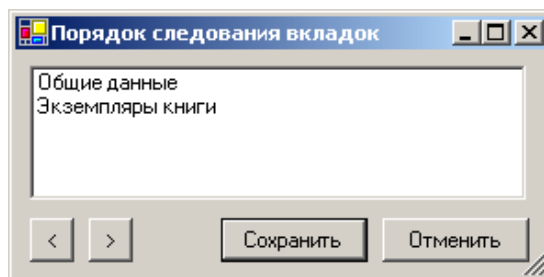


Рис. 40. Форма изменения порядка следования вкладок

По особому указанию администратора возможно *размещение на форме атрибутов родительских объектов* редактируемого объекта. Для этого нужно:

- Выбрать в контекстном меню формы пункт «Отобразить родителя».
- В появившемся диалоговом окне (рис. 41) в левой части выбрать объект (сущность предметной области), для которого необходимо отобразить родительский объект, а в правой части формы – собственно родительский объект для отображения его атрибутов на форме.

Вкладка с атрибутами объекта-родителя, полученная в результате выполнения операции, показана на рис. 42.

Также по особому указанию администратора возможно *размещение на форме дочерних объектов* редактируемого объекта. Для этого нужно:

- Выбрать вкладку или группу элементов управления, на которой нужно поместить список дочерних объектов.
- Выбрать в контекстном меню пункт «Отобразить список».
- В появившемся диалоговом окне (рис. 43) в левой части выбрать объект (сущность предметной области), для которого необходимо отобразить дочерний объект, а в правой части формы – собственно дочерний тип объекта для отображения.
- Если позволяет тип связи между объектами, выбрать тип формы, вызываемой при редактировании дочернего объекта.
- Нажать «Сохранить» для сохранения изменений.

Изменение типа вызываемой для редактирования формы производится через пункт «Изменить тип вызываемой формы» контекстного меню списка дочерних объектов (рис. 44). Вкладка со списком дочерних объектов, полученным в результате выполнения операции, показана на рис. 45.

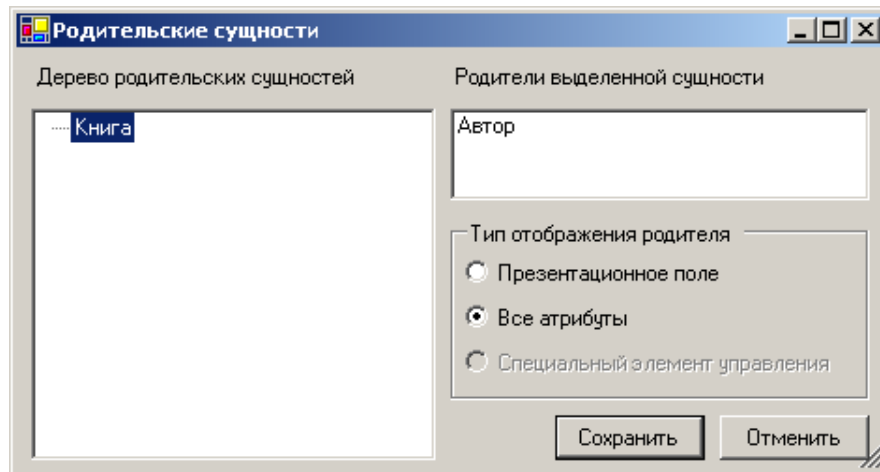


Рис. 41. Форма выбора родительской сущности для отображения ее атрибутов на форме

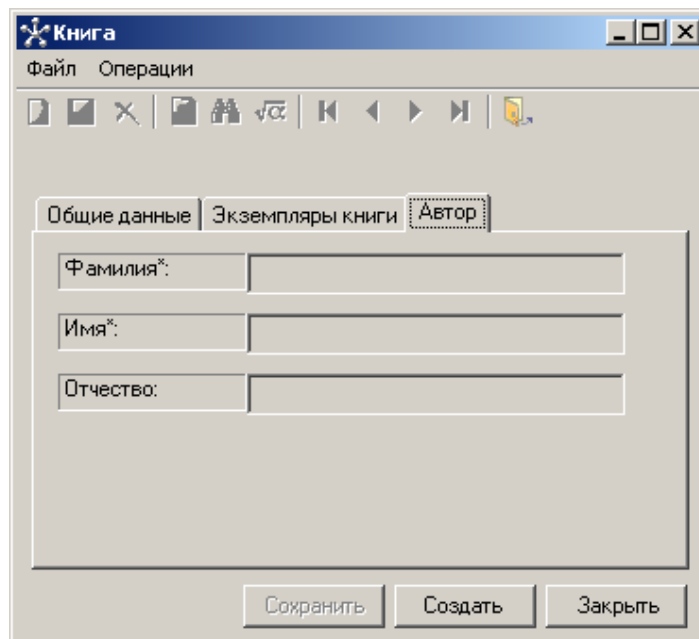


Рис. 42. Форма с атрибутами родительского объекта

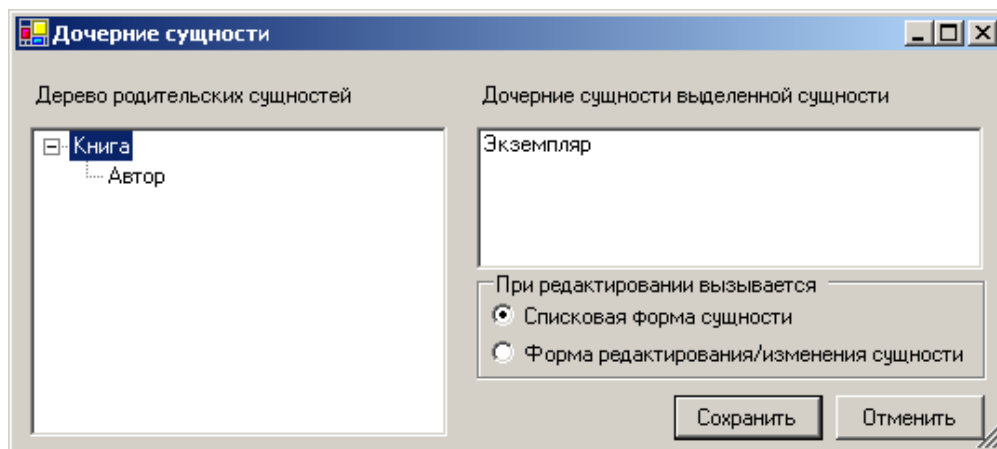


Рис. 43. Форма выбора дочерней сущности для отображения на форме

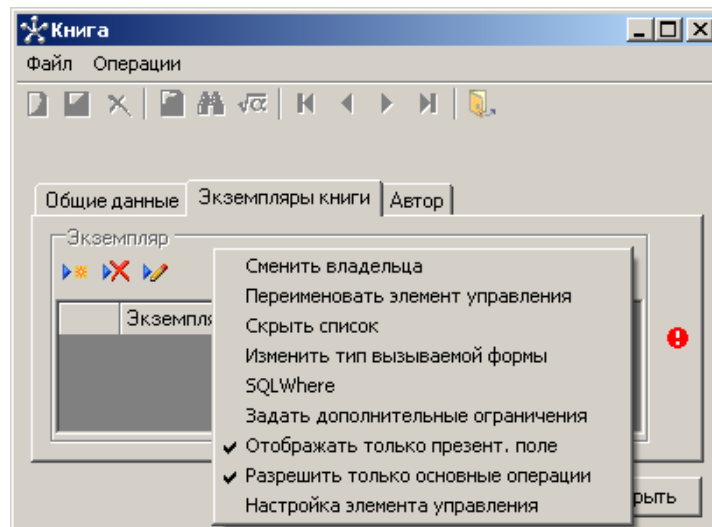


Рис. 44. Контекстное меню списка дочерних объектов

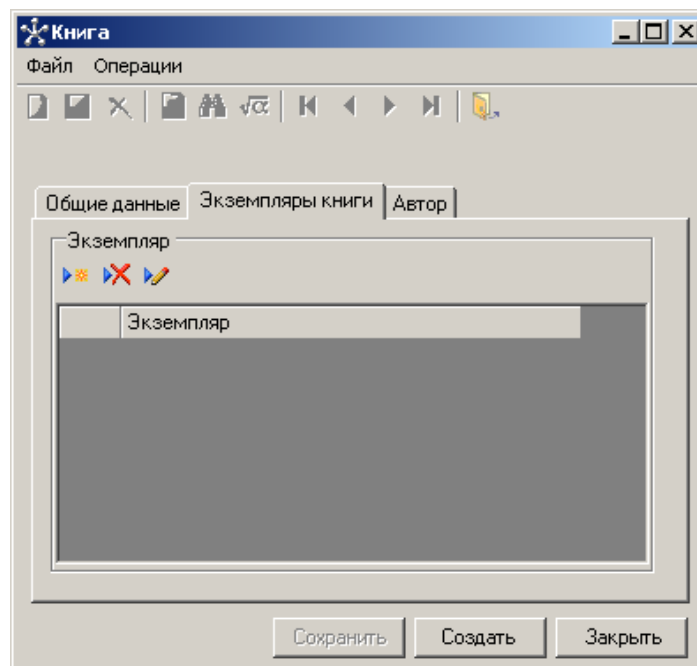


Рис. 45. Форма со списком дочерних объектов

До нажатия кнопки «Сохранить» возможна отмена изменений с помощью кнопки «Отменить».

Для отмены отображения дочерних объектов необходимо выбрать на форме список объектов, который требуется скрыть; выбрать в контекстном меню пункт «Скрыть список».

Для списков дочерних объектов возможна *дополнительная настройка*.

Пункт меню «Изменить тип вызываемой формы» позволяет *изменить тип формы, вызываемой при добавлении нового объекта в список*, если тип связи с

дочерними объектами позволяет использовать и форму редактирования, и списковую форму.

Пункт меню «SQLWhere» вызывает диалоговое окно *ввода условия для отображаемых в списке данных*. При вводе условия используется построитель выражения.

Для задания контекстных ограничений на список дочерних объектов предназначен пункт меню «Задать дополнительные ограничения». При выборе этого пункта на экране появляется диалог для выбора пути, аналогичный форме, изображенной ниже. Выбор пути производится с помощью указателя мыши. При нажатии кнопки «Сохранить» выбранный путь сохраняется. Если для списка дочерних объектов указаны какие-либо пути, то загрузка данных в список происходит с учетом путей, значимых в данном контексте. Контекстом называется цепочка объектов и связей между ними, т.е. цепочка раскрытых вершин дерева и форм, по которой пользователь открыл текущую форму редактирования.

Если установлен флажок «Отображать только презент. поле», то все объекты в списке представлены только своим презентационным полем. Иначе в списке отображаются значения всех атрибутов дочерних объектов.

При установленном флажке «Разрешить только основные операции» в списке объектов пользователь может выполнять только операции добавления, изменения, удаления дочерних объектов. В противном случае в панели инструментов списка появляются дополнительные кнопки (рис. 46), предназначенные для выполнения операций печати, просмотра, копирования, обновления, поиска, сортировки, фильтрации.

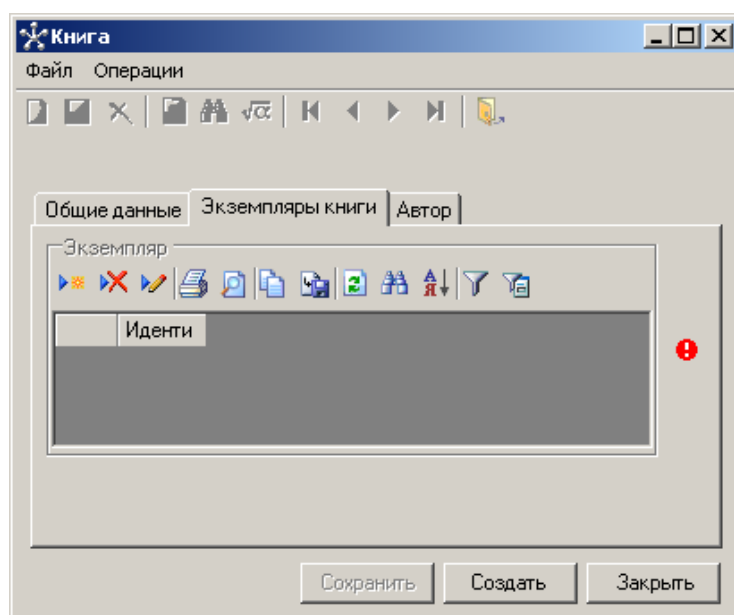


Рис. 46. Список дочерних объектов с полным набором доступных операций

Также дополнительная настройка может быть выполнена для справочников (элементы управления с выпадающим списком). В контекстном меню такого элемента управления имеется пункт «Способ сортировки».

Пункт меню «Настройка элемента управления» позволяет перейти к настройке элемента управления для работы с классификаторами.

Построение и настройка дерева объектов

Для перехода в *режим настройки дерева* в контекстном меню необходимо выбрать пункт «Режим настройки». Аналогичным образом происходит возврат в режим обычной работы: необходимо снять флажок с пункта «Режим настройки» (рис. 47-48).

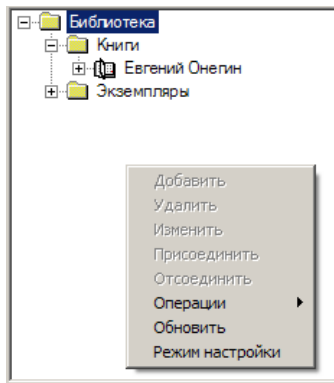


Рис. 47. Переход в режим настройки

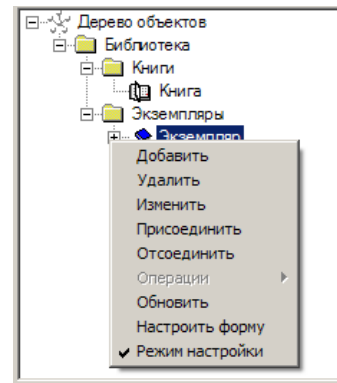


Рис. 48. Дерево в режиме настройки

В дереве представлены два вида вершин: *объектные* и *групповые*. *Групповая* вершина не является объектом предметной области, а представляет собой «папку» (например, «Образовательные учреждения», «Территории», «Ученики»). Групповые вершины используются, чтобы облегчить навигацию по данным, структурировать их (сгруппировать по типам, назначению). *Объектная* вершина – это вершина, представляющая сущности предметной области (каждый экземпляр отображается с помощью вершины такого типа, т.е. это – конкретные *объекты*). Вершины также делятся на *контейнерные* и *терминальные*. Вершина *контейнерная*, если она может содержать в себе другие (дочерние) объекты. В противном случае вершина *терминальная*.

Корневым элементом дерева в режиме настройки будет фиктивная вершина «Дерево объектов». Следующим уровнем будут те типы вершин, у которых атрибут «*Корневая вершина*» установлен. Далее вершины будут раскрываться точно так же, как и при обычном просмотре дерева объектов предметной области. При этом в тех местах, где идет отображение реальных объектов предметной области (т.е. строится и выполняется запрос к БД), просто отображается значок и название сущности в случае объектной вершины или папка с подписью (текст) в случае групповой вершины. Например, вид дерева на

экране пользователя, настраивающего дерево, может быть таким, как показано на рис. 48.

Вершину «Дерево объектов» нельзя удалить или изменить.

Для *создания нового типа* вершин необходимо:

- Выделить вершину, в которой должна содержаться создаваемая вершина.
- В контекстном меню выбрать пункт «Добавить».
- Заполнить поля появившейся на экране формы «Тип вершины» (правила работы с формой см. далее).
- Сохранить изменения с помощью кнопки «Сохранить».

Отмена изменений производится с помощью кнопки «Отмена».

Для *редактирования типа вершины* надо:

- Выделить тип для редактирования.
- В контекстном меню выбрать пункт «Изменить».
- Внести изменения в поля появившейся на экране формы «Тип вершины» (правила работы с формой см. далее).
- Сохранить изменения с помощью кнопки «Сохранить».

Отмена изменений производится с помощью кнопки «Отмена».

Чтобы *удалить тип вершин* из дерева, необходимо:

- Выделить тип для удаления.
- В контекстном меню выбрать пункт «Удалить».
- Подтвердить удаление.

При создании или изменении типа вершин пользователь заполняет поля формы редактирования типа вершины (рис. 49).

Рассмотрим поля этой формы более подробно:

- «Наименование» – уникальное имя вершины.
- «Корневая вершина» – признак, является ли вершина корневой в дереве. Поле не доступно для редактирования пользователем. Определяется местоположением редактируемой или добавляемой вершины.
- «Объектная вершина» – признак, является ли вершина объектной.
- «Сущность» – сущность, которая связана с вершиной. Поле заполняется, если установлен флажок «Объектная вершина». Сущность выбирается из выпадающего списка.
- «Текст» – строка, отображаемая в качестве подписи групповой вершины.
- «Терминальная вершина» – признак, является ли вершина терминальной в дереве.

Рис. 49. Форма редактирования типа вершин

- «Верхняя вершина при составлении подсказки» – вершины, расположенные выше такой вершины в иерархии объектов дерева, не включаются в текст строки подсказки, появляющейся в строке состояния при выделении какой-либо вершины дерева.
- «Текст о количестве дочерних вершин» – текст, который добавляется к информации о количестве дочерних вершин при составлении строки подсказки.
- «Иконка» – картинка, отображаемая в качестве иконки (пиктограммы) вершины. Выбор картинки осуществляется с помощью кнопки «Выбрать» или «Создать». При использовании кнопки «Выбрать» пользователь выбирает картинку из списка уже имеющихся картинок. В случае использования кнопки «Создать» на экране появится диалоговое окно выбора графического файла. Найти нужный файл можно путем перехода между папками. Для выбора файла необходимо щелкнуть по нему указателем мыши, а затем нажать кнопку «Открыть». Выбранная картинка отобразится в поле «Иконка».

- «SQLWhere» – SQL-выражение, используемое для объектных вершин в качестве ограничения на выборку объектов сущности. Выражение может быть построено с помощью построителя выражений.
- «SQLOrder» – SQL-выражение, указывающее для объектных вершин способ сортировки объектов сущности. Также может использоваться построитель выражения.
- «Описание» – краткое описание вершины. Любой текст.
- «При добавлении открывать форму» – переключатель способа загрузки данных в форму при добавлении объектов в вершину в режиме ввода и редактирования данных.
- «При редактировании открывать форму» – переключатель способа загрузки данных в форму при изменении объектов в вершине в режиме ввода и редактирования данных.

Если пользователь устанавливает флажок «Объектная вершина» и выше по иерархии также есть вершина объектного типа, то необходимо *задать путь*, состоящий из последовательности связей между сущностями, соединяющий сущности вершин нужным образом. Для этого необходимо:

- Нажать кнопку «Задать путь».
- В появившемся списке выбрать путь, который соответствует нужному соединению сущностей. Для этого достаточно щелкнуть указателем мыши по выбранному пути (рис. 50).
- Нажать кнопку «Выбрать» для сохранения выбранного пути и возвращения на форму редактирования типа вершин.

