Міністерство освіти і науки України
Запорізький державний університет

До захисту допущений
Зав. кафедрою
_____ММ та ІТ_____

_____
(підпис)

Борю Сергій Юрійович
(прізвище, ім'я, по батькові)

_____
(дата)

ДИПЛОМНА РОБОТА

НА ТЕМУ: **РАЗРАБОТКА МОДЕЛИ ДАННЫХ И ИНФОРМАЦИОННОЙ ПОДДЕРЖКИ ЛОКАЛЬНОГО УРОВНЯ СРЕДЫ ИНТРАНЕТ ЗГУ**

Виконав Плецкій Сергій Юрійович

ст. групи ПМ-2 С. Ю. Плецкій
(шифр) (підпис і дата) (ім'я, по батькові, прізвище)

Керівник доцент В. А. Єрмолаєв
(посада) (підпис і дата) (ім'я, по батькові, прізвище)

Нормоконтролер І. А. Костюшко
(підпис) (ім'я, по батькові, прізвище)

Запоріжжя

1998

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Создание и внедрение программного обеспечения для корпоративных сетей на сегодняшний день является, пожалуй, одной из наиболее сложных проблем, с которыми приходится сталкиваться специалистам в области информационных систем (ИС) и информационных технологий (ИТ). С одной стороны, рассматриваемая предметная область подвержена резким изменениям с течением времени. С другой стороны, представление разработчика какой-либо прикладной системы всегда несколько отличается от представления пользователя о функциях и роли программного обеспечения, с которым ему предстоит иметь дело в повседневной работе. Поэтому основной задачей разработчиков ИС и ИТ является снабжение ПО интеллектуальными виртуальными свойствами. Для этого ИС должна удовлетворять следующим требованиям:

- ИС должна иметь интерфейс, интуитивно понятный пользователю и средства навигации, близкие и естественные для него;
- ИС должна прозрачно и гибко охватывать все программные средства и другие ресурсы, которые необходимы пользователю.

Одним из существующих подходов к решению данной проблемы является на данный момент создание сети Intranet - корпоративной сети, использующей технологию и стандарты Internet. Наиболее существенным преимуществом данного подхода является использование графического интерфейса на основе технологии WWW (World Wide Web).

Наиболее распространенным способом организации сетей Intranet является создание набора HTML страниц, представляющих те или иные узлы Корпоративной системы. Такой подход является неэффективным, так как порождает следующие проблемы:

- неэкономное использование системных ресурсов вследствие необходимости постоянного хранения статического набора HTML страниц;
- отсутствие удобного способа модификации элементов интерфейса в ответ на изменения в предметной области;
- отсутствие унифицированного пользовательского интерфейса.

В Запорожском Государственном Университете разработан проект создания Корпоративной сети Intranet, базирующийся на основе Единого Информационного Пространства Университета (ЕИПУ), описанного в [1]. Единое Информационное Пространство разработано как логическая надстройка над Корпоративной Сетью Университета. При этом используется динамический подход для создания элементов визуального интерфейса с использованием БД для хранения информации, необходимой для динамического создания HTML страниц. Предполагается использование визуального интерфейса, имитирующего структуру и содержание реальных объектов – университет, корпус (здание), подразделение, аудитория, сервер, рабочая станция, группа или сотрудник. Из-за ограниченности этого набора предоставляется возможность разработать объекты интерфейса, связав их с объектами реального мира и испоользовать его для создания конечных объектов интерфейса. Это позволит предоставить пользователю однородное виртуальное средство, обладающее стандартизованными процедурами доступа и средствами навигации.

Целью данной дипломной работы является:

- определение базового набора элементов визуального интерфейса ЕИП;
- создание логической модели БД ЕИП для хранения динамически получаемой информации;
- определение процедур виртуального интерфейса между пользователем и ЕИП;
- разработка алгоритмов работы процедур виртуального интерфейса.

1 ПОДХОДЫ И ЗАДАЧИ МОДЕЛИРОВАНИЯ КОРПОРАТИВНЫХ ИНФОРМАЦИОННЫХ СРЕД

1.1 Корпоративные сети Intranet

В этом разделе рассматривается один из подходов к организации корпоративных сетей – создание среды Intranet, а также рассматриваются применяемые при этом технологии.

1.1.1 Причины возникновения Intranet

Появление глобальной сети Internet ознаменовало новый этап в развити человечества. С этого момента наш век действительно стал информационным, так как Internet позволил осуществлять быстрый обмен информацией между людьми, находящимися в различных местах земного шара, предоставляя свои многочисленные сервисы - электронную почту, телеконференции, передачу файлов (FTP), доступ в режиме удаленного терминала и др. Но все же основным шагом к открытию Internet для массового пользователя стало появление технологии World Wide Web, которая полностью перевернула представления о работе с информацией. Существующий до этого целый набор средств для передачи данных из одной компьютерной системы в другую не учитывал одного из главных моментов - интерфейса взаимодействия человека с информационной системой. Web-технология, опираясь на наиболее естественный для человека способ потребления необходимой ему информации, предоставляет универсальный, интуитивно ясный инструмент для доступа к данным и является наиболее универсальным подходом к интеграции информационных ресурсов. Именно благодаря возможности использования столь удобного графического интерфейса и появились сети Intranet - локальные корпоративные сети, использующие технологию и стандарты Internet. Сети Intranet находят свое приеменение как в сфере бизнеса, так и в системе образования, например, в высших учебных заведениях.

Следующий раздел посвящен краткому обзору технологий, на которых основывается построение корпоративных сетей Intranet.

1.1.2 Технология и стандарты Intranet.

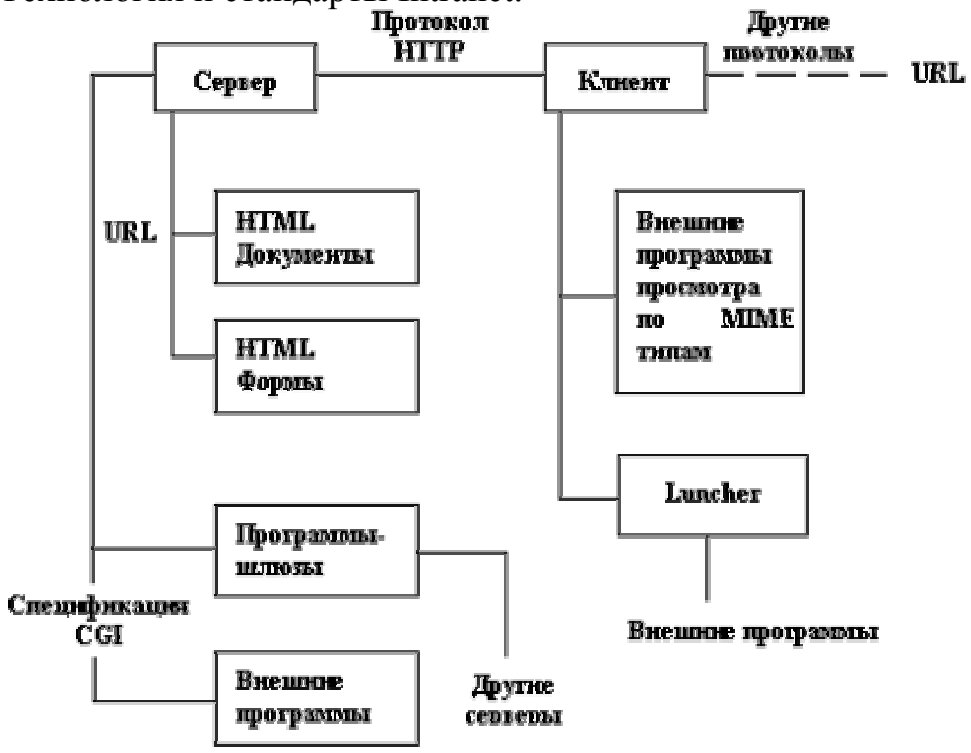

Рисунок 1.1. Архитектура технологии World Wide Web.

Hyper Text Markup Language (HTML) – язык гипертекстовой разметки.

HTML представляет собой стандартный синтаксис описания Web-документов, используемых для передачи информации в среде Internet [2]. Основными целями, которые преследовались при разработке HTML, были следующие:

- дать дизайнерам гипертекстовых баз данных простое средство создания документов;
- сделать это средство достаточно мощным, чтобы отразить имеющиеся на тот момент представления об интерфейсе пользователя гипертекстовых баз данных.

Web-документ представляет собой текстовый файл. Такой файл можно создать в любом текстовом редакторе на любой аппаратной платформе в среде любой операционной системы. К моменту разработки HTML существовал американский стандарт для разработки сетевых информационных систем - Z39.50, в котором в качестве единицы хранения указывался простой текстовый файл в кодировке LATIN1, что соответствует US ASCII.

Версии языка:

- HTML 1.0 - была направлена на представление языка как такового, где описание его возможностей носило скорее рекомендательный характер;

- HTML 2.0 - фиксировала практику использования конструкций языка;
- HTML++ - новые возможности;
- HTML 3.0 - призвана упорядочить все нововведения и согласовать их с существующей практикой.

Таговая модель документа

"ЭЛЕМЕНТ" := <"ИМЯ ЭЛЕМЕНТА" "СПИСОК АТРИБУТОВ">
СОДЕРЖАНИЕ ЭЛЕМЕНТА </"ИМЯ ЭЛЕМЕНТА">

Структура документа

Элемент HTML или гипертекстовый документ состоит из двух частей: заголовка документа (HEAD) и тела документа (BODY).

<HTML>
<HEAD> Содержание заголовка </HEAD>
<BODY> Содержание тела документа </BODY>
</HTML>

С помощью различного рода тегов можно как оформлять внешний вид HTML документа, так и организовывать связь с другими документами.

Common Gateway Interface (CGI) – универсальный шлюзовый интерфейс.

CGI является стандартом интерфейса (связи) внешней прикладной программы с информационным сервером типа HTTP, Web сервером.

Обычно гипертекстовые документы, извлекаемые из WWW серверов, содержат статические данные. С помощью CGI можно создавать CGI-программы, называемые шлюзами, которые во взаимодействии с такими прикладными системами, как система управления базой данных, электронная таблица, деловая графика и др., смогут выдать на экран пользователя динамическую информацию.

Программа-шлюз запускается WWW сервером в реальном масштабе времени. WWW сервер обеспечивает передачу запроса пользователя шлюзу, а она в свою очередь, используя средства прикладной системы, возвращает результат обработки запроса на экран пользователя. Программа-шлюз может быть закодирована на языках C/C++, Fortran, Perl, TCL, Unix Schell, Visual Basic, Apple Script. Как

выполнимый модуль, она записывается в поддиректорий с именем cgi-bin WWW сервера.

