ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ
Государственное образовательное учреждение
высшего профессионального образования
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Факультет информатики

Кафедра теоретических основ информатики

УДК 681.03

ДОПУСТИТЬ К ЗАЩИТЕ В ГАК
Зав. кафедрой, проф., д.т.н.
_____ Костюк Ю.Л.
« »_____2005 г.

Серафимович Ирина Валерьевна

РАЗРАБОТКА ЭЛЕКТРОННОГО УЧЕБНОГО КУРСА «ПРОЕКТИРОВАНИЕ БАЗ ДАННЫХ В РАСШИРЕННОЙ МОДЕЛИ ДАННЫХ «СУЩНОСТЬ – СВЯЗЬ»

Дипломная работа

Научный руководитель,
ст. преп., к.т.н. А.М. Бабанов

Исполнитель,
студ. гр. 1401 И. В. Серафимович

Электронная версия дипломной работы помещена
в электронную библиотеку. Файл
Администратор

Томск – 2005

Реферат

Дипломная работа 52 с., 7 рис., 32 источника, 7 прил.

ЭЛЕКТРОННЫЙ УЧЕБНЫЙ КУРС, ПРОЕКТИРОВАНИЕ БАЗ ДАННЫХ, МОДЕЛЬ ДАННЫХ «СУЩНОСТЬ-СВЯЗЬ», ИНТЕРФЕЙС ПОЛЬЗОВАТЕЛЯ, ER-ДИАГРАММА

Объект исследования – концептуальное проектирование баз данных.

Цель – разработка раздела электронного учебного курса "Проектирование баз данных", посвященного проектированию баз данных в расширенной модели данных «сущность – связь» и содержащего теоретический лекционный материал на основе единого демонстрационного примера и инструмент для формирования навыков построения ER-диаграмм.

Метод проведения работы – проектирование и реализация.

Результат – создан электронный учебный курс по проектированию баз данных в EER-модели и инструмент для рисования ER и EER-диаграмм.

Содержание

Введение

В настоящее время образовательная методика ищет новые способы реализации принципов обучения. Традиционные способы обучения, предполагающие посещение лекций, написание конспектов, чтение бумажных учебников, непосредственный контакт с преподавателем, все еще остаются на первом месте, однако в качестве альтернативы все более распространенным становится применение электронных обучающих комплексов, которые можно классифицировать в соответствии с их назначением, способом подачи материала, проверки усвоения знаний. Использование компьютерных технологий в сфере образования открывает новые возможности как для подачи учебного материала (нелинейная организация курса, использование текстовой, речевой, музыкальной, графической, видеоинформации), так и для контроля за ходом обучения (тесты с различными формами вопросов, практические задания, лабораторные практикумы и пр.). Таким образом, «внедрение новых информационных технологий в образовании позволит от традиционного процесса преподавания перейти к процессу обучения**.** К такому процессу, в котором сам обучаемый определяет ход процесса обучения»[1].

В нашей стране до сих пор не существует единых стандартов, регламентирующих методику составления электронных учебных систем, равно как и отсутствуют серьезные исследования, посвященные разработке концепции таких систем. При этом, однако, выработан некоторый ряд общих методических рекомендаций по разработке электронных учебных курсов, а также ряд требований, касающихся их содержания. В отечественной методике обучения формальный подход к данной проблеме еще настолько не развит, что очень тяжело обнаружить серьезные работы, посвященные данной тематике. Вопросы о построении электронных курсов обсуждаются на конференциях, симпозиумах, и, как правило, находят свое изложение в форме небольших статей. В качестве примера таких статей можно привести [1-6].

В данной дипломной работе был реализован электронный учебный курс, посвященный проектированию баз данных в терминах модели данных "Сущность – связь", включая ее расширенную нотацию, использующую понятия специализаций и категоризаций.

1. Анализ требований к электронному учебному курсу

1.1 Общие определения

В современной практике обучения, как уже говорилось выше, не существует глубоких исследований, позволяющих формально определить общие концепции электронного обучающего курса. В связи с этим не существует общепринятой терминологии для такого рода систем. В статье [2] приводятся следующие определения:

«Учебник – учебное издание, содержащее систематическое изложение учебной дисциплины или ее раздела, части, соответствующее Государственному образовательному стандарту и учебной программе и официально утвержденное в качестве данного вида издания.

Электронный учебник – учебное электронное издание, созданное на высоком научном, методическом и техническом уровне, соответствующее составляющей дисциплине Государственного образовательного стандарта, учебного плана и рабочей программе. Другое определение электронного учебника дано в [7].

Электронный учебный курс (ЭУК) – учебное издание электронного типа, соответствующее учебной дисциплине, частично или полностью заменяющее (дополняющее) базовый учебник; это совокупность графической, текстовой, речевой, музыкальной, видео-, фото- и другой информации, а также печатной документации пользователя».

Таким образом, мы видим, что электронный учебный курс должен представлять собой целостную дидактическую систему, которая в идеале должна обеспечивать обучение студентов по индивидуальным программам и предоставлять возможность тщательного контроля за ходом обучения.

1.2 Назначение и классификация электронных учебных курсов

В зависимости от целей, которые ставит перед собой составитель учебного курса, он может реализовать его в самых разных вариантах. Назначение курса определит те возможности, которые будет иметь обучающийся в ходе освоения курса. К электронным учебным курсам могут относиться электронные курсы лекций, учебные справочники и базы данных, компьютерные сборники упражнений и задач и т. д. При этом курс должен не только давать справочную и теоретическую информацию по той или иной дисциплине, но и предоставлять возможности учащемуся и виртуальному преподавателю следить за качеством усвоения получаемой информации и приобретаемым умением использовать полученные знания в конкретных практических целях.

Куфлей [2] предлагает классификацию электронных обучающих курсов по двум основаниям: по типу изложения материала и по характеру взаимодействия учащегося и программы курса.

На основании первого критерия он выделяет:

1. текстовый учебный курс;

2. гипертекстовый учебный курс (изложение в виде разветвленного “дерева” взаимных ссылок);

3. учебный курс справочного характера (изложение материала в виде справочника со

свободным входом в любую часть учебного материала);

4. игровой учебный курс (изложение материала в виде деловой, ролевой игры).

По типу взаимодействия предложены следующие классы курсов:

1. информационные (изложение в классическом учебном виде);

2 . “вопрос – ответ” (изложение с акцентированием на конкретные вопросы, проблемы и задачи);

3. информационно-контролирующие (чередование учебного материала и проверяющих вопросов);

4. с обратной информационной связью (интерактивный учебный курс, предусматривающий постоянную оценку знаний обучаемого и выдачу рекомендаций по дальнейшему “движению” по учебному материалу);

5. с пороговыми уровнями контроля (переход к очередному разделу учебного материала возможен только после положительного преодоления контрольных испытаний на предыдущем этапе обучения).

1.3 Критерии оценки качества электронного учебного курса

В статье [5] приведен ряд критериев, в соответствии с которыми можно оценить качество электронного учебного курса.

В педагогических экспериментах все критерии принято делить на количественные и качественные. В качестве количественных предложены следующие критерии:

– объем используемых знаний;

– коэффициент усвоения учебного материала, который равен отношению объема учебного материала, усвоенного учащимся в течение определенной единицы времени к материалу, сообщенному учащемуся за то же время;

– коэффициент прочности усвоения учебного материала как отношение запомнившегося материала и материала сообщенного учащимся в процессе обучения за определенный период.

Качественные критерии отождествляются с уровнями знания:

– учебного материала;

– понимания учебного материала;

– овладения учебным материалом (умение фактически использовать усвоенное при решении практических задач);

– овладения интеллектуальными навыками (умение трансформировать усвоенный материал в новых условиях сознательно и оперативно).

Одним из вариантов оценки качества электронного курса являются опросные листы оценки качества.

1.4 Контроль знаний

Одним из главных компонентов обучающей системы является подсистема контроля знаний. В идеальном случае она должна наиболее адекватно и правильно оценивать знания и навыки учащегося в соответствии с требованиями, предъявляемыми к обучающей системе и зависящими от специфики предмета обучения. Так, контроль знаний должен, во-первых, оценить качество и количество усвоенного учащимся материала и определить, достигнут ли требуемый уровень, что в свою очередь будет сигнализировать об успешном прохождении курса. Если специфика предмета предполагает помимо теоретических знаний овладение и усвоение ряда практических навыков, необходимо проверить, решена ли эта задача для каждого конкретного обучающегося. Иногда может потребоваться оценка того, сможет ли человек, прошедший курс, применить полученные знания и навыки при решении схожих или смежных задач, прямо не затрагивавшихся в рамках конкретной обучающей программы.

В [8] автор выделяет два основных аспекта проблем компьютерного контроля знаний: методологический и технический. К первому Зайцева Л. В. относит «планирование и организацию проведения контроля; определение типов вопросов и отбор заданий для проверки знаний студентов; формирование набора вопросов и заданий для опроса; определение критериев оценки выполнения каждого задания и контрольной работы в целом и др.». В качестве технического аспекта рассматриваются «автоматическое формирование набора контрольных заданий на основе выбранного подхода; выбор и использование в системе контроля параметров КЗ; выбор алгоритмов для оценки знаний учащихся и др.». Существует целый ряд работ, посвященных отдельным вопросам контроля знаний, включая разработку и классификацию различных типов вопросов и их параметров, методов проведения контроля знаний и оценки знаний учащихся. На рисунке 1 представлена слегка модифицированная, по сравнению с предлагаемой в [8], схема классификации методов проведения контроля. Согласно рисунку, все методы контроля делятся на:

– неадаптивные

– частично-адаптивные

– адаптивные.

Рисунок 1 – методы проведения контроля

Общим для всех неадаптивных методов контроля является то, что тестовое задание формируется до начала проведения контроля и в ходе него никаким образом не изменяется, вне зависимости от поведения и особенностей студента.

При формировании набора контрольных вопросов в частично-неадаптивных методах используется информация либо из модели студента (в более общем случае ее называют моделью обучающегося), либо из модели учебного материала.

В адаптивных методах подразумевается наиболее тщательный учет самой разнообразной информации из модели студента. Одновременно может использоваться и информация из модели учебного материала. Общий обзор моделей студента рассматривается в частности в [9]. Важной составляющей модели обучающегося является психофизиологическая компонента. Ее выявление и разработка представляют собой отдельную сложную проблему [10].

В [8] приведена следующая таблица методов проведения контроля и используемых моделей.

Таблица 1 Методы проведения контроля и используемые модели

	Метод проведения контроля	Тип метода	Время формирования заданий	Используемые модели и параметры
1	Строгая последовательность	Неадаптивный	До контроля	нет
2	Случайная выборка	Неадаптивный	Непосредственно перед контролем	нет
3	Комбинированный метод	Неадаптивный	Непосредственно перед контролем	нет
4	Случайная выборка с учетом отдельных параметров модели	Частично-адаптивный	Непосредственно перед контролем	Модель студента: уровень подготовленности
5	Контроль на основе ответов студента	Частично-адаптивный	До контроля (и в процессе контроля)	Модель студента: текущие ответы
6	Контроль на основе модели учебного материала	Частично-адаптивный	В процессе контроля	Модели УМ, МС: уровень
7	Модульно-рейтинговый метод	Частично-адаптивный	В процессе контроля	Модель студента: рейтинг студента
8	Контроль по модели студента	Адаптивный	В процессе контроля	Модель студента
9	Контроль по моделям студента и учебного материала	Адаптивный	В процессе контроля	Модель студента, модуль УМ

После того, как выбран метод контроля знаний, необходимо определить, каким образом оценить полученные в результате такого контроля результаты. Рисунок 2 [8] приводит

основную классификацию методов оценки знаний.

Литература, посвященная различным аспектам созданий тестов (методологическим, психологическим), достаточно обширна и разнообразна и представлена как внушительными научными трудами, так и небольшими статьями [8,11].

Рис. 2 – Методы оценки знаний

1.5 Анализ существующих учебных курсов по теме проектирования баз данных

В статье [12] приведена типичная последовательность действий, выполняемых в рамках классической методологии проектирования. И хотя в разных источниках эта последовательность может несколько видоизменяться, в целом ее можно взять за основу. Согласно классической методологии выделяется 3 основных этапа проектирования баз данных: концептуальное проектирование, логическое проектирование и физическое проектирование.

На фазе концептуального проектирования принято говорить о семантическом, или инфологическом моделировании [12], [13]. Целью семантического моделирования является определение предметной области системы и взгляда на нее с точки зрения будущего пользователя базы данных. При этом формируется такое представление предметной области, которое даст возможность хранить всю необходимую информацию и обеспечивать требуемую функциональность. Во внимание не берется эффективность хранения и работы с данными, простота и удобство оперирования хранимой информацией и прочие аспекты, относящиеся к логическому и физическому проектированию.

Фаза логического проектирования подразумевает создание логической модели предметной области в терминах модели данных, реализуемой в выбранной СУБД. Так, если в качестве целевой СУБД выбрана реляционная, необходимо представить предметную область в виде совокупности отношений.

Под физическим проектированием понимается непосредственное создание физических структур базы данных, то есть перевод логической модели в модель хранения. Здесь также решаются вопросы распределения хранимых данных, организации доступа к ним, повышения производительности операций над данными и т. д.

Таким образом, полноценный курс по проектированию баз данных должен давать представление о всех трех фазах проектирования [14]. Очевидно, что такое представление

не может быть исчерпывающим, поскольку объем учебного материала тогда был бы неподъемен ни для написания, ни для изучения. В связи с этим часть существующих курсов акцентируют свое внимание на той или иной фазе проектирования.

Обычно подробно рассматривается логическое проектирование, предваряемое более или менее детальными сведениями о методиках концептуального проектирования. Как правило, рассматривается модель данных «сущность – связь». Очень редко – расширенная модель данных «сущность – связь». Такой подход аргументируется тем, что во многих случаях классическая ER-модель, предложенная Ченом [15], достаточна для адекватного отражения предметной области. Возможно, предполагается, что более сложные системы будут проектироваться с использованием других методологий или моделей данных, например, в рамках унифицированного процесса.