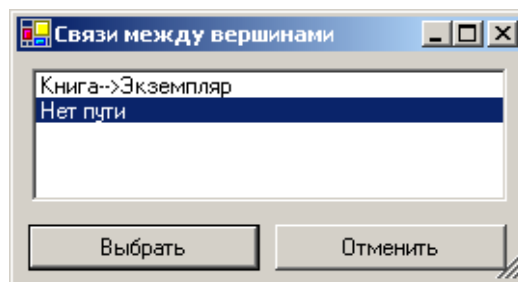


Рис. 50. Форма выбора пути между сущностями

Если сущности не связаны (прямо или через другие сущности, то путь между ними будет отсутствовать).

С помощью кнопки «Отменить» происходит возврат к редактированию полей типа вершины. Путь не сохраняется.

Возможно также задание *дополнительных путей*, связывающих текущую вершину с другими объектными вершинами, расположенными выше по иерархии в дереве. Это позволяет пользователю наиболее удобным способом, с учетом различных связей организовать навигацию по объектам системы.

Для задания дополнительного пути необходимо:

- Нажать кнопку «Дополнительно...».
- В появившемся списке (рис. 51) выбрать тип вершин, с которым должен быть связан текущий тип, и нажать кнопку «Выбрать».
- В появившемся списке (см. выше) выбрать путь, который соответствует соединению сущностей. Для этого достаточно щелкнуть указателем мыши по нужному пути.
- Нажать кнопку «Выбрать» для сохранения выбранного пути и возвращения на форму редактирования типа вершин.

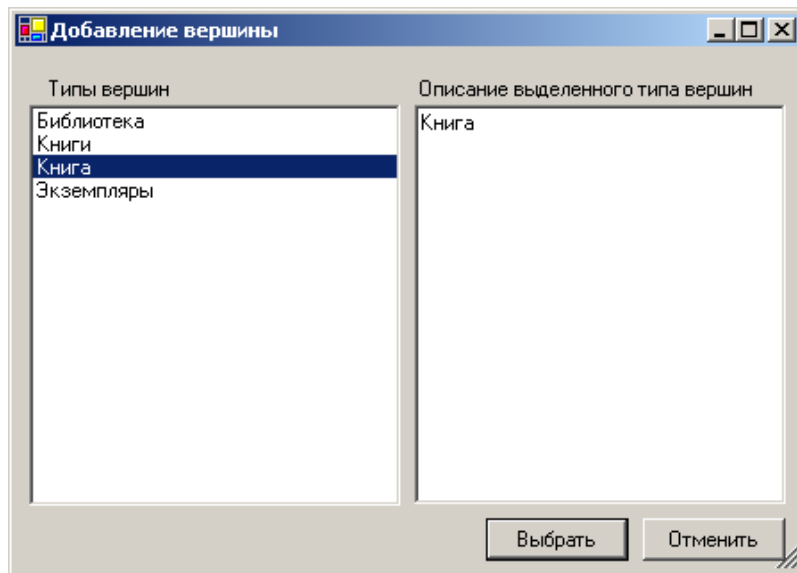


Рис. 51. Список типов вершин дерева объектов

Нажатие кнопки «Отменить» в любой момент позволяет прервать выполнение операции добавления дополнительного пути в дерево.

Для групповых вершин, содержащих в себе объектные вершины, существует возможность настройки *списка объектов* (правая верхняя часть формы «Объекты») и краткой информации о выделенном объекте (правая нижняя часть формы «Объекты»). Вызов формы настройки осуществляется через контекстное меню, пункт «Редактировать поля» (рис. 52).

С использованием формы настройки, изображенной на рис. 53, производится настройка полей отображаемых объектов.

В левой части формы показано дерево, включающее все доступные поля. С помощью двойного щелчка на каком-либо поле можно добавить соответствующее поле в список выбранных полей (средняя часть формы). При необходимости можно задать дополнительные свойства выбранного поля (раздел «Свойства» в правой части формы).

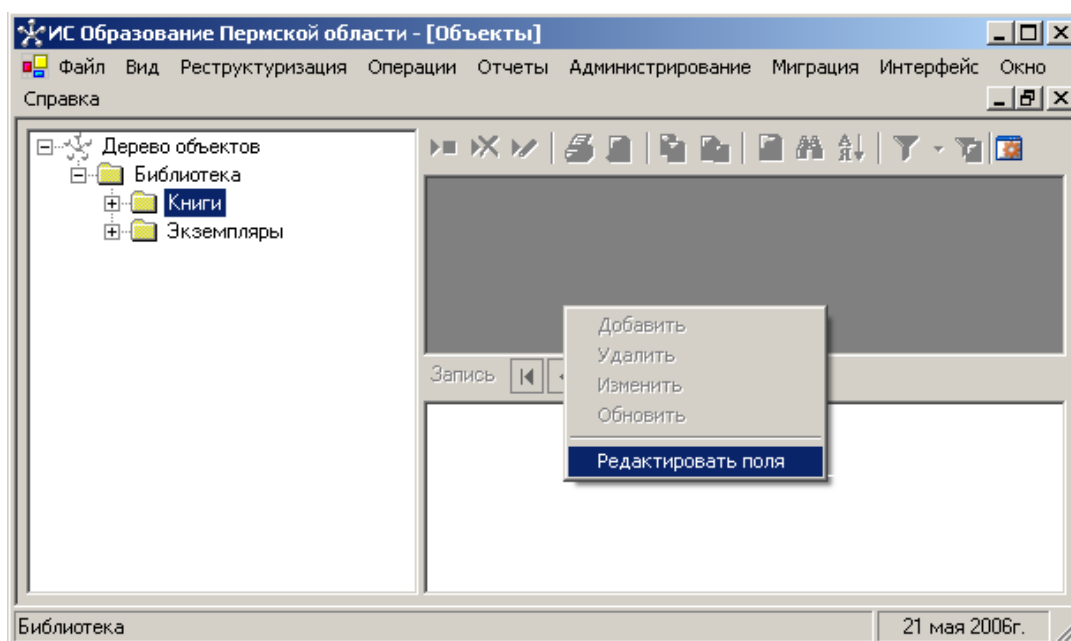


Рис. 52. Контекстное меню списка объектов вершины и краткой информации об объекте в режиме настройки

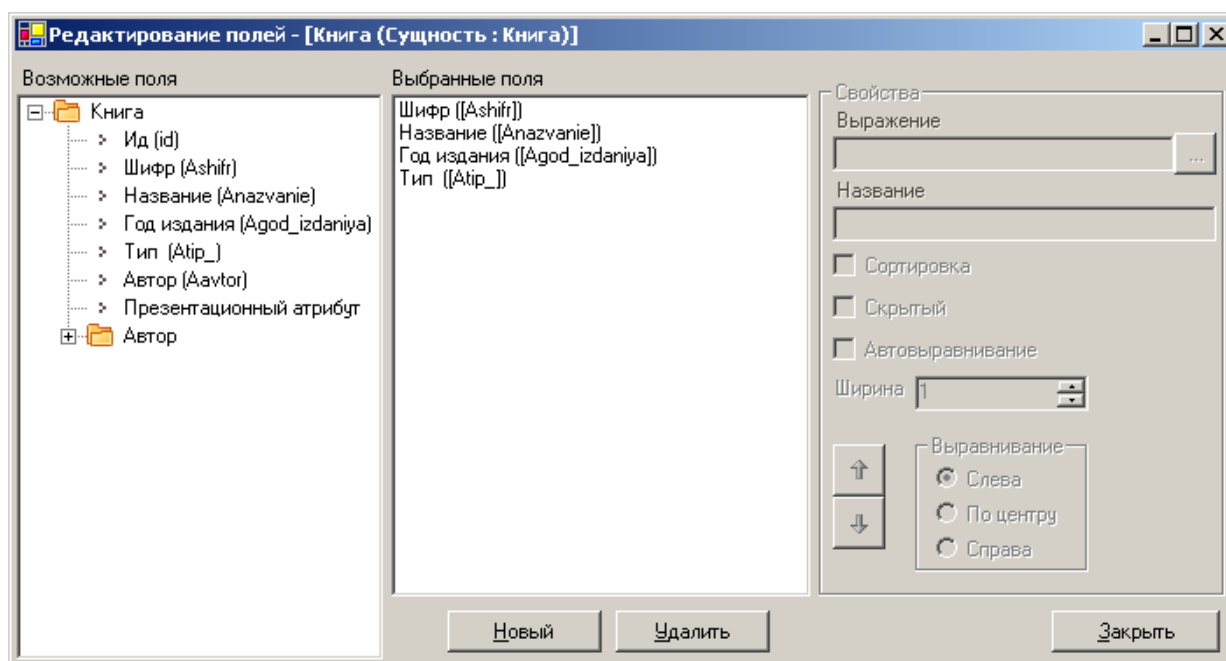


Рис. 53. Форма редактирования полей списка объектов вершины и краткой информации об объекте

Также можно добавить новое поле с помощью кнопки «Новый» и задать выражение для поля с использованием построителя выражений. Удаление выделенного поля производится с помощью кнопки «Удалить».

Подключение нестандартных элементов управления

Подключение нестандартных элементов управления производится через пункт меню «Интерфейс ► Подключение элементов управления» главного окна приложения.

При выборе указанного пункта меню на экране появляется форма, изображенная на рис. 54.

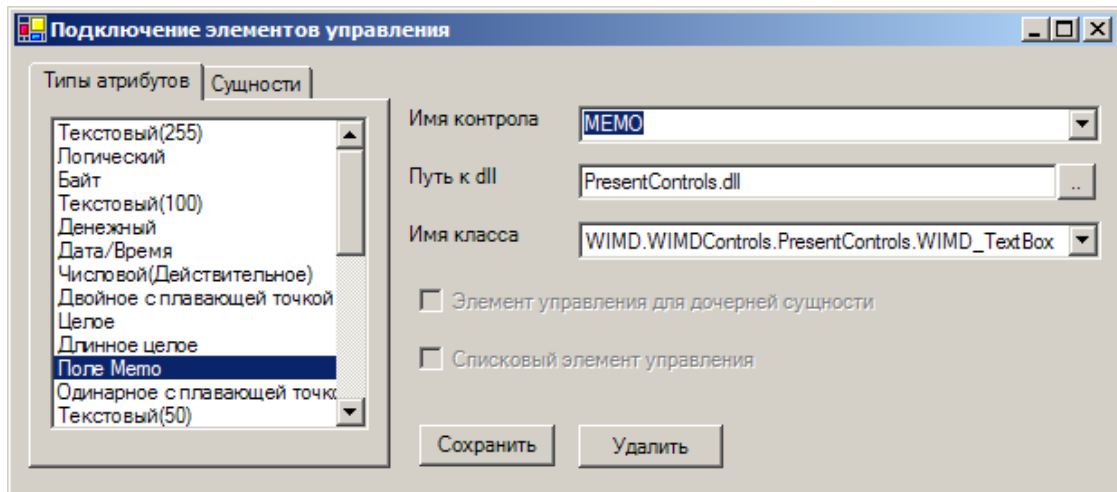


Рис. 54. Подключение элементов управления для типов атрибутов

В левой части формы расположены две вкладки, на которых содержатся списки типов атрибутов и сущностей.

В правой части находятся свойства подключаемого элемента управления. Поле «Имя контрола» содержит уникальное имя элемента управления для работы с выбранным типов атрибута при отображении его значения на форме. При выделении типа атрибута или сущности в список это поля загружаются имена всех элементов управления, связанных с выбранным атрибутом или сущностью соответственно. Можно либо выбрать имя из предложенного списка и отредактировать поля выбранного элемента управления, либо ввести новое имя и заполнить поля нового элемента. Обязательными для заполнения являются поля «Путь к dll» и «Имя класса». Путь к библиотеке dll, содержащей необходимый элемент управления, является строкой, представляющей собой либо полный путь к файлу, либо путь относительно исполняемого файла приложения. Выбор пути можно осуществить через стандартный диалог Windows открытия файла. При выборе пути заполняется список имен классов, которые могут быть использованы в качестве подключаемых элементов управления.

Для элементов управления сущностей доступны дополнительные логические поля «Элемент управления для дочерней сущности» и «Списковый элемент управления» (рис. 55).

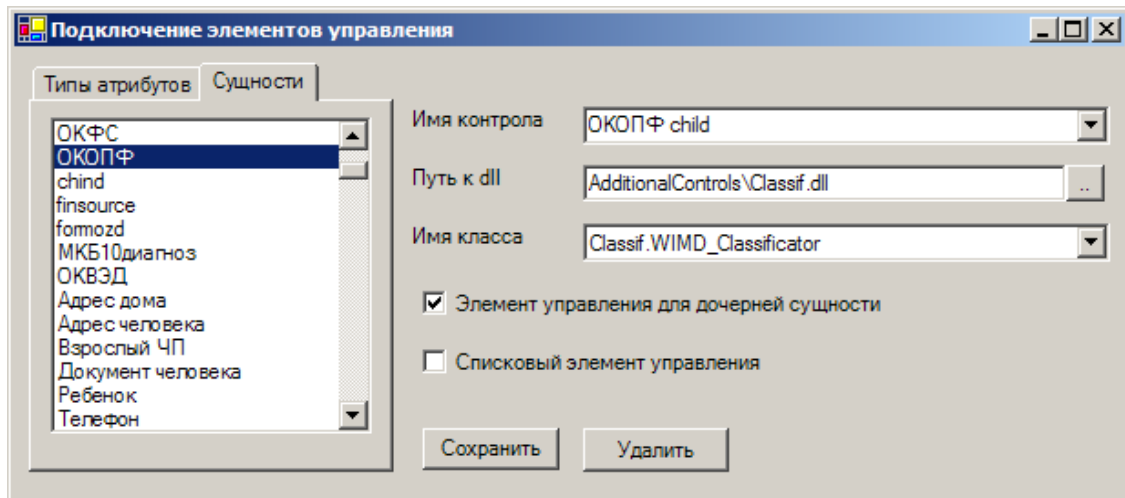


Рис. 55. Подключение элементов управления для сущностей

Выбор элемента управления для отображения сущности производится на основе анализа ее связи с сущностью, для которой рассматриваемая сущность является родительской или дочерней. Если элемент управления предназначен для отображения связи с родителем (тип связи «1:M», выбранная сущность расположена на конце «1»), то последние два поля не заполняются. Если элемент управления отображает дочернюю сущность, то необходимо отметить поле «Элемент управления для дочерней сущности». Если это дочерняя сущность из связи «1:0..1» (частный случай связи «1:M»), то элемент управления не является списковым, поле «Списковый элемент управления» заполнять не надо. Во всех остальных случаях в поле «Списковый элемент управления» необходимо поставить галочку.

Нажатие на кнопку «Сохранить» приводит к сохранению изменений в редактируемом описании элемента управления или к созданию нового описания в случае, если в поле «Имя контрола» было введено не существующее ранее имя. Кнопка «Удалить» позволяет удалить существующее описание подключаемого элемента управления. В случае, когда в поле «Имя контрола» было введено новое имя, при нажатии кнопки «Удалить» никаких действий не производится.

Учет связей между сущностями при отображении взаимосвязанных данных на форме

Если у сущности имеются связи с родительскими сущностями и эти родительские также сущности связаны между собой, то существует возможность использования этой взаимосвязи при редактировании данных. Например:

- при редактировании адреса можно выбрать улицу и территорию без учета принадлежности улицы к определенной территории или сначала выбрать территорию, а затем улицу, принадлежащую к указанной территории;

- при создании записи об образовательном учреждении, выбрав тип учреждения, вид выбираем уже не из всего списка, а только из тех видов, которые относятся к данному типу (на рис. 56 показаны соответствующие связи в окне компонента реструктуризации).

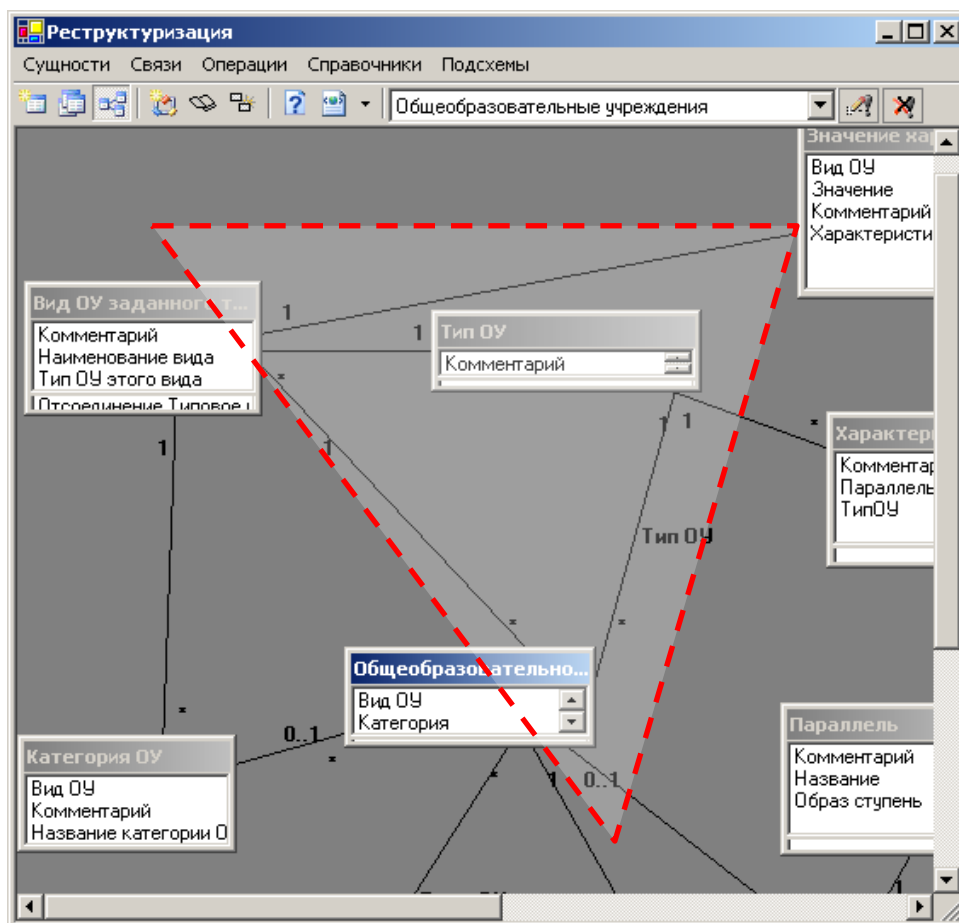


Рис. 56. Связи между сущностями, отображаемые на форме

Для использования взаимосвязи между атрибутами-родителями необходимо сделать соответствующую настройку элементов управления, размещаемых на форме. Для этого нужно:

- открыть окно компонента реструктуризации;
- с помощью контекстного меню открыть окно свойств сущности, имеющей несколько взаимосвязанных родительских объектов;
- щелчком по кнопке «Настройка» открыть диалоговое окно установки связи (рис. 57);
- в списке «Сущность» выбрать сущность, на форме которой размещаются элементы, представляющие взаимосвязанные родительские сущности («Общеобразовательное учреждение»);
- в списке «Главный атрибут» выбрать родительскую сущность, которая является родительской и для второй родительской сущности («Тип ОУ»);

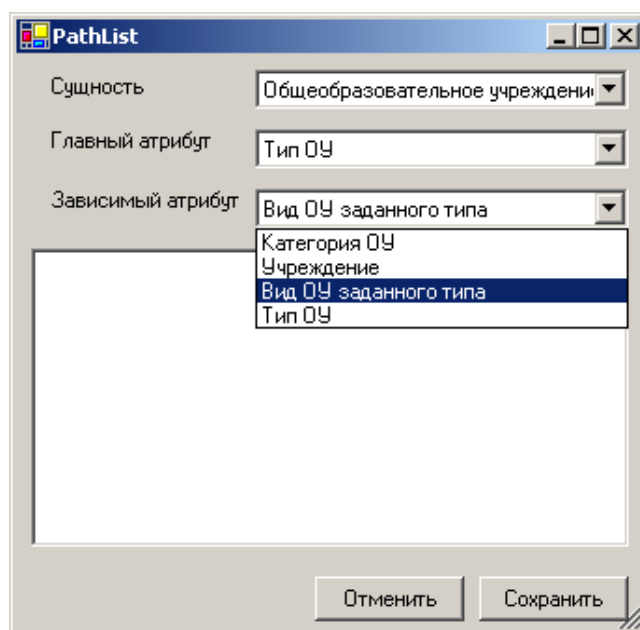


Рис. 57. Окно установки связи элементов управления на форме в соответствии со связями сущностей в модели предметной области

- в списке «Зависимый атрибут» выбрать сущность, которая для выбранной в первом списке сущности является родительской, а для сущности, указанной во втором списке – дочерней;
- в окне (рис. 58) появится список существующих между родительскими сущностями связей (их может быть несколько);
- щелчком по кнопке «Сохранить» сохранить установленную зависимость.