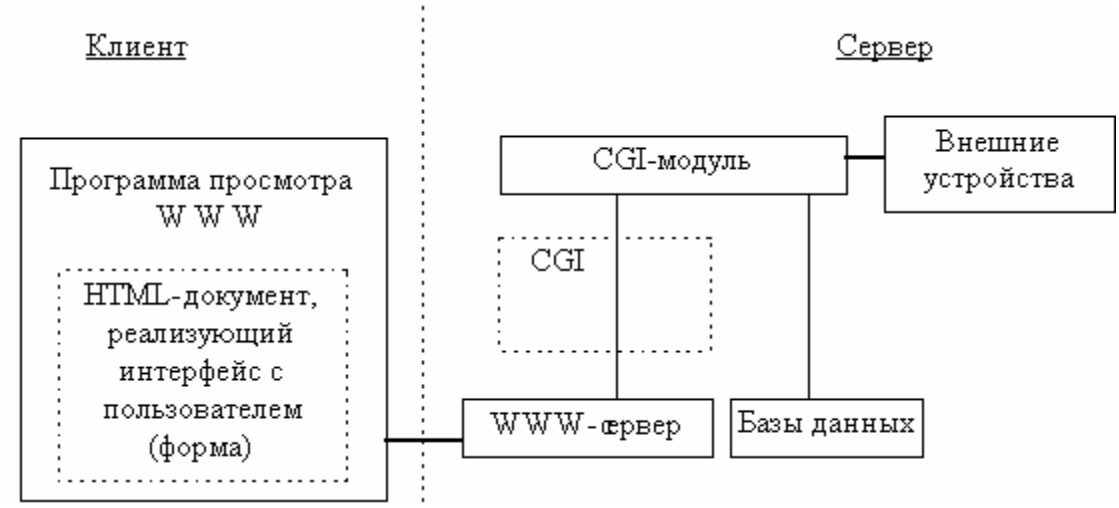

Рисунок 1.2. Клиентская и серверная части интерактивного интерфейса CGI.

Для передачи данных об информационном запросе от сервера к шлюзу, сервер использует командную строку и переменные окружения. Эти переменные окружения устанавливаются в тот момент, когда сервер выполняет программу шлюза. Информация шлюзам передается в следующей форме:

имя=значение&имя1=значение1&..,

где имя- имя переменной (из оператора FORM, например), и значение - ее реальное значение. В зависимости от метода, который используется для запроса, эта строка появляется или как часть URL (в случае метода GET), или как содержимое HTTP запроса (метод POST). В последнем случае, эта информация будет послана шлюзу в стандартный поток ввода.

Java.

Язык Java - это объектно-ориентированный язык программирования, созданный фирмой Sun для разработки программ, распространяемых по сети Internet. Система программирования Java позволяет использовать WWW для распространения небольших интерактивных прикладных программ (апплетов), которые размещаются на серверах Internet, транспортируются клиенту по сети, автоматически устанавливаются и запускаются на месте, как часть документа WWW. При этом апплет имеет весьма ограниченный доступ к ресурсам компьютера клиента, так что он может предоставить произвольный мультимедийный интерфейс и выполнять

сложные вычисления, не привнося при этом риска заражения вирусом или порчи данных.

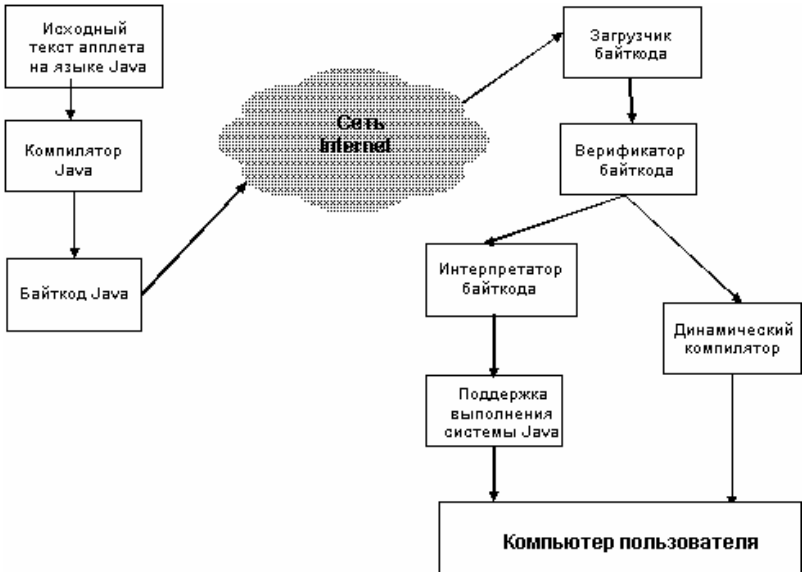

Рисунок 1.3. Выполнение Java-апплета.

Система программирования Java может служить основой для совместной разработки больших программных систем коллективом разработчиков, связанных между собой только через WWW. Ясно, что ведущую роль в обеспечении указанной возможности играет именно Java, так как именно Java позволяет распространять не просто тексты, а работающие программы и их фрагменты (апплеты) по WWW.

Основным свойством апплетов является возможность выполнять их на различных платформах и в различных окружениях, не оказывая вредного влияния на аппаратуру, программы и данные их пользователей. Язык, ориентированный на программирование апплетов, должен, прежде всего, обеспечивать надежность и безопасность. Этого легче всего достичь в интерпретируемом языке, хотя интерпретация, как правило, ведет к существенной потере эффективности программы (она выполняется гораздо медленнее, чем могла бы), причем эти потери растут нелинейно с ростом объема программы и объема обрабатываемых ею данных. Поэтому авторы языка Java предчувствовали возникновение проблемы эффективности Java-программ. Они ввели в язык легковесные процессы (трэды), обеспечив их параллельное выполнение на многопроцессорных компьютерах (которых становится все больше), они сконструировали интерпретатор языка (JavaVM) таким образом, что в Java-программу можно включать фрагменты,

написанные на других языках и выполняемые в объектном коде (native code) соответствующей платформы. Но эти средства языка Java не решают проблемы эффективности в полной мере, так как потери, связанные с интерпретацией намного больше.

1.2 Существующие подходы к организации сетей Intranet

В настоящее время при проектировании сетй Intranet применяются два подхода, которые в зависимости от организации способа получения пользователем HTML-документов условно можно назвать статическим и динамическим.

В первом случае источником интерфейса является HTML-документ, созданный в каком-либо текстовом или HTML-ориентированном редакторе. Данный документ храниится в готовом виде и остается неизменным в течение использования. В качестве примера системы, использующей статический подход можно привести построитель Web-карт PTOLOMAEUS, разработанный на факультете Информатики и Автоматизации Римского Университета (Department of "Informatica ed Automazione" of University "Roma Tre", Rome, ITALY). Этот прграммный продукт предоставляет возможность как создания HTML документов, так и организации гипертекстовых связей между ними. Однако такой подход является весьма неэффективным, поскольку любое изменение в предметной области влечет за собой либо модификацию существующего HTML документа, либо создание нового. Организация же связи с ним даже при наличии такого подручного средства как PTOLEMAEUS представляет собой не самую простую задачу.

Во втором случае источником интерфейса является HTML-документ, сгенерированный CGI-модулем, вследствие чего появляется гибкость в видоизменении интерфейса во время использования. Динамический подход используется, например, при создании поисковыз серверов Internet (Yahoo, AltaVista и др.). В качестве результата на свой запрос пользователь получает динамически созданную HTML-страницу с набором ссылок на документы с интересующей его информацией. Другим примером использования динамического подхода является организация доступа к существующим хранилищам информации

(БД) с использованием WWW-интерфейса. Однако существующие на данный момент подобные разработки не предоставляют пользователю интерфейс, адекватно отражающий предметную область их корпоративной сети, а могут служить лишь средством получения информации справочного характера.

В Запорожском Государственном Университете на данный момент разаботан проект построения корпоративной сети Intranet ЗГУ, авторы которого попытались решить данную задачу путем объединения некоторых концепций:

- концепции визуализации интерфейса;
- концепции федерализации данных;
- концепции создания адаптивных информационных систем.

Далее будет проведен их краткий обзор, а также будут рассмотрены конкретные проекты, основанные на их использовании.

Концепция виртуализации ресурсов.

Концепция виртуализации ресурсов состоит в использовании определенных ресурсов для имитации раьоты с объектами иной природы. При этом предоставляется возможность использования «родного» интерфейса (средств взаимодействия, включающих способы передачи информации и обработки возвращаемой информации) для работы с виртуальным объектом.

Одной из областей, в которой данная концепция преобрела широкое применение, является разработка операционных систем. Такие понятия, как «виртуальная память», «виртуальный диск», «виртуальная ЭВМ» на сегодняшний день имеют достаточно обширное применение [3].

Кроме того, в настоящее время вследствие широкого распространения такого явления, как Internet, виртуализация ресурсов переходит на несколько иной уровень. Существуют попытки организации виртуальных объектов такого класса, как магазины, библиотеки и.т.д. Например, на Web-страницах сервера Citforum (www.citforum.ru) существует ссылка на виртуальный книжный магазин Mistral. Другим примером может служить разработка Дельфийского университета - виртуальная библиотека, в которой для поиска интересующей информации

используется интерфейс, имитирующий содержание реальной библиотеки этой организации.

Концепция федерализации данных.

Во многих крупных организациях очень часто одновременно используются несколько различных систем управления данными. Эти системы и данные, которые они обрабатывают, зачастую разрабатываются независимо друг от друга. Это могут быть как СУБД, так и просто файловые системы, отличающиеся по многим аспектам, таким как используемая модель данных, язык запросов, системная архитектура, и т.д., а также структурой и семантикой управления данными. В этом контексте обычно используется термин «разнородность систем управления данными». Для решения этой проблемы и разрабатываются федеративные системы управления базами данных (ФСУБД).

Основная идея ФСУБД состоит в организации единого интерфейса для разнородных источников данных независимо разработанных систем. При этом обращение к локальным СУД осуществляется с использованием их «родного» интерфейса, поэтому нет необходимости вносить какие–либо изменения в рамках локальной системы. ФСУБД предлагает однородную интегрированную схему. Существует механизм преобразования из интегрированной схемы в локальную. На первом шаге процесса интеграции разнородность модели данных может быть решена путем преобразования локальной схемы в каноническую модель данных ФСУБД. На втором шаге схематическая разнородность решается путем интегрирации схемы. В основном этот шаг представляет собой сложную задачу и приводит к конфликтам различного рода. На данный момент существует ряд теоретических разработок, предлагающих различные технические премы разрешения этих конфликтов.

Концепция построения адаптивных информационных систем.

Данная концепция получила свое развитие в связи с необходимостью изменения модели данных используемой ИС (в частности при создании СУБД) без

внесения изменений в связанный с ней программный код. Перечислим некоторые разработки, касающиеся данной концепции.

A. Rosenthal и D. Reiner [4] представили проект Проектирование Баз Данных и Инструментальные средства проверки – систему, которая использует информацию, содержащуюся внутри себя для предотвращения трансформации используемой модели.

P. Fraternali, L. Tanca [5] добавляют некоторое дополнение к системам Базы Данных - Активный подход Баз Данных. Это основано на правилах ECA и некоторых расширений ECA. Подход к системе Баз Данных определен согласно некоторому набору правил ECA. Другие авторы работают в области Исследований Наборов Активных Правил Базы Данных. Пример - Система Правил Starbust см. A. Aiken, J. M. Hellerstein, J. Widom [6].

В качестве конкретного проекта, основанного на использовании данной концепции можно упомянуть систему «Абитуриент» [7], разработанную и использующуюся в нашем университете. В основе данной системы лежит идея использования Активных Словарей Данных [8].