Многие курсы акцентированы на фазе собственно логического проектирования, причем в качестве наиболее популярной модели данных используется реляционная модель. Другие учебные материалы посвящены физическому проектированию в рамках конкретной СУБД, при этом знание вышележащих аспектов проектирования подразумевается.

Одним из наиболее важных недостатков многих существующих курсов является отсутствие единого демонстрационного примера, иллюстрирующего большую часть тех теоретических аспектов, которые автор стремится донести до обучающихся. Как правило, используются маленькие, не связанные друг с другом примеры. Такой подход не формирует у учащихся целостную картину описываемых концепций. Часто имеет место ситуация, когда, понимая отдельные аспекты, учащийся испытывает сложности при интегрировании разрозненных представлений для проектирования более сложных схем. И даже если учащийся сможет преодолеть эти трудности, необходимо, чтобы на достаточно большом примере он смог оценить, насколько он правильно это сделал, то есть, чтобы ему было с чем сравнить свой результат.

1.6 Создание электронных обучающих систем как проблема

Вышеизложенное позволяет сделать вывод, что создание полноценной обучающей среды является сложной многоаспектной задачей. Для того чтобы решить ее в полной мере, необходимо привлечение самых разных научных дисциплин и методологий проектирования и разработки.

Так, полноценная модель обучающегося не может быть построена без глубоких знаний психологии, педагогики, процесса того, как человек приобретает знания и навыки. Эти же и смежные дисциплины, включая психофизиологию, позволяют сформулировать требования к интерфейсу обучающей программы с тем, чтобы максимальным образом увеличить эффективность подачи и усвоения материала. Проще говоря, обучающемуся должно быть удобно и интересно работать с программой [16-18]. Одновременно с этим появляются требования к дизайну программы как таковому, поскольку не достаточно определить, каким его надо сделать – его надо еще и непосредственно реализовать. Внешний вид программы – это то, на что в первую очередь обращает внимание пользователь, и если он невыразителен, вне зависимости от того, насколько информационно и содержательно насыщенным является курс, провал ему обеспечен.

Для оценки знаний учащихся используются математические и вероятностные модели. Очевидно, что для создания хорошего курса необходимы эксперты в той предметной области, которая преподается.

Далее, необходима высококачественная техническая реализация курса, что является,

в первую очередь, задачей программиста.

Множество литературы посвящено тому, как создать успешный (с точки зрения поставленной цели) электронный обучающий продукт [19-24].

Одной из важнейших задач является задача формулирования требований к обучающей системе. Количество их огромно, и определены они могут быть самыми разными способами. Таким образом, и в этой области нельзя обойтись без упорядочивания, структурирования, стандартизации таких требований. В настоящее время существует целый ряд организаций и объединений, чья деятельность направлена на:

«– создание концептуальной модели стандартизации в системе открытого образования (IEEE); разработку архитектуры технологических систем в образовании AICC, IMS, ISO/IEC JTC1 SC36;

– разработку внутренних стандартов и спецификаций для корпоративного обучения и переподготовки персонала компаний (AICC);

– решение задач в области телематики и мультимедиа в образовании для Европейского Сообщества (ARIADNE, PROMETEUS); формирование учебного контента для учебных заведений, ориентированных на Интернет-обучение (проект SCORM)»[25].

Однако этим спектр возникающих проблем не ограничивается, поскольку сам процесс того, как человек учится, иными словами, как он вдруг научается понимать и оперировать теми понятиями, с которыми он раньше не был знаком, не выяснен до конца. Более того, до сих пор существует путаница в понимании того, что есть информация, данные, знания и что есть процесс обучения как таковой, в связи с чем возникают вопросы подобные следующим: «Что есть знания? Является ли обучение передачей знания? Что такое е-знания? Является ли е-обучение передачей е-знаний?» [26,27].

Таким образом, создание полноценного электронного обучающего курса, удовлетворяющего как объективным, так и специфическим пользовательским требованиям, является непростой задачей и, как правило, не может быть осуществлено одним человеком или небольшой группой людей.

2. Выбор средств для практической реализации

2.1 Анализ существующих средств для создания электронных учебных курсов

В первом разделе отчета было дано определение электронного учебного курса, в соответствии с которым он должен представлять собой «совокупность графической, текстовой, речевой, музыкальной, видео-, фото- и другой информации».

Таким образом, средством для практической реализации простейшего электронного курса является любой инструмент, позволяющий создать такую совокупность. Теоретически это может быть сделано с помощью многих языков программирования высокого уровня, поскольку почти все из них уже обладают возможностями для включения в программу различных мультимедийных компонентов. Выбор именно таких средств имеет ряд недостатков:

– во-первых, выбирая какой-либо язык программирования, в большинстве случаев придется делать выбор, каким именно способом смогут учащиеся получать доступ к создаваемому электронному курсу. В настоящее время существует две основные альтернативы: создавать web-курс для представления его в сети Internet либо реализовывать его как самостоятельное приложение. Для первого варианта, например, можно выбрать реализацию курса в виде набора html-страниц, обрабатываемых каким-либо скриптовым языком, таким как php. Для второго варианта можно использовать визуальные средства разработки на базе языков Delphi, Visual Basic и т. д. Однако в любом случае придется приложить немалые дополнительные усилия (если не писать два разных курса), чтобы обеспечить представление курса в двух разных формах.

– во-вторых, очевидно, что неадекватно большие усилия будут потрачены на реализацию взаимодействия различных частей курса, включая навигацию и отслеживание процесса обучения, поскольку такие возможности не встроены изначально в языки программирования и их надо будет писать самостоятельно практически с нуля. Если для небольших курсов такой подход еще может быть приемлем, то для серьезных обучающих систем он может оказаться чересчур трудоемким, громоздким и нестабильно работающим.

Существует и еще один недостаток. Разработка электронного учебного курса подразумевает выполнение, по крайней мере, трех ролей с различными специализациями, которые совмещаются либо в одном разработчике, либо распределяются по разным:

– необходима роль «методиста» – человека, который отслеживает теоретическое и практическое наполнение курса с точки зрения преподаваемой в нем дисциплины или демонстрируемых навыков. Этот же человек может выполнять роль эксперта при создании подсистемы тестирования определять, какие именно реакции учащегося ожидаются и будут оценены как правильные;

– другая роль принадлежит «дизайнеру» – это человек, отвечающий за интерфейс обучающего курса, его цветовую гамму, расположение компонентов, словом, за художественное оформление. При этом в идеале он должен обладать психологическими и педагогическими навыками, чтобы за счет интерфейса улучшить восприятие учебного материала;

– наконец, третья роль отведена «проектировщику-программисту», который разрабатывает логическую схему курса, обеспечивает взаимодействие его различных частей и берет на себя программную реализацию.

При использовании языков программирования высокого уровня львиная доля работы

оказывается на плечах программиста, а в некоторых средствах вообще тяжело отделить интерфейс курса от его программной части, поэтому дизайнер не может работать отдельно от программиста, что также можно рассматривать как недостаток такого подхода.

Выходом из этой ситуации может рассматриваться использование специализированных средств для создания различных электронных мультимедийных приложений, включая и электронные учебные курсы. Эти средства позволяют устранить вышеперечисленные недостатки.

Во-первых, они, как правило, дают возможность создавать как электронные курсы для internet, так и самостоятельные приложения, распространяемые на различных носителях, причем для этого не надо писать две различные программы.

Во-вторых, они имеют визуальную среду, которая позволят размещать мультимедийные компоненты курса, не используя программный код, а также имеют встроенные средства, упрощающие навигацию в рамках создаваемого курса.

Одновременно с этим они имеют встроенный язык написания скриптов, имеющий действительно большие возможности для функционального наполнения курса, включая взаимодействие с пользователем, сохранение данных на внешних носителях либо пересылку результатов деятельности учащегося на сервер со специализированным серверным программным обеспечением, которое впоследствии осуществляет обработку этих данных.

Одним из самых известных специализированных средств для разработки электронных обучающих курсов, созданных в России, является eAuthor 2.0 компании «ГиперМетод». Согласно информации, размещенной на сайте компании, «с помощью eAuthor v 2.0 – конструктора дистанционных курсов – можно создать целый ряд разнообразных учебных электронных изданий – мультимедийных учебных курсов, тестирующих систем, учебно-методических комплектов, учебных программ различных дисциплин и т.д.» [28]. В качестве основных свойств продукта декларируются следующие:

«– возможность интеграции внешних программ (редакторов) для обработки (правки) мультимедийных объектов;

– ручная и автоматическая расстановка гипертекстовых ссылок;

– ручное и автоматическое разбиение сплошных текстов на части - лекции, занятия и т.п.;

– структурирование и управление учебным материалом - в тексте могут быть указаны определения, примеры, примечания, важные мысли и пр. понятийные единицы - для отображения учебного материала в нормальном, развернутом и конспективном виде;

– блоки тестирования могут быть включены непосредственно в лекционный материал, а также вынесены в отдельный аттестационный блок. Результаты тестирования фиксируются для дальнейшего контроля уровня подготовки» [28].

Одним из наиболее популярных продуктов западного производства, предназначенных для создания полноценных электронных учебных курсов, является продукт компании Macromedia «Authorware». В настоящее время доступна седьмая версия этой системы. «Пакет Authorware предназначен для создания компактных мультимедийных приложений, предусматривающих совместное использование различных форм подачи материала, текста, рисунков, видео и звукового сопровождения. Входящие в состав Authorware средства позволяют практически в полном объеме реализовать современные требования к построению и организации систем электронного обучения» [30].

Существуют и другие программные средства данного класса, но они не так распространены, особенно в России.

Однако такую ситуацию мы имеем, если рассматриваем создание обучающих программ и электронные курсов как бы самих по себе. В более сложном случае электронный курс является интегрированным в систему дистанционного образования, и здесь надо уже говорить не только об инструментах создания курсов, но и о программной платформе в целом [29]. В [29] в качестве основных элементов функционально полной платформы, которую автор определяет как "взаимосвязанный комплекс программ, предназначенный для организации и проведения дистанционного обучения", выделяются:

1. Средства создания контента (то есть, наполнения учебного курса), которые представляют собой инструменты для авторского дизайна, включая дизайн текстов, графики и прочих мультимедийных материалов, а также инструменты для импортирования контента в обучающую среду.

2. Средства управления контентом, которые позволяют изменять, дополнять, удалять контент, а также доставлять его до потребителя – студента.

3. Средства управления и поддержки процесса обучения, включающие возможности регистрации и удаления студентов, создания отчетности о деятельности обучающихся, их успеваемости, "посещаемости", а также средства, которые позволят оценить эту деятельность, то есть тесты, экзамены, различного рода задания.

В настоящий момент тенденция такова, что эти составляющие реализуются и продаются по отдельности. Поскольку все эти продукты, особенно зарубежные, являются дорогостоящими, гораздо удобнее и выгоднее платить только за то, что действительно потребуется, а при необходимости получения дополнительных возможностей данные средства, особенно производимые одним разработчиком, поддерживают тесную интеграцию друг с другом. Таким образом, при выборе инструмента для разработки электронного учебного курса необходимо прежде всего определиться с функциональностью, которая должна поддерживаться.

Одновременно инструменты разработки обучающих систем можно разделить на клиентские и серверные. Как следует из названия, клиентские компоненты ответственны за сбор данных о деятельности пользователя, но обработкой этих данных занимаются серверные компоненты. Как правило, они составляют так называемую систему управления обучением (LMS – Learning Management System или CMI – Computer-Managed Instruction).

Так, Authorware является клиентским компонентом, то есть имеет средства для создания содержания курса, тестовых заданий и отслеживания деятельности учащегося. Последнее реализуется с помощью целого ряда встроенных системных переменных и системных функций. Изменение значений системных переменных происходит автоматически при выполнении студентом некоторых действий с программой и отслеживает самые разные аспекты этой деятельности. Системные функции предназначены для пересылки данных, собранных в переменных, на сервер.

2.2 Выбор инструментов для реализации

Для создания электронного учебного курса был выбран пакет Macromedia Authorware 7.0. Этот выбор был обусловлен тем, что компания Macromedia предоставляет также другие продукты и системы, которые помогают в процессе создания учебного курса. Так, для создания некоторых графических элементов курса использовался пакет для работы с графикой различных форматов Macromedia Fireworks.

Кроме этого, пакет Authorware имеет встроенный язык написания скриптов (при

этом в версии 7.0 в качестве альтернативы появился язык JavaScript, возможности которого дублируют функциональность встроенного языка). Более того, данный продукт поддерживает расширение своей функциональности за счет использования специальных компонентов-расширений, называемых Xtras. Это специализированные компоненты для выполнения определенных функций, разрабатываемые как самой компанией Macromedia, так и сторонними разработчиками. Так, например, с их помощью Authorware поддерживает использование при создании учебных курсов ActiveX компонентов. Это обеспечивает новое измерение в процессе создания обучающего курса и делает возможности разработчика если не бесконечными, то весьма широкими.

К недостаткам данного пакета, хотя весьма субъективным, можно отнести практически полное отсутствие документации на русском языке (так, в русскоязычном Интернете я не обнаружила ни одного сайта, посвященного данному продукту, и в России вышла единственная книга о Macromedia Authorware [30]).

Проектирование баз данных в ER-модели как предмет обучения имеет свою специфику при оценивании понимания учащимся излагаемых концепций. Очевидно, что проверка усвоения материала, заключающаяся исключительно в оценке того, насколько качественно учащийся запомнил названия терминов и обозначений модели, недостаточна. Необходим инструмент, который позволил бы ученику создавать, по крайней мере, простейшие ER-диаграммы. В связи с этим понадобилось выбрать инструмент для решения этой задачи. Этим инструментом стал продукт все той же компании Macromedia, называемый Flash. Его специфика заключается в возможности создавать интерактивные графические приложения за счет одновременного использования графики различной сложности, управляемой довольно мощным встроенным объектно-ориентированным языком написания скриптов ActionScript. Поскольку оба продукта (Flash и Authorware) разработаны одной компанией, довольно функциональным образом поддерживается их интеграция: ролики, созданные на Flash, можно внедрять в проекты Authorware и определенным образом осуществлять взаимодействие между ними.