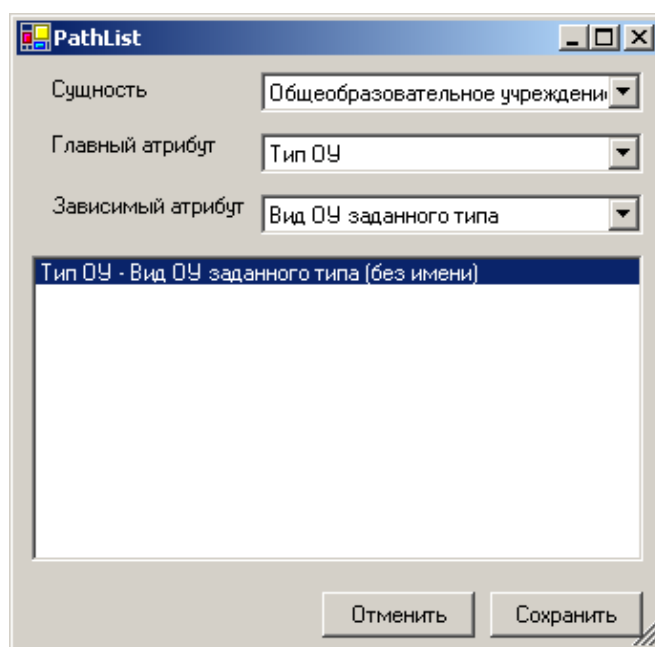


Рис. 58. Список связей сущностей

Установленная связь начнет работать. И главный, и зависимый атрибуты представляют собой связи с сущностями-родителями для указанной сущности (назовем их главной и зависимой сущностями). При вводе или редактировании данных о сущности при задании какого-либо значения для главного атрибута выбор значения для зависимого атрибута будет ограничен только теми объектами зависимой сущности, для которых значение атрибута, представляющего заданную связь с главной сущностью, совпадает с указанным значением главного атрибута.

Средства репортинга: создание запросов и отчетов

Подсистема репортинга позволяет делать запросы к БД без участия разработчика ИС, а также настраивать формы документов для автоматической их генерации на основе данных, размещенных в системе.

Менеджер запросов

Команда «Менеджер запросов» используется для управления пользовательскими запросами. С помощью данного компонента возможны создание, редактирование, удаление и выполнение пользовательских запросов к базе данных информационной системы.

Для запуска Менеджера запросов в главном меню программы необходимо выбрать команду «Отчеты ▸ Менеджер запросов» (рис. 59) (пункты главного меню, доступные на вам, могут отличаться от представленных на рисунке). Для быстрого запуска из главного окна программы можно воспользоваться комбинацией клавиш *Ctrl+Q*.

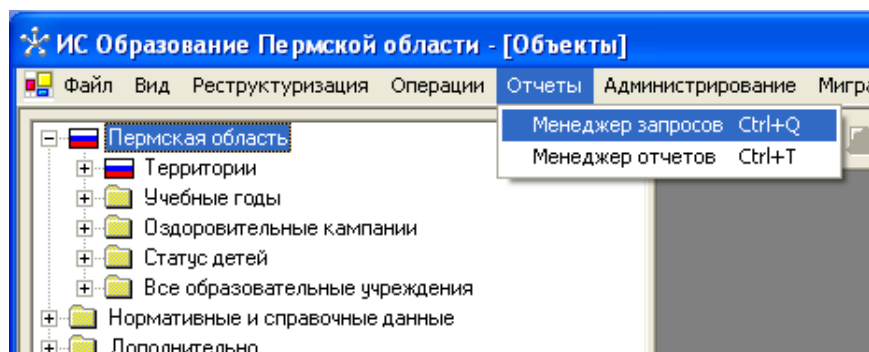


Рис. 59. Команда запуска Менеджера запросов

Окно «Менеджер запросов» состоит из двух частей: *древовидного списка запросов* и *панели инструментов*, представляющей доступ к наиболее часто используемым командам (рис. 60).

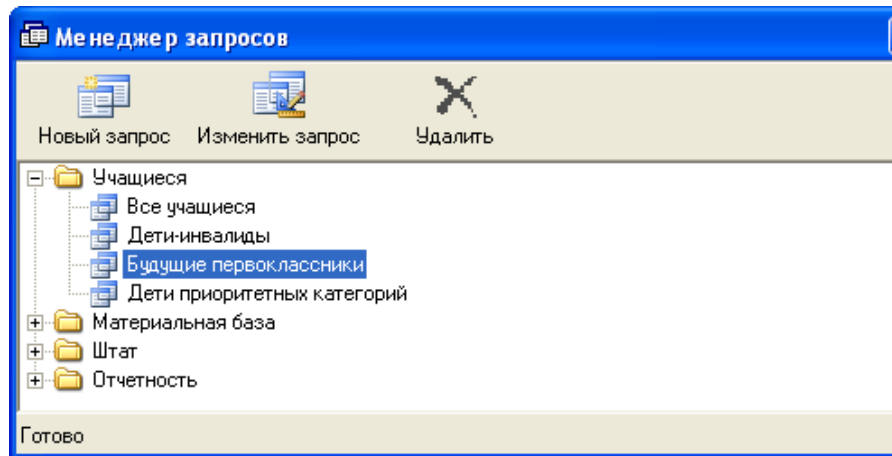


Рис. 60. Окно Менеджера запросов

Для осуществления над запросом какой-либо операции (выполнение, редактирование, удаление) его необходимо *выделить* (выбрать). Выбор запроса можно осуществить с помощью мыши или клавиш управления курсором на клавиатуре.

Для выполнения операций (действий) над выбранным запросом используется панель инструментов и контекстное меню. Команды применяются к выделенному в данный момент объекту. Для вызова контекстного меню необходимо нажать левую кнопку мыши и соответствующую клавишу на клавиатуре.

Запросы хранятся, подобно файлам на диске, в папках. Кроме запросов в папке могут размещаться и другие папки. Для удобства навигации строится *дерево запросов*, отражающее вложенность папок и запросов.

Для *создания* новой папки необходимо выделить папку, в которую будет вложена создаваемая папка, и выбрать в контекстном меню команду «Добавить папку». Новая папка будет добавлена. По умолчанию созданная папка будет названа «Новая папка». Для *переименования* папки по ней необходимо сделать два одиночных щелчка мышью, после появления курсора необходимо ввести новое имя папки. Ввод имени должен быть закончен нажатием клавиши *Enter*. Второй способ изменения имени папки – использования команды «Свойства...» папки.

Для *удаления* папки необходимо ее выделить и выбрать в контекстном меню команду «Удалить». При удалении папки удаляются все находящиеся в ней запросы (см. «Удаление запроса»). Папка не может быть удалена, если не может быть удален хотя бы один находящийся в ней запрос. Если папка не является пустой, на экране появится окно, требующие подтверждения операции удаления.

Для *создания нового запроса* пользователя необходимо в контекстном меню выбрать команду «Добавить запрос...» или на панели инструментов нажать кнопку «Новый запрос» (рис. 61). Запрос будет добавлен в выделенную папку, в случае, если был выделен запрос, новый запрос будет добавлен в папку, содержащую выделенный запрос. Для смены имени запроса, присвоенного по умолчанию, по запросу необходимо сделать на его имени два одиночных щелчка мышью, после появления курсора необходимо ввести новое имя запроса (рис. 62). Ввод имени должен быть закончен нажатием клавиши *Enter*. Второй способ изменения имени – использования команды «Свойства...» запроса.

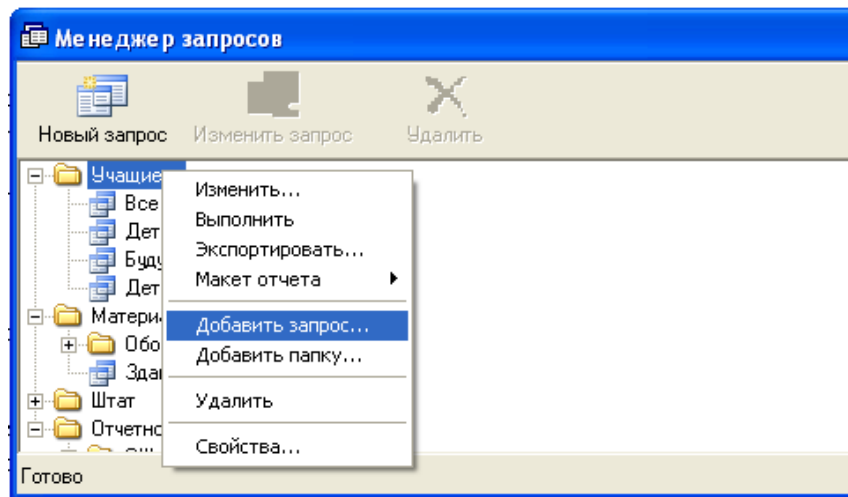


Рис. 61. Выполнение команды создания запроса

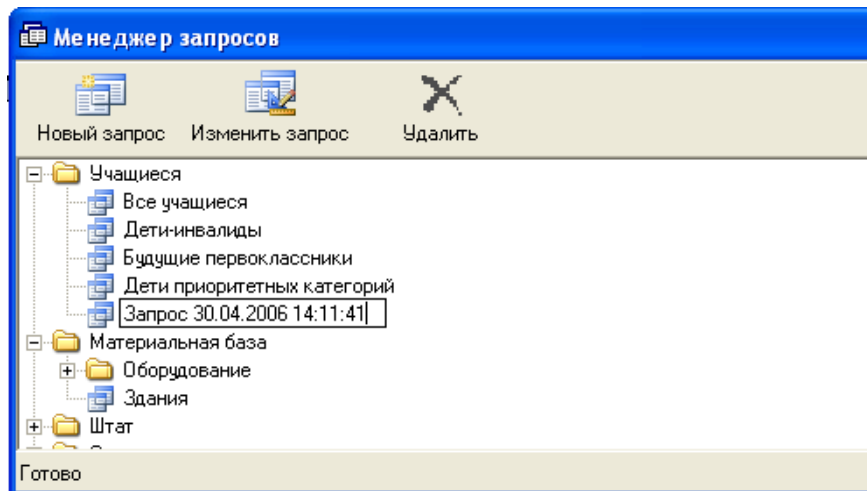


Рис. 62. Переименование запроса

Под *редактированием запроса* понимается изменение структуры запроса (списка выводимых полей, условий и т.д.).

Для редактирования запроса необходимо в контекстном меню запроса выбрать команду «Изменить...» или на панели инструментов нажать кнопку

«Изменить запрос». После выполнения данных команд на экране появится окно «Построитель запроса».

«Построитель запросов» используется для создания и редактирования пользовательских запросов. Параметры запроса устанавливаются на соответствующих вкладках, расположенных в нижней части окна. Верхняя часть окна используется для представления схемы сущностей предметной области, используемых в запросе (рис. 63).

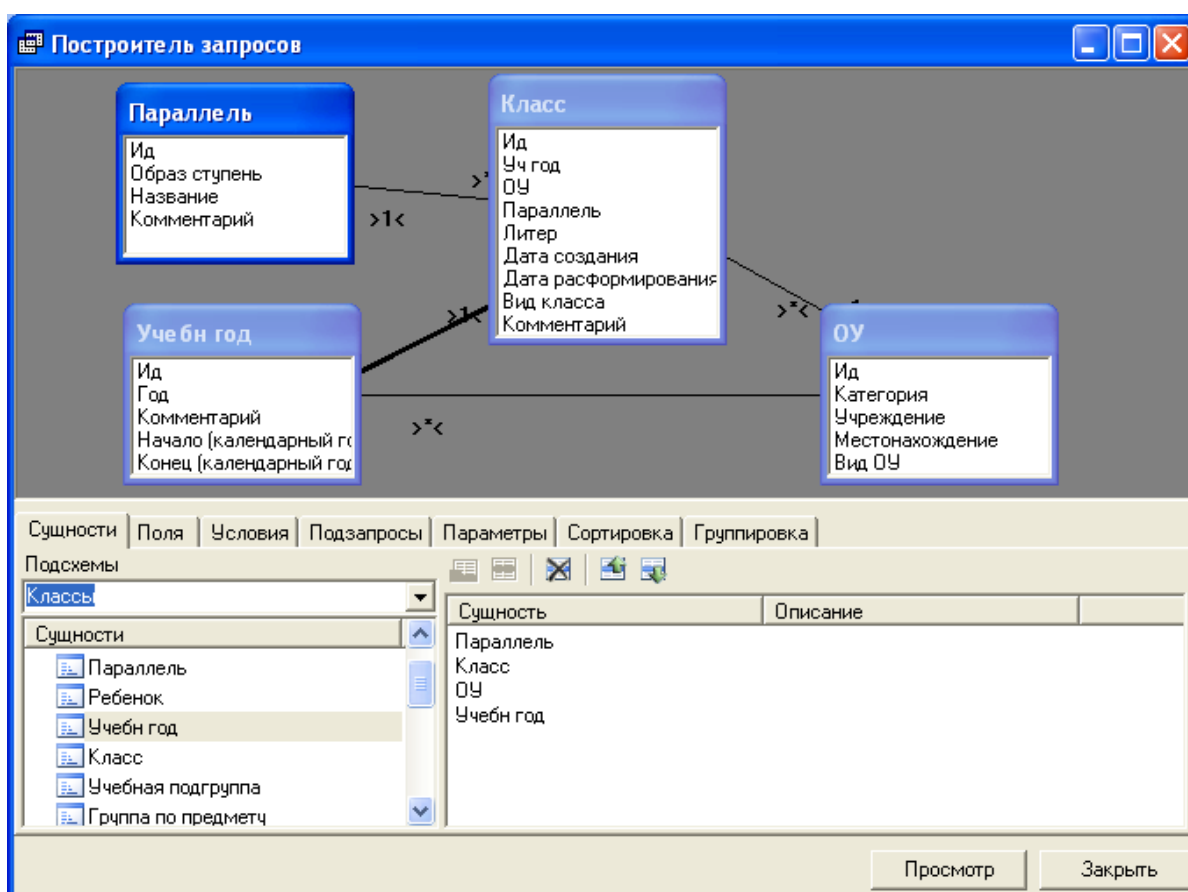


Рис. 63. Окно Построителя запросов

На вкладках располагаются списки для операций над элементами данных:

-  – добавление нового элемента в список;
-  – редактирование выделенного элемента списка;
-  – удаление выделенного элемента списка;
-  – перемещение выделенного элемента списка вверх;
-  – перемещение выделенного элемента списка вниз.

Доступность команд определяется текущим контекстом. Кнопка «Просмотр» используется для вывода результата выполнения сформированного запроса. При нажатии на кнопку «Заккрыть» данное окно закрывается, а параметры запроса сохраняются.

Вкладка «Сущности» (рис. 63) служит для выбора сущностей, используемых в запросе.

В списке «Подсхемы» на ней находятся все доступные в информационной системе подсхемы сущностей. При выборе какой-либо подсхемы в расположенном ниже списке «Сущности подсхемы» перечисляются все сущности выбранной подсхемы. Для просмотра всех сущностей необходимо выбрать подсхему «Все сущности», располагающуюся в самом начале списка подсхем.

В списке «Выбранные сущности» находятся используемые в запросе сущности. Для добавления сущности в запрос необходимо выбрать ее имя в списке «Сущности подсхемы» и с помощью мыши перетащить его в список «Выбранные сущности». Добавленная сущность автоматически добавится в схему сущностей запроса вместе со всеми связями, которыми она связана с уже добавленными в запрос сущностями.

При необходимости ненужные для выполнения запроса связи между сущностями можно удалить. Для этого необходимо выделить связь с помощью мыши и в контекстном меню данной связи выбрать команду «Удалить связь» (это не повлияет на модель, на связи сущностей в ней).

Для удаления сущности из списка сущностей, используемых в запросе, необходимо выделить ее имя и нажать клавишу «Delete» на клавиатуре или нажать кнопку «Удалить» на панели инструментов.

Команда «Свойства» контекстного меню связи позволяет настроить способ объединения таблиц, объединенных данной связью.

Вкладка «Поля» служит для выбора атрибутов сущностей, которые будут представлены в запросе (рис. 64). На вкладке представлено два списка: иерархический список «Сущности», в котором представлены выбранные для запроса сущности с их атрибуты, и список «Поля запроса», содержащий выражения для полей запроса.

Для добавления атрибута сущности в список полей необходимо выбрать его имя в списке «Сущности» и с помощью мыши перетащить его в список «Поля запроса». Изменить параметры поля запроса можно с помощью диалогового окна «Поле», вызываемого двойным щелчком мыши по соответствующему элементу списка.

Для удаления элемента из списка полей необходимо нажать на кнопку «Удалить», расположенную на панели инструментов.

Вкладка «Условия» служит для создания условий, накладываемых на атрибуты сущностей при выполнении запроса (рис. 65). При наложении условий на атрибуты сущности в результате выполнения запроса, в него попадет только информация, удовлетворяющая условиям запроса.

