# ВВЕДЕНИЕ

Потребность выразить и передать информацию привела к появлению речи, письменности, книгопечатания, почтовой связи, телеграфа, телефона, радио, телевидения и многого другого.

С самого раннего детства все мы вовлечены в процессы обмена информацией. Вопросы, ответы, просьбы – все это передача информации. Мы получаем информацию, когда читаем книги, газеты и журналы, слушаем радио или смотрим телевизор.

Общение людей друг с другом – это передача информации: сведений и суждений, данных и сообщений. Любая совместная деятельность людей – работа, учеба – невозможна без обмена информацией. Передаваемая информация обычно касается каких-то предметов или нас самих и связана с событиями, которые происходят в окружающем нас мире.

Выражение мыслей в письменной форме открыло возможность не только передавать сведения и сообщения, но и накапливать человеческие знания в форме рукописей и рукописных книг и тем самым передавать сокровища человеческой мысли от одного поколения к другому.

Совершенно новые возможности для поиска и обработки информации открыло перед людьми изобретение в середине XX века электронных вычислительных машин – ЭВМ. Первоначально ЭВМ создавались для автоматизации вычислений. Затем их научили записывать и хранить информацию на магнитных лентах, печатать ее на бумаге и выводить на экран ЭВМ. По мере развития они стали использоваться для создания архивов, подготовки и редактирования текстов, выполнения чертежных и графических работ, для автоматизации производства и многих других видов человеческой деятельности.

Вычислительная техника все более широко используется не только для автоматизации производства, но и в самых различных его сферах. Подобное вовлечение вычислительной и микроэлектронной техники в деятельность различных производственных систем называется компьютеризацией производства.

Компьютеризация – это основа технического перевооружения производства, необходимое условие повышения его эффективности. На базе ЭВМ и микропроцессоров создаются технологические комплексы, машины и оборудование, измерительные, регулирующие и информационные системы, ведутся проектно-конструкторские работы и научные исследования, осуществляется информационное обслуживание, обучение и многое другое, что обеспечивает повышение общественной и индивидуальной производительности труда, создание условий для всестороннего и гармоничного развития личности.

Для нормального развития и функционирования сложного народнохозяйственного механизма необходимы постоянный обмен информацией между его звеньями, своевременная обработка большого объема данных на различных уровнях управления, что также невозможно без использования ЭВМ. Поэтому от уровня компьютеризации в значительной степени зависит развитие экономики.

В процессе своего развития ЭВМ прошли путь от громоздких машин на электронных лампах, общение с которыми было возможно только на машинном языке, до современных ЭВМ.

Развитие ЭВМ происходит в двух основных направлениях. Первое – создание мощных многопроцессорных вычислительных систем с производительностью сотни и тысячи миллионов операций в секунду. Второе – создание дешевых и компактных микроЭВМ на базе микропроцессоров. В рамках второго направления развивается производство персональных компьютеров, которые становятся мощным универсальным инструментом, существенно повышающим производительность интеллектуального труда специалистов различного профиля. Персональные компьютеры отличает работа в диалоговом режиме с индивидуальным пользователем; небольшие размеры и автономность функционирования; аппаратные средства на базе микропроцессорной техники; универсальность, обеспечивающая ориентацию на широкий круг задач, решаемых одним пользователем с помощью технических и программных средств.

Следует отметить и такой важный факт компьютеризации производства, как широкое распространение собственно микропроцессоров, каждый из которых ориентирован на выполнение одной или нескольких специальных задач. Встраивание таких микропроцессоров в узлы промышленного оборудования позволяет обеспечить решение поставленных задач с минимальными затратами и в оптимальном виде. Использование микропроцессорной техники для сбора информации, регистрации данных или локального управления значительно расширяет функциональные возможности оборудования.

Развитие компьютеризации вызывает потребность в разработке и создании новых средств вычислительной техники. Их характерными особенностями являются: формирование элементной базы на сверхбольших интегральных схемах; обеспечение производи-тельности до 10 миллиардов операций в секунду; наличие искусственного интеллекта, что значительно расширяет возможности ЭВМ в обработке поступающей информации; возможность общения человека с ЭВМ на естественном языке путем речевого и графического обмена информацией.

В ходе развития компьютеризации – создание национальных и межнациональных коммуникационно-вычислительных сетей, баз данных, нового поколения спутниковых систем космической связи, что позволит облегчить доступ к информационным ресурсам.

Важнейшим фактором прогресса является совершенствование форм и методов управления предприятием на основе вычислительной техники и средств связи, которые представляют собой материально техническую базу информационной системы (ИС) предприятия. Под информационной системой понимается комплекс технических и программных средств, выполняющих автоматизированную обработку данных в процессе управления предприятием. Она служит для связи между объектами и субъектами управления.

Основное преимущество автоматизации для данной отрасли заключается в предоставлении читателю более полной и достоверной информации о состоянии библиотечного фонда, возможности быстрой и эффективной ориентации в довольно больших объемах информации. В настоящее время основным направлением развития автоматизации всех библиотек является разработка каталога библиографических описаний. Перспективным направлением - переход к электронным библиотекам. Как правило, внедрение любой автоматизированной системы влечет за собой изменение существующих организационных структур и методов управления, требует более четкой регламентации документооборота и процедур управления, упорядочения используемых и создание новых нормативов, совершенствование условий труда.

Работа информационной системы «Библиотека» реализует ввод информации; ведение баз данных; контроль за своевременным возвратом выданной литературы; учет движения читательского фонда.

Характерными особенностями технологии обработки данных на ПЭВМ, в том числе и в разработанной информационной системе «Библиотека», являются:

* работа пользователя в режиме манипулирования данными. Пользователь должен «видеть» и «действовать», а не «знать» и «помнить».
* сквозная информационная поддержка на всех этапах прохождения информации на основе распределенных баз данных.
* безбумажный процесс отработки документа.
* интерактивный режим решения задач с широкими возможностя-ми для пользователя.
* возможность коллективного оформления документов на группе взаимосвязанных средствами коммуникаций ПЭВМ.

Использование такой технологии снижает в несколько раз затраты на документооборот, повышает скорость и качество обслуживания, упорядочивает организационную структуру документооборота, и в результате повышает эффективность управления.

Библиотеку давно перестали рассматривать как старинное здание, в котором пылятся книжки. Социологический опрос молодежи показал, что за сохранение традиционных функций библиотеки выступает 25% читателей, а 75% хотят работать с компьютером, смотреть видеофильмы, использовать в поиске Internet, а ведь это мнение самой важной части нашего общества. Следовательно, действительно назрела необходимость изменения роли библиотеки, она должна превратиться в центр информационного и коммуникативного обеспечения людей.

Библиотека должна стать не только источником идей, мыслей, технологий в виде журналов, диссертаций, книг, каталогов и др., но и автоматизированным информационным центром, обслуживающим пользователей в локальном и сетевом режимах, центром внедрения новых информационных технологий, создателем собственных баз данных и комплексов автоматизированных услуг благодаря разработке и внедрению библиотечных информационных систем.

## 1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

## Инструментальная поддержка процесса проектирования программного обеспечения

В качестве предметной области автоматизации рассматривается автоматизация рабочего места сотрудников библиотеки.