Таким образом, Flash позволяет Authorware расширить свои возможности в области создания интерактивного оценивания навыков учащихся, а Authorware одновременно помогает снять ограничения Flash, поскольку в отличие от него может связываться с базами данных и сохранять там результаты, полученные при работе с роликом Flash.

Оба продукта снабжены довольно подробной системой помощи разработчику, которая позволяет освоить эти инструменты, даже не прибегая к дополнительной литературе. Мною было использовано всего два источника по данным технологиям ([30,31]).

Из других инструментов реализации был использован Microsoft Visio, в котором создавались ER-диаграммы как составная часть иллюстративного наполнения курса.

Базы данных, необходимые для учебного курса, создавались в СУБД Access.

3. Практическая реализация курса «Проектирование баз данных в EER-модели»

3.1 Общий подход к реализации электронного учебного курса «Проектирование баз данных»

Задачей, разрешение которой было поставлено в рамках дипломной работы, являлось создание электронного учебного курса, посвященного проектированию баз данных в терминах расширенной модели «Сущность – связь».

В качестве основных этапов разработки и проектирования электронного учебного курса «Проектирование баз данных в расширенной модели данных «Сущность-связь» можно выделить следующие:

1. Формулирование цели

2. Формулирование требований

3. Планирование и отбор материала для представления в курсе, включая тексты, рисунки, схемы, диаграммы и пр.

4. Создание материалов, отобранных в пункте 3.

5. Проектирование интерфейса пользователя, включая внешний вид курса, его навигационную структуру

6. Реализация интерфейса пользователя, то есть создание «каркаса» курса в том виде и с той навигацией, которая была спроектирована в пункте 5 (здесь же написание программного кода для такой реализации).

7. Наполнение «каркаса» отобранным и созданным материалом.

8. Проектирование подсистемы контроля знаний (включая создание вопросов теста и разработку функциональных и интерфейсных требований к простому и усложненному заданиям по рисованию ER- и EER-диаграмм).

9. Реализация подсистемы контроля знаний.

3.2 Формулирование цели и требований

Как можно было понять из предыдущих разделов, первое, что необходимо четко определить при создании учебного курса, это цель. При формулировании цели прежде всего необходимо ответить на два взаимосвязанных вопроса.

1. Для кого создается обучающая программа (то есть необходимо определить целевую аудиторию)?

2. Чему, в какой степени мы хотим научить целевую аудиторию, и как мы будем проверять достигнутый уровень?

Второй вопрос можно назвать проблемой выработки требований, где под требованием подразумевается «возможность, которую должна обеспечивать система; это некое свойство ПО, необходимое пользователю для решения проблемы при достижении поставленной цели» [25].

Однако именно от ответа на первый вопрос будет во многом зависеть процесс формулирования ответа на второй.

В связи с этим прежде всего необходимо определить, для кого предназначен электронный курс, описываемый в дипломной работе, и какую цель преследует его использование в рамках учебного процесса.

Предполагается, что целевой аудиторией разработанного курса являются студенты специальностей, имеющих отношение к информационным технологиям. Данный электронный курс используется при изучении предмета, посвященного базам данных. При этом курс не является полноценной заменой лекционным занятиям. Он выступает в качестве дополнительного источника информации по предмету, в котором обобщены наиболее важные сведения по концептуальному проектированию и реализованы средства для приобретения навыков разработки схем баз данных в виде EER-диаграмм. Одновременно он может использоваться преподавателем для проведения тестов в рамках практических занятий по курсу. Оценка за тесты будет являться составной частью оценки по предмету, поскольку подразумеваются и иные способы оценки учащихся, не поддерживающиеся электронным курсом.

При этом изначально не ставится задача изучения какой-либо конкретной СУБД, то есть не имеет места физический аспект проектирования как таковой. Целью курса является создание навыков обучающихся в области концептуального проектирования, навыков, которые позволят создавать схемы данных предметной области в виде, наиболее полно отвечающем требованиям пользователей к базе данных, и при этом максимально непротиворечивым образом.

Для всего курса будет использован полномасштабный пример, основанный на предметной области «Больница». На этом примере будут демонстрироваться описываемые понятия и определения. Таким образом, использование примера делает данный учебный курс не просто набором теоретической информации, подлежащей запоминанию и заучиванию. Он позволит на конкретной предметной области оценить возможности описываемой модели данных с тем, чтобы впоследствии применить полученные знания для решения практических задач в требуемых предметных областях.

3.3. Планирование, отбор и создание материалов для курса

В основу текстового теоретического наполнения легли курсы лекций, читаемых моим научным руководителем Бабановым А. М. на факультете информатики в рамках курса «Базы данных и СУБД». Материал этих лекций был слегка переработан и должным образом структурирован по параграфам для представления в учебном курсе.

Как уже говорилось, в качестве наглядного представления излагаемых концепций во всей главе используются примеры на единой демонстрационной предметной области. Она включает в себя практически все возможности расширенной модели данных «Сущность – связь» и была разработана в рамках курсовой работы, выполненной на третьем курсе. Примеры, приводимые в тексте, иллюстрируются на рисунках, многие из которых представляют собой ту или иную часть общей EER-диаграммы. Диаграммы и ряд других иллюстраций были созданы в пакете Microsoft Visio.

3.4 Проектирование и реализация пользовательского интерфейса

Каждый учащийся, проходящий учебный курс, может быть как зарегистрированным пользователем, так и незарегистрированным. Для регистрации преподаватель должен занести его в специальную базу данных учащихся, предоставить логин и пароль для входа в систему. Для зарегистрированных пользователей предусмотрено отслеживание процесса обучения и сохранение результатов тестирования. Незарегистрированные пользователи смогут пройти курс без сохранения истории и результатов своей учебной деятельности. При этом в качестве логина они должны ввести «Гость» для получения доступа к курсу.

Таким образом, первым элементом интерфейса пользователя является механизм аутентификации, после прохождения которой пользователь получает доступ к учебному материалу.

При наборе пользователем логина и пароля происходит обращение к базе данных учащихся, извлечение и сравнение соответствующей информации. Аутентификация и связь с базой данных реализована на встроенном языке Authorware. Схема базы данных учащихся приведена в приложении А.

Назовем область, в которой расположен сам курс и элементы пользовательского интерфейса, главным окном курса. Итак, главное окно курса поделено на две части: левую и правую. В зависимости от того, где в курсе в данный момент находится пользователь, они могут иметь разную смысловую нагрузку.

В самом начале курса левая часть отражает его содержание. При этом название каждого параграфа является гипертекстовой ссылкой на этот параграф.

При переходе на отдельный параграф правая часть служит для представления текстового материала параграфа, левая – иллюстраций. Рисунков в параграфе может быть несколько, тогда вверху имеются кнопки с порядковыми номерами, при нажатии на которые пользователь переходит на соответствующий рисунок. Ссылки на эти же рисунки присутствуют и в тексте параграфа. Такое дублирование произведено в целях удобства, если пользователь хочет только просмотреть иллюстрации, не обращаясь при этом к тексту параграфа. Одновременно сразу при чтении текста пользователь может без лишних движений перейти к соответствующему рисунку. Для создания переключаемых кнопок и ссылок на рисунки также использовался встроенный язык Authorware.

Когда пользователь находится внутри главы, он имеет возможность перемещаться по ней с помощью панели навигации, которая позволяет:

– перейти к следующей странице;

– перейти к предыдущей странице;

– перейти на первую страницу курса;

– перейти на последнюю страницу курса;

– последовательно переходить на последнюю просмотренную пользователем страницу;

– вывести окно со списком просмотренных пользователем страниц, чтобы дать ему возможность перейти на конкретную страницу из просмотренных.

На панели меню, которая находится вверху главного окна, определено меню "Перейти" с тремя возможными опциями: "Содержание", "Тестирование" и "Выход".

Таким образом, в процессе проектирования и создания пользовательского интерфейса было принято решение о строгом функциональном разделении главного окна курса. Текстовый материал пространственно отделен от иллюстративного, что позволяет пользователю, в зависимости от его потребностей, либо только читать текст, не отвлекаясь на

рисунки, либо только просмотреть рисунки, которые поясняют и раскрывают текстовый материал, либо делать и то, и другое параллельно.

Создана последовательная, логичная и довольно функциональная система навигации в пределах всего курса. Она реализована средствами Authorware, включая программный код на языке написания скриптов. Для работы с базой данных был использован xtra XTinyAdoDB, предоставляемый бесплатно сторонним разработчиком [32].

Используемые элементы пользовательского интерфейса (меню, кнопки, переключатели) интуитивно понятны и удобны в использовании. Иллюстрации, использованные для фона, а также ряд кнопок были разработаны специально для курса с использованием пакета Macromedia Fireworks.

На протяжении всего курса используется спокойная цветовая гамма, не утомляющая зрение обучающегося; текст хорошо различим и читаем.

3.5 Проектирование и реализация подсистемы контроля знаний

Для успешного обучения проектированию в ER-модели учащемуся недостаточно объяснить, какими структурными элементами обладает данная модель, какие возможности она предоставляет, и по каким правилам строятся ER-диаграммы. По возможности надо сформировать у обучающегося навыки самостоятельного создания ER-диаграмм. Однако знания терминологии, правил, концепций и элементов пусть и недостаточны, но необходимы. В связи с этим подсистема контроля знаний была разбита на следующие разделы:

1. Тестовая часть (в виде вопросов)

2. Упрощенное задание, заключающееся в создании ER-диаграммы на основе существующего набора множеств сущностей и связей.

3. Усложненное задание на самостоятельное составление схемы базы данных по предметной области, задаваемой преподавателем.

3.5.1 Тестирование

В процессе тестирования студенту будет предлагаться ряд вопросов. Текст вопросов, варианты ответов к ним и некоторая другая информация хранится в базе данных "Questions". Схема этой базы данных приведена в приложении Б. Количество вопросов, задаваемых в тесте, можно менять. При этом если в базе данных n вопросов, а в тесте предлагается m вопросов (в общем случае $m \leq n$), то каждый раз m вопросов выбираются случайным образом из набора в n вопросов. Это позволяет привнести некоторое разнообразие в предлагаемые тесты.

Вопросы помещаются в окно курса по одному, и переход осуществляется по кнопке «Далее». Возврат к предыдущим вопросам не предусмотрен.

Основными вариантами вопросов являются вопросы с единственным правильным ответом среди альтернатив (single-choice question) и с несколькими правильными вариантами ответов (multiple-choice question).

При ответе на такие вопросы оценивание происходит следующим образом: для каждого вопроса на основе информации из базы данных строится вектор, длина которого рав-

на количеству вариантов ответа на данный вопрос. Если вариант правильный – на соответствующем месте в этом векторе стоит единица, если неправильный – ноль. Создается второй вектор такой же длины, где единицами отмечаются те варианты, которые выбрал пользователь. Когда учащийся нажимает на кнопку «Далее» происходит сравнение векторов и подсчитывается количество «исправлений». Для вопросов с единственным вариантом ответов возможно только одно «исправление» – в случае, если пользователь выбрал неверный вариант ответа. В вопросах со множественным выбором «исправлением» считается и выбор неправильного варианта и невыбор правильного.

За каждый верно выбранный вариант начисляется один балл. Подсчитывается общее количество баллов, «заработанных» учащимся. Одновременно подсчитывается количество вопросов, на которые дан правильный ответ. Если в вопросе допущено хотя бы одно исправление, то считается, что ученик не ответил на вопрос.

После прохождения теста сначала выдается сокращенная статистика: количество вопросов, на которые дан правильный ответ, то же самое, но в процентном отношении, количество набранных баллов в процентном отношении к общему количеству баллов, которое можно было набрать за данный тест. Если хотя бы на один вопрос отвечено неверно, человек может просмотреть более подробную статистику, где будет указано, в каком вопросе сколько ошибок было сделано.

Для самих вопросов можно задавать уровень сложности. Также для каждого вопроса в базе данных хранится общее количество раз, которое он задавался и сколько раз на него был дан правильный ответ. Со временем эта информация может использоваться преподавателем, чтобы оценить, с какими вопросами у студентов возникает больше всего проблем. Кроме того, если на вопрос очень часто отвечают правильно, возможно, необходимо будет пересмотреть и изменить уровень сложности данного вопроса, либо усложнить сам вопрос, либо вовсе удалить, поскольку он мало что оценивает.

3.5.2 Упрощенное задание: создание ER-диаграммы

Для долговременного хранения описания предметной области в рамках расширенной модели данных «Сущность-связь» разработан «шаблон» базы данных в СУБД Access. Схема этого шаблона приведена в приложении В. На основе этого шаблона создаются базы данных, хранящие конкретные, разработанные в EER-нотации, схемы баз данных по различным предметным областям.

Само задание состоит в следующем: учащемуся предлагается на выбор ряд предметных областей, которые заранее заготовлены, выделены использующиеся в них множества сущностей и множества связей. (В данной работе существует только одна предметная область, однако легко могут вводиться другие предметные области, поскольку информация о них хранится в специальных базах данных). Одновременно с этим ему предлагается список, в котором перечислены названия всех сформированных множеств сущностей и связей. Задачей учащегося является выбор из этих названий тех, которые необходимы для данной предметной области, определение того, множеством связей или множеством сущностей будет данное понятие, и если множеством связей, то на каких множествах сущностей оно определено. По построенной пользователем таким образом ER-диаграмме создается база данных на основе разработанного шаблона, в которой структурирована информация о созданной схеме предметной области. Поскольку все предметные области подготовлены заранее, для них уже сформированы такие базы данных. Оценка работы учащегося производится в результате сравнения «с образцом» ER-диаграммы, созданной пользователем.