1.3 Виртуальный Университет

Все аспекты организации доступа к информационным ресурсам, перечисленные в разделе 1.2 решают конкретные задачи и на данный момент нашли свое широкое применение в различных областях. Ниже будет рассмотрена концепция Виртуального Университета, разработанная в ЗГУ и объединяющая все эти выиграшные технологии в рамках одного информационного проекта – построения корпоративной сети Intranet ЗГУ [9].

1.3.1 Концепция Виртуального Университета и Визуальный интерфейс.

Концепция Виртуального Университета Запорожского Государственного Университета формируется на основе Единого Информационного Пространства Университета (ЕИПУ), описанного в [1]. Единое Информационное Пространство разработано как логическая надстройка над Корпоративной Сетью Университета.

Как и любая другая информационная система, Корпоративная Сеть Университета представляет собой сложную разнородную систему, состоящую из элементов различных типов аппаратного и программного обеспечения на разных уровнях. Основная задача ЕИПУ - играть роль однородного виртуального средства, предоставляющего пользователю стандартизованные процедуры доступа и средства навигации, системному разработчику - необходимые свойства адаптируемости и гибкости а системному администратору - эффективное средство контроля.

Основное внимание при разработке данного проекта уделяется созданию для пользователя интуитивно понятного интерфейса. Поэтому одним из основных элементов ВУ является Unified Visual Intranet Interface (UVII) – Унифицированный Визуальный Интерфейс Intranet. Современное информационное окружение настолько сложно, что попытка обзора всей доступной информации представляет собой весьма трудоемкую задачу. Решение этой задачи в рамках данного проекта основывается на использовании визуального интерфейса, имитирующего структуру и содержание реальных объектов – университет, корпус (здание), подразделение, аудитория, сервер, рабочая станция, группа или сотрудник. Так как набор этих объектов ограничен, то предлагается разработать объекты интерфейса, связав их с объектами реального мира, что будет первым шагом на пути преодоления существующего семантического разрыва.

В следующем разделе будут описаны некоторые аспекты разработки архитектуры ЕИП. Так как главной идеей первого уровня ЕИП является автоматическое генерирование HTML кода для элементов интерфейса, то следующий шаг состоит в получении необходимых данных из БД ЕИП.

1.3.2 Архитектура Единого Информационного Пространства.

Для описания архитектуры ЕИП рассмотрим его отдельные уровни и интерфейсы между ними. Первым уровнем является уровень Пользователя или Внешний уровень. Он был описан в предыдущем разделе. Элементами интерфейса данного уровня являются: запрос Пользователя к ЕИП и результат в форме HTML кода, возвращаемый ЕИП броузеру пользователя. Следующий уровень -

Корпоративная Информационная Система. Задача этого уровня состоит в том, чтобы преобразовать запрос пользователя в виртуальный запрос к Корпоративной ИС. После обработки виртуального запроса его результат передается Обработчику Результата запроса, который генерирует соответствующий HTML код и возвращает его на Внешний уровень (Уровень Пользователя). Архитектура Внешнего уровня и уровня Корпоративной ИС схеметически показана на рисунке 1.4.

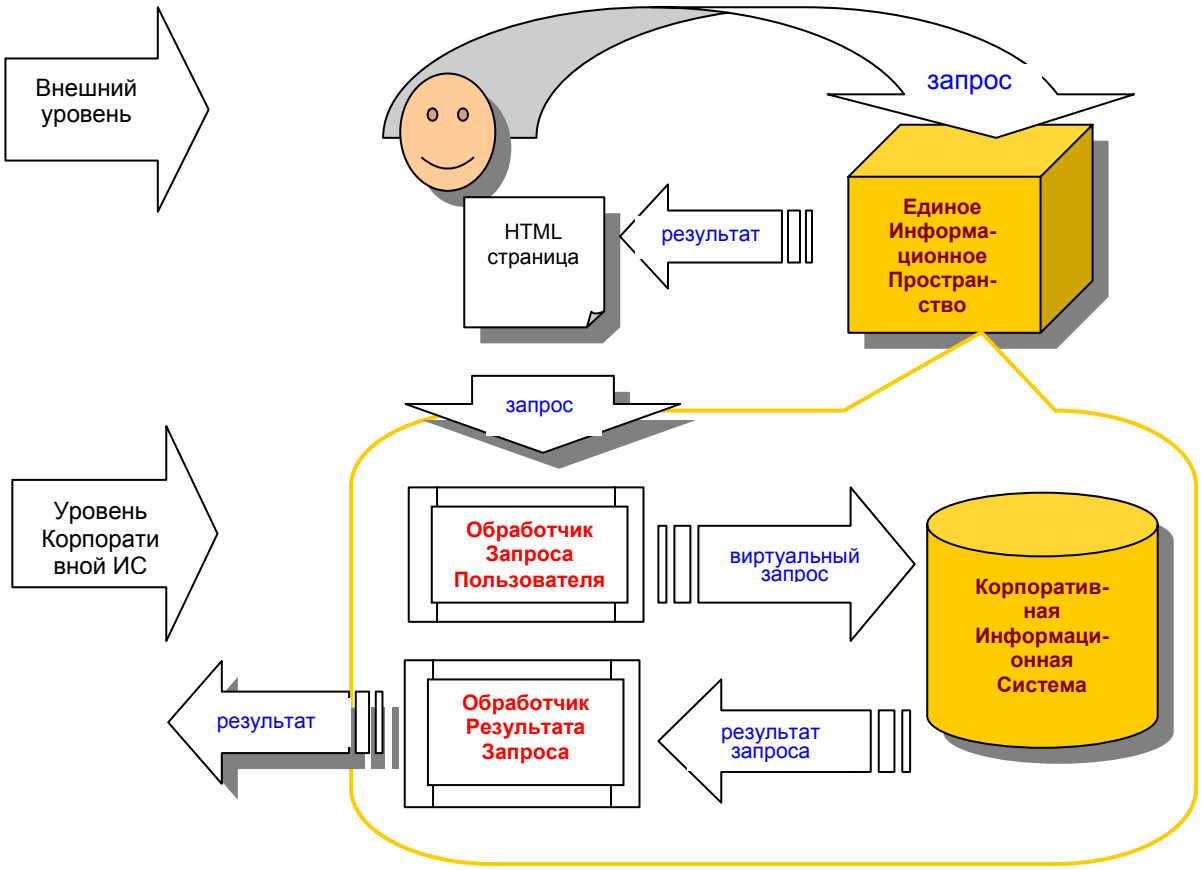

Рисунок 1.4. Внешний уровень и уровень Корпоративной ИС Архитектуры ЕИП.

После того, как запрос пользователя будут преобразован в запрос к КИС, возникает возможность описания UVII в тиерминах модели данных, объектов и отношений и рассматривать запрос как запрос в смысле СУБД. Для осуществления задач этого уровня архитектуры нам необходима модель данных для описания элементов интерфейса, СУБД, обработчик запроса пользователя и обработчик результата запроса к КИС.

Далее, при осуществлении виртуального запроса возникает ряд проблем. КИС представляет собой достаточно сложную структуру, включающую ряд Локальных Информационных Систем и Функциональных Серверов, распределенных по всему сетевому окружению и управляющих локальной информацией, приложениями и ресурсами. Данные и функциональные характеристики приложений и ресурсов зачастую имеют семантические пересечения. Одним из методов разрешения этой проблемы является использование Федеративных моделей данных и СУБД. В ЕИП планируется использование метода и модели, разработанных G. Saake и его коллегами [10], с использованием Активных Словарей Данных [8]. Таким образом, Федеративный уровень архитектуры ЕИП может быть представлен схемой, изображенной на рисунке 1.5.

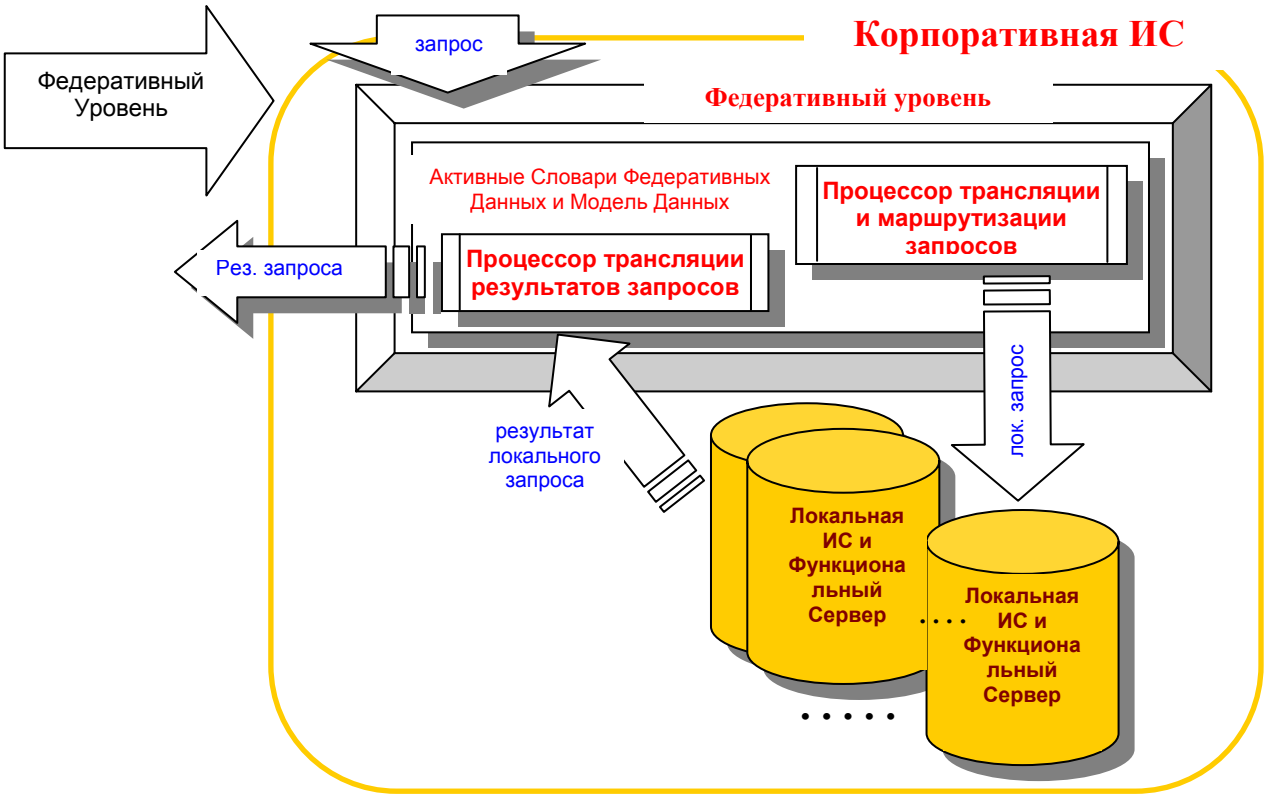

Рисунок 1.5. Федеративный уровень Единого Информационного Пространства.

Этот уровень разделяется на два подуровня с соответствующими интерфейсами – Федеративный сервер и набор серверов локальных информационных ресурсов. Федеративный уровень осуществляет преобразование поступающего виртуального запроса в набор локальных запросов и перенаправляет их к элементам локального уровня архитектуры.