Основное преимущество автоматизации для данной отрасли заключается в предоставлении читателю более полной и достоверной информации о состоянии библиотечного фонда, возможности быстрой и эффективной ориентации в довольно больших объемах информации. В настоящее время основным направлением развития автоматизации всех библиотек является разработка каталога библиографических описаний.

Как правило, внедрение любой автоматизированной системы влечет за собой изменение существующих организационных структур и методов управления, требует более четкой регламентации документооборота и процедур управления, упорядочения используемых и создание новых нормативов, совершенствование условий труда.

Представленная в данной работе автоматизированная информационная библиотечная система разработана с целью обеспечения информативности и ускорения ориентации в объемах библиотечного фонда.

Информационную систему определяют, как систему информационных, математических, программных, языковых, организационных и технических средств, предназначенных для централизованного накопления и коллективного многоаспектного использования данных для получения необходимой информации.

Данные – информация, представленная в определенной форме, пригодной для последующей обработки, хранения и передачи.

Структура данных – способ объединения нескольких элементов данных в один.

Предметная область – часть реального мира, подлежащая изучению с целью организации управления и последующей автоматизации.

Предметная область определена, если известны существующие в ней объекты, их свойства и отношения.

Модель данных – представление о предметной области в виде данных и связей между ними.

Выбор модели данных зависит от объема информации, сложности решаемых задач и имеющегося технического и программного обеспечения.

Автоматизация технологического процесса – совокупность методов и средств, предназначенная для реализации системы или систем, позволяющих осуществлять управление самим технологическим процессом без непосредственного участия человека, либо оставления за человеком права принятия наиболее ответственных решений.

Инсталляция – процесс установки программного обеспечения на компьютер конечного пользователя.

Веб-сервер – это сервер, принимающий HTTP-запросы от клиентов, обычно веб-браузеров, и выдающий им HTTP-ответы, обычно вместе с HTML-страницей, изображением, файлом, медиа-потоком или другими данными.

База данных (БД) – совокупность данных конкретной предметной области. Они организованы по определенным правилам, предусматривающим общие принципы описания, хранения и манипулирования, и не зависят от программ обработки.

Система управления базами данных (СУБД) – набор программных средств (программная система или пакет), обеспечивающих создание и обслуживание баз данных и выполнение операций над данными базами данных (доступ к ним и обработку).

### 1.2 Информационная система

Информационная система (ИС) представляет собой коммуникационную систему по сбору, передаче и обработке информации о заданной предметной области, снабжающую всех своих пользователей необходимой информацией [2].

Основными компонентами ИС являются:

* собственно база данных (БД), содержащая необходимую информацию и описание структуры хранимых данных;
* система управления базой данных (СУБД), выполняющая типовые процедуры управления данными;
* прикладная программа (приложение пользователя), реализующая требуемый алгоритм ведения диалога пользователя с информационной системой для обслуживания базы данных и решения всего комплекса задач обработки данных.

Автоматизация предметной области может осуществляться в различных масштабах – от отдельных процедур обработки данных, задач и комплексов задач управления до создания функционально полной автоматизированной информационной системы (АИС).

Процессу проектирования автоматизированной информационной системы предшествует этап анализа предметной области. На этом этапе осуществляется:

* определение, исходя из потребностей всех пользователей, информации, которая будет храниться в базе данных, ее источника;
* выявление имеющихся и перспективных задач обработки данных;
* документирование результатов анализа.

Следовательно, процесс построения (разработки) информационно программного комплекса автоматизированной информационной системы можно разбить на две части:

* определение состава решаемых задач;
* создание интегрированной базы данных.

Эти две части взаимосвязаны в том смысле, что структура базы данных зависит от задач обработки данных заданной предметной области.

Таким образом, база данных – это некоторая модель предметной области, т.е. в базе данных находят отражение только те факты о предметной области, которые необходимы для функционирования информационной системы.

Этап разработки базы данных предшествует этапу разработки приложения пользователя или проходит параллельно с ним; при этом фазы проектирования и реализации могут перекрываться.

Если база данных спроектирована, то алгоритм обработки данных конкретной задачи отражает процесс формирования выходных документов на основании выборок из базы данных.

Реализация конкретной задачи осуществляется с помощью средств СУБД, ориентированных на конечного пользователя: запросы, экранные формы, отчеты, макросы, стандартные программы.

При разработке алгоритмов решения каждой задачи следует ориентироваться на укрупненные операции обработки данных, легко реализуемые запросами или отчетами. Диалоговое приложение пользователя (прикладная программа) объединяет всю технологию обработки данных предметной области, включая загрузку, ведение базы данных и решение всего комплекса задач.

Программа должна иметь «дружественный» интерфейс и содержать набор команд решения конкретных задач:

* создание нового набора (таблицы, списка) данных;
* добавление, удаление, изменение, просмотр данных;
* сортировка данных;
* выбор данных по запросам;
* формирование выходных документов (с возможностью просмотра на экране и вывода на печать).

Интерфейс пользователя должен строиться на основе иерархических меню с использованием диалоговых окон, кнопок, «горячих клавиш» и т.п.

При реализации информационной системы следует использовать инструментальные средства, предоставляемые системой управления базами данных:

* генераторы (конструкторы, мастера) экранных форм и отчетов для создания диалоговых окон для ввода данных и выходных (отчетных) документов;
* языки запросов для поиска и фильтрации данных;
* языки программирования для реализации алгоритмов обработки информации.

Выбор того или иного языка программирования для реализации модулей, реализующих сетевые решение, чаще всего напрямую зависит от выбранной технологии. Но и здесь есть исключения.

* 1. **Среда разработки Microsoft Visual Studio**

VisualStudio 2008 представляет собой полный набор средств, помогающих ускорить процесс реализации замысла разработчика. Это решение было создано чтобы обеспечить поддержку проектов создания программного обеспечения для Интернета (включая ASP.NET AJAX), WindowsVista, WindowsServer 2008, выпуска 2007 системы MicrosoftOffice, SQL Server 2008 и устройств под управлением WindowsMobile. Число платформ, на которые должны ориентироваться разработчики в соответствии с бизнес-требованиями, быстро увеличивается. VisualStudio 2008 ProfessionalEdition предоставляет интегрированный набор средств, позволяющих учесть все эти требования путем расширения функциональности, доступной в VisualStudio 2008 StandardEdition.

Современным разработчикам приходится ориентироваться на широкий спектр платформ, создавая приложения, позволяющие организациям быстро получать ожидаемый результат. Встроенные в VisualStudio конструкторы и возможности языков программирования позволяют создавать приложения, способные связываться с удаленными базами данных и отвечающие чаяниям сегодняшнего бизнеса, а использование преимуществ среды .NET Framework 3.5 помогает сократить время разработки. Из возможных языков программирования, предоставляемых VisualStudio 2008, для разработки визуальной оболочки был выбран C#.

C# элегантный, строго типизированный объектно-ориентированный язык, предназначенный для разработки разнообразных безопасных и мощных приложений, выполняемых в среде .NET Framework. На языке C# можно разрабатывать обычные клиентские приложения Windows, веб-службы XML, распределенные компоненты, приложения типа “сервер-клиент”, приложения баз данных и многие-многие другие. В Visual C# 2008 имеется расширенный редактор кода, конструкторы с удобным пользовательским интерфейсом, встроенный отладчик и многие другие средства, призванные упростить разработку приложений на языке C# версии 3.0 и .NET Framework версии 3.5.