Кроме того, по окончании задания созданная диаграмма сохраняется в виде рисунка в папке результатов.

Для реализации описанной задачи был разработан инструмент для рисования простейших ER-диаграмм (то есть диаграмм, не включающих категоризации и специализации).

Итак, когда пользователь попадает в данный раздел учебного курса, он будет действовать следующим образом. Пользователь выбирает близкую ему из предложенных предметную область (в данный момент реализована только одна). В списке он выбирает название множества, переключателем устанавливает, хочет ли он создать множество сущностей или множество связей, и нажимает на кнопку «Создать». Соответствующее графическое представление в ER-диаграмме появляется в области рисования и имеет выбранную из списка пометку (то есть, это либо ромб, либо прямоугольник). Использованное название удаляется из списка. После создания множества связей учащемуся необходимо определить, на каких множествах сущностей его надо задать. Для этого он переходит в режим рисования дуг и соединяет прямоугольники, соответствующие требуемым множествам сущностей, с ромбом множества связей, после чего выходит из режима рисования дуг. Имеется возможность удаления множеств связей и множеств сущностей (с инцидентными им дугами) и возможность удаления отдельных дуг. Для этого в созданном инструменте существуют кнопки для перехода в соответствующие режимы. По окончании работы пользователь нажимает на кнопку «Диаграмма готова» и ожидает результата работы.

Сам инструмент для выполнения этого задания написан как ролик Flash. Затем он встроен в курс в среде Authorware. Событие нажатия кнопки «Диаграмма готова» отлавливается в Authorware, из ролика извлекается информация о созданных множествах сущностей и связей, как они связаны друг с другом, и записывается в базу данных. Затем происходит сравнение с базой данных – образцом, по результатам которого выдается оценка. Максимальная оценка устанавливается, если отчеты совпали. В противном случае, в зависимости от количества отличий, она снижается.

Описание программной части этого инструмента приводится в приложении Г.

3.5.2 Усложненное задание: самостоятельное создание EER-диаграммы

Это задание состоит в самостоятельной разработке EER-диаграммы для выбранной предметной области. По окончании работы пользователя, как и в упрощенном задании, созданные им множества связей и множества сущностей, а также ограничения целостности будут специальным образом преобразованы в структуры для хранения в базе данных. Одновременно в виде рисунка сохраняется нарисованная диаграмма, поскольку ее проще воспринимать, чем информацию, отраженную в базе данных. Преподаватель будет сам просматривать полученные результаты и по ним определять оценку учащегося, поскольку не существует формализованных методов для оценки качества схемы базы данных с точки зрения ее семантической значимости.

Инструмент для выполнения этого задания является модифицированной и усложненной версией инструмента, описанного в предыдущем пункте. По сравнению с ним, здесь появляется возможность создания специализаций (пересекающихся и непересекающихся, с частичным и полным участием) и категоризаций (с частичным и полным участием) с одновременной возможностью их удаления в ходе разработки диаграммы. Кроме того, здесь отсутствует список предопределенных названий и вместо него имеется поле для ввода текста – названия множества сущностей или множества связей.

Соответственно, усложнен процесс сбора информации о схеме в конце работы и запись ее в базу данных.

4. Демонстрационная предметная область: глоссарий

Данная предметная область разработана в качестве полноценного, максимально иллюстративного и непротиворечивого примера, на котором можно будет продемонстрировать большую часть лекционного материала, лежащего в основе электронного курса.

Учащиеся имеют возможность изучить этот пример, и если они ставят перед собой цель как можно лучше понять и запомнить теоретический материал и применить его на практике, ограничиться поверхностным просмотром будет нельзя. Необходима глубокая и детальная проработка примера. Прежде всего, надо определить предметную область в тех ее границах, в которых она будет использована.

EER-схема предметной области приводится в приложении Д.

Предметной областью демонстрационного примера является больница. В ней будут выделены следующие сущности:

4.1 Описание сущностей

1. Человек – интуитивно понятно, синонимы: личность, персона.

2. Работник – человек, работающий в больнице

3. Медицинский работник – работник, имеющий право оказывать медицинские услуги, то есть услуги, связанные со здоровьем пациентов

4. Обслуживающий работник – работник, выполняющий немедицинские работы (уборка помещений, обслуживание в гардеробе и пр.)

5. Вспомогательный работник – представитель медицинского персонала, не являющийся врачом (санитар, медсестра и пр.)

6. Врач – представитель медицинского персонала, имеющий право заниматься врачебной деятельностью, в частности проводить осмотры, ставить диагнозы, оперировать и пр.

7. Ординатор – человек, проходящий практику в больнице и готовящийся получить диплом врача; имеет право заниматься врачебной деятельностью в пределах, установленных администрацией больницы

8. Дипломированный врач – человек, имеющий диплом, который дает ему право заниматься врачебной деятельностью в соответствии со своей специальностью

9. Пациент – человек, пользующийся услугами, которые предоставляет больница

10. Амбулаторный пациент – приходящие пациенты и пациенты на дому

11. Стационарный пациент – пациент, лежащий в больнице

12. Диагноз – сущность и особенности болезни пациента, определенные в результате медицинского обследования

13. Анализ – установленное правилами больницы количество некоторой среды организма пациента (кровь и пр.), исследуемое в лаборатории для определения химического состава, позволяющего прояснить состояние здоровья пациента.

14. Лаборатория – учреждение, занимающееся исследованием анализов пациентов.

15. Отделение – часть больницы, где размещены стационарные пациенты в зависимости от специфики своего заболевания

16. Палата – комната, в которой лежат стационарные пациенты

17. Услуга – действие, которое может быть выполнено при обслуживании пациентов

18. Бесплатная услуга – услуга, за которую пациенты не платят

19. Платная услуга – услуга, за которую пациенты должны внести определенную больницей плату

20. Материал для услуг – материал, за который платят пациенты при оказании им платных услуг

21. Оборудование – использующиеся в больнице устройства

22. Медикамент – лекарственный препарат

Примечание 1: «Диагноз» и «Анализ» по сути дела представляют собой частные случаи услуг. При этом кажется целесообразным выделить их в отдельные сущности, поскольку они являются ключевыми элементами в сфере отношений «Врач-пациент», обладают специфическими атрибутами, а сущность «Анализ» также имеет свою связь с сущностью «Лаборатория», в то время как другие услуги такой связи не имеют.

Примечание 2: когда слово «Анализ» употребляется в контексте названия сущности, имеется в виду некоторый физический объект. Не следует путать слово «анализ», когда оно употребляется в прочих контекстах и может означать процесс исследования соответствующего физического объекта в лаборатории.

4.2 Специализации и их ограничения целостности

1. Человек: Работник, Пациент

Сущность из множества сущностей Персонал может одновременно являться пациентом, поэтому специализация пересекающаяся.

Не каждая сущность множества сущностей Человек должна быть отнесена либо к персоналу, либо к пациенту (потенциально возможно существование в больнице людей, выполняющих отличные от персонала и пациента роли), поэтому специализация с частичным участием.

2. Персонал: Медицинский работник, Обслуживающий работник

Каждая сущность суперкласса «Персонал» может входить либо в подкласс «Медицинский персонал», либо в подкласс «Обслуживающий персонал», поскольку человек, оказывающий медицинский услуги, не имеет права заниматься обслуживанием, и наоборот. Следовательно, данная специализация является непересекающейся.

Любая сущность множества сущностей «Персонал» должна быть отнесена либо к медицинскому, либо к обслуживающему персоналу, поэтому специализация с полным участием.

3. Медицинский работник: Вспомогательный работник, Врач

Врач не может быть одновременно вспомогательным персоналом, поэтому специализация непересекающаяся.

В больнице возможно существование медицинского персонала, не являющегося ни врачом, ни вспомогательным персоналом, поэтому специализация с частичным участием.

4. Врач: Ординатор, Дипломированный врач

Каждый врач данной больницы может либо только готовиться к получению диплома, то есть быть ординатором, либо уже иметь диплом, то есть быть дипломированным врачом. Таким образом, данная специализация является непересекающейся.

Специализация с частичным участием, так как не каждая сущность множества сущностей Врач должна быть представлена в одном из множеств сущностей Ординатор или Дипломированный врач.

5. Пациент: Амбулаторный пациент, Стационарный пациент

Любой пациент может присутствовать в базе данных и как амбулаторный, и как стационарный. В связи с этим специализация является пересекающейся.

Каждый пациент может быть зарегистрирован в базе данных либо как амбулаторный, либо как стационарный. Никаких других статусов для пациента не предусматривается. Это означает, что данная специализация с полным участием.

4.3 Категоризации и их ограничения целостности

1. Услуга: Платная услуга, Бесплатная услуга

Все экземпляры множеств сущностей «Платная услуга» и «Бесплатная услуга» должны быть представлены в категории «Услуга», поэтому категоризация с полным участием.

2. Материал для услуги: Оборудование, Медикамент

В больнице возможно существование оборудования и медикаментов, которые не используются при оказании услуг пациентам, следовательно, данная категоризация с частичным участием.

4.4 Множества сущностей и их атрибуты

Вынесены в приложение Е.

4.5 Множества связей и их атрибуты

1. Оказание услуг (Медицинский персонал, Услуга) M:N

Семантика: данная связь позволяет определить, какие вообще услуги могут оказываться каждым представителем медицинского персонала. Очевидно, что один и тот же представитель медперсонала вправе оказывать целый ряд услуг. При этом одна и та же услуга может предоставляться разными медицинскими сотрудниками. Выделение такой связи будет удобно, например, при составлении указателя для клиентов больницы, в соответствии с которым они смогут получить представление о диапазоне услуг, предоставляемых больницей, и узнать, к какому сотруднику обратиться за желаемой услугой.

2. Оказание услуг пациенту (Медицинский персонал, Пациент, Услуга) 1:1:M

Семантика: кажется целесообразным выделить данную связь, которая позволит установить, когда, каким сотрудником и какому пациенту была оказана определенная услуга. Для этого данная связь имеет атрибут «Дата оказания услуги». Таким образом, в отличие от множества связей 1. мы имеем более детальную информацию, а также возможность для получения различного рода статистических расчетов, позволяющих оценить загруженность медперсонала, частоту оказываемых клиенту услуг и другие параметры.

Атрибут связи: дата оказания услуги

3. Услуга-материал (Платные услуги, Материал для услуги) M:N

Семантика: категоризация «Материалы для услуги: медикаменты, оборудование» является категоризацией с частичным участием. В категориях «Медикаменты» и «Оборудование» одним из атрибутов является общая стоимость данного материала. Если же некоторый материал используется при оказании платной услуги, то в соответствующей связи указывается стоимость материала для данной услуги. Таким образом, можно будет подсчитать доходы больницы от оказания платных медицинский услуг.

Атрибут связи: стоимость материала

4. Направление на анализ (Врач, Пациент, Анализ) 1:1:M

Семантика: направление на анализ является частным случаем услуги, поэтому семантика очень близка к семантике связи «Оказание услуг пациенту». Данная связь позволяет установить, какой врач и какого пациента на какой анализ направлял.

Атрибут связи: дата направления

5. Постановка диагноза (Врач, Пациент, Диагноз) 1:1:M

Семантика: постановка диагноза является частным случаем услуги, поэтому семантика данной связи близка к семантике связи «Оказание услуг пациенту». Данная связь позволяет установить, какой врач и какому пациенту какой диагноз ставил.

Атрибут связи: дата постановки

6. Направленный анализ (Анализ, Лаборатория) M:1

Семантика: каждый анализ отправляется на обработку в одну из лабораторий. Таким образом, данная связь устанавливает отношение между анализом и лабораторией, в которой он исследуется.

7. Размещение (Стационарный пациент, Палата) M:1

Семантика: устанавливает отношение между стационарным пациентом и палатой, в которой он располагается. В каждой палате может одновременно лежать несколько пациентов, но пациент может находиться только в одной палате.

Атрибуты связи:

дата размещения в палате

предполагаемый срок лечения

8. Расположение (Палата, Отделение) M:1

Семантика: в каждом отделении может располагаться несколько палат, причем каждое отделение имеет свою нумерацию палат. При этом палата не может одновременно относиться сразу к нескольким отделениям.

9. Персонал-отделение (Медицинский персонал, Отделение) M:1

Семантика: все медицинские сотрудники больницы приписаны к отделению, причем только к одному. Соответственно каждое отделение может иметь много медицинских сотрудников.

10. Управление (Персонал, Персонал) 1:M

Семантика: для представителей персонала может быть определен его начальник.

11.Лечащий врач (Врач, Стационарный пациент) 1:M

Семантика: когда пациент помещается в больницу, ему назначается лечащий врач, который закрепляется на все время пребывание пациента в стационаре.

Заключение

Таким образом, результатом дипломной работы является создание электронного учебного курса, который призван, во-первых, познакомить студента с основными концепциями модели «Сущность – связь», включая ее расширенную нотацию, во-вторых, одновременно применить эти концепции для моделирования предметной области при проектировании баз данных, и, в-третьих, сформировать навыки построения ER и EER-диаграмм в рамках такого проектирования. Для выполнения последней задачи создан инструмент для создания и простейшего редактирования ER-диаграмм в простой и расширенной нотациях.

Созданный электронный учебный курс рекомендован к применению в рамках изучения предмета, посвященного базам данных, для студентов технических специальностей. Курс является дополнительным источником сведений по предмету, а также может использоваться для проведения тестирования в рамках практических занятий по предмету.