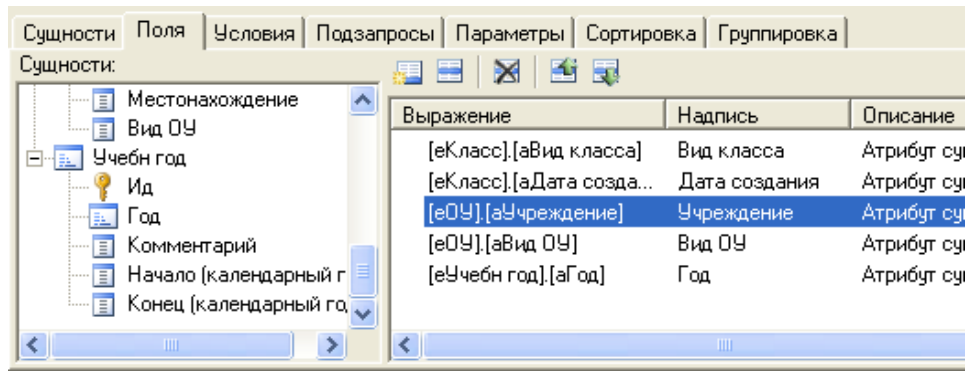


Рис. 64. Выбор атрибутов для формирования запроса

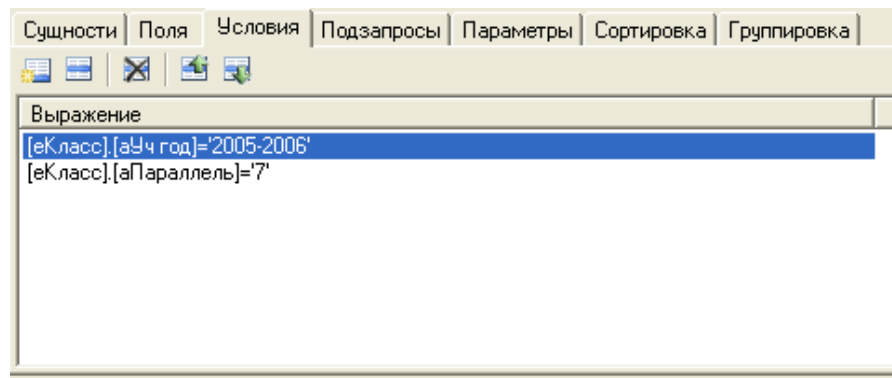


Рис. 65. Условия запроса

Для редактирования условия используется «Построитель выражений» (рис. 66). В выражении могут быть использованы атрибуты сущностей, параметры и функции SQL. Для выбора значения из дерева необходимо дважды щелкнуть мышкой по вершине.

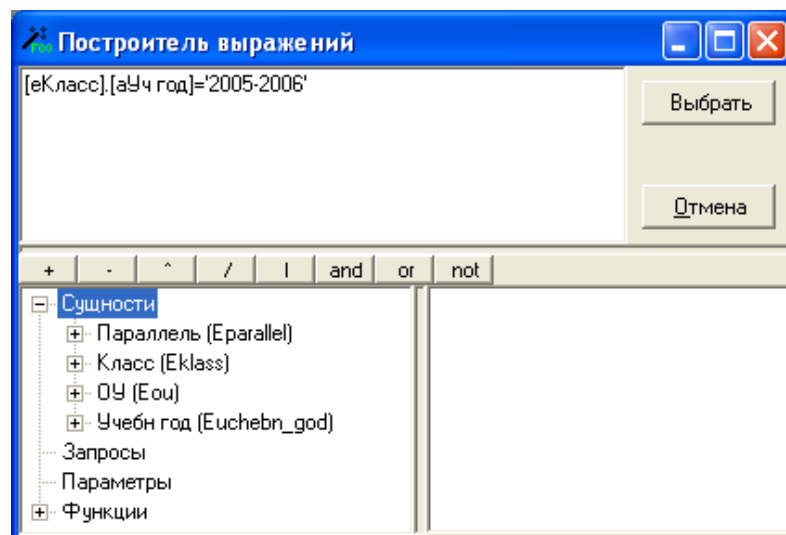


Рис. 66. Окно построителя выражений

Параметры в запросе используются для облегчения многократного использования запроса. Параметры могут быть использованы в формулировке условия, и тогда при каждом выполнении запроса у пользователя будет запрошено конкретные значения параметров.

Для работы с параметрами запроса служит соответствующая вкладка «Параметры» (рис. 67).

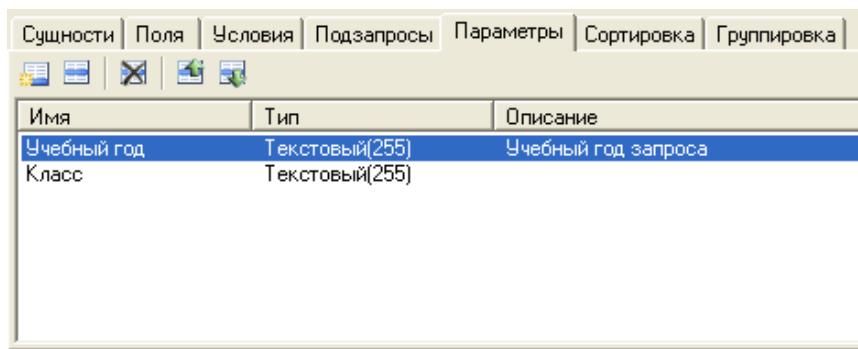


Рис. 67. Вкладка «Параметры»

При добавлении запроса необходимо указать его имя, тип данных и описание (рис. 68). Также в поле «Запрос» можно указать уже существующий запрос, из результатов выполнения которого можно будет выбрать значение параметра.

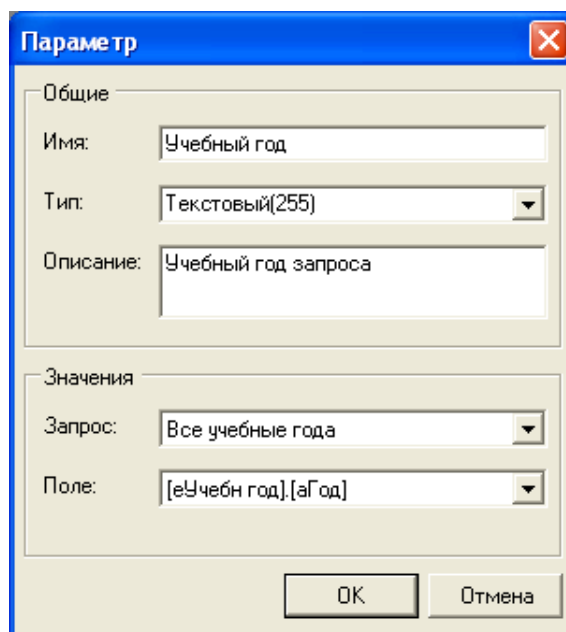


Рис. 68. Описание параметра запроса

Вкладка «Сортировка» служит для задания *упорядочивания строк* результата запроса. Можно задать несколько выражения для сортировки результата. Тогда результирующие строки запроса будут отсортированы по значениям первого выражения, затем строки, имеющие одинаковое значение первого выражения, будут отсортированы по значению второго выражения и т.д.

Результаты запроса можно *сгруппировать* для вычисления промежуточных итогов. Вкладка «Группировка» служит для задания параметров группировки результатов запроса.

Для *удаления запроса* необходимо в контекстном меню запроса выбрать команду «Удалить» или на панели инструментов нажать кнопку «Удалить». После чего необходимо подтвердить удаление, нажав на кнопку «Да» в появившемся диалоговом окне.

Удаление запроса может быть не выполнено по следующим причинам:

- Запрос связан с каким-либо шаблоном отчета. В данной ситуации сначала необходимо удалить все связи шаблона с запросом.
- Запрос является источником данных для значения параметра другого запроса. В данном случае необходимо удалить связь параметра с запросом.

Для *выполнения запроса* (вывода на экран таблицы данных, удовлетворяющих условиям запроса) необходимо в контекстном меню выбрать команду «Выполнить». После выполнения данной команды на экране появится окно «Результат запроса» (рис. 69).

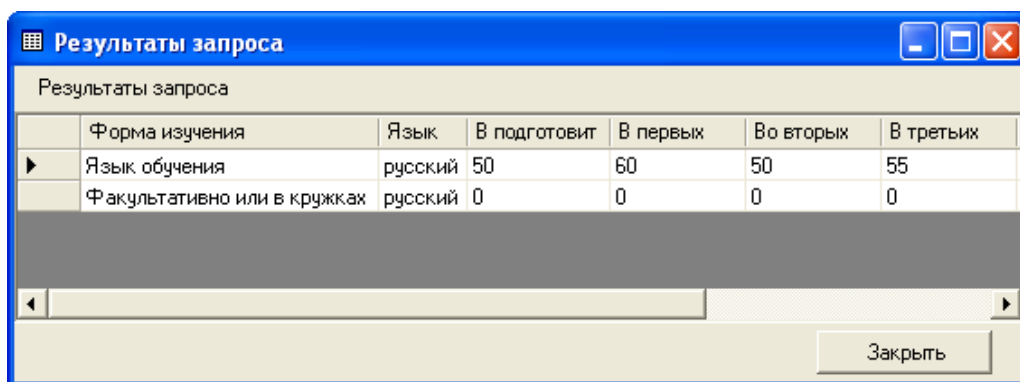


Рис. 69. Вывод результатов запроса

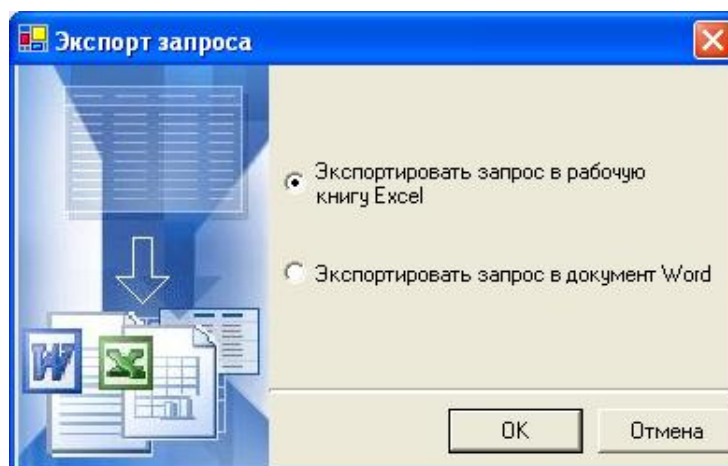


Рис. 70. Выбор варианта экспорта запроса

Результата выполнения запроса можно *экспортировать* в рабочую книгу Microsoft Excel или в документ Word. Для экспорта результатов запроса необходимо в контекстном меню выбрать команду «Экспортировать...». После выполнения данной команды на экране появится окно мастера «Экспорта запроса» (рис. 70).

Построитель запросов может функционировать в *двух режимах* «Стандартном» и «Расширенном». Переключение между двумя данными режимами осуществляется с помощью соответствующих переключателей, находящихся в окне свойств запроса (рис. 71). «Стандартный» режим позволяет установить *основные параметры запроса*. В «Расширенном» режиме возможно управление всеми параметрами запроса, включая возможность непосредственного введения текста запроса на языке SQL.

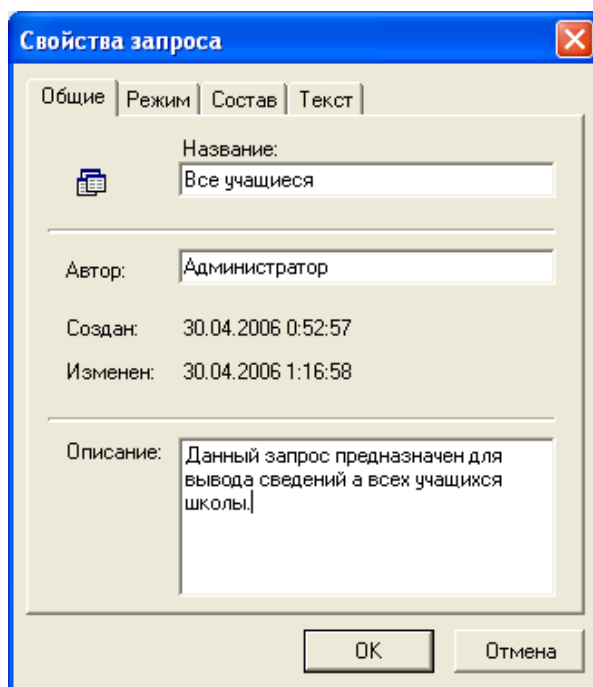


Рис. 71. Окно свойств запроса

Команды *создания макета отчета* используются для разметки шаблонов отчетов (в шаблоне документа Word или книги Excel создается размеченные диапазоны, куда могут быть размещены результаты запросов).

Для начала разметки шаблона необходимо открыть существующий или создать новый шаблон. С помощью команд «Шаблон Word», «Шаблон Excel» можно создать экземпляр документа требуемого приложения. Результатом выполнения данных команд будет открытие нового документа. Команда контекстного меню запроса «Макет отчета ▸ Открыть шаблон...» позволяет внести разметку в уже существующий шаблон. В диалоговом окне, появившемся

после выполнения команды, необходимо выбрать шаблон. Шаблоны документов Word и рабочих книг Excel имеют расширения *.dot и *.xlt соответственно. Создать шаблон можно из любого документа путем его сохранения в соответствующем формате.

Для добавления диапазона под размещением конкретного запроса необходимо в созданном документе установить курсор в то место документа, где предполагается поместить диапазон. Затем следует выбрать команду «Добавить разметку». Описанные выше действия можно произвести для каждого запроса, который планируется вставить в создаваемый шаблон. После завершения разметки созданный файл необходимо сохранить на диске как шаблон (*.xlt, *.dot), воспользовавшись командой «Сохранить и закрыть».

Менеджер отчетов

Команда «Менеджер отчетов» используется для управления документами, создаваемыми в ИС на основе данных, размещенных в ее БД, подготовкой и генерацией отчетов. С помощью данного компонента выполняется создание, редактирование, удаление отчетов, генерируемых системой.

Для запуска Менеджера отчетов в главном меню программы необходимо выбрать команду «Отчеты ► Менеджер отчетов». Для быстрого запуска из главного окна программы можно воспользоваться комбинацией клавиш *Ctrl+R*.

Окно «Менеджер отчетов» состоит из двух частей: *древовидного списка доступных пользовательских документов (дерева)* и *панели инструментов*, представляющей доступ к наиболее часто используемым командам (рис. 72).

Для осуществления с какой-либо операции с объектом его необходимо *выделить* (выбрать). Выбор отчета можно осуществить с помощью мыши или клавиш управления курсором на клавиатуре.

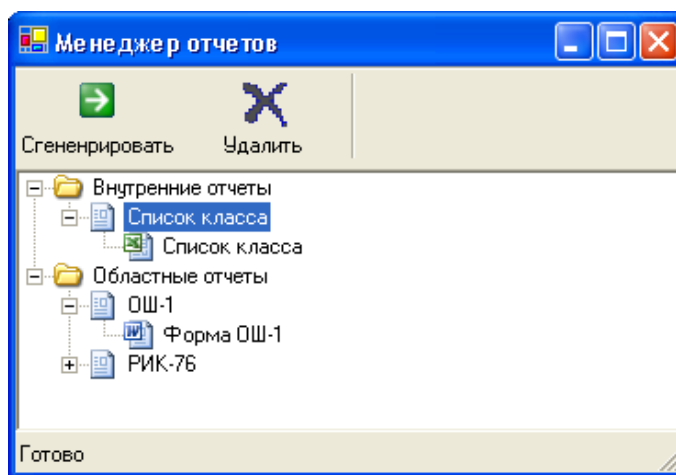


Рис. 72. Окно Менеджера отчетов

Для выполнения операций (действий) над выбранным запросом используется панель инструментов и контекстное меню. Команды применяются к выделенному в данный момент объекту. Для вызова контекстного меню необходимо нажать левую кнопку мыши и соответствующую клавишу на клавиатуре (рис. 73).

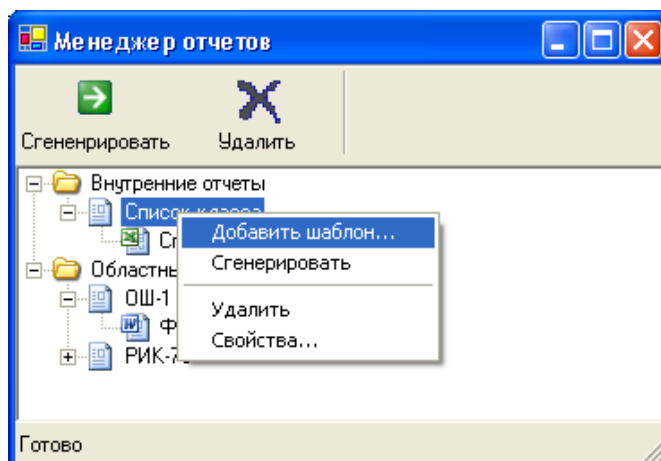


Рис. 73. Выбор команды меню

Документы хранятся, подобно файлам на диске, в *папках*. Кроме документов в папке могут находиться и другие папки.

Документ в свою очередь может содержать *разделы*, для каждого из которых создаются собственные шаблоны офисных документов (шаблон документа Word, шаблон рабочей книги Excel). Например, документ может содержать один основной шаблон и несколько шаблонов для генерации его приложений.

Для *создания новой папки* необходимо выделить папку, в которую будет вложена создаваемая папка, и выбрать в контекстном меню команду «Добавить папку». Новая папка будет добавлена. По умолчанию созданная папка будет названа «Новая папка». Для переименования папки по ней необходимо сделать два одиночных щелчка мышью, после появления курсора необходимо ввести новое имя папки. Ввод имени должен быть закончен нажатием клавиши «Enter». Второй способ изменения имени папки – использования команды «Свойства...» папки.

Для *удаления папки* необходимо ее выделить и выбрать в контекстном меню команду «Удалить». При удалении папки удаляются все находящиеся в ней документы. Если папка не является пустой, на экране появится окно, требующие подтверждения операции удаления.

Для *создания нового документа* пользователя необходимо в контекстном меню папки выбрать команду «Добавить документ». После выполнения данной

команды на экране появится окно «Свойства документа», в котором можно задать имя и описание документа.

Для *смены имени документа* можно также непосредственно изменить текст его вершины. Для этого необходимо сделать два одиночных щелчка мышью по имени документа, а после появления курсора необходимо ввести новое имя запроса. Ввод имени должен быть закончен нажатием клавиши *Enter*. Еще один способ изменения имени – использования команды «Свойства...» контекстного меню документа.

Для *добавления нового шаблона* к документу необходимо выбрать команду «Добавить шаблон...» в контекстном меню документа (рис. 74).

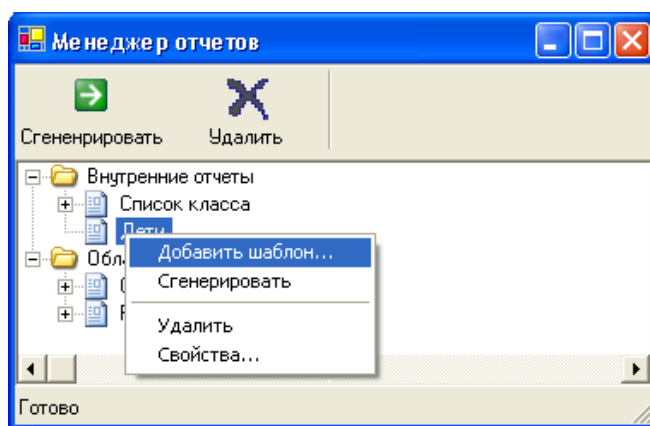


Рис. 74. Добавление шаблона документа

Результатом выполнения данной команды будет запуск мастера «Создание шаблона». Кнопки мастера «Далее» и «Назад» используются для перемещения между этапами работы мастера, кнопка «Отмена» прекращает работу мастера.

На первом этапе работы мастера необходимо указать расположения файла (шаблон рабочей книги Excel или шаблон документа Word), на базе которого будет создан шаблон документа. Импортируемые шаблоны предварительно должны быть размечены соответствующим образом. Используйте кнопку «Обзор» для указания пути к файлу шаблона (рис. 75). При нажатии на кнопку «Далее» происходит анализ разметки шаблона, после завершения которого будет выведен список всех доступных диапазонов.