Синтаксис C# очень выразителен, но прост в изучении. Все, кто знаком с языками C, C++ или Java с легкостью узнают синтаксис с фигурными скобками, характерный для языка C#. Разработчики, знающие любой из этих языков, как правило, смогут добиться эффективной работы с языком C# за очень короткое время. Синтаксис C# делает проще то, что было сложно в C++, и обеспечивает мощные возможности, такие как типы значений Nullable, перечисления, делегаты, лямбда-выражения и прямой доступ к памяти, чего нет в Java. C# поддерживает универсальные методы и типы, обеспечивая более высокий уровень безопасности и производительности, а также итераторы, позволяющие при реализации коллекций классов определять собственное поведение итерации, которое может легко использоваться в клиентском коде. В C# 3.0 выражения LINQ (Language-IntegratedQuery) делают строго-типизированный запрос первоклассной конструкцией языка.

Как объектно-ориентированный язык, C# поддерживает понятия инкапсуляции, наследования и полиморфизма. Все переменные и методы, включая метод Main – точку входа приложения – инкапсулируются в определения классов. Класс может наследовать непосредственно из одного родительного класса, но может реализовывать любое число интерфейсов. Для методов, которые переопределяют виртуальные методы в родительском классе, необходимо ключевое слово override, чтобы исключить случайное повторное определение. В языке C# структура похожа на облегченный класс: это тип, распределяемый по стопкам, реализующий интерфейсы, но не поддерживающий наследование.

В дополнение к основным описанным объектно-ориентированным принципам, язык C# упрощает разработку компонентов программного обеспечения благодаря нескольким инновационным конструкциям языка, в число которых входят следующие:

* инкапсулированные подписи методов, называемые делегатами, которые поддерживают строго-типизированные уведомления о событиях;
* свойства, выступающие в роли методов доступа для закрытых переменных-членов;
* атрибуты с декларативными метаданными о типах во время выполнения;
* встроенные комментарии XML-документации;
* LINQ (Language-IntegratedQuery), предлагающий встроенные возможности запросов в различных источниках данных.

Если потребуется обеспечить взаимодействие с другим программным обеспечением Windows, таким как объекты COM или собственные библиотеки DLL Win32, в языке C# можно использовать процесс, который называется "Interop." Процесс Interop позволяет программам на C# выполнять практически любые действия, которые может выполнять исходное приложение на C++. Язык C# поддерживает даже указатели и понятие "небезопасного" кода для тех случаев, когда прямой доступ к памяти имеет крайне важное значение.

Процесс построения C# по сравнению с C и C++ прост и является более гибким, чем в Java. Нет отдельных файлов заголовка, а методы и типы не требуется объявлять в определенном порядке. В исходном файле C# может быть определено любое число классов, структур, интерфейсов и событий.

## СИСТЕМНОЕ ПРОГРАММРОВАНИЕ

Значительным этапом при разработке любого программного обеспечения является этап системного анализа и моделирования предметной области. Данный этап позволяет ответить на вопрос, какие функции должна выполнять система, определить связи и взаимодействия между ними, помогает выявить недостатки системы и построить логическую структуру программного обеспечения.

ERwinDataModeler – CASE-средство для проектирования и документирования баз данных, которое позволяет создавать, документировать и сопровождать базы данных, хранилища и витрины данных. Модели данных помогают визуализировать структуру данных, обеспечивая эффективный процесс организации, управления и администрирования таких аспектов деятельности предприятия, как уровень сложности данных, технологий баз данных и среды развертывания.

AllFusionERwinDataModeler (ERwin) предназначен для всех компаний, разрабатывающих и использующих базы данных, для администраторов баз данных, системных аналитиков, проектировщиков баз данных, разработчиков, руководителей проектов. AllFusionERwinDataModeler позволяет управлять данными в процессе корпоративных изменений, а также в условиях стремительно изменяющихся технологий.

BPwin – инструмент для моделирования, анализа, документирования и оптимизации бизнес-процессов. AllFusionProcessModeler 7 можно использовать для графического представления бизнес-процессов. Графически представленная схема выполнения работ, обмена информацией, документооборота визуализирует модель бизнес-процесса. Графическое изложение этой информации позволяет перевести задачи управления организацией из области сложного ремесла в сферу инженерных технологий. Средства автоматизированного анализа, моделирования и разработки систем получили название CASE-средств (Computer-AidedSoftware/SystemEngineering – компьютерная поддержка проектирования программного обеспечения/систем). Одним из таких средств является BPwin – оптимально подходящее для разработки, анализа и оптимизации бизнес-процессов. Диаграммы, разработанные в BPwin, позволяют четко увидеть предметную область, проверить связи между собой, оптимизировать систему.

BPwin поддерживает три методологии моделирования: функциональное моделирование (IDEF0); описание бизнес-процессов (IDEF3); диаграммы потоков данных (DFD). Наиболее подходящей методологией является IDEF0. В ее основе лежат диаграммы, состоящие из блоков и стрелок. Блоки означают какое-то действие, процесс или задачу, а стрелки показывают взаимодействие блоков друг с другом или внешним миром. У каждой стороны блока свое назначение: левая сторона – показывает, что подается на вход, правая - что получается на выходе, верхняя - что влияет на результат, нижняя - при помощи чего получается результат. База данных – это совокупность взаимосвязанной информации, организованной определенным образом и расположенной на внешнем носителе.

Запрос на выбор нужной книги или читателя вводится с клавиатуры и может содержать следующие данные: № читательского билета, Ф.И.О. читателя и его адрес, название и автор книги. Допускается ввод неполных данных.

В составе модели должна присутствовать контекстная диаграмма A-0, которая содержит только один блок. Номер единственного блока на контекстной диаграмме A-0 должен быть 0.

Не контекстные диаграммы должны содержать не менее трех и не более шести блоков. Эти ограничения поддерживают сложность диаграмм на уровне, доступном для чтения, понимания и использования. Диаграммы с количеством блоков менее трех вызывают серьезные сомнения в необходимости декомпозиции родительской функции. Диаграммы с количеством блоков более шести сложны для восприятия читателями и вызывают у автора трудности при внесении в нее всех необходимых графических объектов и меток.

Каждый блок, подвергнутый декомпозиции, должен иметь ссылку на дочернюю диаграмму; ссылка (например, узловой номер, C-номер или номер страницы) помещается под правым нижним углом блока.

Имена блоков (выполняемых функций) и метки стрелок должны быть уникальными. Если метки стрелок совпадают, это значит, что стрелки отображают тождественные данные.

При наличии стрелок со сложной топологией целесообразно повторить метку для удобства ее идентификации.

Следует обеспечить максимальное расстояние между блоками и поворотами стрелок, а также между блоками и пересечениями стрелок для облегчения чтения диаграммы. Одновременно уменьшается вероятность перепутать две разные стрелки.

Блоки всегда должны иметь хотя бы одну управляющую и одну выходную стрелку, но могут не иметь входных стрелок.

Если одни и те же данные служат и для управления, и для входа, вычерчивается только стрелка управления. Этим подчеркивается управляющий характер данных и уменьшается сложность диаграммы.