Список использованных источников

1. Ланина Э. П. Электронный учебный курс и его идеология [Электронный ресурс]. – Режим доступа: http://www.cctpu.edu.ru

2. Куфлей О. В. Разработка электронных учебных курсов [Электронный ресурс]. – Режим доступа к журн.: http://www.dlnet.unesco.kz

3. Тимкин С. Л. Структурирование материала электронного учебника на примере мультимедийного курса «История естествознания» [Электронный ресурс]. – Режим доступа к журн.: http://www.vvsu.ru

4. Околелов О. Электронный учебный курс // Высшее образование в России [Электронный ресурс]. – 1999. – №04. – Режим доступа к журн.: http://www.dvgu.ru

5. Андреев А. А., Лупанов К. Ю., Солдаткин В. И. Электронный учебные средства и оценка качества сетевого обучения [Электронный ресурс]. – Режим доступа к журн.: http://ict.edu.ru

6. Овчинникова К. Р., Соколинский Л. Б. Электронный учебный курс в системе открытого образования [Электронный ресурс]. – Режим доступа: http://tm.ifmo.ru

7. Тыщенко О. В. Новое средство компьютерного обучения – электронный учебник // Компьютеры в учебном процессе. – 1999. – №10. – С. 89-92

8. Зайцева Л. В., Прокфьева Н. О. Модели и методы адаптивного контроля знаний [Электронный ресурс] // Educational Technology & Society. – 2004. – №7(4). – Режим доступа к журн.: http://www.e-library.ru

9. Буль Е. Е. Обзор моделей студента для компьютерных систем обучения [Электронный ресурс] // Educational Technology & Society. – 2003. – №6(4). – Режим доступа к журн.: http://www.e-library.ru

10. Филатова Н. А., Тулова С. А., Ахремчик О. Л. Разработка и исследование программно-методического комплекса для построения ПФК модели обучаемого [Электронный ресурс] // Educational Technology & Society. – 2004. – №7(1). – Режим доступа к журн.: http://www.e-library.ru

11. Смолин Д. В. Методология создания компьютерного теста (найти данные)

12. Зиндер Е.З. Проектирование баз данных: новые требования, новые подходы// Системы управления базами данных [Электронный ресурс]. – 1996. – №3. – Режим доступа: http://rema.44.ru

13. Кириллов В. В. Основы проектирования реляционных баз данных [Электронный ресурс]. – Режим доступа: http://khpi-iip.mipk.kharkiv.edu

14. Коннолли Т., Бегг К., Страчан А. Базы данных: проектирование, реализация и сопровождение. Теория и практика. – М.: Вильямс, – 2000. – 1120 с.

15. Чен П. Модель «сущность – связь» – шаг к единому представлению данных // СУБД [Электронный ресурс]. – 1995. – №03. – Режим доступа к журн.: http://www.osp.ru

16. Надточий И. Л., Кафтанников И. Л. Методология и средства повышения интеллектуализации IT-учебного процесса [Электронный ресурс] // Educational Technology & Society. – 2003. – №6(3). – Режим доступа к журн.: http://www.e-library.ru

17. Григорьев В. К., Антонов А. А., Закаблуков Д. А. Исследование обобщенного интерфейса тьюторной обучающей системы [Электронный ресурс] // Educational Technology

& Society. – 2004. – №7(3). – Режим доступа к журн.: http://www.e-library.ru

18. Соловьев А. В. Дидактический анализ проблематики электронного обучения (найти данные)

19. Steiner M. Critical strategies for a successful e-learning project [Электронный ресурс]. – Режим доступа: http://www.macromedia.com

20. Bardzell J. Introducing the technologies of web learning [Электронный ресурс]. – Режим доступа: http://www.macromedia.com

21. Bardzell J. Seven key features of e-learning environments [Электронный ресурс]. – Режим доступа: http://www.macromedia.com

22. Bruce B., Fallon C., Horton W. Getting started with e-learning [Электронный ресурс]. – Режим доступа: http://www.macromedia.com

23. Foley A., Regan B. Best practices for web accessibility design and implementation [Электронный ресурс]. – Режим доступа: http://www.macromedia.com

24. Brogan P. Using the web for interaction teaching and learning [Электронный ресурс]. – Режим доступа: http://www.macromedia.com

25. Мишнев Б., Герасимова Л. Управление требованиями для разработки и эксплуатации обучающей системы TSI [Электронный ресурс] // Educational Technology & Society. – 2004. – №7(4). – Режим доступа к журн.: http://www.e-library.ru

26. Лавров О. А. Что есть знание? Является ли обучение передачей знания? Что такое е-знання? Является ли е-обучение передачей е-знания? [Электронный ресурс] // Educational Technology & Society. – 2003. – №6(3). – Режим доступа к журн.: http://www.e-library.ru

27. Лавров О. А. Обучение как передача знаний. Тихно-философское задание к размышлению [Электронный ресурс] // Educational Technology & Society. – 2003. – №6(3). – Режим доступа к журн.: http://www.e-library.ru

28. http://www.learnware.ru

29. Лавров О. А., Агапонов С. В. Выбор программной платформы для дистанционного обучения как проблема [Электронный ресурс] // Educational Technology & Society. – 2004. – №7(1). – Режим доступа к журн.: http://www.e-library.ru

30. Гультяев А. К. Macromedia Authorware 6.0. Разработка мультимедийных учебных курсов. – Спб.: Учитель и ученик: КОРОНА принт, 2002. – 400 с.

31. Китинг, Дж. Flash MX. Искусство создания web-сайтов. – Спб.: ООО ДиаСофтЮП, 2003. – 848 с.

32. http://www.xtra-ucd.com

Приложение А.
Схема базы данных "Students"

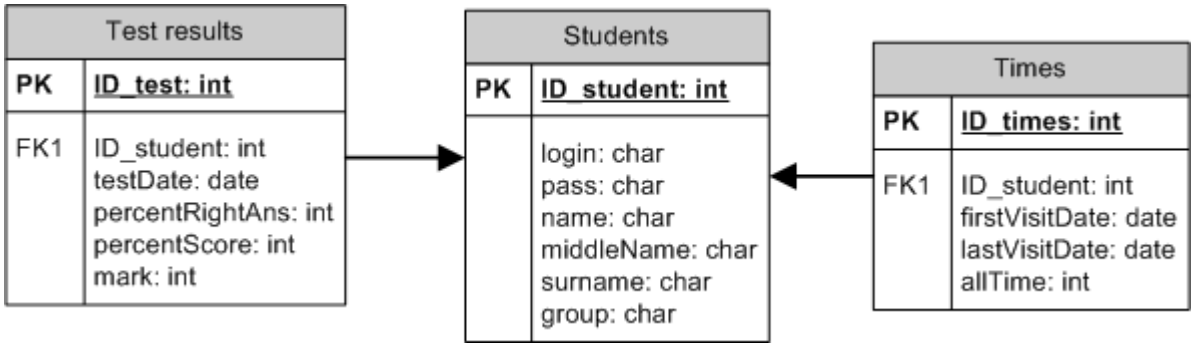


Таблица Students, атрибуты:
ID_student – первичный ключ
login – логин студента
pass – пароль студента
name – имя студента
middleName – отчество студента
surname – фамилия студента
group – номер группы студента

Таблица Test results, атрибуты:
ID_test – первичный ключ
ID_student – внешний ключ
testDate – дата проведения теста
percentRightAns – количество правильных ответов в процентах
percentScore – количество набранных баллов в процентах
mark – оценка за тест

Таблица Times, атрибуты:
ID_times – первичный ключ
ID_student – внешний ключ
firstVisitDate – дата первого посещения курса
lastVisitDate – дата последнего посещения курса
allTime – общее время работы с курсом (в количестве минут)

Приложение Б.

Схема базы данных "Questions"

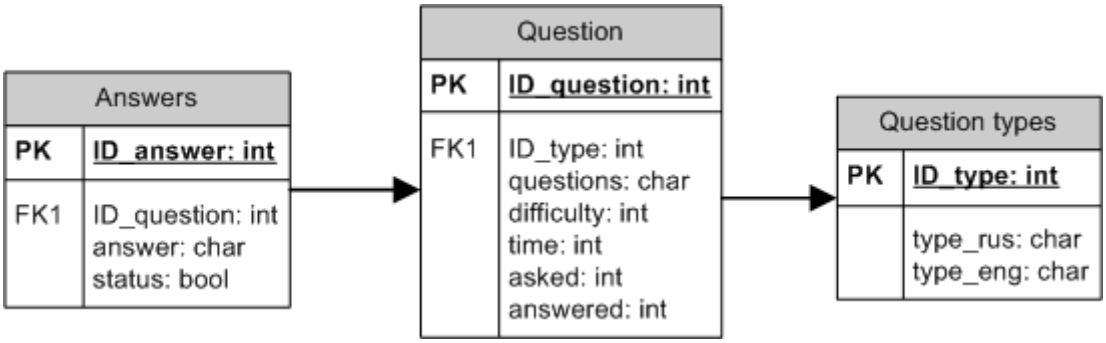


Таблица Question, атрибуты:

ID_question – первичный ключ

ID_type – внешний ключ

question – текст вопроса

difficulty – сложность вопроса (целое число в диапазоне от 1 до 5. Чем больше, тем выше сложность)

time – количество секунд, требуемых для ответа на вопрос (для задания временных ограничений при проведении тестов)

asked – количество раз, которое задавался вопрос

answered – сколько раз на вопрос был дан верный ответ

Таблица Question types определяет типы вопросов, атрибуты:

ID_type – первичный ключ

type_rus – название типа вопроса на русском языке

type_eng – английский эквивалент

Таблица Answers содержит варианты ответов на вопросы, атрибуты:

ID_answer – первичный ключ

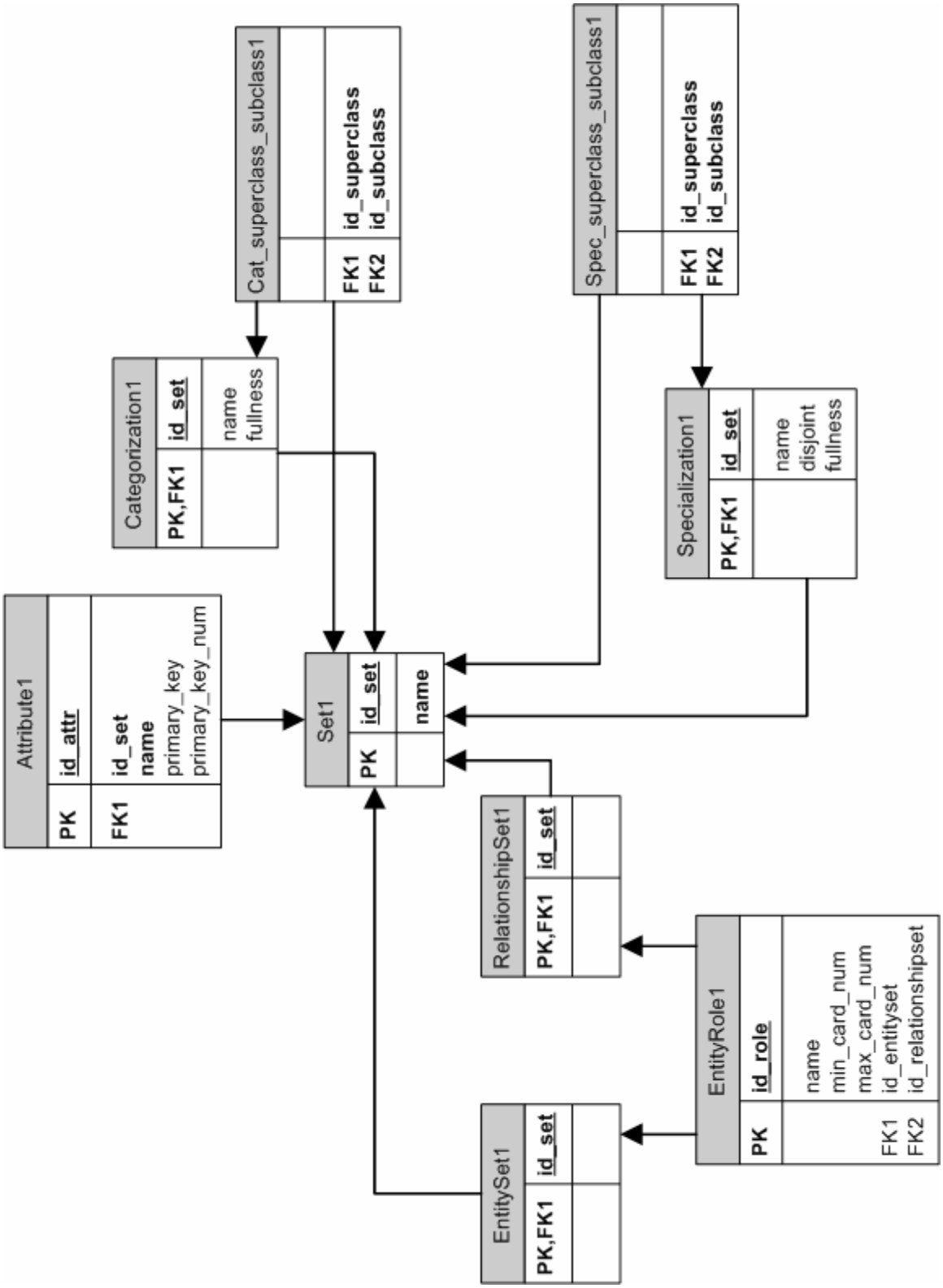
ID_question – внешний ключ

answer – текст варианта ответа

status – логическое значение, true – данный вариант правильный, false – неверный.

Приложение В.

Схема базы данных "EER-schema"



Приложение Г.

Описание программы

В данном приложении описывается программная реализация инструмента для создания простейших ER-диаграмм для упрощенного задания и усложненная версия этого инструмента для усложненного задания. Этот инструмент реализован как фильм Flash, созданный соответственно в среде Flash MX. В этой среде код на языке ActionScript может прикрепляться к кадрам, клипам и кнопкам. Все клипы и кнопки хранятся в библиотеке проекта Flash с определенными разработчиком именами, а на сцену помещаются только их экземпляры, которые также могут иметь название, используемое как имя переменной экземпляра.

Ромбы и прямоугольники как графические представления множеств связей и множеств сущностей в ER-диаграмме реализованы в виде клипов. В дальнейшем при описании переменных и функций вместо выражения "экземпляр клипа, соответствующий множеству сущностей или множеству связей" будет говориться просто "клип", если из контекста понятно, о чем идет речь.