На следующем этапе работы мастера «Настройка диапазонов» (рис. 76) следует проверить параметры найденных диапазонов шаблона. При необходимости изменения параметров конкретного диапазона следует выбрать команду контекстного меню «Изменить», после выполнения которой на экране появится окно настройки параметров диапазона (рис. 77).

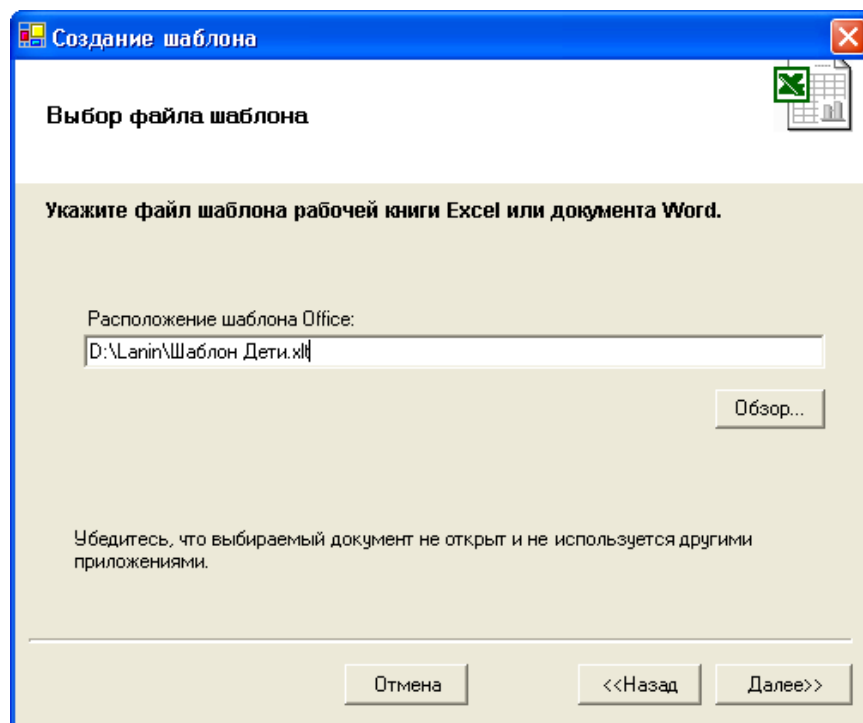


Рис. 75. Мастер создания шаблона: выбор файла

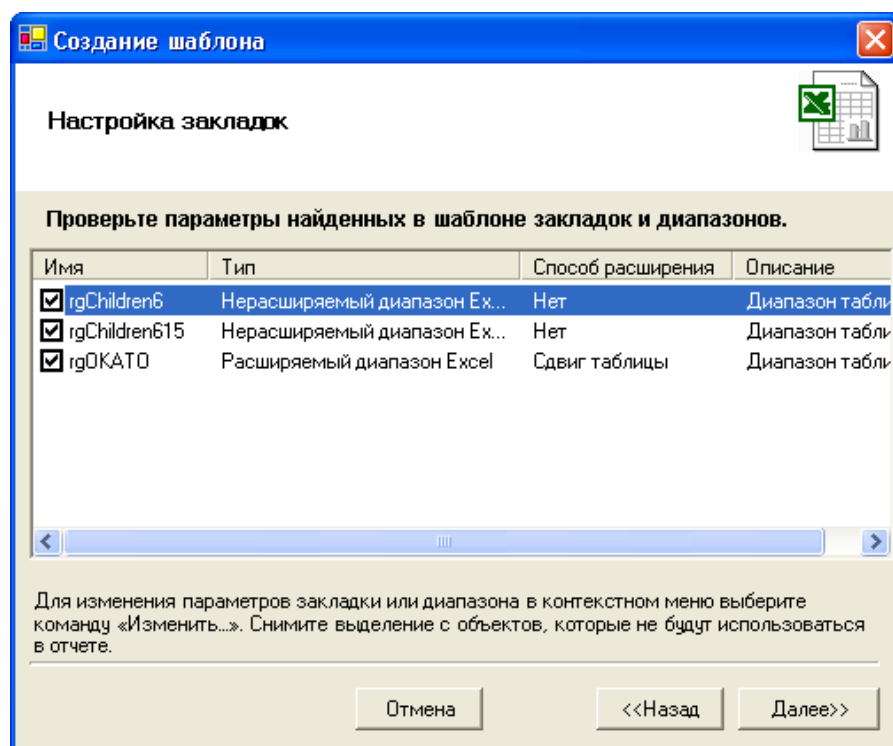
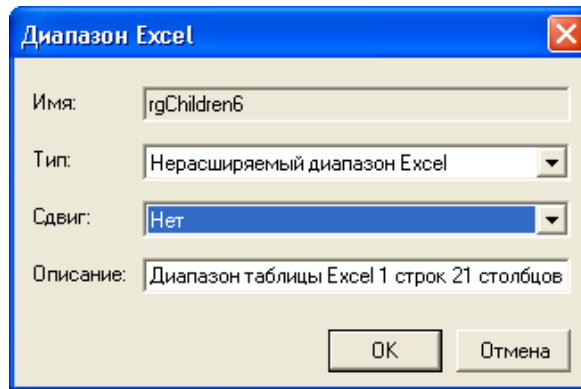
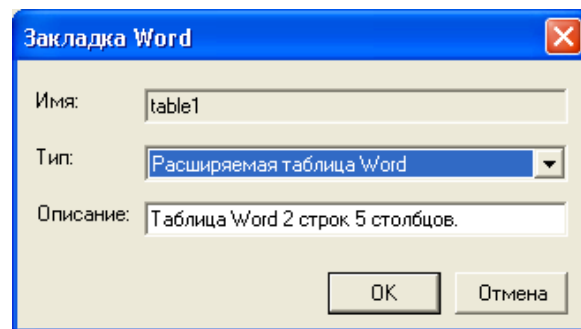


Рис. 76. Список размеченных диапазонов



а



б

Рис. 77. Настройка параметров диапазонов в Excel (а) и Word (б)

Типы диапазонов для шаблонов Word и Excel представлены в табл. 1.

Таблица 1. Типы диапазонов в шаблонах

Диапазоны Excel	
Ячейка таблицы Excel	Диапазон представляет одну ячейку таблицы Excel. Используется для вывода одного значения. Расширение не предусмотрено.
Расширяемый диапазон Excel	Данный вид диапазонов используется для размещения результатов запросов, количество строк которого заранее неизвестно, а число столбцов (полей) фиксировано. Размеры диапазона автоматически изменяются для размещения необходимого числа строк.
Нерасширяемый диапазон Excel	Диапазон представляет фиксированную область таблицы Excel. Используется для вывода данных, размерность которых известна заранее. Расширение не предусмотрено.
Закладки Word	
Закладка Word	Диапазон представляет одну закладку Word, не содержащую таблицы. Данный тип закладок используется для вставки в документ однородного текстового или числового значения.
Расширяемая таблица Word	Данный вид закладок Word используется для размещения результатов запросов, количество строк которого заранее неизвестно, а число столбцов (полей) фиксировано. Закладка содержит в себе таблицу, размеры которой автоматически изменяются для размещения необходимого числа строк.
Нерасширяемая таблица Word	Диапазон представляет собой таблицу с фиксированным количеством строк и столбцов. Используется для вывода данных, размерность которых известна заранее. Расширение не предусмотрено.

Диапазон Excel имеет дополнительный параметр «Сдвиг». Если значение данного параметра для расширяемого диапазона установлено в «Сдвиг таблицы», то при расширении диапазона смещаются вниз все ячейки листа расположенные ниже диапазона. Если значение параметра установлено в «Сдвиг ячеек», то при расширении диапазона смещаются вниз только ячейки расположенные ниже диапазона.

На следующем этапе нужно указать *название шаблона*. Название должно быть уникальным в пределах данного документа.

На завершающем этапе работы мастера в базу данных будет сохранен файл шаблона.

Работа мастера завершается нажатием на кнопку «Готово». При необходимости можно запустить мастер связи запросов с диапазонами шаблона сразу после завершения работы мастера.

Мастер связей

Мастер связей предназначен для установки связи диапазонов шаблонов и запросов. При генерации документа в диапазон документа помещается результат выполнения связанного с ним запроса. Для запуска мастера необходимо выполнить команду «Мастер связей...» в контекстном меню шаблона.

Окно мастера (рис. 78) состоит из трех основных частей. В нижней части располагаются дерево запросов и список диапазонов данного шаблона. Верхнюю часть окна занимает список связанных запросов и диапазонов.

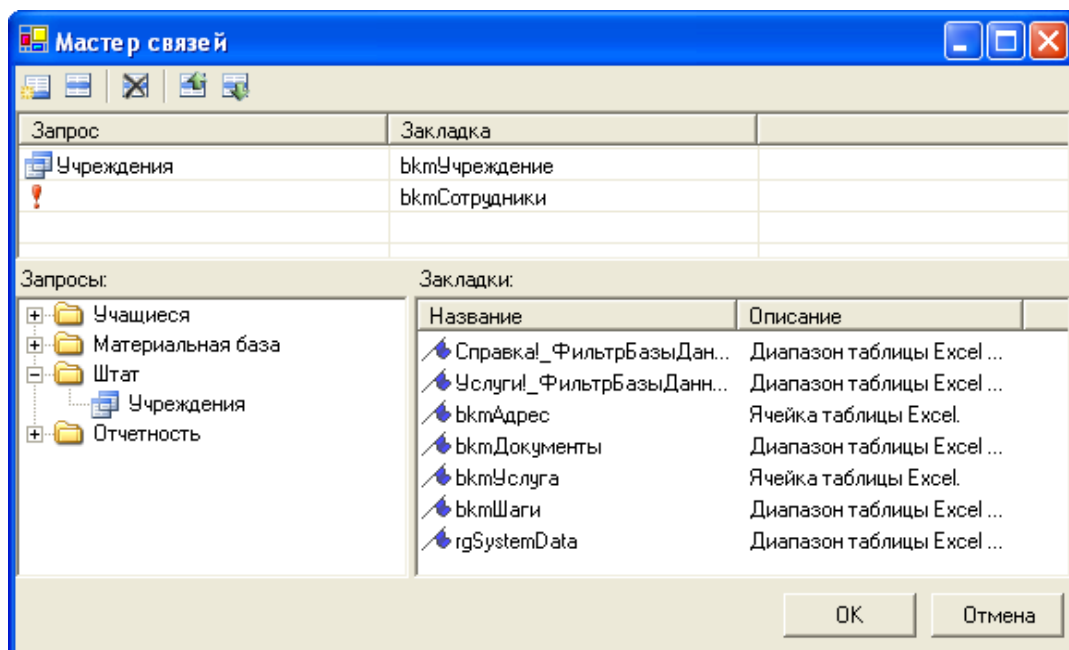


Рис. 78. Окно Мастера связей

Для создания связи диапазона с запросом необходимо поместить их в одну строку списка. Переместить элементы нижних списков в верхний можно с помощью метода «Drag&Drop», т.е. путем перетаскивания элементов мышью. При связывании диапазона он удаляется из списка, т.к. в один диапазон может быть помещен результат только одного запроса. При необходимости один запрос может быть использован при необходимости более чем один раз.

Для разрыва связи запроса и диапазона необходимо удалить связывающую их строку нажатием клавиши *Delete* или с помощью контекстного меню.

Генерация документа

Под генерацией документа подразумевается заполнение диапазонов шаблонов документа результатами выполнения связанных с ними запросов.

Для выполнения генерации документа необходимо его выделить, затем в контекстном меню выбрать команду «Сгенерировать» или нажать соответствующую кнопку на панели инструментов. После выполнения данной команды на экране появится диалоговое окно «Генерация документа». Нажимая клавишу «Далее» необходимо указать требуемые параметры генерации документа: шаблоны, необходимость визуализации процесса генерации, действия при окончании генерации отчета. Визуализация процесса заметно увеличивает время генерации документа, поэтому при работе с большими документами ее лучше не использовать.

Средства тиражирования данных

ИС дает возможность передавать во внешние подсистемы данные, выбираемые из локальной БД системы, подмодели модели данных, настроенные запросы и шаблоны отчетов.

Менеджер схем тиражирования

Для создания копии данных, связанных с некоторым объектом, необходимо выбрать пункт меню «Тиражирование – Создать копию...» Появится диалоговое окно выбора схемы тиражирования (рис. 79).

Схема тиражирования включает в себя объект, информацию о котором необходимо сгенерировать, список связанных с ним сущностей и соответствующие им идентифицирующие атрибуты.

Выбрав схему в списке, пользователь может ее изменить, удалить или переименовать через соответствующие кнопки или контекстное меню (рис. 80).

Создание и изменение схем производится в окне редактора схем тиражирования, представленном на рис. 81.

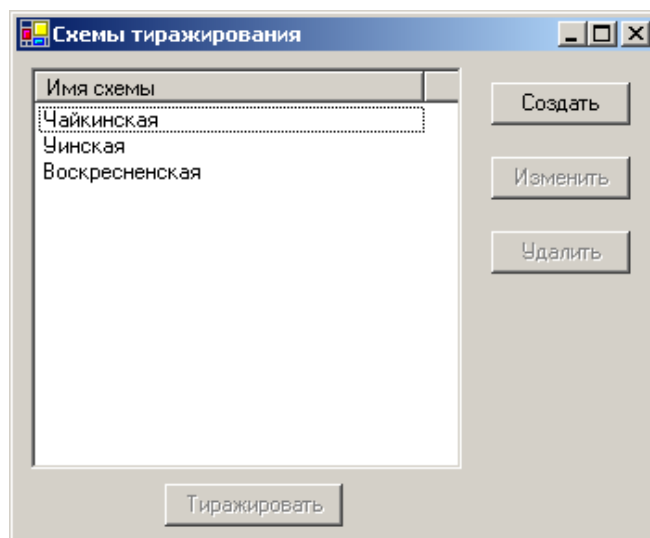


Рис. 79. Выбор схемы тиражирования

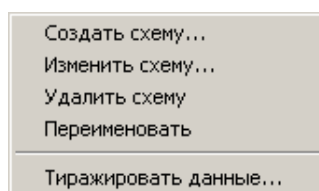


Рис. 80. Операции над схемами

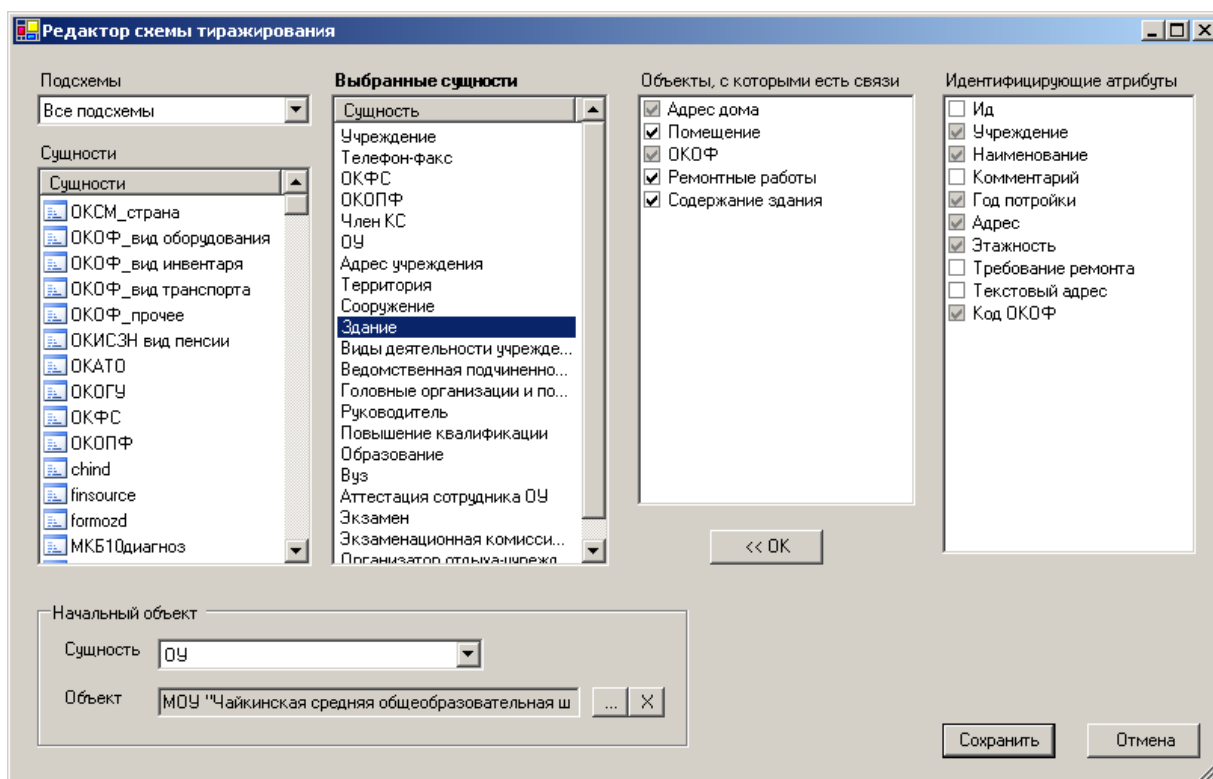


Рис. 81. Окно редактора схем тиражирования

Верхняя часть окна служит для удобного составления списка сущностей, выбираемых для тиражирования. Первый список содержит сущности выбранной подсхемы модели ИС или все сущности сразу (при выборе специального элемента «все подсхемы»). Список посередине – результирующий. Посредством механизма drag-and-drop в него можно добавлять сущности из первого списка. Нажатие клавиши *Delete* или выбор команды «Удалить» контекстного меню позволяет убрать сущность из результирующего списка.

При выделении имени сущности справа отображаются сущности, с которыми связана данная и соответствующие ей идентифицирующие атрибуты. При этом если сущность уже имеется в результирующем списке, то в список сущностей, с которыми выделенная сущность имеет связи, она уже не попадет.

Пользователь должен пройти по этому списку, расставляя «галочки» напротив тех сущностей, который он хочет добавить в результирующий список. «Галочки», которые нельзя снять (отображаются на сером фоне) означают, что данная сущность является родительской и потому должна быть добавлена обязательно. После просмотра списка, при нажатии на кнопку , все отмеченные сущности перемещаются в результирующий список.

По умолчанию каждую сущность идентифицирует набор обязательных атрибутов. Они отмечены «галочками», которые нельзя снять (на сером фоне) в списке «идентифицирующие атрибуты».

Внизу окна редактора схемы задается корневой объект – объект, информацию о котором необходимо сгенерировать. Вначале нужно выбрать сущность этого объекта (одну из сущностей результирующего списка), а затем нажать на кнопку «...» Откроется диалоговое окно выбора начального объекта (рис. 82), которое содержит в виде таблицы все объекты указанной ранее сущности. Сброс выбранного объекта осуществляется нажатием на кнопку .