## 2.1 Построение моделиIDEF0

Контекстная диаграмма – вид IDEF0-диаграммы. Это диаграмма, расположенная на вершине древовидной структуры диаграмм, представляющая собой самое общее описание системы и ее взаимодействие с внешней средой (как правило, здесь описывается основное назначение моделируемого объекта).

Диаграмма IDEF0 наиболее удобный язык моделирования бизнес-процессов, где система представляется как совокупность взаимодействующих работ или функций. Такая чисто функциональная ориентация является принципиальной — функции системы анализируются независимо от объектов, которыми они оперируют. Это позволяет более четко смоделировать логику и взаимодействие процессов организации.

Процесс моделирования системы в IDEF0 начинается с создания контекстной диаграммы — диаграммы наиболее абстрактного уровня описания системы в целом, содержащей определение субъекта моделирования, цели и точки зрения на модель.

Контекстная диаграмма A-0 должна содержать краткие утверждения, определяющие точку зрения должностного лица или подразделения, с позиции которого создается модель, и цель, для достижения которой ее разрабатывают. Эти утверждения помогают руководить разработкой модели и ввести этот процесс в определенные рамки. Точка зрения определяет, что и в каком разрезе можно увидеть в пределах контекста модели. Изменение точки зрения приводит к рассмотрению других аспектов объекта. Аспекты, важные с одной точки зрения, могут не появиться в модели, разрабатываемой с другой точки зрения на тот же самый объект.

Изучение состояния проблемы является обязательной частью любого проекта создания или развития системы. Построение функциональной модели «КАК ЕСТЬ» позволяет четко зафиксировать, какие процессы осуществляются в системе, какие информационные объекты используются при выполнении процессов и отдельных операций. Функциональная модель «КАК ЕСТЬ» является отправной точкой для анализа существующей системы, выявления проблем и «узких» мест и разработки проекта совершенствования имеющейся системы представлена на рисунке 2.1.

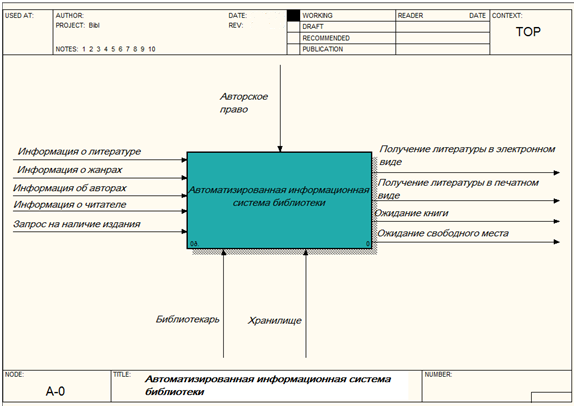


Рисунок 2.1 – Контекстная диаграмма модели «КАК ЕСТЬ»

На данной диаграмме отображены основные подпроцессы основного бизнес-процесса отдела. В качестве входных потоков представлены:

* информация о читателях;
* информация о книгах.

В качестве выходных потоков представлены:

* выходная информация.

Управляющими воздействиями являются такие, как требования законодательства РБ и устав библиотеки.

В качестве механизмов обозначены сотрудники и читатели.

Декомпозиция контекстной диаграммы представлена на рисунке 2.2.

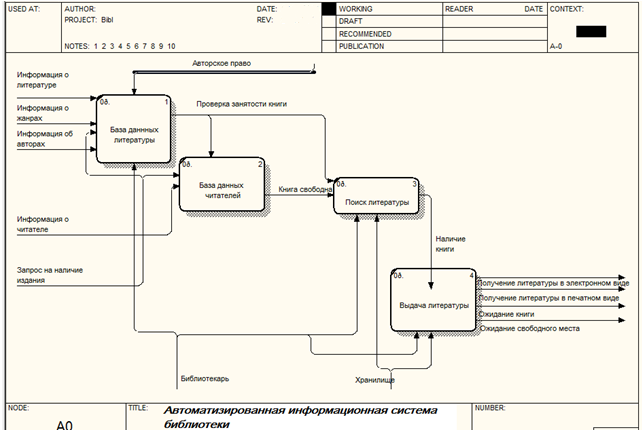


Рисунок 2.2 – Декомпозиция модели «КАК ЕСТЬ»

# ЛОГИЧЕСКОЕ МОДЕЛИРОВАНИЕ

RationalRose – популярное средство визуального моделирования объектно-ориентированных информационных систем компании RationalSoftwareCorp. Работа продукта основана на универсальном языке моделирования UML (UniversalModelingLanguage). Благодаря уникальному языку моделирования RationalRose способен решать практически любые задачи в проектировании информационных систем: от анализа бизнес процессов до кодогенерации на определенном языке программирования. Только Rose позволяет разрабатывать как высокоуровневые, так и низкоуровневые модели, осуществляя тем самым либо абстрактное проектирование, либо логическое. Только RationalRose имеет весь необходимый набор визуальных средств проектирования. Только Rose поможет решить проблемы с кодогенерацией на определенном языке программирования. Только RationalRose осуществляет такие подходы, как прямое и обратное проектирование, а также RoundTripEngineering. Такой арсенал позволит не только проектировать новую систему, но и доработать старую, произведя процесс обратного проектирования. Работа пользователя с базами данных в ПС реализована посредством Работа пользователя с настройками приложения и базами данных в ПС реализована посредством графического интерфейса. Именно этот блок объединят все модули ПС и предоставляет пользователю возможность управления процессом. Блок экспорта отвечает за процесс экспорта, управление процессом экспорта, визуализацию процесса экспорта, индикацию процесса экспорта. Процесс разделен и протекает в несколько этапов. Соответственно блок экспорта имеет в своем составе блок экспорта метаданных и блок экспорта данных, а каждый из этих блоков имеет в своем составе блоки извлечения, структурирования, форматирования и выгрузки. Каждый блок отвечает за работу одного этапа экспорта БД. Блок управления профилями настроек экспорта предоставляет возможность создавать, хранить, редактировать и использовать несколько экземпляров настроек экспорта баз данных. Блок конфигурирования настроек экспорта баз данных отвечает за редактирование профилей экспорта БД. Так как настроек довольно много, и их структура довольно сложна, блок имеет в своем составе множество блоков, каждый из которых выполняет часть задачи. Блоки настроек экспорта метаданных и экспорта данных имеют в своем составе блоки настроек извлечения, настроек структурирования и настроек форматирования. Каждый из этих блоков отвечает за предоставление пользователю удобного интерфейса и функционала по создание и редактированию настроек, влияющих на работу определенного этапа работу процесса экспорта базы данных. Работа с настройками соединения с БД и настройками выгрузки выделена в отдельные блоки, так как в дополнение к задачам, которые выполняют остальные блоки, эти блоки взаимодействуют еще и с внешними системами. Блок редактирования общих настроек служит для редактирования общих настроек приложения. Блок обработки ошибок и исключений отвечает за обработку штатных исключений, предусмотренных разработкой ПС и внештатных ошибок. Блок настройки диалогов предоставляет доступ к списку диалогов, которые выдает ПС для пользователя и позволяет исключить либо же наоборот вернуть определенные диалоги в работу ПС. Обоснование выбора среды разработки:

* Microsoft Visual Studio 2012
* Microsoft Visio 2010
* CA ERwin Process Modeler 7.3
* Subversion + TortoiseSVN

Основное назначение логического представления состоит в анализе структурных и функциональных отношений между элементами модели системы. Различные элементы логического представления, такие как классы, ассоциации, состояния, сообщения, не существуют материально или физически. Они лишь отражают наше понимание структуры физической системы или аспекты ее поведения.