Версия инструмента для простого задания позволяет создавать множества сущностей и множества связей и дуги между ними, но не поддерживает специализаций и категоризаций.

Сам фильм это версии инструмента находится в файле ERD_011.fla. В этом фильме 3 слоя:

1) workField – здесь расположен одноименный клип workField. Это область, на которую помещаются элементы ER-диаграммы.

2) UI components – здесь расположены клипы и кнопки, использующиеся как элементы пользовательского интерфейса:

– список (list box), в котором находятся предлагаемые на выбор учащемуся названия множеств сущностей и множеств связей; имя переменной списка – lb1

– группа из двух переключателей (radio-buttons), с помощью которых пользователь определяет, хочет ли он создать множество сущностей или множество связей. Имена переменных для этих кнопок: rb1 и rb2. Обе кнопки имеют одинаковое имя группы (Group Name) – radioGroup1. Свойство label для rb1 – "Множество сущностей", для rb2 – "Множество связей". По умолчанию выбранной кнопкой является кнопка rb1.

– одноименный экземпляр кнопки beginButton. Пользователь нажимает на эту кнопку с надписью "Начать задание", чтобы приступить к выполнению задания. При этом в список загружаются предлагаемые названия множеств сущностей и связей, причем они передаются как значение переменной из Authorware.

– одноименный экземпляр кнопки createSetButton. Это кнопка с надписью "Создать", при нажатии на которую в области рисования появляется графическое изображение множества сущностей или множества связей, в зависимости от состояния переключателей.

– экземпляр клипа arcButton с названием drawArcs. По логике это кнопка, имеющая два состояния: нажата или отжата. При нажатии на кнопку пользователь переходит в режим рисования дуг: в углах всех прямоугольников и ромбов на ER-диаграмме появляются ограничивающие квадратики. Когда пользователь нажмет на один квадратик некоторого множества, а затем на квадратике другого множества, между ними будет проведена прямая линия, которая будет трактоваться как дуга между множествами. При отжатии этой кнопки пользователь возвращается в режим создания множеств связей и сущностей.

– одноименный экземпляр кнопки doneButton. При нажатии на эту кнопку происходит формирование строковой переменной, содержащей результат работы, и вызывается функция, событие вызова которой отлавливается в Authorware, трактуется как завершения задания и подвергается дальнейшей обработке.

3) Actions – этот слой разработчики Flash рекомендуют создавать, чтобы в нем определять все специфические действия для фильма или клипа на языке ActionScript. У меня здесь также определены глобальные переменные и глобальные функции, используемые в различных клипах для организации взаимодействия. Весь программный код этого слоя привязан к первому кадру фильма.

Функции и переменные, определенные в слое Actions:

_root.rectNumToName – массив, где на i-том месте лежит название клипа, соответствующего i-тому прямоугольнику на сцене

_root.rombNumToName – массив, где на i-том месте лежит название клипа, соответствующего i-тому ромбу на сцене

_root.counter – общее количество когда-либо созданных множеств, включая удаленные. Это своеобразный счетчик, чтобы формировать уникальные названия клипов множеств, присоединяемых к сцене, а также номер глубины Z, на котором помещать новый клип, чтобы они не стирали один другой

_root.numSet – общее количество множеств

_root.numRect – количество множеств сущностей

_root.numRomb – количество множеств связей

_root.clickCount – при режиме рисования дуг в этой переменной хранится, сколько раз было нажато уголков. Возможные значения: 0 – выбрано первое множество, 1 – выбрано второе множество, надо рисовать дугу

_root.arcDirection – массив; здесь будут храниться направления (соответствующие ссылкам) для одной дуги

_root.clips – массив; здесь хранятся ссылки на два клипа, между которыми надо провести дугу

_global.X,_global.Y – эти координаты надо будет отнимать от соответствующих координат при рисовании линий на клипе workField, т.к. координаты клипов будут высчи-

тываться относительно всего окна, а не клипа workField

_root.modeNames – массив, в котором хранятся строковые константы, описывающие названия возможных режимов в программе.

_global.Point=function (x,y) – глобальная функция-конструктор объектов типа Point (точка)

Параметры:

x – абсцисса точки

y – ордината точки

_global.Line=function (x1,y1,x2,y2) – глобальная функция-конструктор объектов типа Line (отрезок)

Параметры:

x1 – абсцисса начала отрезка

y1 – ордината начала отрезка

x2 – абсцисса конца отрезка

y2 – ордината конца отрезка

_global.pushSet=function(goalSet,dirIndex,pushSet) – глобальная функция; добавляет в клип по ссылке, соответствующей направлению, ссылку на другой клип. Проверяется, чтобы между данными клипами (соответствующими либо множеству сущностей, либо множеству связей) не было двух разных дуг.

Параметры:

goalSet – клип, в который добавляем ссылку

dirIndex – целое число от 0 до 3, соответствующее направлению, в котором надо добавить ссылку: 0 – левая , 1 – верхняя, 2 – правая, 3 – нижняя

Возвращаемое значение:

true – если клип добавлен по соответствующей ссылке;

false – в противном случае (если между этими клипами уже есть одна дуга)

_global.drawArcsForSet=function(set) – глобальная функция; рисует все дуги для заданного множества сущностей или связей

Параметры:

set – название клипа, для которого надо нарисовать дуги

_global.clearArcsForSet=function(set) – глобальная функция; стирает все дуги для заданного множества сущностей или связей

Параметры:

set – название клипа, для которого надо стереть все дуги

_global.coorOnDirection=function(clip,dir) – глобальная функция; по клипу и направлению ссылки возвращает координаты точки, из которой или в которую надо проводить дугу

Параметры:

clip – клип (множество сущностей или связей), координаты точки дуги которого надо получить

dir – целое число от 0 до 3, соответствующее направлению дуги: 0 – левая дуга, 1 – верхняя, 2 – правая, 3 – нижняя

Возвращаемое значение:

массив из двух целых чисел, соответствующих координатам x и y точки

_global.setNewConnectionPoint=function(goalSet,changedSet,newx,newy) – глобальная функция; меняет координаты точки одного конца дуги между двумя множествами. Вызывается, когда некоторое множество поменяло свое положение на области рисования и следовательно изменились координаты одного конца дуги.

Параметры:

goalSet – клип, из которого идет дуга (тот который остался на месте, но у него есть ссылка на координаты точки прикрепления дуги ко второму клипу, которые надо обновить)

changedSet – клип, который изменил свое месторасположение

newx – координата x нового конца дуги

newy – координата y нового конца дуги

_global.drawAllArcs=function() – глобальная функция; рисует дуги для всех множеств сущностей и связей в области рисования

_global.makeReport=function() – глобальная функция; создает содержимое файла отчета

Возвращаемое значение:

строка следующего формата:

"*Множества сущностей:*

множество сущностей 1

……

множество сущностей N

Множества связей:

множество связей 1 (мн_сущ$_{11}$, …, мн_сущ$_{1m1}$)

……..

множество связей K (мн_сущ$_{k1}$, …, мн_сущ$_{kmk}$)"

_global.isArcBetweenClips=function(clip1, dir1, clip2, dir2) – глобальная функция, проверяет, если ли между двумя клипами дуга в конкретных направлениях

Параметры:

clip1 – первый проверяемый клип

dir1 – направление, из которого исходит дуга для первого клипа

clip2 – второй проверяемый клип

dir2 – направление, из которого исходит дуга для второго клипа

Возвращаемое значение:

Массив из трех элементов. Первый элемент – логический (истина, если дуга между клипами найдена, если нет – ложь). Если дуга найдена, то второй элемент – индекс в списке по направлению dir1 для клипа clip1, где он ссылается на второй клип, третий - наоборот, индекс в списке по направлению dir2 в клипе clip2, где он ссылается на первый клип. Если дуга не найдена, второй и третий элементы равны -1

_global.clipOnSquareNumber=function(num) – глобальная функция, которая по порядковому номеру данного клипа-квадратика определяет, на каком клипе-множестве он находится, и возвращает ссылку на этот клип-множество

Параметры:

num – порядковый номер данного клипа (он извлекается из названия этого клипа)

_global.findNumOnName=function(name) – глобальная функция, по названию клипа определяет его порядковый номер на сцене

Параметры:

name – название клипа

Возвращаемое значение:

целое число – порядковый номер клипа на сцене

_global.deleteSet=function(set) – глобальная функция, удаляет множество со сцены и ссылки на это множество из других множеств, а также перенумеровывает оставшиеся множества (меняет их порядковый номер на сцене)

Параметры:

set – удаляемый клип множества

_global.makeButtonsInvisible=function(ar) – глобальная функция, вызывается при переходе в некоторый режим. Делает некоторые кнопки невидимыми.

Параметры:

ar – массив, в котором единицы стоят на местах, соответствующих кнопкам, которые надо сделать невидимыми, а 0 соответствуют кнопкам, которые оставить видимыми

_global.makeButtonsVisible=function(ar) – глобальная функция, вызывается при переходе в некоторый режим. Делает некоторые кнопки видимыми.

Параметры:

ar – массив, в котором единицы стоят на местах, соответствующих кнопкам, которые надо сделать видимыми, а 0 соответствуют кнопкам, которые надо оставить невидимыми.

Клип Rect

Графически этот клип представляет собой прямоугольник с текстовым полем, значение которого хранится в свойстве клипа label. В нем также определен слой Actions, в котором приписано "поведение" экземпляров данного клипа. Здесь определены следующие переменные и функции:

this.curPoints – массив точек, из которых выходят дуги для данного множества

this.sets – двумерный массив; фактически состоит из четырех списков, соответствующих каждому направлению, в каждом из которых хранятся ссылки на соседние по дугам множества в данном направлении

this.setPoints – двумерный массив; фактически состоит из четырех списков, соответствующих каждому направлению, в каждом из которых хранятся точки с координатами. Эти точки соответствуют концам дуг, которые входят в соседние по дугам множества в данном направлении.

Список 0 соответствует направлению "левое"; 1 – "верхнее"; 2 – "правое"; 3 – "нижнее".

Таким образом, this.sets[1][2] означает ссылку на клип множества, которое соединено дугой с данным, исходящей из верхней стороны, причем данный клип является третьим (нумерация идет с нуля) "соседом", который соединяется с данным через "верхнюю дугу". При этом this.setPoints[1][2] соответствует точке с координатами того конца дуги, которая заходит в "соседа" this.sets[1][2].

onPress=function() – функция, которая вызывается при нажатии на клип мышью(клип начинает перетаскиваться, все его дуги стираются)

onRelease=function() – функция, которая вызывается, когда отпускается нажатая на клипе кнопка мыши (при этом перетаскивание клипа заканчивается, находятся все "соседи" по дугам, у них обновляются координаты концов дуг, обновленные дуги рисуются заново)

Клип romb

Графически этот клип представляет собой ромб с текстовым полем, значение которого хранится в свойстве клипа label. В нем также определен слой Actions, в котором приписано "поведение" экземпляров данного клипа. Здесь определены те же переменные и функции, что и в клипе Rect.

Клип arcButton

Графически этот клип представляет собой кнопку с надписью "Нарисовать дуги". В нем также создан слой Actions, в котором определены следующие переменные и функции:

this.clicked – это логическая переменная-свойство, может принимать два возможных значения: true – если кнопка нажата, false – если кнопка не нажата

this.initialDepth – целочисленная переменная-свойство, в которой хранится первоначальная глубина клипа. Она необходима, когда приходится менять глубину клипа и возвращаться к исходной.

onPress=function() – функция, которая вызывается, когда пользователь нажимает мышкой на кнопке. Если кнопка не была нажата, следовательно надо ее "нажать" и перейти в режим рисования дуг. При этом для всех ромбов и прямоугольников на сцене с четырех сторон добавляются ограничивающие указатели, из которых можно "чертить" дуги. Если кнопка была нажата, ее надо вернуть в исходное состояние и выйти из режима рисования дуг, при этом все ограничивающие указатели убираются.

Клип deleteArcs

Графически этот клип представляет собой кнопку с надписью "Удалить дуги". В нем определен слой Actions, в котором присутствуют следующие переменные и функции:

this.clicked – это логическая переменная-свойство, может принимать два возможных значения: true – если кнопка нажата, false – если кнопка не нажата

this.initialDepth – целочисленная переменная-свойство, в которой хранится первоначальная глубина клипа. Она необходима, когда приходится менять глубину клипа и возвращаться к исходной.

onPress=function() – функция, которая вызывается, когда пользователь нажимает мышкой на кнопке. Если кнопка не была нажата, следовательно надо ее "нажать" и перейти в режим удаления дуг. При этом для всех ромбов и прямоугольников на сцене с четырех сторон добавляются ограничивающие указатели, с помощью которых можно определить, какую дугу надо удалить. Если кнопка была нажата, ее надо вернуть в исходное состояние и выйти из режима удаления дуг, при этом все ограничивающие указатели убираются.

Клип squareButton

Графически этот клип представляет собой маленький квадрат, который помещается в качестве граничного указателя на стороны клипов для множеств в режиме рисования дуг. В нем определен слой Actions, в котором содержатся следующие переменные и функции:

this.clicked – это логическая переменная-свойство, может принимать два возможных значения: true – если кнопка нажата, false – если кнопка не нажата

onRollOver = function () – функция, которая вызывается, когда указатель мыши находится над данным клипом. При этом клип переходит на кадр, где вместо белого цвета квадратик рисуется красным.

onRollOut = function () – функция, которая вызывается, когда указатель мыши выходит за пределы данного клипа. При этом клип снова возвращается на первый кадр, где квадрат имеет белый цвет.

onRelease = function () – функция, которая вызывается, когда пользователь отпускает кнопку мыши, нажатую на данном клипе. При этом если выбрано первое множество для начала дуги, оно запоминается в глобальных переменных, если второе, запоминается оно, и делается взаимная ссылка по соответствующим направлениям этих клипов друг на друга.