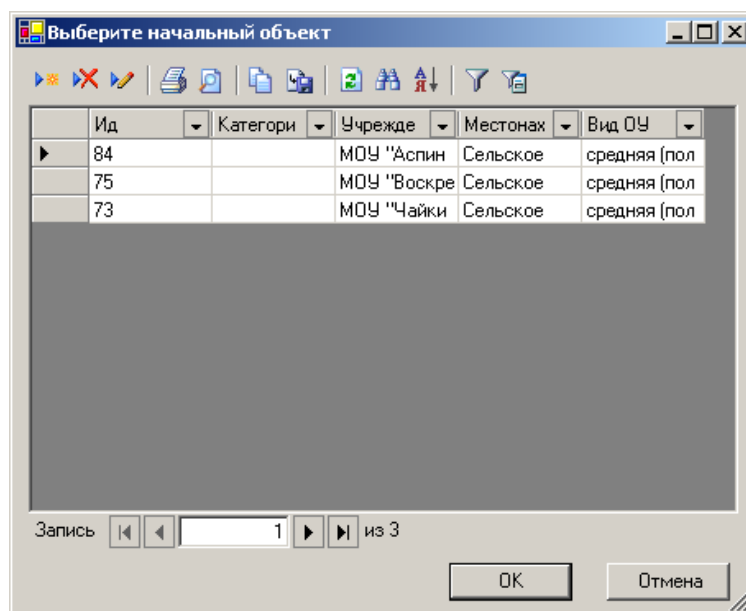


Рис. 82. Выбор основного объекта для реплицирования

Если корневой объект оставить пустым, то он будет запрошен непосредственно при создании копии.

Кнопка «Сохранить» редактора схем тиражирования позволяет принять все изменения и закрывает окно.

Создание резервной копии данных

Чтобы создать резервную копию, необходимо выбрать схему тиражирования и нажать кнопку «Тиражировать» или воспользоваться соответствующей командой контекстного меню.

В появившемся окне (рис. 83) нужно задать имя копии и путь к папке, в которой требуется ее сохранить.

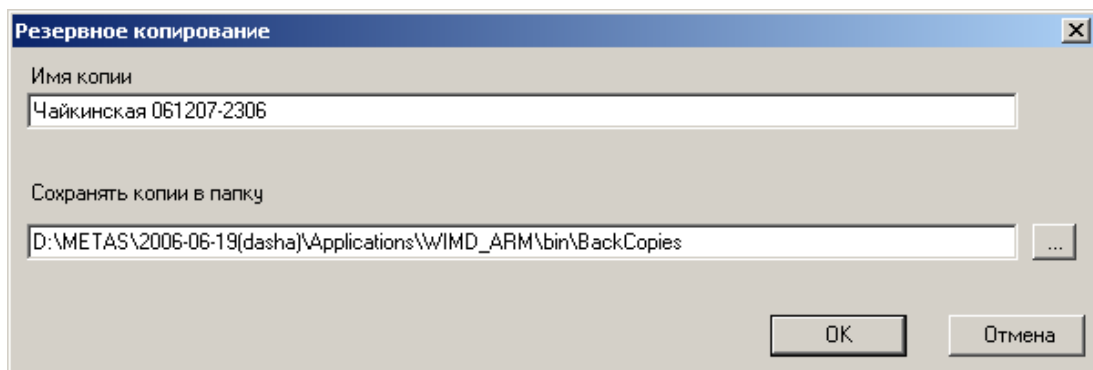


Рис. 83. Задание файла и места для размещения копии

Поскольку создание копии предполагает создание нескольких файлов, имя копии обозначает имя папки, которая будет создана по указанному пути.

Нажатие кнопки «ОК» может привести к длительным вычислениям (время определяется сложностью модели данных ИС и объемом тиражируемых данных). Пользователь должен дождаться сообщения об успешном создании копии.

Этапы создания копии:

1. Вначале происходит сохранение модели данных, образованной указанной схемой тиражирования. В нее включаются структура выбранных сущностей и связи между ними. Сохранение подмодели необходимо при реплицировании данных в адаптируемой системе, допускающей реструктуризацию БД, т.к. в процессе функционирования модель ИС могла быть изменена и при передаче данных во внешнюю систему необходимо контролировать соответствие данных модели.
2. Начальный объект и все объекты, с ним связанные, будут копироваться во временную БД, которая имеет название BackupDB. Поэтому пользователь должен иметь права на создание БД.
3. Далее компонент тиражирования выполняет обход всех сущностей, связанных с начальной.

4. Теперь временная БД BackupDB сформирована. На этом шаге выполняется ее упаковка средствами СУБД.
5. Далее выполняется сохранение описания схемы тиражирования, в соответствии с которой была создана копия.

Файлы, представляющие копию

Созданная копия данных (реплика) состоит из нескольких файлов (табл. 2), которые располагаются в папке с именем, заданным пользователем в поле «имя копии».

Таблица 2. Файлы, входящие в реплику

Имя файла	Описание
BackupInfo.xml	Служебная информация для компонента восстановления. Содержит порядок обхода сущностей при формировании копии.
SchemeDescription.xml	Содержит описание схемы, по которой была создана копия
Replic_PrM_EF.xml	Метаданные презентационного уровня
BackupDB.dat	Непосредственно данные. Файл представляет собой backup-файл СУБД
Replic_entities.md	Метаданные логического уровня (структура сущностей и связи между ними)

Восстановление копии

Для восстановления копии необходимо выбрать команду меню «Тиражирование – Восстановить...». В появившемся окне нужно выбрать папку, которая содержит копию, и нажать кнопку «ОК». Восстановление данных пройдет в автоматическом режиме. При этом будет создана (или обновлена) схема тиражирования, по которой создавалась данная копия.

Этапы восстановления копии:

1. Вначале происходит восстановление модели данных. Для успешной работы алгоритма схема данных на узле-источнике и узле-приемнике должна быть одинакова.
2. Далее средствами СУБД происходит распаковка БД во временную БД с именем RestoreDB.
3. Компонент восстановления данных формирует порядок обхода физических таблиц этой временной БД. В соответствии с этим порядком, происходит копирование объектов в рабочую БД системы.
4. Заключительным этапом является создание (или замещение уже существующей) схемы тиражирования на узле-приемнике, по которой была создана копия на узле-источнике. Схема будет иметь то имя, которая она имела при создании копии.

Средства защиты от несанкционированного доступа

В системе реализованы средства защиты, математической моделью которых является матрица прав доступа, усложненная реализацией средств эквивалентности прав и наследования. Кроме того, реализованы средства контроля непротиворечивости назначаемых прав.

Категории пользователей системы

Все пользователи системы делятся на две категории: «пользователи» и «администраторы». Полномочия *пользователя* в системе определяются в соответствии с правами доступа, назначенными пользователю *администратором*, а также теми *ролями*, которые он исполняет.

Для администрирования системы используется пункт меню «Администрирование». Данный пункт меню доступен только для «администраторов» системы, что позволяет полностью скрыть от обычных пользователей всю часть системы, связанную с администрированием.

Для оперирования пользователями системы используется интерфейс, аналогичный интерфейсу администрирования операционных систем семейства Windows (рис. 84). В левой части формы находится дерево субъектов, вершинами которого являются пользователи и роли системы.

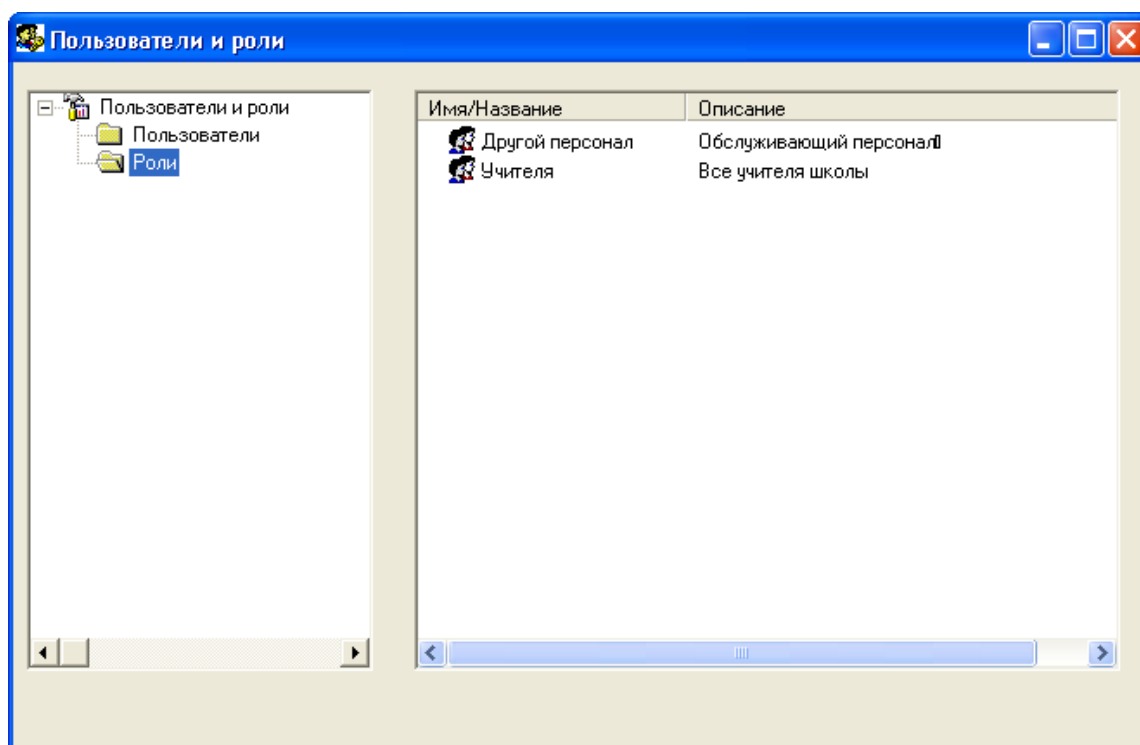


Рис. 84. Форма администрирования

Для удобства работы с *субъектами*, они подразделяются в дереве на две ветви: *пользователи* и *роли*. При выборе одной из этих ветвей в правой части формы отображаются все субъекты из этой ветви с кратким описанием, которое назначается субъекту при создании или в процессе изменения личных данных субъекта.

Пользователи

При регистрации нового пользователя создается запись о нем в БД подсистемы защиты. Необходимой информацией является идентификатор (имя пользователя) и пароль (рис. 85). Эти данные будут вводиться пользователем при входе в систему. Кроме того, можно указать дополнительную информацию, которая не только облегчает манипулирование пользователями (дополнительная информация), но и предоставляет дополнительные возможности с точки зрения защиты (срок действия пароля, категория пользователя).

Рис. 85. Создание пользователя

Для того чтобы просмотреть и/или изменить данные о субъекте необходимо выделить его в списке и нажатием правой кнопки мыши вызвать контекстное меню (рис. 86).

Для того чтобы сменить пароль пользователя необходимо выбрать пункт контекстного меню «Сменить пароль».

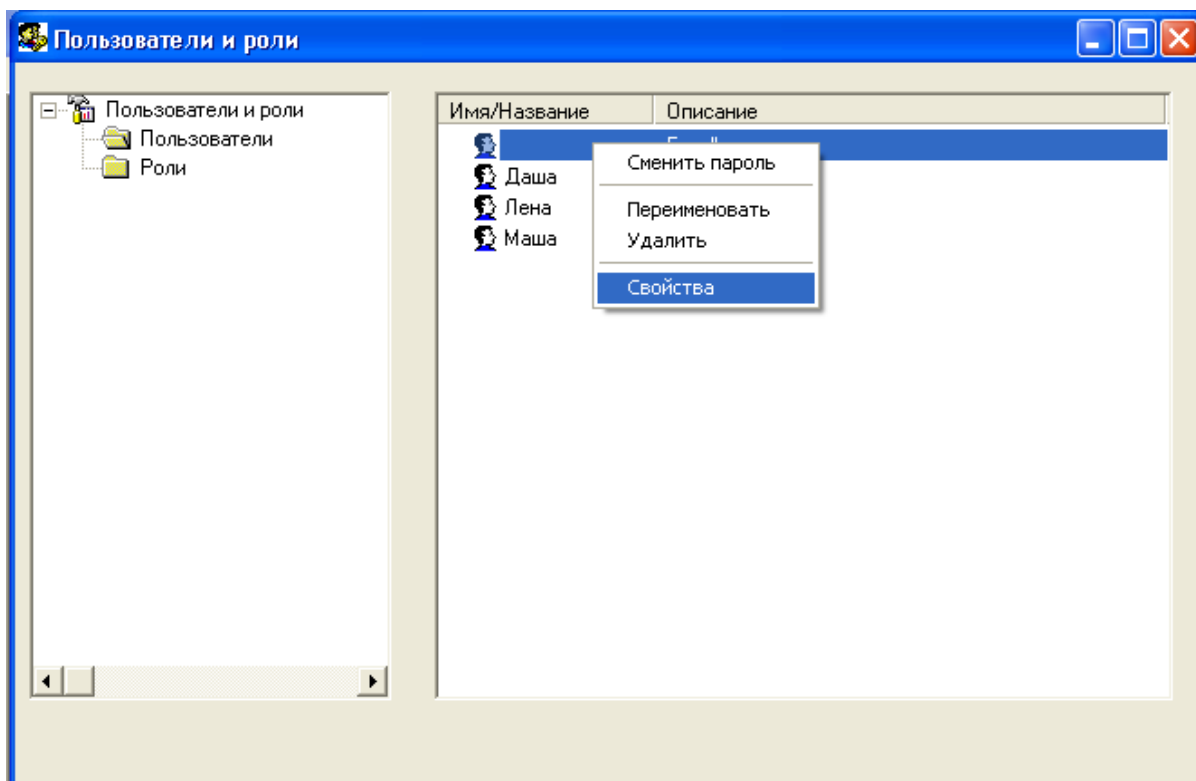


Рис. 86. Изменение информации о пользователе

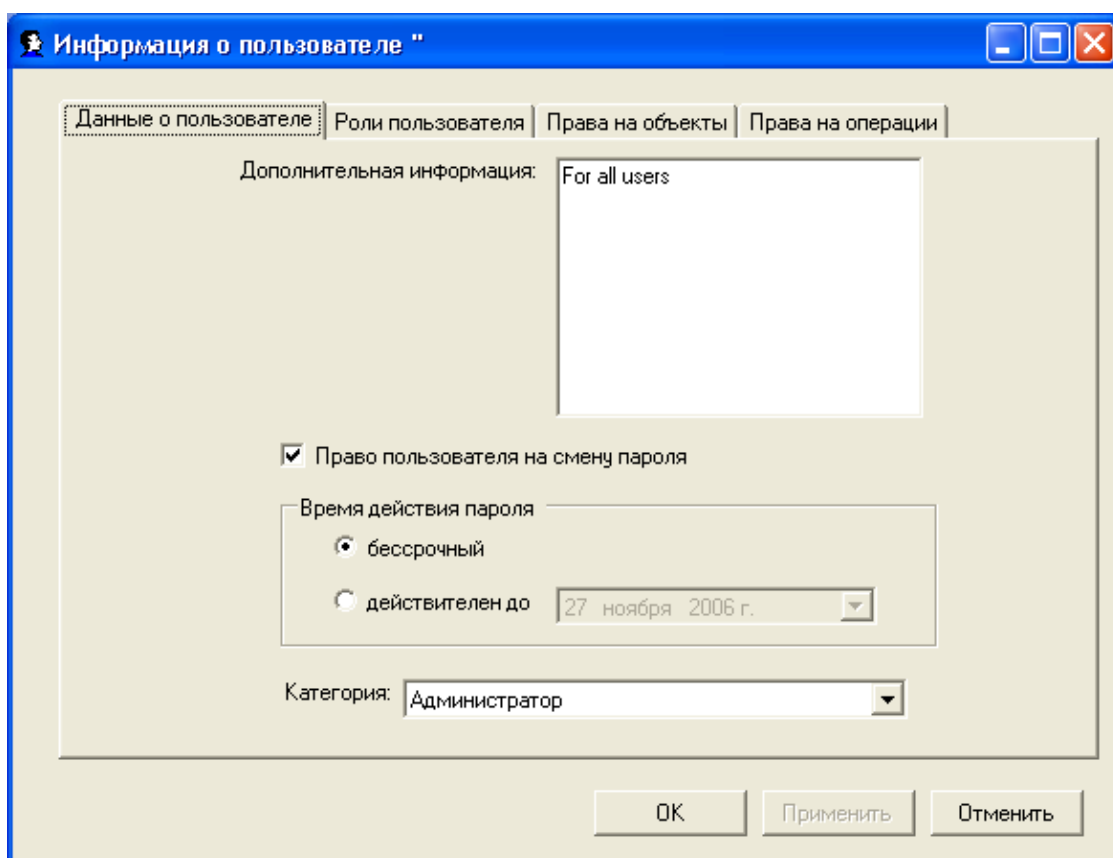


Рис. 87. Данные о пользователе

Для смены идентификатора (имени пользователя) используется пункт контекстного меню «Переименовать».

Пункт «Удалить» используется для удаления пользователя из системы.

Для всех остальных изменений данных о пользователя, а также для просмотра этих данных используется пункт контекстного меню «Свойства».

При выборе пункта «Свойства» администратор работает с формой, представленной ниже (рис. 87). Кнопка «ОК» сохраняет все изменения в форме и закрывает форму, кнопка «Применить» используется для сохранения изменений без закрытия формы, а кнопка «Отмена» - для закрытия формы без сохранения.

На данной форме имеются четыре *вкладки*:

- «Данные о пользователе» – вкладка для просмотра и модификации личных данных о пользователе. На этой вкладке помимо описания (дополнительной информации) пользователя, имеются поля для оперирования паролем (срок действия, право на смену пароля), а также поле для выбора категории пользователя: «администратор», «пользователь». Сама по себе категория уже является ограничением прав, поскольку весь интерфейс и права по администрированию доступны только субъектам с категорией «администратор».
- «Роли пользователя» – вкладка для изменения ролей пользователя. На данной вкладке расположен список всех ролей пользователя, а также кнопка «Изменить» для вызова формы модификации ролей пользователя (рис. 88). На форме назначения пользователю ролей расположены два списка (список всех ролей в системе, а также список ролей назначенных данному пользователю). Для переноса ролей между списками используются кнопки «Добавить» и «Удалить». Кнопка «ОК» используется для сохранения, а кнопка «Отмена» для отмены всех совершенных изменений.
- «Права на объекты» – для оперирования правами пользователя на сущности, атрибуты и операции над сущностями (рис. 89-90). Для назначения прав используются таблицы, в строках которой представлены объекты, а в столбцах – операции которые могут быть выполнены над данным типом объекта. На пересечении строки и столбца находится ячейка, содержащая права на совершение данной операции над данным объектом. Для того чтобы назначить право доступа необходимо в списке прав («>», «есть», «нет», «условное») выбрать нужное право. «>» указывает на то, что право не назначается явно, а наследуется от ролей пользователя.