## 3.1 Модель вариантов использования

Модель вариантов использования предназначается для определения требований к системе. Она включает в себя актеров, варианты использования и связи между ними. Для отображения этой модели язык UML предлагает использовать диаграммы UseCase (вариант использования) совместно с моделями StateDiagram (диаграммы состояний) и ActivityDiagram (диаграммы деятельности/активности). Последние используются для конкретизации вариантов использования системы.

Разработка диаграммы вариантов использования преследует цели:

* определить общие границы и контекст моделируемой предметной области на начальных этапах проектирования системы;
* сформулировать общие требования к функциональному поведению проектируемой системы;
* разработать исходную концептуальную модель системы для ее последующей детализации в форме логических и физических моделей;
* подготовить исходную документацию для взаимодействия разработчиков системы с ее заказчиками и пользователями.

### 3.1.1 Варианты использования

Работа с вариантами использования является одной из самых важных на стадии уточнения. Каждый вариант использования — это требование к системе. Диаграммы отображают взаимодействие между вариантами использования системы и действующими лицами, передающими информацию в данную систему. Такой пояснительный текст получил название примечания или сценария.

### 3.1.2 Диаграмма вариантов использования

На диаграмме вариантов использования показывают взаимодействия между всеми действующими лицами и вариантами использования. Диаграмма должна показывать, какие действующие лица инициируют варианты использования, а также должна отображать, когда действующие лица получают информацию от вариантов использования.

Основные взаимодействия между действующими лицами и вариантами использования задаются с помощью связи коммуникации. Задается в виде простой стрелки. Направление стрелки показывает, кто инициирует связь (всегда действующее лицо) и какой вариант использования отправляет информацию внешнему действующему лицу. Диаграмма вариантов использования изображена на рисунке 3.1

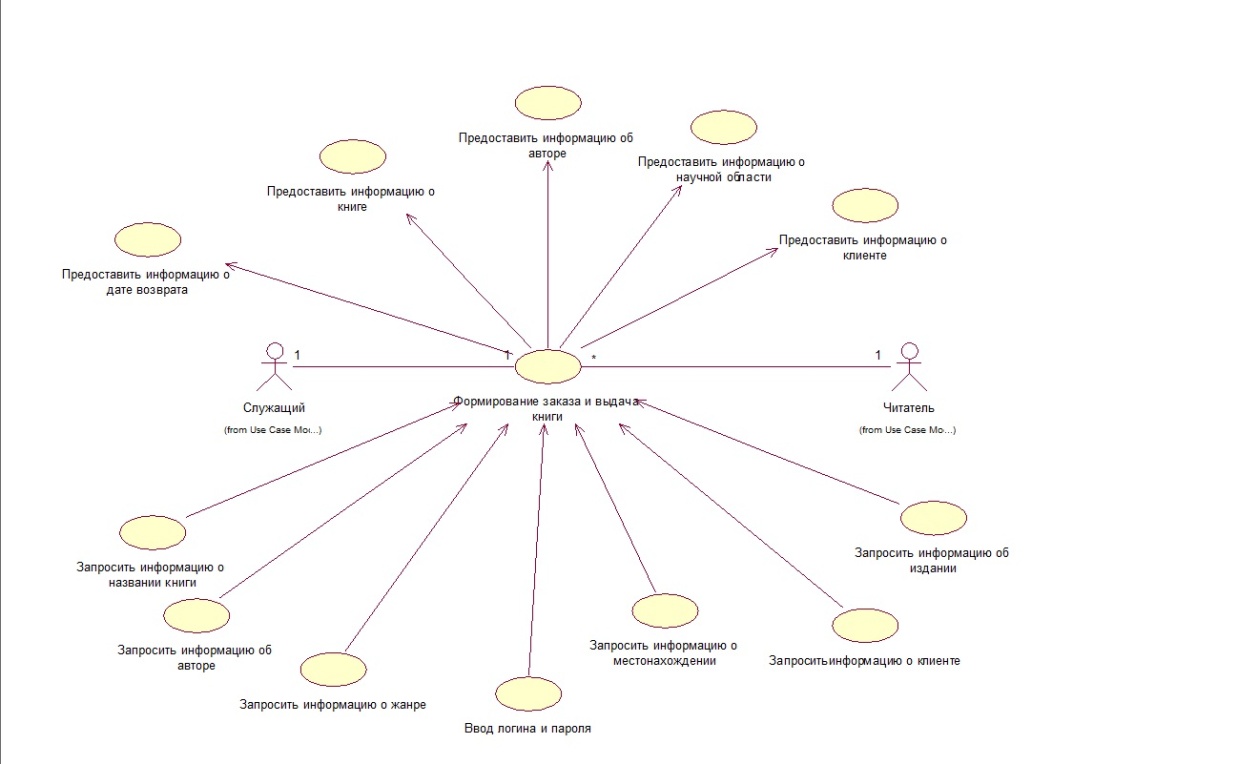


Рисунок 3.1 – Диаграмма вариантов использования

Каждый вариант использования показывает, как конкретный актер использует систему. Для того чтобы начать выполнять операции, в системе необходимо ее запустить.

### 3.1.3 Описание вариантов использования

Вариант использования «Вход в систему»:

* назначение: данный вариант использования описывает вход пользователя в систему;
* основной поток событий: данный вариант используется, когда пользователь хочет войти в систему:

1) система запрашивает имя пользователя и пароль;

2) пользователь вводит имя и пароль;

3) система проверяет имя и пароль, после чего открывает доступ в систему.

Вариант использования «Использование записей БД»:

* назначение: данный вариант использования служит для просмотра, добавления, редактирования, удаления информации, а так же печати необходимой документации;
* основной поток событий: данный вариант использования начинает выполняться, когда пользователь начинает работу с базой данных для ее редактирования:

1) система запрашивает требуемое действие (просмотреть запись, добавить запись, удалить запись, изменить запись, распечатать документацию);

2) после того как пользователь выбирает действие, выполняемое с базой данных, начинает выполняться один из подчиненных потоков: просмотр, добавление, редактирование удаление, печати.

Добавление записи:

а) система предоставляет специальную форму для добавления записи;

б) пользователь вводит данные;

в) система проверяет корректность введенной информации;

г) система сохраняет новость в базе данных.

Удаление записи:

а) сотрудник выбирает необходимую запись для удаления;

б) система предоставляет специальную форму для удаления записи;

в) сотрудник подтверждает удаление;

г) система производит удаление записи и вносит изменения в базу данных.

## 3.2 Идентификация классов

### 3.2.1 Классы анализа

Диаграмма классов является основным средством моделирования структуры в UML, а класс, соответственно, основной структурной единицей. Это совсем не удивительно и вполне естественно, поскольку UML является в большой степени объектно-ориентированным языком. Диаграммы классов наиболее информационно насыщены по сравнению с другими типами канонических диаграмм UML, инструменты генерируют код в основном по описанию классов, структура классов точнее всего соответствует окончательной структуре кода приложения.