Главной архитектурной особенностью локального уровня является метод контроля Локальной ИС. Предполагается использование Активного словаря данных в качестве оболочки (shell) контроля ИС и Модели данных в качестве средства (engine) контроля. Таким образом, локальный уровень представлен тремя уровнями: Уровнем Программного Кода, Уровнем Модели Данных и Уровнем Данных (рисунок 1.6). Словарь Данных с его активными функциями, контролируемый Моделью Данных, предоставляет гибкий интерфейс между верхними уровнями и локальными данными и/или ресурсами.

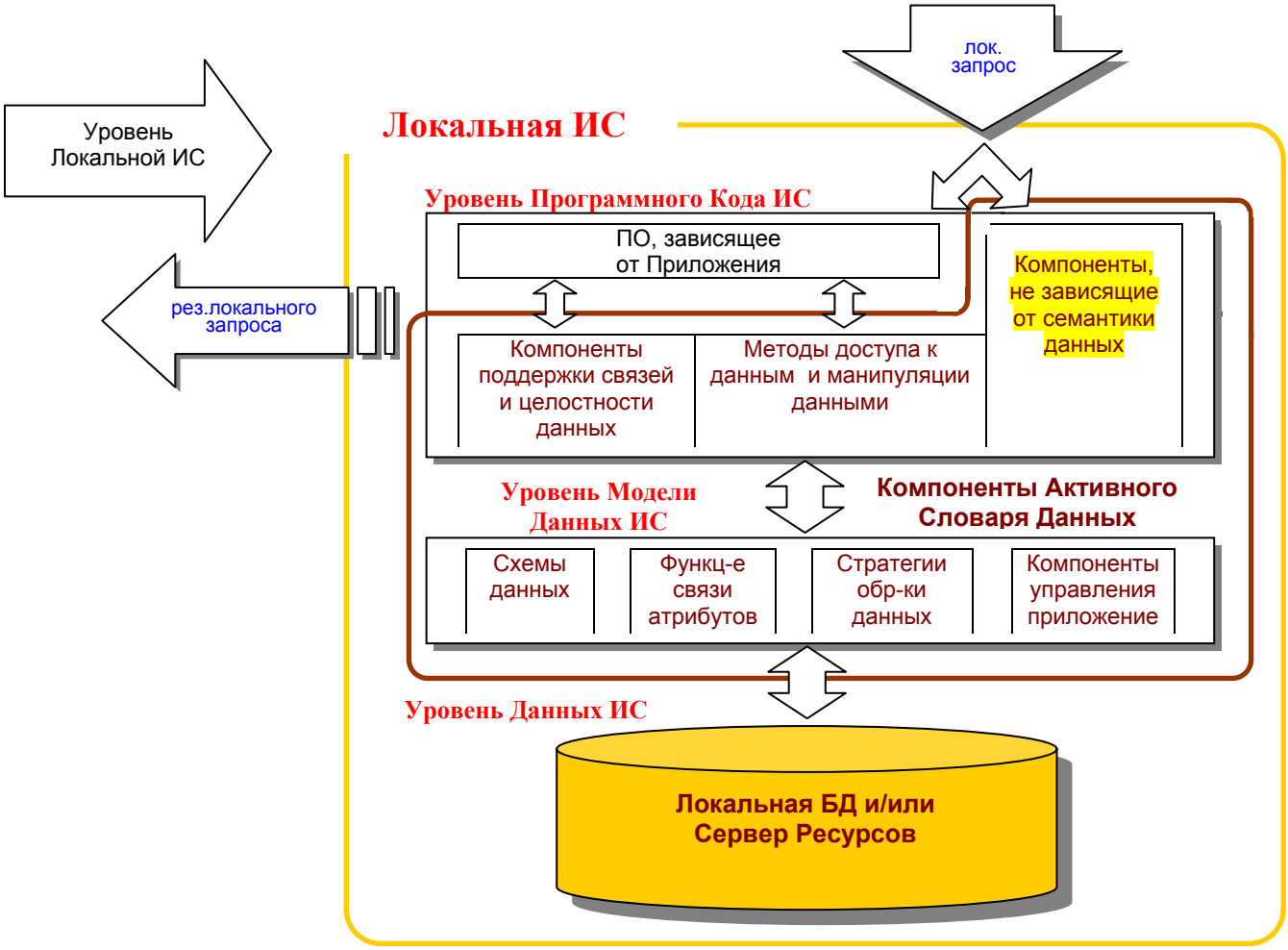

Рисунок 1.6. Уровень Локальной ИС Единого Информационного Пространства.

1.4 Постановка задачи

Проанализировав содержание предыдущих разделов, мы приходим к следующим выводам:

- идея объединения в рамках одного проекта ВУ таких концепций, как виртуализация ресурсов, федерализация данных, построение адаптивных информационных систем нигде раньше не использоваалась и представляет собой новый подход к задаче построения корпоративной сети Intranet;
- основной идеей использования компонентов визуального интерфейса как части ЕИП является использование ограниченного набора стандартных элементов интерфейса, которые могут быть легко созданы автоматически;
- интерфейс ЕИП может рассматриваться, как до сих пор не использовавшийся в применении к такому объекту, как университет, и является новым в Украине.

ЦЕЛЬ РАБОТЫ:

В данной дипломной работе предполагается решение части задач, поставленых в рассмотренном проекте, а именно задач Корпоративного Уровня:

- определение базового набора элементов интерфейса ЕИП;
- создание логической модели данных ЕИП ЗГУ на основе базового набора элементов, позволяющей отразить любые объекты предметной области – материальные, информационные и административные ресурсы;
- определение процедур виртуального интерфейса между пользователем и ЕИП – Обработчика Запроса Пользователя и Обработчика Результата Запроса;
- разработка алгоритмов работы ОЗП и ОРЗ.

2 РАЗРАБОТКА МОДЕЛЕЙ И АЛГОРИТМОВ ЭЛЕМЕНТОВ ВИРТУАЛЬНОГО УНИВЕРСИТЕТА

2.1 Определение базового набора элементов интерфейса ЕИП

В этом разделе проводится анализ предметной области, разрабатывается схема для описания произвольного объекта интерфейса ВУ и на ее основании описывается иерархия классов объектов интерфейса.

2.1.1 Структура Web-документа, моделирующего объект ВУ

Для описания объектов реального университета предполагается использование ограниченного набора связанных элементов интерфейса ВУ. Поэтому из данных, описывающих произвольный объект, можно выделить данные, являющиеся общими для всех объектов данного класса (объектов, описанных с помощью одного элемента). Эти данные будут определять структуру данного класса (шаблон объекта). Остальные данные будут уникальными для каждого отдельного объекта (данные экземпляра класса). Данные класса описывают определенный тип объекта реального мира, тогда как данные экземпляра класса описывают сам объект. Для осуществления доступа к определенному классу введем дополнительные классовые данные – идентификатор класса (ИК). Аналогично, для доступа к данным экземпляра класса введем идентификатор экземпляра класса (ИЭК). Таким образом, для получения доступа к конкретному элементу интерфейса ВУ необходимо определить ИК и ИЭК данного объекта.

Далее, кроме информации, описывающей текущий объект, на Web-странице присутствуют элементы, соответствующие другим реальным объектам, связанным с текущим и используемые для перехода к соответствующим им HTML страницам. Условно их можно назвать элементами навигации. Кроме того, в исходном тексте HTML документа присутствует информация, необходимая для осуществления связи с другими документами и представляет собой ссылки на них в формате URL. Поэтому все данные, составляющие исходный текст гипертекстового документа

можно условно разбить на собственно данные (информационные данные) и на данные-связи (включающие элементы навигации плюс адреса связанных объектов).

Таким образом, мы приходим к следующим выводам. При описании элемента интерфейса ВУ используются:

- данные класса, часть которых описывает текущий объект, а часть - связанные объекты (ИК и соответствующие элементы навигации);

- данные экземпляра класса, часть которых содержит информацию о текущем объекте, а часть - о связанных объектах (ИЭК и информация, относящаяся к элементам навигации).

Описанная схема объекта интерфейса может быть представлена с помощью таблицы 2.1.

Таблица 2.1. Схема объекта интерфейса ВУ.

	ДАННЫЕ КЛАССА	ДАННЫЕ ЭКЗЕМПЛЯРА КЛАССА
Текущий объект	ИК	ИЭК
	…	…
Связанные объекты	ИК	ИЭК (элемент навигации)
	…	…

2.1.2 Модель хранения данных ВУ

Выше была описана схема объекта ВУ, получаемого пользователем после обращения к ВУ – схема готовой Web-страницы. Далее будет рассмотрен вопрос о том, каким образом следует организовать хранение необходимой для ее создания информации.

Так как данные класса произвольного объекта интерфейса ВУ являются общими для всех объектов данного класса, а набор классов – ограничен, то эти данные можно хранить в одном экземпляре отдельно от данных экземпляра класса. При этом информационные данные являются шаблоном гипертекстового документа

и определяют его внешний вид. Данные-связи используются в процессе создания Web-страниц для определения классов связанных объектов.

Данные экземпляра класса представляют собой именнно ту часть информации, которая должна быть получена из БД ЕИП в ответ на запрос пользователя. Для доступа к этим данным кроме ИЭК необходимо наличие ИК, к которому принадлежит данный объект. Все вышесказанное проиллюстрированно на рисунке 2.1.

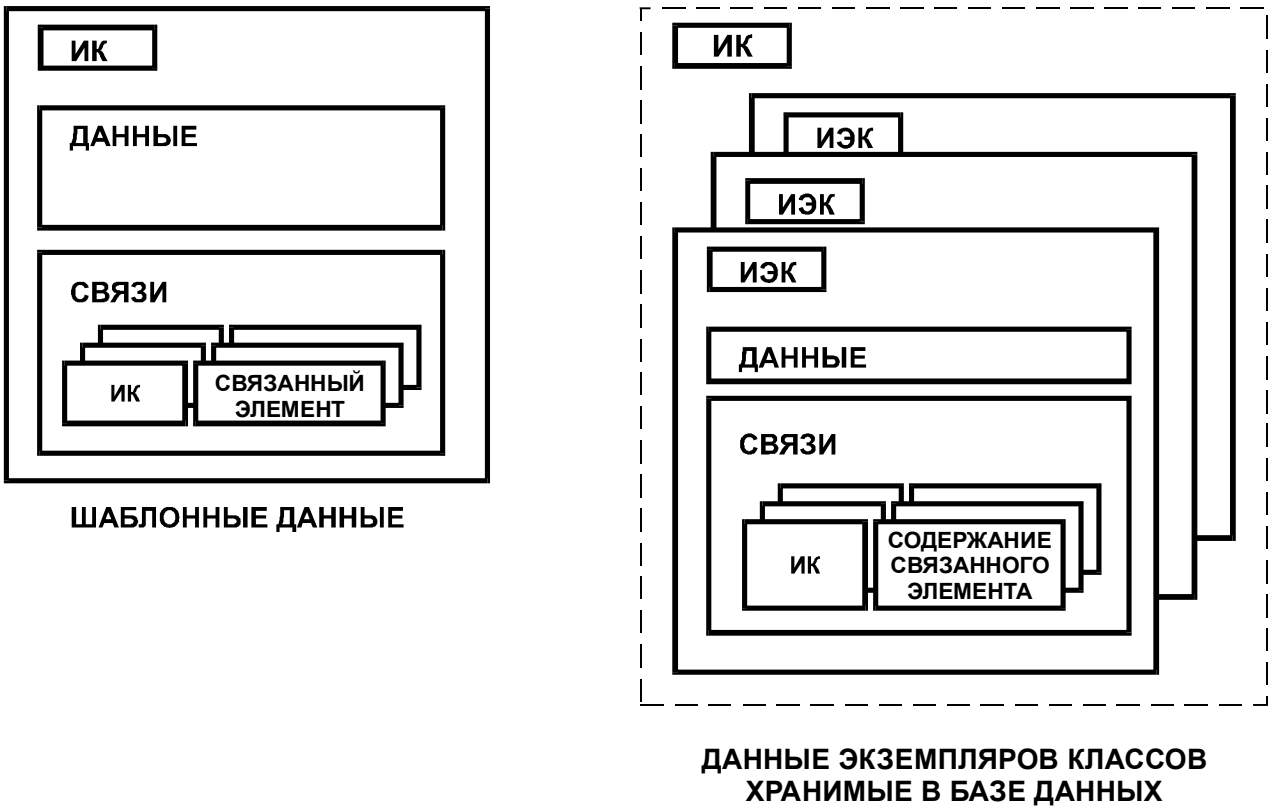

Рисунок 2.1. Модель хранения данных, описывающих элемент интерфейса ВУ.