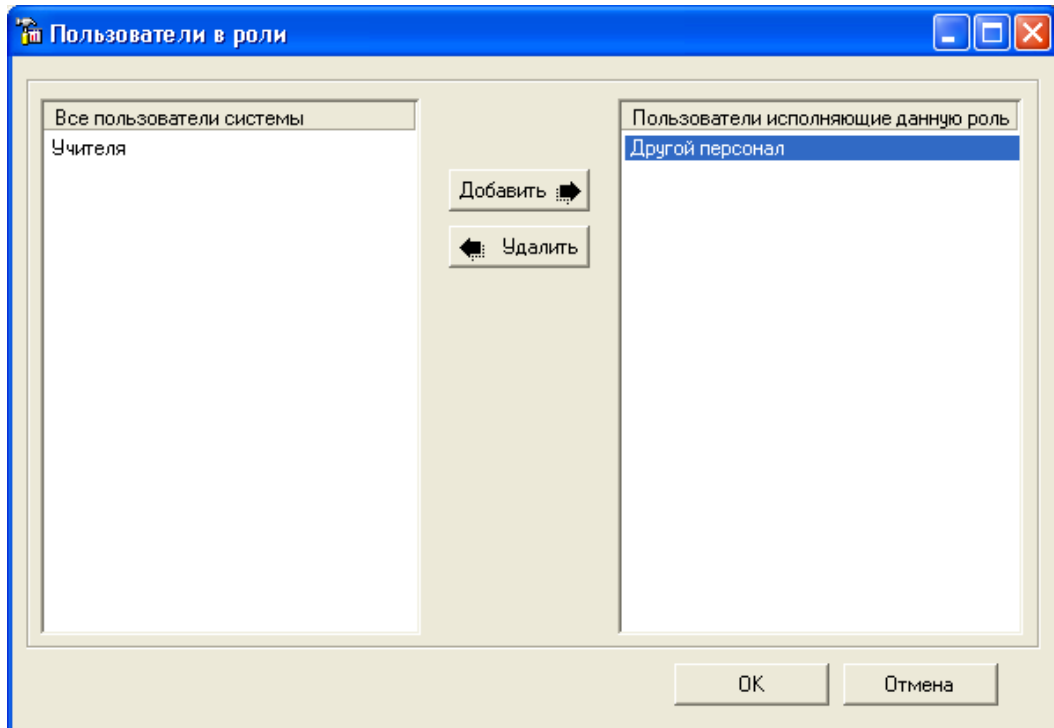


Рис. 88. Назначение пользователю ролей

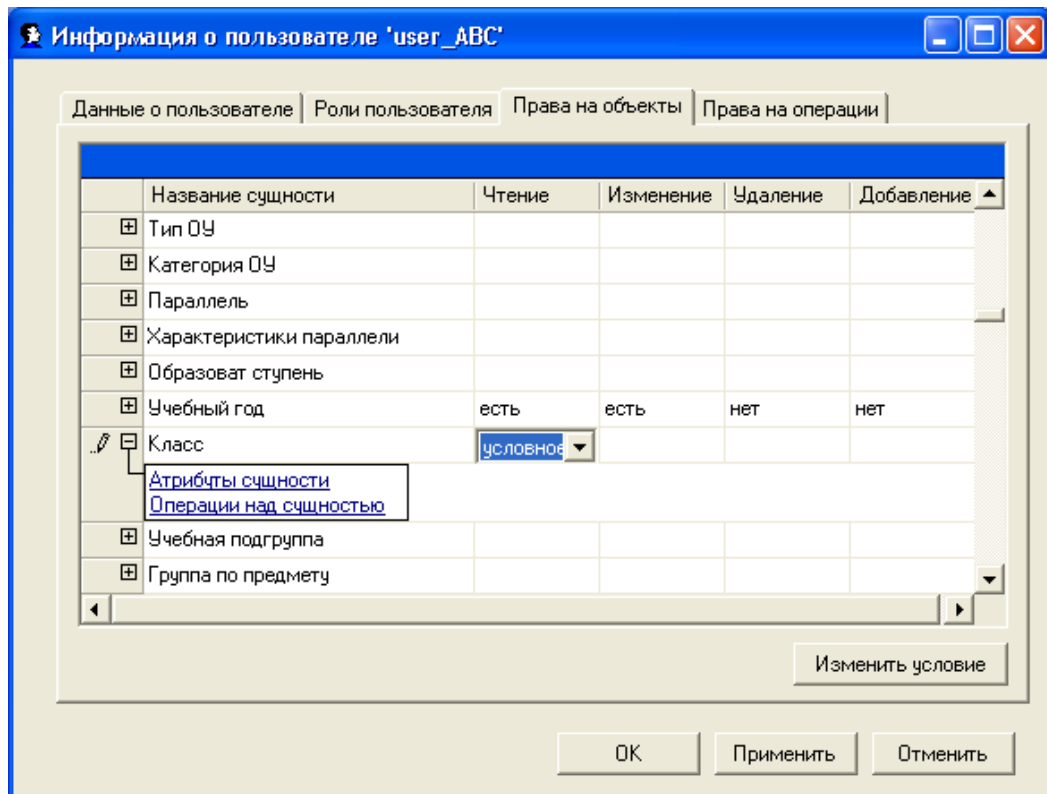


Рис. 89. Назначение прав на сущность

- «Права на операции» – для оперирования правами пользователя на выполнение операций. Права назначаются аналогично правам на сущности, атрибуты и операции над сущностями.

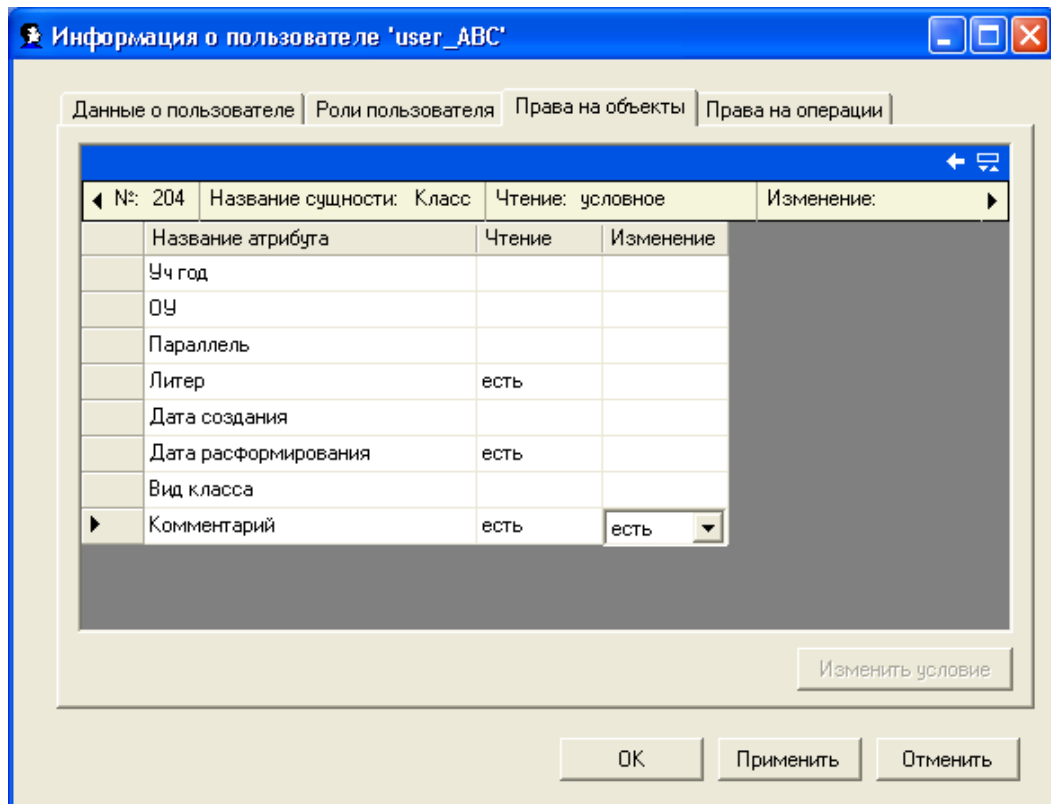


Рис. 90. Назначение прав на атрибуты сущности

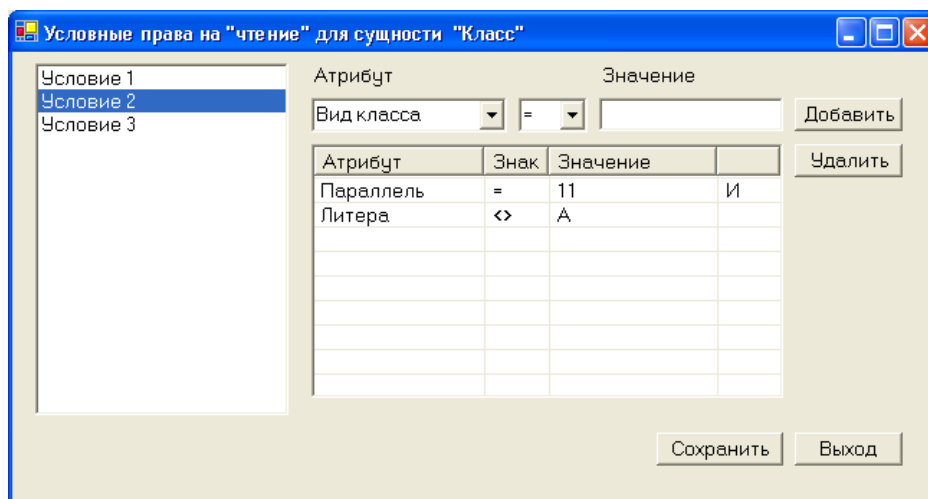


Рис. 91. Назначение условных прав

- *Условные права* используются для назначения прав, зависящих от некоторых условий. Для редактирования этих условий используется кнопка «Изменить условие» (рис. 91). Каждое условное право может включать несколько условий. Для того чтобы у пользователя было право доступа на объект должно выполняться хотя бы одно из условий (т.е.

условия соединяются логической связкой «ИЛИ»). Каждое условие может включать несколько подусловий, которые соединяются логической связкой «И».

Роли

Изменение и просмотр данных о роли происходит аналогичным образом. Единственное серьезное отличие – форма создания роли (рис. 92). На данной форме помимо названия роли и описания роли указывается список пользователей, исполняющих эту роль.

Рис. 92. Создание роли

Порядок назначения прав

Права (полномочия) пользователь на доступ к ресурсам информационной системы определяются их должностными обязанностями. При внедрении системы должны быть выполнены организационные мероприятия, создана нормативная база для работы пользователей, определяющая их права и обязанности при работе с ИС, регламентирующая их работу.

Администратор системы должен организовать работу пользователей, назначив им необходимые для выполнения их обязанностей права.

2.2. Применяемые программные средства

Операционная система и программная платформа

Выбор операционной системы

Выбор ОС (операционной системы) определяется следующими требованиями:

- Поддержка многопоточности.
- Наличие для ОС СУБД серверного типа и программного обеспечения.
- Навыки работы с ОС со стороны потенциальных пользователей.
- Стоимость закупки, установки и настройки ОС.
- Поддержка новейших технологий со стороны ОС.

На сегодняшний день распространены настольные ОС такие как: Семейство ОС Windows, Семейство ОС Unix, Семейство ОС OS/2.

Исходя из требований, приведенных выше, семейство ОС Unix и ОС OS/2 не подходят для реализации клиентской части приложений ИС, так как очень мало пользователей, которые работали бы на данных ОС. У ОС Windows на сегодняшний день также более широкая поддержка новейших технологий.

В минимальной конфигурации все программные компоненты ИС устанавливаются на единственный компьютер, что диктует выбор ОС от одного разработчика.

Исходя из вышеперечисленных требований, выбор ОС очевиден – это операционная система из семейства Microsoft Windows (с Windows 98).

Выбор программной платформы

Ниже приводится ряд требований для программной платформы:

- Поддержка принципов открытых систем.
- Возможность переноса на другую аппаратную или программную платформу.
- Высокая надежность кода.
- Расширяемость.
- Компонентная технология.
- Эффективная защита как Windows-приложений, так и WEB-приложений.

На сегодняшний день существуют две конкурирующие платформы – это JAVA и .NET Framework, которые удовлетворяют вышеперечисленным требованиям. Эти технологии достаточно близки.

Основные отличия JAVA и .NET Framework:

- В .NET реализованы развитые средства проектирования.
- В .NET включена поддержка COM, что позволяет использовать ранее написанные компоненты и визуальные элементы управления.
- .NET – направление которое “родилось” под MS Windows.

- В .NET поддерживаются развитые средства защиты
- В .NET инфраструктура политик и разрешений может быть как собственной, так и полагаться на инфраструктуру политик и разрешений ОС класса Windows NT.
- .NET – мощная интегрированная среда, включающая в себя все необходимое для реализации корпоративных решений и WEB-решений высокого уровня.

Из перечня отличительных особенностей видно, что предпочтение в данной ситуации следует отдать технологии .NET Framework.

Компонентная технология

На сегодняшний день реализация открытой системы немыслима без использования компонентных технологий. В компоненте инкапсулируется его функциональность. Через предоставление внешнего набора интерфейсов обеспечивается сборка компонентов в один программный комплекс, состоящий из многих из функциональных блоков.

Сейчас наиболее законченными и распространенными являются следующие технологии:

- COM – компонентная модель объектов, являющаяся детищем Microsoft и поддерживаемая только на ОС класса MS Windows.
- CORBA – открытая спецификация, разработанная группой поддержки объектной технологии OMG (Object Management Group). Для ее поддержки необходимо установить соответствующее программное обеспечение. Поддерживается большинством распространенных ОС.
- JB – Java Beans – компонентная технология от Java, являющаяся открытой спецификацией. Для работы необходима виртуальная Java-машина. Спектр поддерживаемых ОС также широк. Java Beans ориентирована в основном на графические компоненты.
- EJB – Enterprise Java Beans – продолжение развития Java Beans. Отличие состоит в том, что это уже корпоративные компоненты, в которые инкапсулирована логика приложений и данные, а не только графический интерфейс.
- Компоненты .NET:
 1. Для локального использования (Assemblies – сборки). Компоненты могут предоставлять сервис для нескольких приложений (общедоступные), так и для одного приложения.
 2. .NET Remoting – распределенные компоненты для удаленного доступа. Обладают большей гибкостью и надежностью по сравнению с DCOM. Имеется настройка поведения и создания

компонента, настройка транспортных протоколов и использование своего протокола.

3. WEB Services – компоненты, предоставляющие сервисы для WEB-приложений, причем определение классов ведется на языке программирования C# или VB.NET и потом генерируется в WSDL-документ (WEB-методы ASP.NET описываются на WSDL). WEB-сервисы работают по высокоуровневому протоколу SOAP, причем для данного протокола определен стандарт WS-Security для комплексной защиты передаваемого сообщения между клиентом и сервером независимо от того, какие протоколы используются и используют ли они защищенную передачу.

Исходя из выбора платформы .NET Framework в предыдущем разделе, выбор ложится в основном на компоненты .NET и на COM-технологии (для связи с унаследованными системами).

Системы и языки программирования

Исходя из выбора операционной системы и компонентной технологии программирования, следует остановить свой выбор на системах программирования, поддерживающих технологию .NET Framework. Данной системой на сегодняшний день является Visual Studio .NET. Данная система способна генерировать высокоуровневый код MSIL, являющийся независимым от аппаратной платформы. Система сочетает в себе три языка программирования, основанные на общей библиотеке типов. Dot NET – мощная интегрированная среда, включающая в себя все необходимое для разработки корпоративных решений и WEB-решений высокого уровня.

Выбор языка программирования определяется только личными предпочтениями разработчиков, языки отличаются, в основном, только синтаксисом. Из трех языков, включенных в Visual Studio Dot Net (VC++, C#, VB), выбор ложится на VB и C#. При написании приложений можно использовать как управляемый (managed), так и неуправляемый (unmanaged) код. При реализации данного проекта используется именно управляемый код, который характеризуется повышенной надежностью, менее подвержен ошибкам при написании программ за счет того, что во время выполнения среда CLR (common language runtime) обеспечивает нужную защиту памяти программы.

СУБД и методы доступа к данным

По той причине, что планируется работа на простых ОС класса рабочих станций, а главным образом – для обеспечения адаптируемости и расширяемости, в частности, за счет смены СУБД, желательно обеспечить возможность выбора между СУБД. Поэтому интегрированные системы,

сочетающие в себе СУБД и системы программирования не подходят, так как впоследствии смена СУБД без перезаписи всего программного кода практически невозможна, так как смешана логика приложения и работа с данными. Если исходить из современного трехуровневого подхода к написанию программ, то такой подход смешивает уровень данных и уровень бизнес логики приложения. Поэтому необходимо работать с некоторой технологией доступа к данным.

Выбор технологии доступа к данным

Для разработки приложений, работающих под управлением ОС MS Windows, распространены следующие технологии:

- ODBC (Open Database Connectivity) – открытый низкоуровневый интерфейс для работы с реляционными данными.
- OLE DB – открытый низкоуровневый интерфейс, построенный в соответствии со спецификациями ODBC. OLE DB определяет открытый стандарт доступа к любым данным, причем как к реляционным, так и не к реляционным источникам информации (в отличие от ODBC).
- DAO – предоставляет модель объектов для доступа к локальным базам данных или базам данных SQL через ядро Jet. DAO позволяет обращаться к базам данных Microsoft Jet (напрямую или через ODBC) и к таким источникам данных ISAM, как FoxPro, Paradox или Lotus. В сравнении с более новыми технологиями DAO работает довольно медленно.
- ADO (ActiveX Data Objects) – объектная модель данных на основе технологии COM, предоставляет модель объектов для доступа к данным любых типов через OLE DB. Используя ADO в языках программирования, поддерживающих COM, можно обращаться к любым источникам данных OLE DB – от реляционных баз данных до систем электронной почты.
- ADO.NET – следующий шаг развития ADO, теперь это – улучшенная модель для работы на платформе .NET Framework. ADO.NET обеспечивает мощную поддержку XML и автономную обработку данных.

Наиболее подходящей технологией является ADO.NET по следующим причинам:

- ADO.NET позволяет обращаться к данным любых типов, а DAO – только к реляционным.
- Модель объектов в ADO.NET проще, чем в DAO, и с ней легче работать.
- ADO.NET сочетает в себе лучшие качества DAO и, в конечном счете, полностью заменит их.

- ADO.NET – стандартная модель объектов для доступа к данным во всех средствах разработки, поддерживающих COM, включая Visual Basic, Access, VBA (в Office), Visual Studio .NET.
- ADO.NET – улучшенная версия ADO для платформы .NET Framework.
- ADO.NET – гармоничное сочетание с ASP.NET, при реализации WEB-интерфейса для доступа к данным.
- ADO.NET имеет хорошую поддержку XML.

Таким образом, получаем весь спектр СУБД, для которых существуют провайдеры данных OLE DB или ODBC.

Ограничения при проектировании базы данных

Ограничения, которые необходимо соблюдать при проектировании БД для обеспечения возможности беспрепятственного перехода на другие СУБД:

- Использование базовых типов данных (пересечения типов данных всех современных СУБД). Каждый «экзотический» тип данных сужает область приемлемых для системы СУБД.
- Не использовать хранимые запросы и процедуры, т.е. уровень бизнес-логики при работе с данными необходимо выносить из СУБД, т.к. хранимые запросы и процедуры специфичны для каждой СУБД и их вообще может не быть (пример: в MS Access есть хранимые параметризованные запросы, но нет процедур). При этом сервер приложений должен находиться в достаточной близости к данным. Если все же необходимо использовать эти возможности, то круг СУБД так же, как и в первом случае, сужается.
- Использование стандарта SQL2, без специфических расширений отдельных СУБД.
- Не использовать булевские поля (логический тип), т.к. в разных СУБД их значения кодируются и интерпретируются по-разному. (Пример: MS Access использует константы “TRUE” и “FALSE”, значения которых – “0” и “-1”. MS SQL Server 7.0 не использует константы, а использует значения соответственно “0” и “любое не 0, обычно – -1”.