На диаграммах классов в качестве сущностей применяются, прежде всего, классы, как в своей наиболее общей форме, так и в форме многочисленных стереотипов и частных случаев: интерфейсы, типы данных, активные классы и др. Кроме того, на диаграмме классов могут использоваться (как и везде) пакеты и комментарии.

В этом разделе мы рассматриваем сущности, используемые на диаграммах классов, а в [следующем разделе](http://book.uml3.ru/sec_3_3.html#p0) ‒ отношения между этими сущностями.

Классы анализа отражают функциональные требования к системе и моделируют объекты предметной области. Совокупность классов анализа представляет собой начальную концептуальную модель системы. Классы анализа следует рассматривать как начальные прототипы уровня разработки. Класс анализа представляет собой абстракцию одного или более классов в проекте системы. Диаграмма классов представлена на рисунке 3.2.

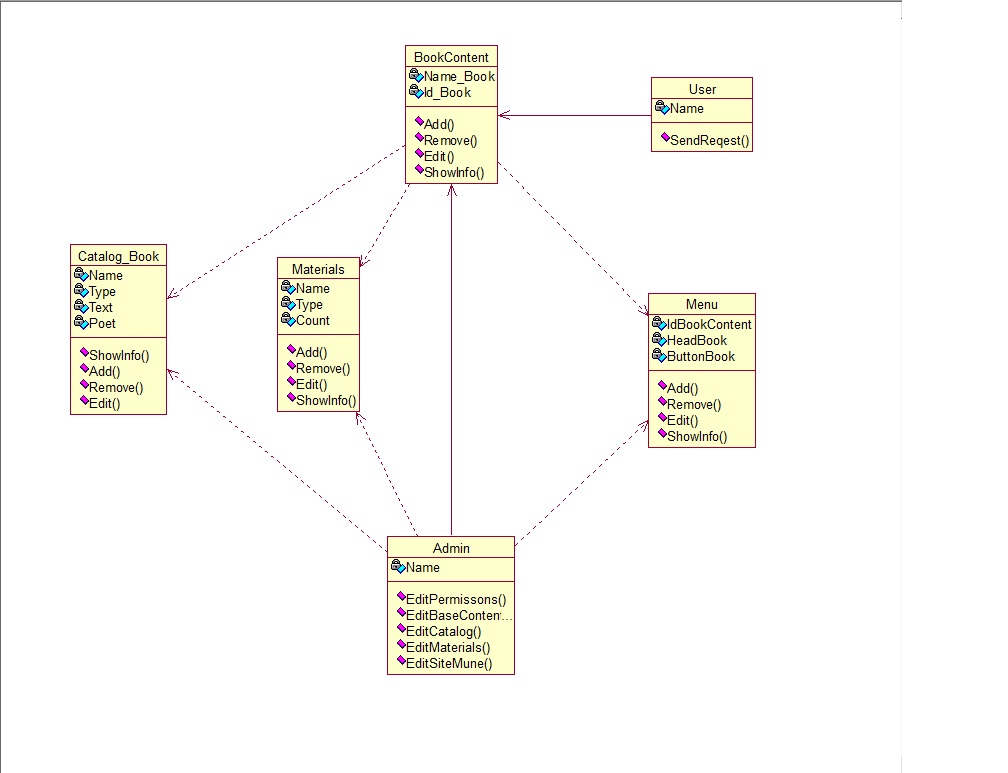


Рисунок 3.2 – Диаграмма классов

**3.3 Проектирование логической модели данных**

Перед проектированием физической модели данных необходимо смоделировать логическую модель. Логическая модель описывает всю базу данных как единое целое. Кроме того, необходимое пользователю логическое представление данных может существенно отличаться от общей модели данных.

Программистам полезно ясно представлять себе все бизнес-процессы компании, которые будут затронуты их новой системой. В данном случае у компании еще есть бизнес-процесс обработки заявок, который уже работает и есть у заказчика, и его также нужно понять. Иначе может оказаться, что упущена какая-то важная деталь, которая не позволяет новой системе полноценно выполнять свои функции.

Итак, главной сущностью этого типа диаграмм является активность (activity) - активное состояние системы, в котором она выполняет некоторую работу. После ее завершения происходит переход в другую активность. Возможны и более сложные случаи переходов между активностями.

На диаграмме должны присутствовать символы начала (start) и конца (finish).

Далее, на диаграмме может использоваться параллельный разветвитель (fork), который запускает несколько одновременно работающих веток. Такие ветки могут объединяться (все или только часть) конструкцией под названием параллельный соединитель (join).

Наконец, на диаграмме могут использоваться символы логического ветвления и логического соединения (decision). На ветках, идущих из логического ветвления, обозначаются условия перехода.

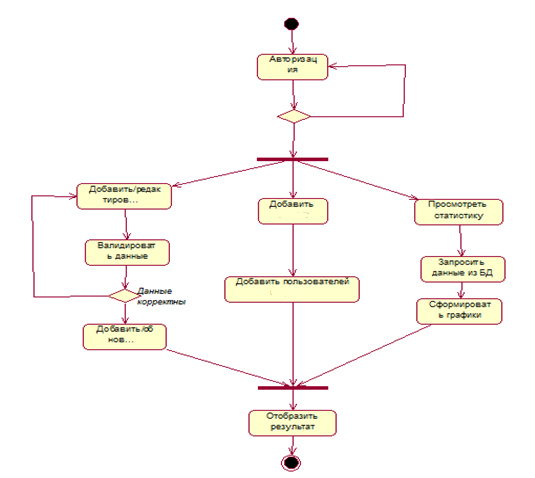
****

Рисунок 3.3 – Диаграмма активности

**3.4 Диаграммы взаимодействия**

На них представлены связи между объектами, показаны, в частности, сообщения, которыми объекты могут обмениваться. Обычно рассматриваются два частных случая это вида диаграмм: диаграммы последовательностей (sequence diagram), которые отражают временную упорядоченность сообщений, и диаграммы кооперации (collaboration diagram), на которых показана структурная организация обменивающихся сообщениями объектов.

Диаграммы последовательности имеют две размерности: вертикальная представляет время, горизонтальная - различные объекты. Оси могут меняться местами, так что ось времени может располагаться горизонтально, слева направо, а список объектов располагаться вертикально.

Объект на диаграмме изображается в виде прямоугольника на вершине вертикальной пунктирной линии, называемой линией жизни объекта (lifeline). Она представляет собой фрагмент жизненного цикла объекта в процессе взаимодействия. Если объект создается или уничтожается на отрезке времени, представленном на диаграмме, то его линия жизни начинается и заканчивается в соответствующих точках, в противном случае линия жизни объекта проводится от начала до конца диаграммы. Символ объекта рисуется в начале его линии жизни; если объект создается не в начале диаграммы, то сообщение о создании объекта рисуется со стрелкой, проведенной к символу объекта. Если объект уничтожается не в конце диаграммы, то момент его уничтожения помечается большим крестиком "Х". При сообщении, вызывающем уничтожение объекта (или самоуничтожение), в конце возвращается сообщение об уничтожении объекта. Линия жизни может разветвляться в две (и более) параллельные линии, показываемые условно. Каждая ответвляющаяся линия соответствует переходу в потоке сообщений. Линии жизни могут объединяться в некоторой последующей отметке.Диаграмма последовательностей представлена на рисунке 3.4.