2.1.3 Классы объектов интерфейса ВУ, иерархия классов

В этом пункте мы рассмотрим все интересующие нас объекты реального университета и, пользуясь разработанной схемой, опишем соответствующие им объекты ВУ. Затем выясним, каким образом они могут быть связаны между собой.

УНИВЕРСИТЕТ

Так как вся интересующая нас информация, относящаяся к университету, находится в пределах какого-либо конкретного здания, то первым элементом можно считать университет, представленный с помощью плана. Поэтому первым классом в

иерархии ВУ будет УНИВЕРСИТЕТ. Поскольку мы рассматриваем модель конкретного университета - ЗГУ, то данные класса УНИВЕРСИТЕТ существуют в одном экземпляре, поэтому ИЭК отсутствует.

ЭТАЖ КОРПУСА

Так как каждый корпус университета состоит из этажей (одного или нескольких), то следующим элементом интерфейса будем считать один этаж какого-либо корпуса. Основными свойствами этого элемента являются изображение плана, элементы навигации, связывающие его с другими объектами: аудиториями, следующими этажами данного корпуса.

АУДИТОРИЯ

Аудиторию представляют две группы объектов реального мира: сетевые ресурсы - рабочие станции и сервера, и люди, относящиеся к определенным подразделениям.

ЭВМ

Следующим классом в нашей иерархии будет класс "ЭВМ", представляющий как сервера, так и рабочие станции, находящиеся в пределах аудитории. Задачей произвольного сетевого узла является предоставление пользователю доступа к функциям, которые этот ресурс в состоянии выполнять и/или к хранимым ресурсам.

ПОЛЬЗОВАТЕЛЬ

Кроме самих ЭВМ, нас могут заинтересовать люди, работающие с ними. Отсюда мы получаем еще один класс - "ПОЛЬЗОВАТЕЛЬ" данной ЭВМ. Причем имеет смысл рассматривать лишь тех пользователей, которые в силу тех или иных причин жестко закреплены за данной машиной. Элементы этого уровня предоставляют средства связи с данным человеком (электронная почта, телефонная связь) и связи с элементами, относящимися к данному человеку и формирующими функциональную модель объекта реального мира.

Кроме описанных выше объектов реального университета мы можем рассмотреть и такие объекты, как, например, факультет, кафедра, бухгалтерия и т.д., то есть какие-то административные подразделения. Они могут нас заинтересовать лишь в том случае, если в их состав входят какие-либо рассматриваемые нами

информационные ресурсы. Например, в бухгалтерии используются ЭВМ с соответствующим программным обеспечением. Или некоторые пользователи ЭВМ являются сотрудниками определенных подразделений и занимают там определенную должность. Кроме того, можно объединить по тем или иным причинам непосредственно и сами ресурсы. Во всех этих случаях было бы хорошо иметь возможность каким-то образом получить доступ ко всем объединенным ресурсам одновременно. Для этого вводится еще один объект ВУ – класс “ВИРТУАЛЬНЫЙ ОБЪЕКТ”, находящийся на произвольном уровне описанной

Рисунок 2.2. Иерархия объектов ЕИП ВУ.

иерархии и объединяющий объекты следующего уровня иерархии. При этом сохраняется шаблон представления объектов. Этот подход дает возможность комбинировать объекты произвольным образом для осуществления более гибкого доступа к ним. Например, “ЭТАЖ ВИРТУАЛЬНОГО КОРПУСА” представляет собой объединение этажей, которые на самом деле принадлежат различным корпусам реального университета. На каждом этаже такого “корпуса” пользователь может получить доступ лишь к тем аудиториям, которые принадлежат данному объединению.

Графически описанная нами иерархия объектов ВУ представлена на рисунке 2.2, а соответствующие схемы элементов даны в Приложении 1.

2.2 Концептуальная модель ВУ.

В этом разделе определяется выбор модели данных для создания логической модели ЕИП, проводится описание используемых при моделировании базовых средств а также разрабатывается конечная схема БД ЕИП.

2.2.1 Выбор модели данных.

В настоящее время существуют три основных подхода к организации системы базы данных, каждый из которых имеет свою сферу применения. Это иерархический, сетевой и реляционный подходы.

Мы будем использовать реляционный подход, так как он обладает следующими преимуществами:

1. Простота использования, т.е. время на разработку программ и время формулирования запросов оптимально, что ведет к широкой применяемости и высокой производительности реляционных баз данных, особенно малых и средних размеров.
2. Из трех указанных моделей только реляционная имеет в своей основе теоретические предложения, предопределяющие любые правильные реализации.
3. Небольшое число понятий в описательной части реляционной модели данных, причем различные понятия четко отделяются и не переплетаются сложными взаимосвязями.
4. Однородность данных - все данные рассматриваются как хранимые в таблицах, в которых каждая строка имеет один и тот же формат. Каждая строка в таблице представляет некоторый объект реального мира или соотношение между объектами.
5. Физический и логический уровни проектирования разделены, поэтому проблемы, связанные с разными уровнями, решаются по отдельности.

6. Эффективность реализации, т.е. простота перевода спецификаций концептуальной схемы в реализацию, эффективную с точки зрения необходимого производства и обработки запросов.

2.2.2 Описание базовых средств для информационного моделирования.

Приведем некоторые определения элементов реляционной модели данных, используемой в в качестве основы проектирования модели БД ЕИП [11].

Определение 1.

Схемой отношений R называется конечное множество имен атрибутов $\{A_1, A_2, ..., A_n\}$. Каждому имени атрибута A_i ставится в соответствие множество D_i называемое доменом атрибута A_i, $1 \le i \le n$. Домен атрибута A_i будем обозначать $dom(A_i)$. Домен является произвольным непустым конечным или счетным множеством. Пусть $D = D_1 \cup D_2 \cup ....\cup D_n$. Отношение r со схемой R - это конечное множество отображений $\{t_1, t_2, ..., t_p\}$ из R в D, причем каждое отображение $t \in r$ должно удовлетворять следующему ограничению $t(A_i) \in D_i$, $1 \le i \le n$. Эти отношения называют кортежами.

Определение 2.

Ключ отношения r со схемой R является подмножеством $K = \{B_1, B_2, ..., B_m\} \subseteq R$ со следующими свойством: для любых двух кортежей t_1 и t_2 в r существует такое $B \in K$ что $t_1(B) \ne t_2(B)$, другими словами не существует двух кортежей имеющих одно и то же значение на всех атрибутах из К. Это условие записывается так: $t_1(K) \ne t_2(K)$. Таким образом достаточно знать К значение кортежа чтобы однозначно идентифицировать кортеж. Более точно ключ отношения r(R) является подмножеством $K \subseteq R$, таким что для любых различных кортежей t_1 и t_2 из r выполняется $t_1(K) \ne t_2(K)$, и ни одно собственное подмножество $K' \subset K$ не обладает этим свойством. Множество К является суперключом относительно r, если К содержит ключ отношения r.

Определение 3.

Пусть r - отношение со схемой R, X и Y - подмножества R. Отношение r удовлетворяет функциональной зависимости $X \to Y$, если $\pi_Y(\sigma_{X=x}(r))$ имеет не более чем один кортеж для каждого X значения x. Где $\sigma_{X=x}(r)$ обозначает операцию

выбора (выбрать в r кортеж в котором значение X равно x), а $\pi_X(\sigma(r))$ - проекция r на X, то есть отношение r'(X), полученное вычеркиванием соответствующих атрибутам в R - X и исключением из оставшихся столбцов повторяющихся строк.

Определение 4.

Пусть U - множество атрибутов, каждый из которых соотнесен с определенным доменом, схемой отношений реляционной базы данных R над U называется совокупность схем отношений $\{R_1,R_2,...,R_p\}$, где $Ri=\{S_i,K_i\}$, $1 \le i \le p$ $\bigcup_{i=1}^{p} S_i = U$, $S_i \neq S_j$ при $i \neq j$.

Определение 5.

Схема отношения R находится в первой нормальной форме (1НФ) если значения в dom(A) являются атомарными для каждого атрибута A в R, т.е. значение в домене не являются ни списком ни сложным значением. Преимущество 1НФ в том, что 1НФ позволяет выражать F-зависимости с той степенью детализации с какой нам требуется, что невозможно без 1НФ.

Определение 6.

Для данной схемы отношения R, подмножества X множества R, атрибута Y в R и множества функциональных зависимостей F атрибут Y называется частично зависимым от X, если существует $X' \subset X$, такое, что $X' \to Y$ - также F-зависимость, которой удовлетворяет отношение r. Иначе - атрибут Y полностью зависит от X.

Определение 7.

Схема отношений R находится во второй нормальной форме (2НФ) относительно множества функциональных зависимостей F, если она находится в 1НФ и ни один из непервичных атрибутов в R не является частично зависимым от ключа для R. Схема базы данных R имеет вторую нормальную форму относительно F, если каждая схема отношения R из R находится в 2НФ относительно

2.2.3 Описание схемы БД ЕИП

При описании схем отношений, составляющих схему отношений БД будем придерживаться следующего синтаксиса:

Имя отношения {Атрибут_1, Атрибут_2, …, Атрибут_N},

где подчеркиванием выделяется первичный ключ отношения.

Каждое отношение будет описывать определенный класс объектов ВУ. Поэтому в качестве имен отношений можно для удобства использовать ИК соответствующих классов. Интересующей нас информацией является Информация Экземпляра Класса а также информация, относящаяся к элементам навигации.

Рассмотрим конкретные схемы отношений.

Схема 1. Университет {Название, Описание, План университета}

Данная схема находится во 2НФ.