Все эти ограничения должны выполняться при разработке ПО системы.

Замечание: Хотя и декларируется не использовать хранимые процедуры и запросы, в некоторых случаях, критичных к производительности, все-таки их использование допускается. К примеру, выполнение длительных расчетов, которые оперируют данными из большого числа таблиц и/или оперируют большими наборами данных, – в этом случае такую расчетную функцию лучше разместить на СУБД в виде хранимой процедуры. Реализация же таких функций может быть оформлена как нестандартная бизнес-логика системы.

Выбор СУБД

Исходя из выбора технологии для доступа к данным и условий работы, необходимо выбрать СУБД по следующим критериям:

- Работа в среде MS Windows.
- Поддержка репликации.
- OLAP и Data Mining.
- Использование реляционной СУБД, т.к. с использованием объектно-ориентированных СУБД (ОО СУБД) связан определенный риск.

Наиболее распространенными серверными СУБД, удовлетворяющими перечисленным требованиям, являются:

- MS SQL Server – серверная СУБД, имеющая средства OLAP, XML, многомерных баз данных, хранилищ данных, репликации данных, хранимых процедур, хранимых параметризованных запросов, OLE DB провайдера в ОС.
- Oracle – серверная СУБД, имеющая средства: OLAP, XML, многомерных баз данных, хранилищ данных, репликации данных, хранимых процедур, OLE DB провайдера в ОС.

По перечисленным выше характеристикам данные СУБД являются примерно одинаковыми по возможностям. Но наиболее подходящей является все-таки MS SQL Server, по причине того, что работа ведется с продуктами компании Microsoft и система разработки MS Visual Studio Dot NET имеет встроенную поддержку и интеграцию с данной СУБД, но иметь совместимость с ORACLE, для хранения базы данных, также необходимо.

Технологии для интеграции приложений

Для информационной связи как внутри организации, так и с внешними организациями необходимо выбрать ПО. Данное ПО должно обеспечивать обмен данными и документами (отчеты, приказы, поручения, справки и т.д.)

Основные требования:

- Связь по различным протоколам, причем обязательны протоколы HTTP, E-mail, File.
- XML – как основной формат данных при обмене.
- Возможность подключения внешних модулей сторонних производителей, для обеспечения расширяемости и интеграции данного ПО, таких как:
 - компоненты расширения базового набора протоколов,
 - компоненты нестандартной логики при обработке документа,
 - компоненты для изменения формата отправляемого или получаемого документа.
- Надежность при отправке и получении документов.

- Нетребовательность к программным и аппаратным ресурсам.
- Мониторинг обработанных документов, ошибок и событий системы.
- Маршрутизация документов.

На сегодняшний день для организации документооборота существует новейшая технология BizTalk Framework, основанная на XML, SOAP и открытости систем.

Как альтернатива Microsoft BizTalk Server, менее требовательная к аппаратным и программным ресурсам, предлагается приложение “Simple BizTalk Server”, разработанное в рамках создания технологии, имеющее следующие характеристики:

- Нетребовательность к программным и аппаратным ресурсам.
- Универсальность представления связей между приложениями.
- Расширяемость.
- Настраиваемость практически на любую предметную область, за счет универсального представления и возможности подключения внешних модулей.
- Мониторинг документов, ошибок и событий системы.
- Надежность обработки документа в системе.
- Поддержка основных транспортных протоколов, таких как: HTTP, POP3, SMTP, COM/DCOM, File.
- Преобразование документов в каналах.
- Полная поддержка XML.

Установка программного комплекса METAS 2.0

Программный комплекс METAS имеет различные варианты установки, описанные ниже. Для его функционирования необходимо обеспечить программное окружение (табл. 3.)

Таблица 3. Требования к программному окружению

Компонент	Сервер (Мин/Рек)	Клиент (Мин/Рек)
<i>Операционная система</i>	Win98 SE / Win 2000 Server и выше	Win98 SE / Win2000 Professional и выше
<i>.NET Framework</i>	версия 1.1 и версия 2	версия 1.1 и версия 2
<i>СУБД</i>	MS SQL Server	MS SQL Server Express
<i>IExplorer</i>	IE 6.0	IE 6.0
<i>Офисное ПО</i>		Office 97/2000/XP/2003

Установка должна выполняться системным администратором, который должен выполнить следующие действия:

1. Установить серверное программное обеспечение, удовлетворяющее требованиям для конкретного варианта установки.
2. Развернуть базы данных.

3. Установить программное обеспечение METAS для сервера.
4. Установить клиентское программное обеспечение, удовлетворяющее требованиям для конкретного варианта установки.
5. Установить программное обеспечение METAS для клиента.
6. Настроить подключения клиентов к серверу.

2.3. Используемые технические средства и требования к аппаратуре

При установке программного обеспечения рассматривается два варианта, требования которых описаны ниже.

Установка пакета разработчика METAS Developer

Этот вариант установки предназначен для установки полной системы METAS со всеми CASE-инструментами. Он обеспечивает пользователю-разработчику визуальную среду проектирования на основе UML-диаграмм. Минимальные требования к аппаратуре приведены в табл. 4.

Таблица 4. Аппаратные требования для установки METAS Developer

Компонент	Сервер (Мин/Рек)	Клиент (Мин/Рек)
<i>Компьютер</i>	PC совместимый	PC совместимый
<i>Процессор</i>	500 / 1000 (МГц)	600 / 1500 (МГц)
<i>ОЗУ</i>	128 / 256 (Мб)	128 / 256 (Мб)
<i>Жесткий диск</i>	7200 / 10000 (об/с.)	5400 / 7200 (об/с.)
<i>Место на диске</i>	300 / 800 (Мб)	200 / 500 (Мб)
<i>Видеокарта и монитор</i>	SVGA 800×600	SVGA 800×600 / 1024×768

Установка системы без средств разработки

При установке программного обеспечения, разработанного с помощью CASE-системы METAS, без средств разработки требования приведены в табл. 5.

Таблица 5. Аппаратные требования

Компонент	Сервер (Мин/Рек)	Клиент (Мин/Рек)
<i>Компьютер</i>	PC совместимый	PC совместимый
<i>Процессор</i>	500 / 1000 (МГц)	166 / 500 (МГц)
<i>ОЗУ</i>	128 / 256 (Мб)	64 / 128 (Мб)
<i>Жесткий диск</i>	7200 / 10000 (об/с.)	3300 / 5400 (об/с.)
<i>Место на диске</i>	300 / 800 (Мб)	200 / 500 (Мб)
<i>Видеокарта и монитор</i>	SVGA 800×600	SVGA 800×600 / 1024×768

3. Специальные условия применения и требования организационного, технического и технологического характера

Для работы с программами нет необходимости создания специальных условий применения и выполнения особых требований организационного, технического и технологического характера. Условия применения и требования определяются требованиями к применяемому программному и аппаратному обеспечению, перечисленными выше, а также выполнением лицензионных соглашений при использовании необходимого для работы программ программного обеспечения.

Требования к квалификации пользователей определяются выполняемыми ими обязанностями. Минимальные требования:

- навыки работы в среде Microsoft Windows;
- навыки работы с приложениями пакета Microsoft Office (Word и Excel).

Разработчики, использующие для создания ИС CASE-технология METAS должны удовлетворять следующим квалификационным требованиям:

- знания и навыки в области проектирования реляционных баз данных, использования языка UML для создания моделей предметной области (в частности, диаграмм классов);
- умение создавать документы Microsoft Office (Word и Excel), использовать средства их автоматизации;
- знания и навыки программирования для создания программных компонентов, расширяющих возможности системы, реализующих нестандартную логику, специфичную для конкретной предметной области;
- знания и навыки администрирования операционных систем Microsoft Windows, используемых для создания БД информационной системы СУБД.

4. Условия передачи программной документации или ее продажи

Продажа и передача программ и программной документации возможна на основе договора с АНО науки и образования «Институт компьютеринга». Условия передачи и/или продажи полностью определяются договором, заключаемым заказчиками/покупателями с АНО «Институт компьютеринга».

Представителем АНО «Институт компьютеринга», имеющим право заключать договоры на передачу и/или продажу программного обеспечения, на разработки программ с использованием представленных в данном документе средств, является заместитель директора Лядова Л.Н. (e-mail: lnlyadova@mail.ru; тел.: +7 (342) 222-37-95).

Библиографический список

1. Бек К. Экстремальное программирование. СПб: Питер, 2002.
2. Борисова Д.А. Применение графовых моделей метаданных при создании компонентов CASE-системы METAS // В кн.: Математика программных систем: Межвузовский сборник научных трудов / Перм. ун-т – Пермь, 2006. С. 14-21.
3. Борисова Д.А., Лядова Л.Н. Иерархическая модель данных как основа реализации информационной системы, управляемой метаданными // В кн.: Математика программных систем: Межвузовский сборник научных трудов / Перм. ун-т – Пермь, 2006. С. 4-13.
4. Борисова Д.А., Лядова Л.Н. Решение задачи автоматизированного создания (реинжиниринга) информационных систем в CASE-системе METAS // В кн.: Технологии Microsoft в теории и практике программирования. Материалы конференции / Под ред. проф. Р.Г. Стронгина. Нижний Новгород: Изд-во Нижегородского госуниверситета, 2006. С. 34-36.
5. Вендров А.М. CASE-технологии. Современные методы и средства проектирования информационных систем; [<http://www.infocity.kiev.ua>]; 2002.
6. Гайсарян С.С. Объектно-ориентированное проектирование; [<http://www.citmgu.ru>]; 2001.
7. Гамма Э., Хелм Р., Джонсон Р., Влисситес Дж. Примеры объектно-ориентированного проектирования. Паттерны проектирования. СПб: Питер, 2001.
8. Еремина М.Е. Операции над моделями данных и их отображение в презентационной модели CASE-системы METAS // В кн.: Технологии Microsoft в теории и практике программирования. Материалы конференции / Под ред. проф. Р.Г. Стронгина. Нижний Новгород: Изд-во Нижегородского госуниверситета, 2006. С. 56-65.
9. Еремина М.Е., Лядова Л.Н. Моделирование предметной области и верификация моделей в информационных системах, основанных на технологии METAS // В кн.: Математика программных систем: Межвузовский сборник научных трудов / Перм. ун-т – Пермь, 2006, с. 22-30.
10. Золотухин П.И. Адаптируемые приложения: ActiveX Scripting как альтернатива VBA; [<http://www.gotdotnet.ru/LearnDotNet/NETFramework/44748.aspx>]; 2004.
11. Калянов Г.Н. CASE-технологии: консалтинг в автоматизации бизнес-процессов. М: Горячая линия–Телеком, 2002.

12. *Куделько Е.Ю.* Операции над моделями данных и их отображение в презентационной модели CASE-системы METAS // В кн.: Математика программных систем: Межвузовский сборник научных трудов / Перм. ун-т – Пермь, 2006. С. 56-65.
13. *Куделько Е.Ю., Лядова Л.Н.* Модель объектов пользовательского интерфейса METAS // В кн.: Математика программных систем: Межвузовский сборник научных трудов / Перм. ун-т – Пермь, 2006, с. 40-55.
14. *Ланин В.В., Лядова Л.Н., Рыжков С.А.* Технологии создания информационных систем для поддержки оперативного мониторинга в сфере образования // Современные информационные технологии в образовательном процессе и научных исследованиях. Материалы Международной научно-практической конференции / Под ред. В.Н. Федосеева. – Шуя: Изд-во «Весть» ГОУ ВПО «ШГПУ», 2004. С. 165-169.
15. *Лядова Л.Н., Рыжков С.А.* CASE-технология METAS // Математика программных систем: Межвуз. сб. научн. тр. / Перм. ун-т. Пермь, 2003. С. 4-18.
16. *Лядова Л.Н., Рыжков С.А.* METAS – технология создания информационных систем сферы образования // XIV Международная конференция-выставка «Информационные технологии в образовании»: Сборник трудов участников конференции. Часть IV. – М.: МИФИ, 2004. С. 38-40.
17. *Лядова Л.Н., Рыжков С.А.* Технология создания информационных систем, управляемых метаданными // Математические методы и информационные технологии в экономике, социологии и образовании: Сборник статей XIII Международной научно-технической конференции / Пенза: Пензенская государственная технологическая академия, Приволжский дом знаний, 2004. С. 316-319.
18. *Лядова Л.Н.* Архитектура информационной системы «Образование Пермской области» // Математика программных систем: Межвуз. сб. научн. тр. / Перм. ун-т. Пермь, 2002. С. 25-35.
19. *Лядова Л.Н.* Метамоделирование и многоуровневые метаданные как основа технологии создания адаптируемых информационных систем // Advanced Studies in Software and Knowledge Engineering. International Book Series “Information Science & Computing”, Number 4. Supplement to the International Journal “Information Technologies & Knowledge. Volume 2, 2008. Institute of Information Theories and Applications FOI ITHEA, Sofia, Bulgaria. Pp. 125-132. Papers are selected from Proceedings of the International Conferences of the Join International Events of Informatics “ITA 2008” (“e.TECH 2008”). Varna, Bulgaria, 2008.

20. *Лядова Л.Н.* О результатах создания и перспективах развития Региональной информационной системы образования и науки Пермского края // В кн.: Математика программных систем: Межвузовский сборник научных трудов / Перм. ун-т – Пермь, 2006, с. 147-163.
21. *Лядова Л.Н.* Региональная информационная система «Образование Пермской области» // Образовательная среда сегодня и завтра: Материалы II Всероссийской научно-практической конференции // М.: Рособразование, 2005. С. 337-339.
22. *Лядова Л.Н.* Технология создания динамически адаптируемых информационных систем // Труды международных научно-технических конференций «Интеллектуальные системы» (AIS'07) и «Интеллектуальные САПР» (CAD-2007). Научное издание в 4-х томах. Т. 2. – М.: Физматлит, 2007. С. 350-357.
23. *Лядова Л.Н., Мороз А.А.* Модель защиты программного обеспечения от несанкционированного распространения // Сборник трудов Второй международной научно-технической конференции «Инфокоммуникационные технологии в науке, производстве и образовании» (Инфоком-2) / Кисловодск, 2006. С. 120-124.
24. *Миков А.И., Лядова Л.Н., Воронцова Т.В.* Информационная система «Образование Пермской области» // XII конференция-выставка «Информационные технологии в образовании»: Сб. тр. конф. Часть IV. М.: МИФИ, 2002. С. 204-207.
25. Разработка концепции информационного развития;
[<http://www.blogic.ru/benefits/default.asp?cl=1&pl=3>]; 2004.
26. *Рыжков С.А.* Концепция метаданных в разработке информационных систем // Математика программных систем: Межвуз. сб. научн. тр. / Перм. ун-т. Пермь, 2002. С. 36-44.
27. *Рыжков С.А.* Программное обеспечение интеграции приложений на основе технологии BizTalk Framework // Математика программных систем: Межвуз. сб. науч. трудов / Перм. ун-т. Пермь, 2002. С. 45-59.
28. *Санблэд С., Санблэд П.* Разработка масштабируемых приложений для Microsoft Windows. Мастер-класс / Пер. с англ. М.: Издательско-торговый дом «Русская Редакция», 2002.
29. *Таненбаум Э., Стен М.* Распределенные системы. Принципы и парадигмы. СПб.: Питер, 2003.
30. *Цыбин А.В., Лядова Л.Н.* Автоматическая генерация документации пользователя в информационных системах, управляемых метаданными // Математика программных систем: Межвуз. сб. науч. трудов / Перм. ун-т. Пермь, 2002. С. 178-188.

31. *Якобсон А., Буч Г., Рамбо Дж.* Унифицированный процесс разработки программного обеспечения. СПб: Питер, 2002.
32. *Chichagova M.V., Lyadova L.N.* Access Rights Inheritance in Information Systems Controlled by Metadata // The Fourth International Conference “Information Research and Applications” (i.TECH’2006). Proceedings of conference / Varna (Bulgaria), June 20-25, 2006. Pp.169-173.
33. *Chichagova M.V., Lyadova L.N.* Access Rights Inheritance in Information Systems Controlled by Metadata // International Journal “Information Theories & Applications”. Volume 14/2007, Number 2. Pp. 189-192.
34. *Lyadova L.N.* Technology for Development of Adaptable Information System // Proceedings of the International Scientific Conferences “Intelligent Systems” (AIS’07) and “Intelligent CAD’s” (CAD’2007) / Scientific publication in 4 volumes. V. 4. – Moscow: Physmathlit, 2007. P. 113.
35. *Lyadova L.N.* The Multilevel Metadata as the Basis of Technology for Creation of Information Systems // Information Technologies and Telecommunications in Science and Education (IT&T ES’2005). Materials of the International Scientific Conference / Turkey, 2005. Moscow: VIZCOM, 2005. .Pp. 83-86.

Приложение 1. Основные обозначения и сокращения

- БД – база данных.
- БМД – база метаданных.
- ИС – информационная система.
- СУБД – система управления базами данных.
- CASE (Computer-Aided Software/System Engineering) –
Автоматизированная разработка программного обеспечения.
- DSL – (Domain Specific Language) – предметно-ориентированный
язык.
- IIS – Internet Information Server.
- METAS (METAdata System) – система, основанная на метаданных.
- UML (Unified Modeling Language) – унифицированный язык
моделирования.

СОДЕРЖАНИЕ

1. ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ ПРОГРАММЫ, ОБЛАСТЬ ЕЕ ПРИМЕНЕНИЯ, ЕЕ ОГРАНИЧЕНИЯ	1
1.1. <i>Назначение программы</i>	2
1.2. <i>Область применения программы</i>	3
1.3. <i>Ограничения использования программы</i>	4
2. ТЕХНИЧЕСКОЕ ОПИСАНИЕ ПРОГРАММЫ	5
2.1. <i>Структура программного продукта</i>	5
Метаданные системы	6
Средства визуального проектирования	7
Интерпретатор метаданных	8
Средства адаптации	9
Модели и программные компоненты METAS	10
Архитектура информационной системы	12
Методология и схема разработки ИС	15
Средства реструктуризации	16
Средства настройки интерфейса пользователя	33
Средства репортинга: создание запросов и отчетов	54
Средства тиражирования данных	69
Средства защиты от несанкционированного доступа	74
2.2. <i>Применяемые программные средства</i>	81
Операционная система и программная платформа	81
Компонентная технология	82
Системы и языки программирования	83
СУБД и методы доступа к данным	83
Технологии для интеграции приложений	86
Установка программного комплекса METAS 2.0	87
2.3. <i>Используемые технические средства и требования к аппаратуре</i>	88
Установка пакета разработчика METAS Developer	88
Установка системы без средств разработки	88
3. СПЕЦИАЛЬНЫЕ УСЛОВИЯ ПРИМЕНЕНИЯ И ТРЕБОВАНИЯ ОРГАНИЗАЦИОННОГО, ТЕХНИЧЕСКОГО И ТЕХНОЛОГИЧЕСКОГО ХАРАКТЕРА	89
4. УСЛОВИЯ ПЕРЕДАЧИ ПРОГРАММНОЙ ДОКУМЕНТАЦИИ ИЛИ ЕЕ ПРОДАЖИ	89
Библиографический список	90
Приложение 1. Основные обозначения и сокращения	94