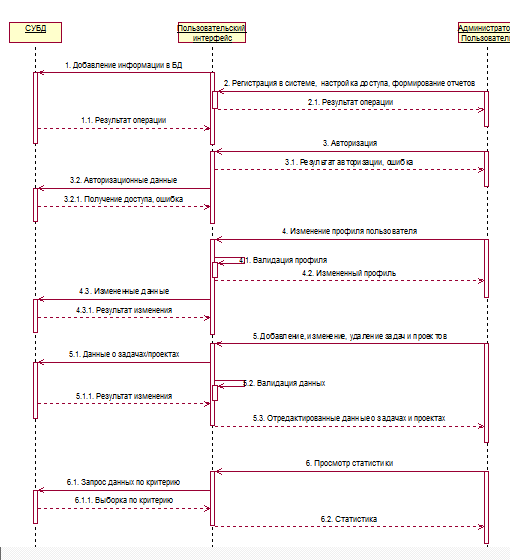


Рисунок 3.4 – Диаграмма последовательностей

Взаимодействие между объектами в системе представляются диаграммами взаимодействия (interaction diagrams). Диаграммы взаимодействия подразделяются на два основных типа диаграмм: диаграммы последовательности (sequence diagrams) и кооперативные диаграммы (collaboration diagrams).

Как правило, диаграмма взаимодействия используется для описания поведения в рамках одного варианта использования. На такой диаграмме изображается ряд объектов и те сообщения, которыми они обмениваются в рамках этого варианта использования.

Диаграммы последовательности и кооперативные диаграммы несут в себе одну информацию, но выраженную разными способами. Диаграммы последовательности показывают взаимодействие объектов во времени и отражают последовательность происходящих событий. На диаграмме не отражаются связи между объектами. Кооперативные диаграммы позволяют пространственно располагать объекты, для того чтобы лучше представить взаимодействие между объектами. Временная последовательность передаваемых сообщений отражается при помощи нумерации сообщений.

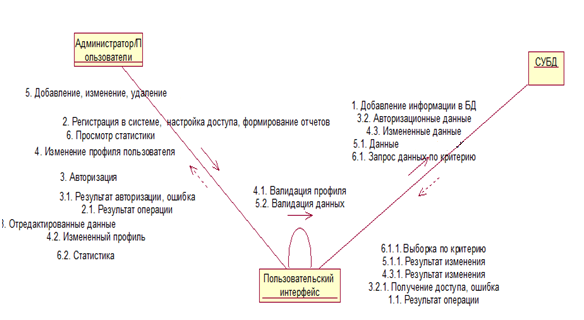


Рисунок 3.5 – Диаграмма взаимодействия

Диаграммы компонентов - это один из двух видов диаграмм, применяемых при моделировании физических аспектов объектно-ориентированной системы. Они показывают организацию наборов компонентов и зависимости между ними.

Диаграммы компонентов применяются для моделирования статического вида системы с точки зрения реализации. Сюда относится моделирование физических сущностей, развернутых в узле, например, исполняемых программ, библиотек, таблиц, файлов и документов. По существу, диаграммы компонентов - это не что иное, как диаграммы классов, сфокусированные на системных компонентах.

Диаграммы компонентов важны не только для визуализации, специфицирования и документирования системы, основанной на компонентах, но и для создания исполняемых систем путем прямого и обратного проектирования.

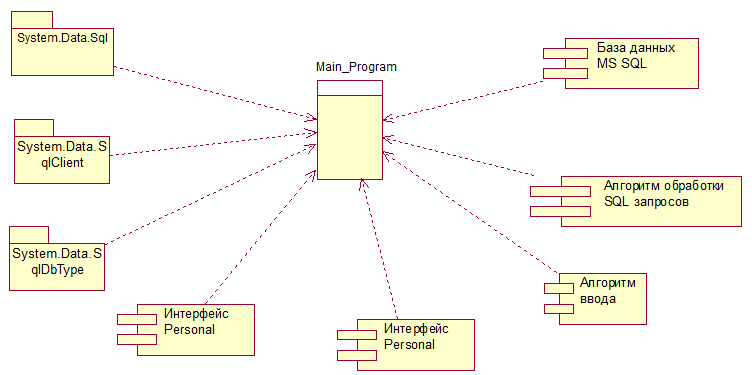


Рисунок 3.6– Диаграмма компонентов

Диаграмма развертывания (deploymentdiagram) отражает физические взаимосвязи между программными и аппаратными компонентами системы. Она является хорошим средством для того, чтобы показать маршруты перемещения объектов и компонентов в распределенной системе.

Каждый узел на диаграмме размещения представляет собой некоторый тип вычислительного устройства – в большинстве случаев, часть аппаратуры. Эта аппаратура может быть простым устройством или датчиком, а может быть и мэйнфреймом.

Диаграмма развертывания показывает физическое расположение сети и местонахождение в ней различных компонентов.



Рисунок 3.7 – Диаграмма развертывания

**3.5 Подготовка модели к генерации кода**

Одним из наиболее важных свойств программы IBM RationalRose является возможность генерации программного кода на нескольких языках программирования, которая может быть использована разработчиком после построения модели. Для этой цели в среде IBM RationalRose присутствует достаточно большой выбор языков программирования и схем баз данных. Однако возможность генерации текста программы на том или ином языке программирования зависит от установленной версии IBM RationalRose.

(object Petal

version 45

\_written "Rose 7.6.0109.2314"

charSet 204)

(object Design "Logical View"

is\_unit TRUE

is\_loaded TRUE

attributes (list Attribute\_Set

(object Attribute

tool "Java"

name "IDE"

value "Internal Editor"))

quid "587CFEF8030D"

defaults (object defaults

rightMargin 0.250000

leftMargin 0.250000

topMargin 0.250000

bottomMargin 0.500000

pageOverlap 0.250000

clipIconLabels TRUE

autoResize TRUE

snapToGrid TRUE

gridX 16

gridY 16

defaultFont (object Font

size 10

face "Arial"

charSet 204

bold FALSE

italics FALSE

underline FALSE

strike FALSE

color 0

default\_color TRUE)

showMessageNum 1

showClassOfObject TRUE

notation "Unified")

root\_usecase\_package (object Class\_Category "Use Case View"

quid "587CFEF80315"

exportControl "Public"

global TRUE

logical\_models (list unit\_reference\_list)

logical\_presentations (list unit\_reference\_list

(object UseCaseDiagram "Main"

quid "587CFEFC02C0"

title "Main"

zoom 100

max\_height 28350

max\_width 21600

origin\_x 0

origin\_y 0

items (list diagram\_item\_list))))

root\_category (object Class\_Category "Logical View"

quid "587CFEF8030E"

exportControl "Public"

global TRUE

subsystem "Component View"

quidu "587CFEF80316"

logical\_models (list unit\_reference\_list)

statemachine (object State\_Machine "State/Activity Model"

quid "587D08D402DE"

states (list States

(object State "$UNNAMED$0"

quid "587D08F601DB"

type "StartState")