Схема 2. Этаж корпуса {№ корпуса, Описание корпуса, Расположение корпуса на плане университета, № этажа, Описание этажа, План этажа, Расположение этажа на планах смежных этажей}

Данная схема находится в 1НФ и не находится во 2НФ, так как атрибуты {Описание корпуса} и {Расположение корпуса на плане университета} зависят лишь от первичного атрибута {№ корпуса}, и не зависят от второго первичного атрибута {№ этажа}, то есть частично зависят от ключа {№ корпуса, № этажа}. Рассмотрим такие схемы.

Схема 3. Корпус {№ корпуса, Описание, Расположение на плане университета}

Данная схема находится во 2НФ, так как в ней присутствует один первичный атрибут, вследствие чего исключается возможность частичной зависимости непервичных атрибутов.

Схема 4. Данная схема находится во 2НФ.

Этаж корпуса {№ корпуса, № этажа, Описание, План, Расположение на планах смежных этажей}

Данная схема находится во 2НФ, так как из нее исключены атрибуты, нарушавшие условие 2НФ в схеме 2.

Схема 5. Аудитория {№ корпуса, № аудитории, № этажа, Расположение на плане этажа}

Данная схема находится во 2НФ.

Схема 6. ЭВМ {№ корпуса, № этажа, № аудитории, №ЭВМ, Изображение}

Данная схема находится во 2НФ.

Схема 7. Ресурс ЭВМ {№ корпуса, № этажа, № аудитории, №ЭВМ, Путь}

Схема 8. Пользователь {Код Пользователя, № корпуса, № этажа, № аудитории, №ЭВМ, ФИО, Титул, Описание, Изображение}

Данная схема находится во 2НФ.

Схема 9. Ресурс Пользователя {№ корпуса, № этажа, № аудитории, №ЭВМ, Сетевой Путь}

Данная схема находится во 2НФ.

Схема 10. Виртуальные Корпуса {№ виртуального корпуса, Описание}

Данная схема находится во 2НФ.

Схема 11. Аудитории Виртуальных Корпусов {№ виртуального корпуса, № корпуса, № этажа, № аудитории}

Данная схема находится во 2НФ.

Схема 12. Виртуальные аудитория {№ виртуальной аудитории, Описание}

Данная схема находится во 2НФ.

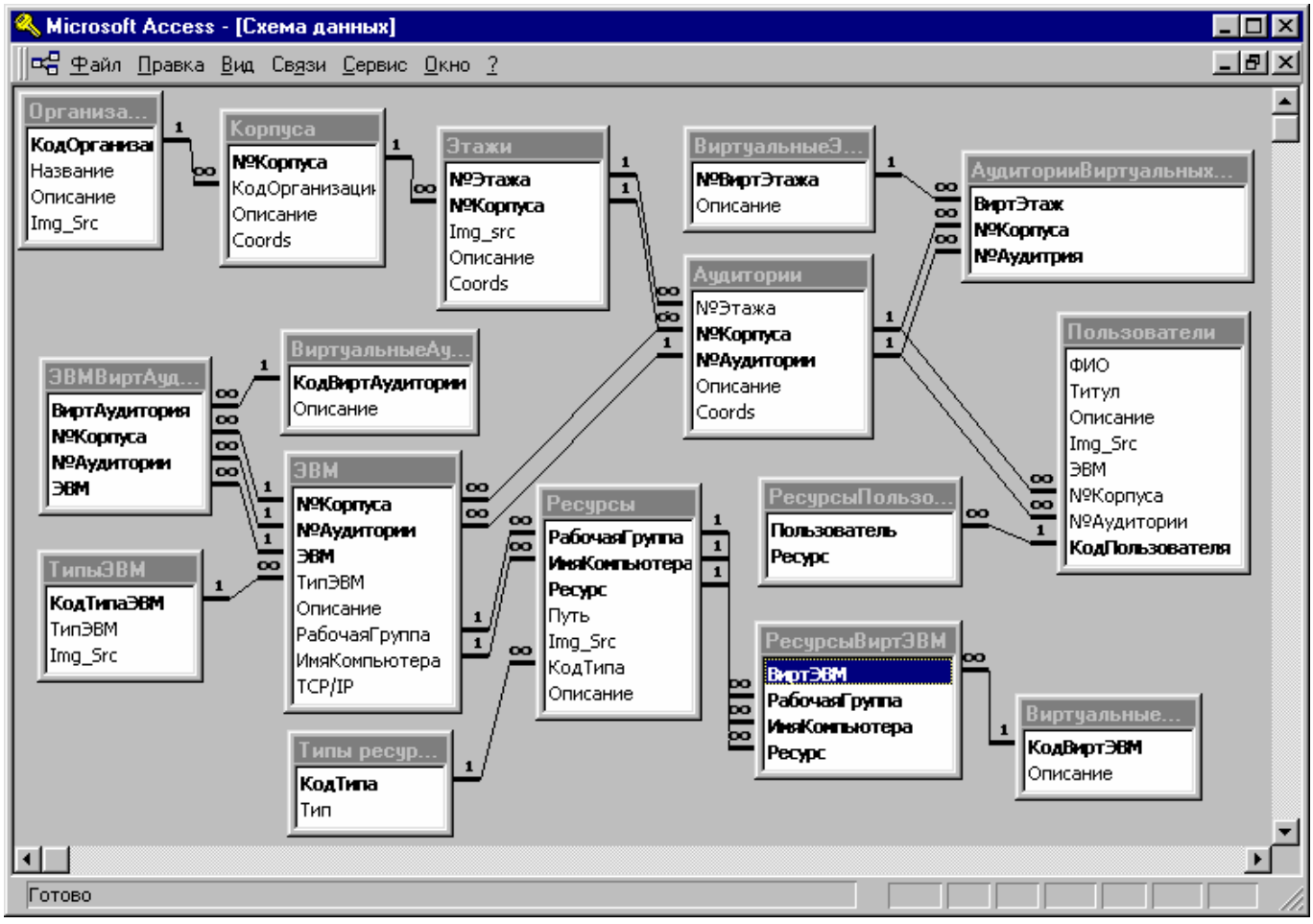

Рисунок 2.3. Схема базы данных Единого Информационного Пространства.

Схема 13. ЭВМ виртуальной аудитории {№ виртуальной аудитории, № корпуса, № этажа, № аудитории, №ЭВМ}

Данная схема находится во 2НФ.

Схема 14. Виртуальная ЭВМ {№ виртуальной ЭВМ, Описание}

Данная схема находится во 2НФ.

Схема 15. Ресурс виртуальной ЭВМ {№ виртуальной ЭВМ, Сетевой Путь}

Данная схема находится во 2НФ.

Таким образом, мы приходим к схеме БД, находящейся во второй нормальной форме (рисунок 2.3).

2.3 Определение процедур виртуального интерфейса и разработка их алгоритмов

Далее будут определены процедуры виртуального интерфейса между пользователем и Корпоративной ИС – Обработчик Запроса Пользователя и Обработчик Результата Запроса, описаны их входные/выходные данные, а также описаны алгоритмы работы.

2.3.1 Обработчик Запроса Пользователя

Запрос пользователя к ЕИП представляет собой выбор им определенного элемента интерфейса. Выше было сказано, что для доступа к определенному объекту необходимо наличие его идентификатора класса и идентификатора экземпляра класса. Поэтому входными параметрами Обработчика Запроса Пользователя можно считать ИК и ИЭК. Выходными параметрами ОЗП должен быть запрос к БД ЕИП. Так как для описания схемы БД мы выбрали реляционную модель, то запрос может быть представлен в виде SQL-запроса (Structured Query Language – структурированный язык запросов), так как язык SQL предназначен специально для работы с реляционными базами данных [11,12].

Из БД нам необходимо получить следующую информацию:

- данные, описывающие конкретный объект класса, определяемый ИК и ИЭК на входе ОЗП (эти данные существуют в одном экземпляре);
- данные, позволяющие установить связь между создаваемым документом и всеми документами, описывающими объекты ВУ, связанные с данным.

Каждый класс связан с несколькими другими классами (эта информация хранится в классовых данных). Кроме того, может существовать несколько конкретных объектов определенного класса, связанных с данным. Следовательно, количество записей, возвращенных в ответ на запрос к БД и описывающих связанные объекты, будет разным для каждого связанного класса. Поэтому для получения информации о каждом связанном классе целесообразно генерировать отдельный запрос.

Для установления связи с конкретным объектом необходимо наличие ИК и ИЭК этого объекта. Кроме этих данных нам необходимо получить ту информацию, на основе которой будет создан элемент навигации, представляющий собой элемент интерфейса, связанный с требуемым объектом.

Таким образом, мы приходим к следующим формальным SQL-запросам.

Запрос на получение данных, описывающих объект, передаваемый пользователю:

SELECT [$ИК_{НА\ ВХОДЕ}$.ДАННЫЕ ЭКЗЕМПЛЯРА КЛАССА]
FROM [$ИК_{НА\ ВХОДЕ}$]
WHERE [$ИК_{НА\ ВХОДЕ}$.ИЭК]=$ИЭК_{НА\ ВХОДЕ}$;

Здесь [$ИК_{НА\ ВХОДЕ}$] обозначает имя таблицы, в которой хранится информация об объекте, [ДАННЫЕ ЭКЗЕМПЛЯРА КЛАССА] представляют собой набор полей таблицы [$ИК_{НА\ ВХОДЕ}$], описывающих объекты данного класса, $ИК_{НА\ ВХОДЕ}$.ИЭК – поля, являющиеся ключевыми.

Запрос на получение данных, описывающих связанные объекты:

SELECT [$ИК_{СВЯЗАННЫЙ\_ОБЪЕКТ}$.ИЭК], [$ИК_{СВЯЗАННЫЙ\_ОБЪЕКТ}$.Элемент навигации]
FROM

[ИК$_{\text{НА ВХОДЕ}}$] INNER JOIN [ИК$_{\text{СВЯЗАННЫЙ ОБЪЕКТ}}$] ON

[ИК$_{\text{НА ВХОДЕ}}$.ИЭК$_{\text{НА ВХОДЕ}}$] = [ИК$_{\text{СВЯЗАННЫЙ ОБЪЕКТ}}$.ИЭК$_{\text{НА ВХОДЕ}}$]

WHERE [ИК$_{\text{НА ВХОДЕ}}$.ИЭК]=ИЭК$_{\text{НА ВХОДЕ}}$;

Так как некоторые объекты интерфейса могут быть описаны с помощью различного количества таблиц и для выбора необходимой информации будут применяться различные условия, то структура конкретных запросов будет несколько отличаться от описанных выше схем. Поэтому для каждого класса необходимо хранить свой набор SQL-запросов. Кроме того, необходимо хранить набор таблиц, из которых извлекается информация по каждому запрашиваемому элементу интерфейса, с описанием структуры каждой таблицы, то есть конкретным набором требуемых полей (атрибутов).

Пример реализации набора SQL-запросов будет дан в 3 главе.

Итак, алгоритм работы ОЗП можно представить следующим образом:

На входе: ИК и ИЭК запрашиваемого объекта.

На выходе: набор SQL-запросов.

а) Определить класс, к которому принадлежит требуемый элемент интерфейса по ИК$_{\text{НА ВХОДЕ}}$.