Клип workField

Графически этот клип представляет собой прямоугольное поле, в котором будет создаваться ER-диаграмма. Здесь определена единственная переменная:

this.initialDepth – это целочисленная переменная-свойство, в которой сохраняется первоначальная глубина данного клипа.

Клип deleteSet

Графически этот клип представляет собой кнопку с надписью "Удалить множества". В слое Actions этого клипа определены следующие функции и переменные:

this.clicked – это логическая переменная-свойство, может принимать два возможных значения: true – если кнопка нажата, false – если кнопка не нажата

onPress=function() – функция, которая вызывается, когда пользователь нажимает на кнопку. Если кнопка не была нажата, тогда мы переходим в режим удаления множеств, при этом меняется строка названия режима в окне программы и ряд кнопок делается невидимыми. Если кнопка была нажата до этого, то мы выходим из режима удаления множеств, меняется строка названия режима в окне программы и кнопки, которые были неви-

димыми, снова становятся видимыми.

Кнопка beginButton

Для экземпляра этой кнопки на сцене определена следующая функция:

on(press) – это обработчик события нажатия на кнопку. В нем содержимое переменной, передаваемое из Authorware и состоящее из названий множеств сущностей и множеств связей, разделенных знаком равенства, преобразовывается в массив строковых переменных, значениями которых заполняется список lb1. После этого кнопка делается невидимой.

Кнопка createSetButton

Для экземпляра этой кнопки на сцене определена следующая функция:

on(press) – это обработчик события нажатия на кнопку. В нем в зависимости от состояния кнопок переключателей к текущему фильму присоединяется либо клип-прямоугольник, либо клип-ромб, значению свойства label которого присваивается выбранное в списке название. Затем это название удаляется из списка.

Кнопка doneButton

Для экземпляра этой кнопки на сцене определена следующая функция:

on(release) – это обработчик события отпускания кнопки. В нем вызывается глобальная функция makeReport(), результат которой передается в свою очередь во встроенную функцию getURL(). Событие вызова этой функции будет отловлено в Authorware и будет означать конец задания.

На основании файла ERD_011.fla создается непосредственно файл ролика (или фильма) – ERD_011.swf, который и будет проигрываться в Authorware.

Инструмент для усложненного задания хранится в файле EERD_012.fla. Он является модифицированной версией только что описанного инструмента, поэтому для него необходимо указать только те функции и переменные, которые были добавлены.

_root.circleNumToName – массив, где на i-том месте лежит название клипа, соответствующего i-тому ромбу на сцене

_root.numCircle – количество кружков специализаций и категоризаций на сцене

_root.inputText – строковая переменная, хранит введенное в поле для ввода название множества сущностей или множества связей

_global.requestOnDouble=function() – глобальная функция, вызывается, когда проводят дугу между кружком специализации, у которой еще не определен суперкласс, и множеством связей, либо между кружком категоризации, у которой еще не определен под-

класс, и множеством связей. Она выводит модальное окно диалога для запроса, является ли множество сущностей, к которому идет дуга, суперклассом или подклассом.

_global.makeReport=function() – глобальная функция, куда помимо множеств сущностей и связей, как в простом инструменте, добавляется формирование отчета по категоризациям и специализациям

Возвращаемое значение:

строка, начальный вид которой смотри в описании этой функции к простому инструменту, а продолжение имеет следующий формат:

"Специализации:

Множество_сущностей_суперкласс (множество_сущностей_подкласс$_{11}$, …,множество_сущностей_подкласс$_{1k1}$) [непересекающаяся | пересекающаяся, с полным участием | с частичным участием]

……

Множество_сущностей_суперкласс (множество_сущностей_подкласс$_{n1}$, …,множество_сущностей_подкласс$_{nkn}$) [непересекающаяся | пересекающаяся, с полным участием | с частичным участием]

Категоризации:

Множество_сущностей_подкласс (множество_сущностей_суперкласс$_{11}$, …,множество_сущностей_суперкласс$_{1l1}$) [с полным участием | с частичным участием]

……..

Множество_сущностей_подкласс (множество_сущностей_суперкласс$_{m1}$, …,множество_сущностей_суперкласс$_{mlm}$) [с полным участием | с частичным участием]
"

_global.drawDoubleArc1=function(clip1,dir1,clip2,dir2) – глобальная функция, рисует двойную дугу между двумя клипами из выбранных направлений

Параметры:

clip1 – первый клип

dir1 – направление, из которого исходит дуга в первом клипе

cli2 – второй клип

dir2 – направление из которого исходит дуга во втором клипе

_global.drawDoubleArc2=function(p1,p2,dir1,dir2) – глобальная функция, делает то же самое, что и предыдущая функция, но имеет другие параметры

Параметры:

p1 – первая точка, из которой рисуется дуга

p2 – вторая точка, из которой рисуется дуга

dir1 – направление для первой точки

dir2 – направление для второй точки

_global.clearDoubleArc1=function(clip1,dir1,clip2,dir2) – глобальная функция, стирает двойную дугу между двумя клипами в выбранных направлениях

Параметры:

clip1 – первый клип

dir1 – направление, из которого исходит дуга в первом клипе

cli2 – второй клип

dir2 – направление из которого исходит дуга во втором клипе

_global.clearDoubleArc2=function(p1,p2,dir1,dir2) – глобальная функция, делает то же самое, что и предыдущая, но имеет несколько другие параметры:

Параметры:

p1 – первая точка, из которой стирается дуга

p2 – вторая точка, из которой стирается дуга

dir1 – направление для первой точки

dir2 – направление для второй точки

_global.deleteSet=function(set) – глобальная функция, аналогичная той, которая описана в простом инструменте, только добавлено удаление кружков специализаций

Клип Specialization

Графически этот клип представляет собой кружок с текстовым полем, значение которого хранится в свойстве клипа label. В нем также определен слой Actions, в котором приписано "поведение" экземпляров данного клипа. Здесь определены те же переменные и функции, что и в клипе Rect.

Клип Categorization

Графически этот клип представляет собой кружок с текстовым полем, значение которого хранится в свойстве клипа label. В нем также определен слой Actions, в котором приписано "поведение" экземпляров данного клипа. Здесь определены те же переменные и функции, что и в клипе Rect.

Клип clipModalWindow

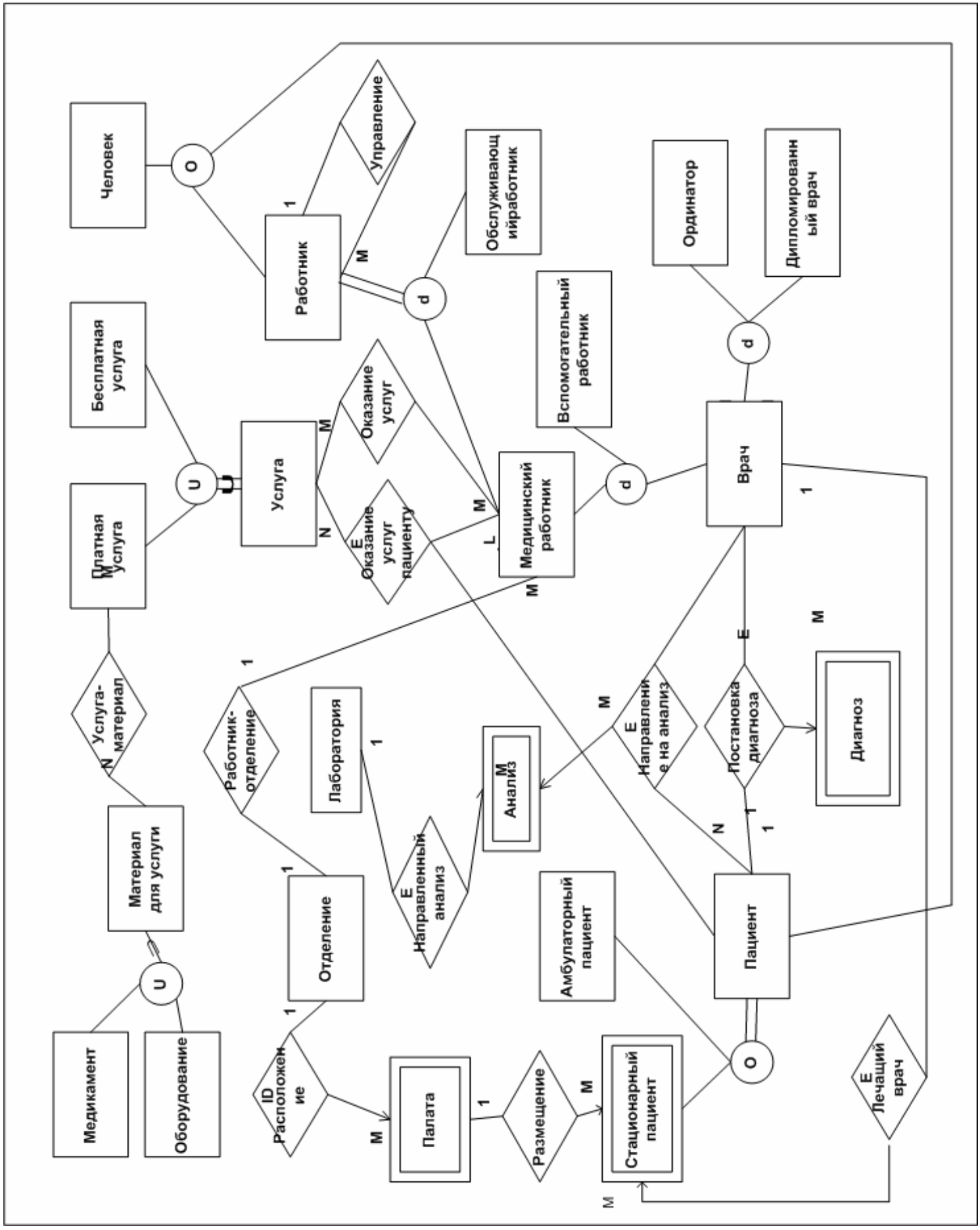
Графически этот клип представляет собой окно с вопросом, текст которого определяется значением свойства клипа label, и двумя кнопками: "Да" и "Нет".

Кнопка "Да" имеет функцию on(release), которая вызывается при нажатии на этой кнопке. Здесь отмечается факт того, что некоторое множество связей получило статус суперкласса или подкласса и в зависимости от обстоятельств рисуется двойная или одинарная дуга между двумя множествами на сцене.

Кнопка "Нет" имеет функцию on(release), которая вызывается при нажатии на этой кнопке и рисует дугу между двумя множествами сущностей на сцене.

Приложение Д.

EER-схема демонстрационной предметной области



Приложение Е.

Множества сущностей и их атрибуты

1. Человек

фамилия

имя

отчество

адрес

дата рождения

пол

возраст (виртуальный атрибут)

номер страхового полиса (прим. Человеку, не имеющему страхового полиса, не могут предоставляться бесплатные медицинские услуги)

2. Работник

должность

смена

зарплата

3. Медицинский работник

4. Обслуживающий работник

5. Вспомогательный работник

6. Врач

специальность

7. Ординатор

учебное заведение

предполагаемая дата окончания ординатуры (прим. Поскольку после окончания ординатуры человек становится врачом, реальная дата окончания ординатуры будет указана во множестве сущностей «Врач» как дата получения диплома врача

дополнительная информация

8. Дипломированный врач

стаж

дата получения диплома врача

9. Пациент

номер медицинской карты (первичный ключ)

10. Амбулаторный пациент

11. Стационарный пациент

номер койки

фактическая дата выписки

Дополнительную информацию о сущности типа «Стационарный пациент» можно будет узнать из атрибутов соответствующей связи типа «Размещение», а именно дату размещения в палате и предполагаемый срок лечения.

12. Диагноз

тип диагноза

осложнения

предупреждающая информация

13. Анализ

назначенная дата

назначенное время

номер направления

состояние

результат анализа

14. Лаборатория

название

адрес

телефон

15. Отделение

название

телефон

16. Палата

номер палаты

название

число коек

17. Услуга

18. Бесплатная услуга

название

19. Платная услуга

название

стоимость консультации

20. Материал для услуги

21. Оборудование

название

назначение

общая стоимость

22. Медикамент

название

описание

способ приема

дозировка

общая стоимость

В отношении «Анализ» атрибут «дата_направления» означает дату, когда врач направил пациента на анализ. Назначенное время и дата, как правило, определяются лабораторией как время, когда анализ реально будет готов.

Приложение Ж.

Описание применения

Разработанный электронный учебный курс предназначен для выполнения в операционных средах Windows 98SE/Windows XP, при разрешении экрана 1024*768.

Для установки программы необходимо скопировать папку программы в требуемое место.

Для запуска программы необходимо запустить файл с расширением .exe.

При запросе логина, если пользователь его не знает, он может войти под логином "Гость". При запросе пароля, зарегистрированный пользователь должен ввести пароль. В левой стороне окна будет выведено содержание всего курса. В процессе прохода по страницам главы в правой части окна будет представлен текст, а в левой – иллюстрации.