б) Определить ИК связанных объектов.

в) Сгенерировать SQL-запросы, соответствующие данному классу, используя информацию в соответствующих таблицах, применяя в качестве условия отбора информации значение ИЭК$_{\text{НА ВХОДЕ.}}$

г) Конец алгоритма.

2.3.2 Обработчика Результата Запроса

Входной информацией ОРЗ является набор таблиц данных, сгенерированный в ответ на запрос к БД. Структура этого набора определяется исходя из шаблона запрашиваемого объекта – данные экземпляра класса плюс данные по каждому связанному классу. Структура каждой конкретной таблицы определяется исходя из

шаблонных таблиц, используемых при создании SQL-запроса перечислением всех полей.

Выходной информацией является готовая HTML-страница, передаваемая пользователю. Шаблон этой страницы является статическим документом. Для преобразования шаблона в готовый документ в его исходном тексте необходимо наличие управляющих элементов, позволяющих заполнить его необходимой информацией. Для создания этих управляющих элементов воспользуемся таговой моделью, описанной в разделе 1.1.2.

"ЭЛЕМЕНТ" := <"ИМЯ ЭЛЕМЕНТА" "СПИСОК АТРИБУТОВ">

СОДЕРЖАНИЕ ЭЛЕМЕНТА

</"ИМЯ ЭЛЕМЕНТА">

Введем следующий управляющий элемент (таг):

<ЗАПРОС ИМЯ> </ЗАПРОС>

При использовании этого тага его содержание будет представлять из себя следующее:

<ЗАПРОС ИМЯ=имя запроса>

<!-- элементы шаблона -->…<!-- элементы шаблона -->

%ПОЛЕ_1%

<!-- элементы шаблона -->…<!-- элементы шаблона -->

…

<!-- элементы шаблона -->…<!-- элементы шаблона -->

%ПОЛЕ_N%

<!-- элементы шаблона -->…<!-- элементы шаблона -->

</ЗАПРОС>

Здесь ИМЯ определяет таблицу, из которой должна извлекаться информация, %ПОЛЕ_N% - поле данной таблицы. Если данная таблица имеет несколько записей, Блок <ЗАПРОС>…</ЗАПРОС> дублируется в тексте HTML-докуменгта в соответствии с количеством записей, возвращенных в ответ на данный запрос.

Таким образом, алгоритм работы ОРЗ можно описать следующим образом.

На входе: ИК создаваемого объекта, набор таблиц.

На выходе: HTML-документ.

а) Определить структуру входных таблиц на основании шаблонов таблиц для построения SQL-запросов, соответствующих объекту с ИК=$ИК_{НА\ ВХОДЕ.}$

б) Выбрать первую/следующую таблицу.

в) Если таблица существует, то перейти к г), иначе перейти к е).

г) Выбрать первую/следующую запись.

д) Если запись не пуста, то заменить элементы %ПОЛЕ_N% блока <ЗАПРОС>…</ЗАПРОС> значениями соответствующих полей и перейти к г), иначе перейти к б).

е) Вернуть пользователю HTML-страницу.

ж) Конец алгоритма.

3 ПРАКТИЧЕСКАЯ РЕАЛИЗАЦИЯ ЭЛЕМЕНТОВ ВУ И МОДЕЛИРУЮЩИХ ЗАПРОСОВ

3.1 Элемент интерфейса ВУ.

Рассмотрим в качестве примера HTML-документ, описывающий первый этаж

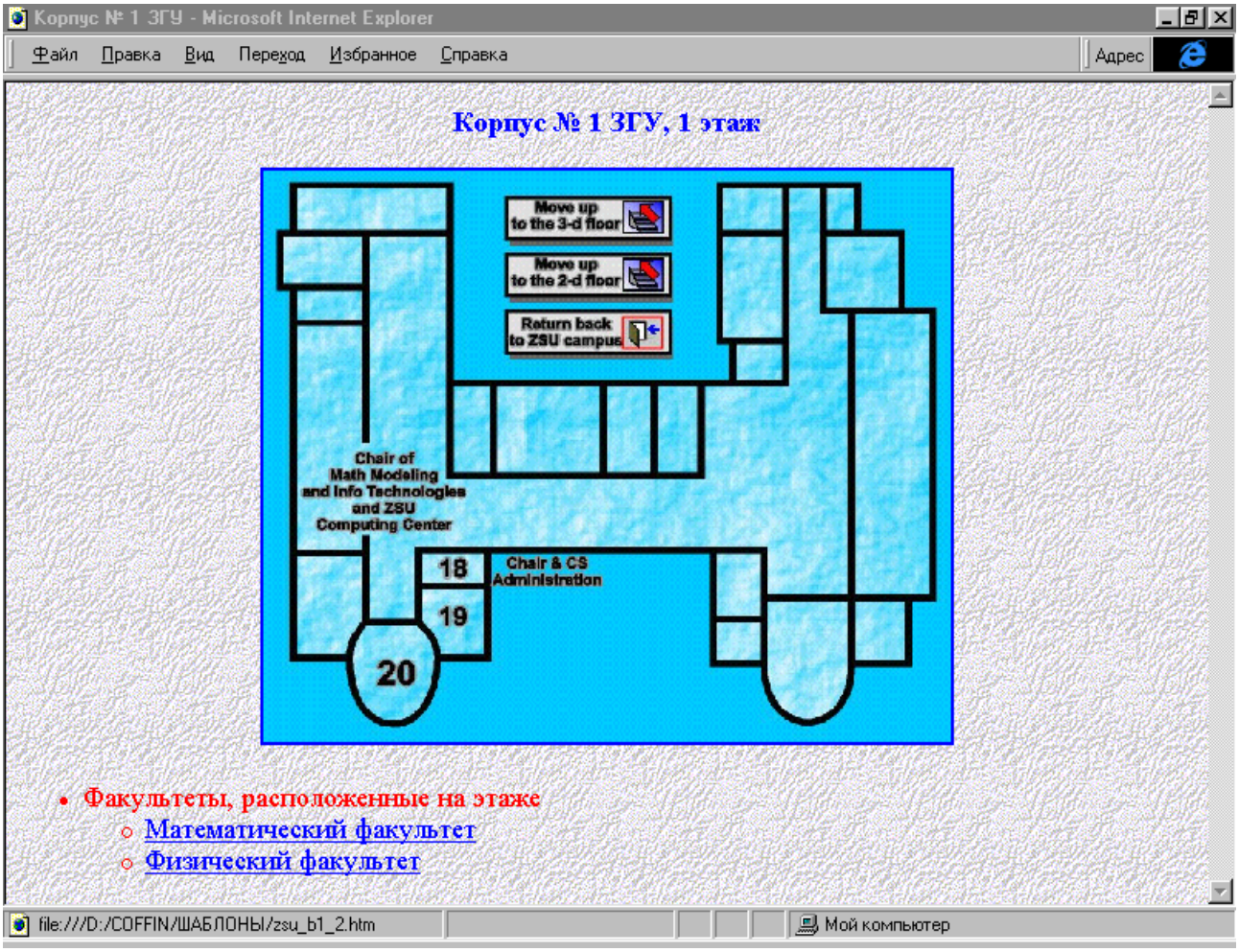

Рисунок 3.1. Первый этаж первого корпуса ЗГУ.

первого корпуса ЗГУ (рисунок 3.1). В иерархии объектов ВУ ему соответствует «ЭТАЖ КОРПУСА». Рассмотрим его элементы параллельно с соответствующими им HTML-кодами.

План этажа представляет собой рисунок в формате JPEG. Для его внедрения в текст HTML-документа используется следующий таговый элемент:

<img src="имя_файла" height="высота" width="ширина">

где "имя_файла" – имя файла в формате JPEG или GIF, "высота" и "ширина" – размеры изображения в пикселах по вертикали и горизонтали соответственно.

Для связи с другими этажами данного корпуса (смежными), перехода к плану всего университета, а также длясвязи с аудиториями, находящимися на этаже используются «активные зоны» - определенные участки самого рисунка, организованные с помощью следующей конструкции:

<map name ="имя_карты">

<area shape ="rect" coords ="x1,y1,x2,y2" href =URL>

</map>

Здесь атрибут name определяет имя создаваемой карты и используется в дальнейшем для наложения на рисунок. Элемент area определяет конкретную «активную зону», форма которой задается с помощью атрибута shape, а координаты – с помощью атрибута coords. Атрибут href указывает на URL-адрес связанного документа. При вставке в документ рисунка на него накладывается карта следующим образом:

<img usemap=”#имя_карты” src="ZSU_1-1.jpg" height="563" width="646">

то есть ко всем выше перечисленным атрибутам добавляется атрибут usemap с именем соответствующей карты.

Для связи с факультетами, аудитории которых находятся на данном этаже (факультет представлен объединением объектов «ЭТАЖ ВИРТУАЛЬНОГО КОРПУСА»), организовывается ненумерованный список:

<UL><LI>Название списка

<UL>

<LI>Первый элемент списка

…

</UL></UL>

Таким образом, для динамического создания данного документа должна быть получена следующая информация:

- имя файла, содержащего рисунок с планом этажа, размеры рисунка;
- координаты, составляющие карту, накладываемую на рисунок, соответствующие выходу из корпуса, смежным этажам и аудиториям;
- названия факультетов;
- первичные ключи соответствующих записей в БД для организации ссылок.

3.2 Шаблон элемента интерфейса ВУ.

Пользуясь разработанной ранее схемой, опишем шаблон данного документа.

Во-первых, вся интересующая нас информация будет храниться в следующих таблицах:

Этаж корпуса {№ корпуса, № этажа, Описание, План, Расположение на планах смежных этажей}

Аудитория {№ корпуса, № аудитории, № этажа, Расположение на плане этажа}

В данном случае поле “План” содержит имя файла с изображением плана, а поля “Расположение на планах смежных этажей” и “Расположение на плане этажа”– координаты в пикселах относительно верхнего левого угла изображения. Активная зона, соответствующая выходу из здания также может быть построена по данным поля “Расположение на планах смежных этажей”, так как нет необходимости переходить с текущего этажа на текущий этаж и эти данные не используются.

Для определения информации о факультетах необходимы еще две таблицы - Виртуальные Корпуса {№ виртуального корпуса, Описание} и Аудитории Виртуальных Корпусов {№ виртуального корпуса, № корпуса, № этажа, № аудитории}

Вместо URL-адресов связанных HTML-страниц будет использоваться имя программы (CGI-скрипта, условно назовем его cgi) с передачей ему необходимых параметров - ИК и ИЭК создаваемого объекта. В качестве последних будут использоваться имена и значения первичных атрибутов соответствующих таблиц. Параметры передаются по методу POST.

Готовый шаблон будет выглядеть следующим образом:

```
<html>
<head> <title>Корпус №
<ЗАПРОС ИМЯ=”Этаж корпуса”> %№корпуса% </ЗАПРОС>
ЗГУ</title> </head>
<body background="DC3.gif">
<p align="center">
<font color="#0000FF"><b><big>
```

```
<ЗАПРОС ИМЯ=”Этаж корпуса”>
Корпус № %№корпуса% ЗГУ, %№корпуса% этаж
</ЗАПРОС></big></b></font></p>
<map name ="План">
<ЗАПРОС ИМЯ=”Университет”>
<area shape ="rect" coords ="%coords%" href =../cgi?ИК=УНИВЕРСИТЕТ>
</ЗАПРОС>
<ЗАПРОС ИМЯ=”Этажи корпусов”>
<area shape=”rect” coords="%coords%"
href../cgi?ИК=ЭТАЖ&КОРПУСА&№КОРПУСА=%№КОРПУСА%&№ЭТАЖА=%
№ЭТАЖА%>
</ЗАПРОС>
<ЗАПРОС ИМЯ=”Аудитории”>
<area shape ="rect" coords="%coords%"
href../cgi?ИК=АУДИТОРИЯ&№КОРПУСА=%№КОРПУСА%&№ЭТАЖА=%№ЭТА
ЖА%&№АУДИТОРИИ=%№АУДИТОРИИ%>
</ЗАПРОС></map>
<p align="center">
<ЗАПРОС ИМЯ=”Этаж корпуса”>
<img usemap="#План" src="%План%" height="%height%" width="%width%">
</ЗАПРОС></p>
<font color="#ff0000" size=4>
<UL><LI>Факультеты, расположенные на этаже
<UL>
<ЗАПРОС ИМЯ=”Виртуальные корпуса”>
<LI><a href../cgi?ИК=ЭТАЖ&ВИРТ.КОРПУСА&
№ВИРТ.КОРПУСА=%№ ВИРТ. КОРПУСА%&№КОРПУСА=%№КОРПУСА%&
№ЭТАЖА=%№ЭТАЖА%>%Описание%</a>
</ЗАПРОС>
</UL></UL></font></body></html>
```

3.3 Пример построения набора SQL-запросов.

Опишем набор SQL-запросов, необходимый для создания описанной выше HTML-страницы. Входными параметрами для ОЗП в данном случае буде следующая информация: ИК - ЭТАЖ КОРПУСА, ИЭК - № корпуса=1, № этажа=1.

Запрос 1. Этаж корпуса (описание текущего этажа).

Набор полей – Этажи {№Корпуса, №Этажа, Img_src}

```
SELECT Этажи.[№Корпуса], Этажи.[№Этажа], Этажи.Img_src
FROM Этажи
WHERE (((Этажи.[№Корпуса])=[№ корпуса:]) AND
((Этажи.[№Этажа])=[№ этажа:]));
```

Запрос 2. Университет.

Набор полей – Этажи {№Корпуса, №Этажа, Coords}

```
SELECT Этажи.[№Корпуса], Этажи.[№Этажа], Этажи.Coords
FROM Этажи
WHERE (((Этажи.[№Корпуса])=[№ корпуса:]) AND ((Этажи.[№Этажа])=[№
этажа:]));
```

Запрос 3. Этажи корпусов (смежные этажи).

Набор полей – Этажи {№Корпуса, №Этажа, Coords}

```
SELECT Этажи.[№Корпуса], Этажи.[№Этажа], Этажи.Coords
FROM Этажи
WHERE (((Этажи.[№Корпуса])=[№ корпуса:]) AND ((Этажи.[№Этажа])<>[№
этажа:]));
```

Запрос 4. Аудитории.

Набор полей - Аудитории {№Корпуса, № Аудитории, Coords}

```
SELECT Аудитории.[№Корпуса], Аудитории.[№Аудитории], Аудитории.Coords
FROM Аудитории
WHERE (((Аудитории.[№Корпуса])=[№ корпуса:]) AND
((Аудитории.[№Этажа])=[№ этажа:]));
```

Запрос 5. Этажи виртуальных корпусов (факультеты).

Набор полей - Этажи {№Корпуса, №Этажа},
Этажи Виртуальных Корпусов {№Вирт.Этажа, Описание}

```
SELECT DISTINCT Этажи.[№Корпуса], Этажи.[№Этажа],
ВиртуальныеЭтажи.[№ВиртЭтажа], ВиртуальныеЭтажи.Описание
FROM ВиртуальныеЭтажи INNER JOIN (Этажи INNER JOIN (Аудитории INNER
JOIN АудиторииВиртуальныхЭтажей ON (Аудитории.[№Аудитории] =
АудиторииВиртуальныхЭтажей.[№Аудитрия]) AND (Аудитории.[№Корпуса] =
АудиторииВиртуальныхЭтажей.[№Корпуса])) ON (Этажи.[№Этажа] =
Аудитории.[№Этажа]) AND (Этажи.[№Корпуса] = Аудитории.[№Корпуса])) ON
ВиртуальныеЭтажи.[№ВиртЭтажа] = АудиторииВиртуальныхЭтажей.ВиртЭтаж
WHERE (((Этажи.[№Корпуса])=[№ корпуса:]) AND ((Этажи.[№Этажа])=[№
этажа:]));
```

Результаты, возвращаемые в ответ на данные запросы:

Таблица 3.1 Этаж корпуса

Корпус	№Этажа	Height	Width	Img_src
1	1	563	646	1_1.jpg

Таблица 3.2 Университет

Корпус	№Этажа	Coords
1	1	11,11,11,11

Таблица 3.3 Этажи корпуса

Корпус	№Этажа	Coords
1	2	12,12,12,12
1	3	13,13,13,13

Таблица 3.4 Аудитории

№Корпуса	№Аудитории	Coords
1	13	13,13,13,13
1	18	18,18,18,18
1	19	19,19,19,19
1	20	20,20,20,20

Таблица 3.5 Виртуальные корпуса

Корпус	№Этажа	№ВиртЭтажа	Описание
1	1	1	Математический факультет
1	1	2	Физический факультет

Готовый HTML-код будет выглядеть следующим образом:

```
<html> <head> <title>Корпус № 1 ЗГУ</title> </head>
<body background="DC3.gif"><p align="center">
<font color="#0000FF"><b><big> Корпус № 1, 1 этаж </big></b></font></p>
<map name ="План">
<area shape ="rect" coords ="11,11,11,11" href =../cgi?ИК=УНИВЕРСИТЕТ>
<area shape=”rect” coords="12,12,12,12"
href../cgi?ИК=ЭТАЖ&КОРПУСА&№КОРПУСА=1&№ЭТАЖА=2>
<area shape=”rect” coords="13,13,13,13"
href../cgi?ИК=ЭТАЖ&КОРПУСА&№КОРПУСА=1&№ЭТАЖА=3>
<area shape ="rect" coords="13,13,13,13"
href../cgi?ИК=АУДИТОРИЯ&№КОРПУСА=1&№ЭТАЖА=1&№АУДИТОРИИ=13>
<area shape ="rect" coords="18,18,18,18"
href../cgi?ИК=АУДИТОРИЯ&№КОРПУСА=1&№ЭТАЖА=1&№АУДИТОРИИ=18>
<area shape ="rect" coords="19,19,19,19"
href../cgi?ИК=АУДИТОРИЯ&№КОРПУСА=1&№ЭТАЖА=1&№АУДИТОРИИ=19>
<area shape ="rect" coords="20,20,20,20"
href../cgi?ИК=АУДИТОРИЯ&№КОРПУСА=1&№ЭТАЖА=1&№АУДИТОРИИ=20>
</map>
<p align="center"><img usemap="#План" src="1_1.jpg" height="563" width="646">
</p><font color="#ff0000" size=4>
<UL><LI>Факультеты, расположенные на этаже<UL>
<LI><a href../cgi?ИК=ЭТАЖ&ВИРТ.КОРПУСА&
№ВИРТ.КОРПУСА=1&№КОРПУСА=1&№ЭТАЖА=1>
Математический факультет</a>
<LI><a href../cgi?ИК=ЭТАЖ&ВИРТ.КОРПУСА&
```

№ВИРТ.КОРПУСА=2&№КОРПУСА=1&№ЭТАЖА=1>
Физический факультет
</UL></UL></body></html>

Другие примеры HTML-документов, их шаблоны и соответствующие SQL-запросы даны в приложениях 2,3,4.

ВЫВОДЫ

В процессе дипломной работы были решены следующие задачи и получены следующие результаты:

- исследованы существующие подходы к решению задачи организации среды Intranet для корпоративных информационных сетей. В результате анализа существующих подходов сделан вывод о преимуществах методики динамического формирования элементов модели информационной среды ВУЗа на базе запросов к базе данных единого информационного пространства;
- рассмотрен проект создания Виртуального Университета - среды Intranet для информационного окружения Запорожского государственного университета, для реализации части которого определен базовый набор элементов визуального интерфейса для ВУ;
- создана логическая реляционная модель Элементов виртуального интерфейса единого информационного пространства ЗГУ, разработанная модель нормализована до второй нормальной формы;
- разработаны алгоритмы: преобразования HTML - запроса пользователя к базе данных единого информационного пространства в запрос на языке SQL и преобразования результатов SQL - запроса к базе данных единого информационного пространства в динамическую HTML - модель элемента Виртуального Университета.

СПИСОК ССЫЛОК

1. V. A. Tolok, S. U. Borue, V. A. Ermolayev, A. I. Kubushkaites, Development of the Consept and Implementation of the First Line of the Integrated Network at ZSU, Research Work Intermediate Report, State Reg. No 0197y012776, Ministry of Education of Ukraine, Zaporozhye State Univ., Zaporozhye, 1997, 28 p.

2. Chuck Musciano & Bill Kennedy, HTML: The Definitive Guide, pp. 126-157

3. Г.Шилдт, Программирование на C и C++ для Windows 95 - К.: Торгово-издательское бюро BHV, 1996 - 400 с.: ил.

4. Rosenthal A. and Reiner D. Tools and Transformations - Rigorous and Otherwise - for Practical Database Design, ACM TODS, Vol. 19, No 2 (June 1994), pp. 167-211

5. Fraternali P., Tanca L.. A Structured Approach for the Definition of the Semantics of Active Databases, ACM TODS, Vol. 20, No 4 (Dec. 1995), pp. 414-471

6. Aiken A., Hellerstein J. M., Widom J.. Static Analysis Techniques for Predicting the Behavior of Active Database Rules, ACM TODS, Vol. 20, No 1 (Mar. 1995), pp. 3-41

7. Филобок А. П., Ермолаев В.А. Проект модели данных для информационной системы “АБИТУРИЕНТ”. Тезисы сообщений научной конференции студентов. Том 6, часть 1, Запорожский государственный университет, Запорожье, Украина, 1996.

8. V. A. Ermolayev, Object Oriented Dynamic Data Modelling and Active Data Dictionaries - Some Crosspoints . - to appeare in "Journal of Metrology and Certification" Vol 1, No 1, (Jul.-Dec. 1997)

9. V. A. Ermolayev, Visual Intranet Interfaces and Architecture of Unified Information Space in the Concept of Virtual University at ZSU, …

10. I. Schmitt, G. Saake, Merging Inheritance Hierarches for Schema Integration Based on Concept Lattices, Fakultät für Informatik, Otto-von-Guericke-Universität, Magdeburg, Preprint Nr. 2, 1997

11. Мейер Д. Теория реляционных баз данных. -М.: Мир,-1987.-608 с.

12. Д.Вейскас, Эффективная работа в Microsoft Access 2.0, -М.: Мир,-1992.-432 с.