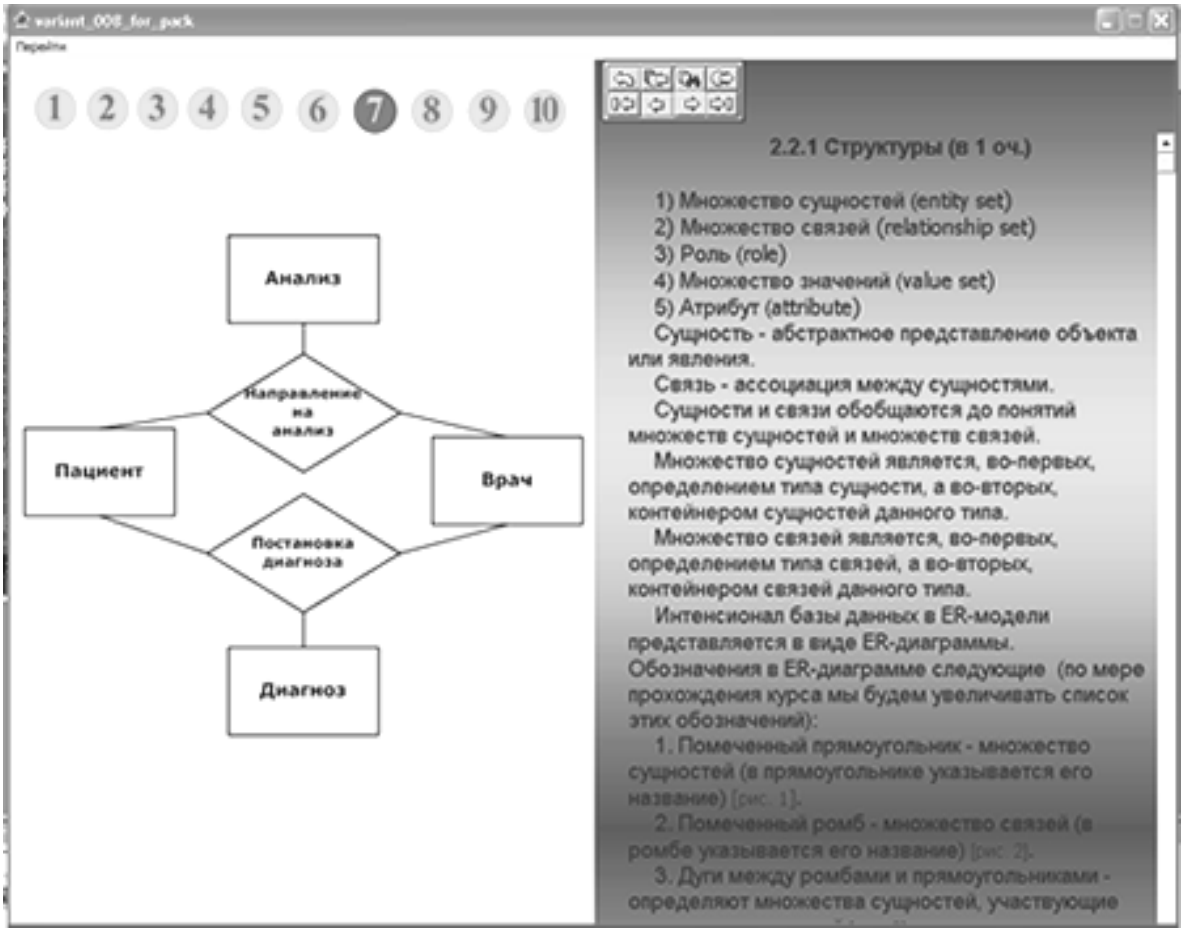


Рисунок Ж.1 – Главное окно программы

В программе доступно меню «Перейти», которое имеет подменю:

1. Содержание. Соответственно происходит переход на содержание курса

2. Тестирование. Студент переходит на страницу, где ему предлагается три возможных варианта: тестирование вопросами, упрощенное и усложненное задания.

3. Выход. При выборе этого пункта меню происходит выход из программы.

Тестирование вопросами

В этом разделе студенту предлагается ответить на двадцать вопросов, выбор которых происходит случайным образом из набора имеющихся вопросов. На экране по одному появляются вопросы, необходимо выбрать один или несколько вариантов ответа и нажать на кнопку «Далее». Возврат к предыдущим вопросам не предусмотрен. По окончании выдается статистика.

Упрощенное задание

В появившемся окне можно увидеть область рисования диаграммы, список предложенных названий множеств сущностей и связей и ряд кнопок. Вверху области рисования идет строка с названием режима, в котором пользователь в данный момент находится.

Для того чтобы приступить к работе, надо нажать кнопку "Начать задание". В самом начале пользователь находится в режиме создания множеств сущностей и связей, о чем ему напоминает строка режима. Чтобы создать множество, надо выделить в списке соответствующее название, поставить переключатель либо на "множество сущностей", либо на "множество связей" и нажать кнопку "Создать". Получаемые прямоугольники или ромбы можно перетаскивать в любое место в границах области рисования.

Чтобы нарисовать дуги между множествами сущностей и множествами связей, необходимо нажать на кнопку "Нарисовать дуги". При этом кнопка станет бледной, все посторонние кнопки станут невидимыми, и пользователь перейдет в режим рисования дуг.

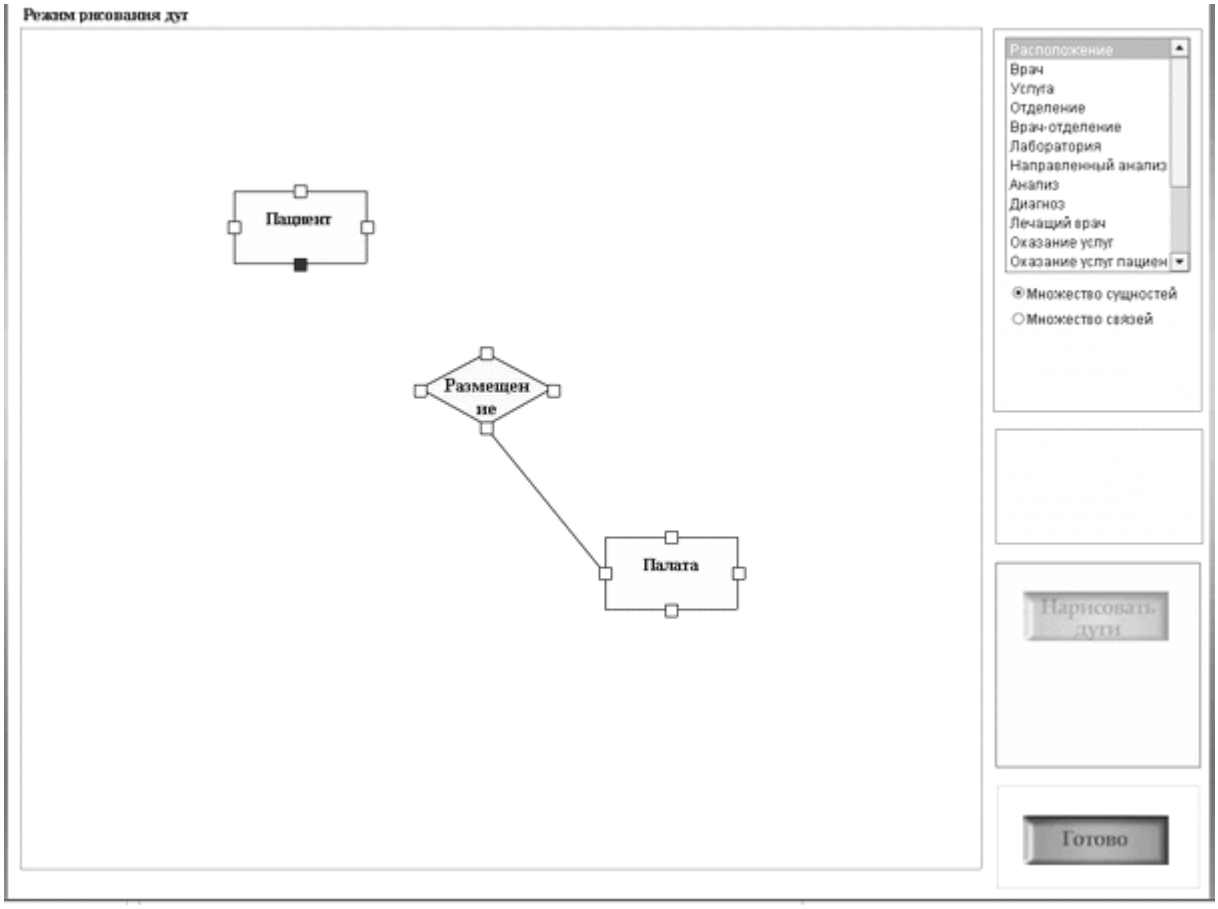


Рисунок Ж.2 – Упрощенное задание

Для рисования дуги необходимо щелкнуть сначала по одному ограничивающему квадратику, затем по второму. Между ними будет нарисована дуга.

Для удаления дуг необходимо нажать на кнопку «Удалить дуги». При этом данная кнопка станет бледной, остальные исчезнут и на всех элементах диаграммы появятся ограничивающие квадратики. Это режим удаления дуг. Чтобы удалить дугу, надо щелкнуть по одному и затем по второму квадратику, которые являются концами дуги.

Для того чтобы удалить множество, надо нажать на кнопку «Удалить множество». Она станет бледной, остальные кнопки исчезнут, пользователь переходит в режим удаления множеств. Щелкая мышкой по множеству на экране, пользователь удаляет его вместе со всеми инцидентными дугами.

После того, как все будет закончено, надо нажать на кнопку «Готово». Будет выведено окно, в котором пользователю в виде текста будет написано, какие множества сущностей и множества связей он создал, и на каких множествах сущностей последние были определены. При этом в папке результатов создается база данных с результатами работы и рисунок с созданной диаграммой для будущего просмотра преподавателем.

Усложненное задание

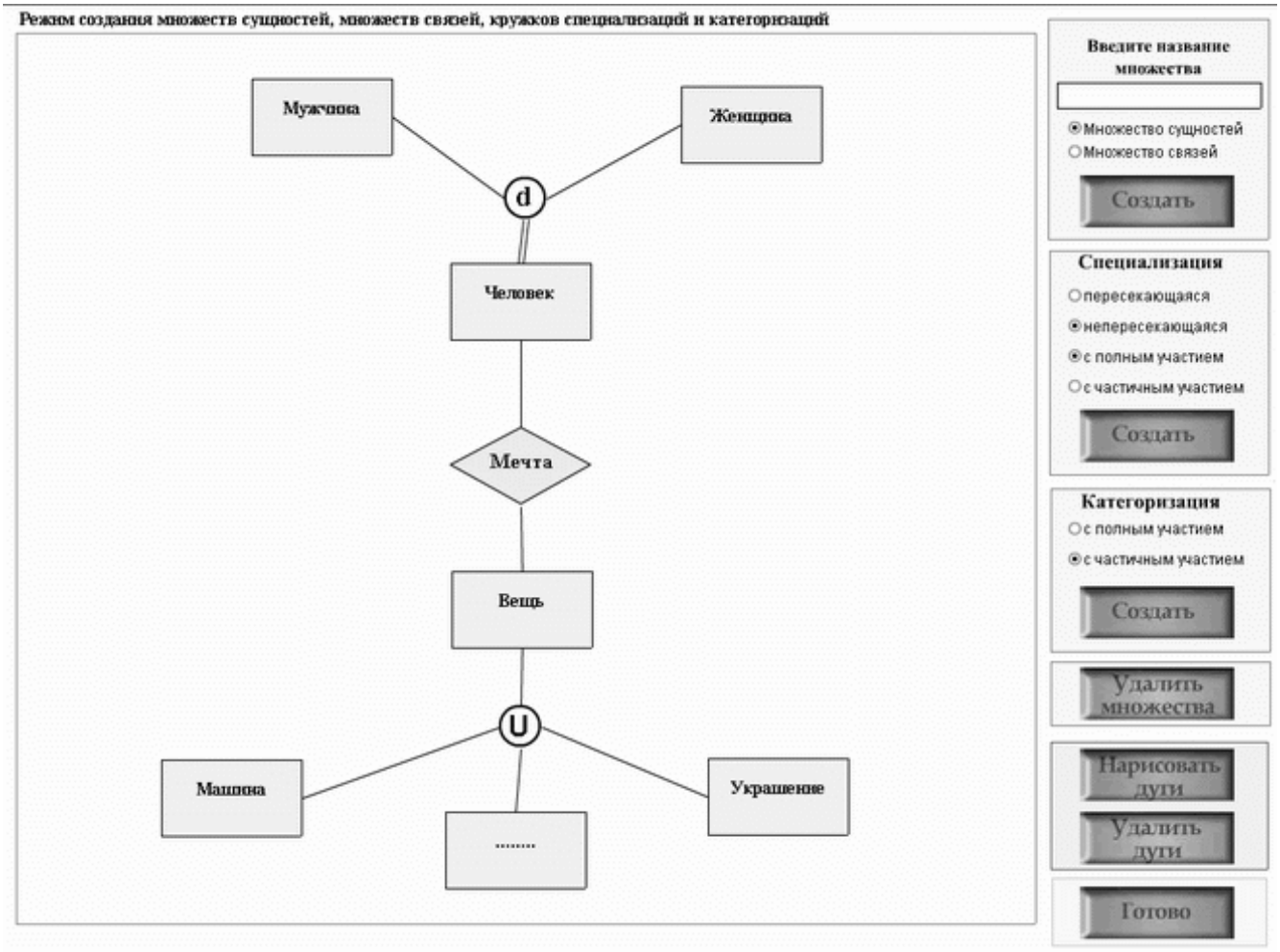


Рисунок Ж.3 – Усложненное задание

В этом задании нет списка с предложенными названиями, есть поле для текстового ввода названия множества сущностей или связей.

Чтобы создать специализацию, надо выбрать, какой она будет (пересекающейся или непересекающейся, с частичным или полным участием) и нажать на кнопку «Создать».

Аналогично для создания категоризации. При этом создаются только кружки специализации или категоризации. Для того чтобы определить полноценную специализацию, надо соединить кружок дугами с соответствующими множествами сущностей. Если для специализации еще не определен суперкласс, а для категоризации подкласс, то при рисовании дуги между множеством сущностей и кружком появится диалоговое окно. В этом окне спрашивается, идет ли данная дуга от суперкласса (для специализаций) или подкласса (для категоризаций). Соответственно, необходимо нажать на кнопку либо «Да», либо «Нет».

Выход из программы можно осуществить, только выбором соответствующего пункта в меню «Перейти».

Для всех пунктов меню предусмотрены «горячие клавиши»:

Содержание – Alt+C

Тестирование – Alt+T

Выход – Alt+X.