(object State "Авторизация"

quid "587D08FA00AC"

transitions (list transition\_list

(object State\_Transition

quid "587D093C01B1"

supplier "$UNNAMED$1"

quidu "587D090802A8"

sendEvent (object sendEvent

quid "587D093C01B4")))

type "Normal")

(object Decision "$UNNAMED$1"

quid "587D090802A8"

transitions (list transition\_list

(object State\_Transition

quid "587D094000D0"

supplier "$UNNAMED$2"

quidu "587D090D0224"

sendEvent (object sendEvent

quid "587D094000D3"))

(object State\_Transition

quid "587D0955013B"

supplier "Авторизация"

quidu "587D08FA00AC"

sendEvent (object sendEvent

quid "587D0955013E"))))

(object SynchronizationState "$UNNAMED$2"

quid "587D090D0224"

transitions (list transition\_list

(object State\_Transition

quid "587D092201F2"

supplier "Добавить/редактировать"

quidu "587D0912018F"

sendEvent (object sendEvent

quid "587D092201F5"))

(object State\_Transition

quid "587D09310339"

supplier "Добавить проект"

quidu "587D0915011D"

sendEvent (object sendEvent

quid "587D0931033C"))

(object State\_Transition

quid "587D09350065"

supplier "Просмотреть статистику"

quidu "587D0917035B"

sendEvent (object sendEvent

quid "587D09350068"))))

(object State "Добавить/редактировать"

quid "587D0912018F"

transitions (list transition\_list

(object State\_Transition

quid "587D09ED01A1"

supplier "Валидировать данные"

quidu "587D09A7004B"

sendEvent (object sendEvent

quid "587D09ED01A4")))

type "Normal")

(object State "Добавить проект"

quid "587D0915011D"

transitions (list transition\_list

(object State\_Transition

quid "587D0A34031C"

supplier "Добавить пользователей к проекту"

quidu "587D09AD03CA"

sendEvent (object sendEvent

quid "587D0A34031F")))

type "Normal")

(object State "Просмотреть статистику"

quid "587D0917035B"

transitions (list transition\_list

(object State\_Transition

quid "587D0A380316"

supplier "Запросить данные из БД"

quidu "587D09BA024F"

sendEvent (object sendEvent

quid "587D0A380319")))

type "Normal")

(object State "Валидировать данные"

quid "587D09A7004B"

transitions (list transition\_list

(object State\_Transition

quid "587D0A1C0137"

supplier "Данные корректны"

quidu "587D0A18036F"

sendEvent (object sendEvent

quid "587D0A1C013A")))

type "Normal")

(object State "Добавить пользователей к проекту"

quid "587D09AD03CA"

transitions (list transition\_list

(object State\_Transition

quid "587D0A7F0082"

supplier "$UNNAMED$3"

quidu "587D0A760274"

sendEvent (object sendEvent

quid "587D0A7F0085")))

type "Normal")

(object State "Запросить данные из БД"

quid "587D09BA024F"

transitions (list transition\_list

(object State\_Transition

quid "587D0A5203D7"

supplier "Сформировать графики"

quidu "587D0A430075"

sendEvent (object sendEvent

quid "587D0A5203DA")))

type "Normal")

(object Decision "Данные корректны"

quid "587D0A18036F"

transitions (list transition\_list

(object State\_Transition

quid "587D0A2A037A"

supplier "Добавить/редактировать"

quidu "587D0912018F"

sendEvent (object sendEvent

quid "587D0A2A037D"))

(object State\_Transition

quid "587D0A660240"

supplier "Добавить/обновить"

quidu "587D0A580360"

sendEvent (object sendEvent

quid "587D0A660243"))))

(object State "Сформировать графики"

quid "587D0A430075"

transitions (list transition\_list

(object State\_Transition

quid "587D0A8302DC"

supplier "$UNNAMED$3"

quidu "587D0A760274"

sendEvent (object sendEvent

quid "587D0A8302DF")))

type "Normal")

(object State "Добавить/обновить"

quid "587D0A580360"

transitions (list transition\_list

(object State\_Transition

quid "587D0A7B00D6"

supplier "$UNNAMED$3"

quidu "587D0A760274"

sendEvent (object sendEvent

quid "587D0A7B00D9")))

type "Normal")

(object SynchronizationState "$UNNAMED$3"

quid "587D0A760274"

transitions (list transition\_list

(object State\_Transition

quid "587D0AA30386"

supplier "Отобразить результат"

quidu "587D0A9201CB"

sendEvent (object sendEvent

quid "587D0AA30389"))))

(object State "Отобразить результат"

quid "587D0A9201CB"

transitions (list transition\_list

(object State\_Transition

quid "587D0AAD001C"

supplier "$UNNAMED$4"

quidu "587D0AA901C8"

sendEvent (object sendEvent

quid "587D0AAD001F")))

type "Normal")

(object State "$UNNAMED$4"

quid "587D0AA901C8"

type "EndState"))

partitions (list Partitions)

objects (list Objects)

statediagrams (list StateDiagrams

(object ActivityDiagram "NewDiagram"

quid "587D08D402DF"

title "NewDiagram"

zoom 100

max\_height 28350

max\_width 21600

origin\_x 1262

origin\_y 950

items (list diagram\_item\_list

(object Swimlane "" @1

line\_color 3342489

width 450)

(object StateView "StartState" "$UNNAMED$0" @2

Parent\_View @1

location (1968, 80)

font (object Font

size 10

face "Arial"

charSet 204

bold FALSE

italics FALSE

underline FALSE

strike FALSE

color 0

default\_color TRUE)

icon\_style "Icon"

line\_color 3342489

quidu "587D08F601DB"

autoResize TRUE)

(object StateView "Normal" "Авторизация" @3

Parent\_View @1

location (1936, 288)

font (object Font

size 10

face "Arial"

charSet 204

bold FALSE

italics FALSE

underline FALSE

strike FALSE

color 0

default\_color TRUE)

label (object ItemLabel

Parent\_View @3

location (1936, 277)

fill\_color 13434879

anchor\_loc 1

nlines 2

max\_width 204

justify 0

label "Авторизация")

icon\_style "Icon"

line\_color 3342489

fill\_color 13434879

quidu "587D08FA00AC"

autoResize TRUE)

(object DecisionView "$UNNAMED$1" @4

Parent\_View @1

location (1936, 480)

icon\_style "Icon"

line\_color 3342489

fill\_color 13434879

quidu "587D090802A8"

autoResize TRUE)

(object TransView "" @5

stereotype TRUE

line\_color 3342489

quidu "587D093C01B1"

client @3

supplier @4

line\_style 0

x\_offset FALSE)

(object TransView "" @6

stereotype TRUE

line\_color 3342489

quidu "587D0955013B"

client @4

supplier @3

vertices (list Points

(2011, 480)

(2268, 478)

(2268, 278)

(2086, 283))

line\_style 0

x\_offset FALSE)

(object StateView "Normal" "Добавить/редактировать" @7

Parent\_View @1

location (1552, 928)

font (object Font

size 10

face "Arial"

charSet 204

bold FALSE

italics FALSE

underline FALSE

strike FALSE

color 0

default\_color TRUE)

label (object ItemLabel

Parent\_View @7

location (1552, 908)

fill\_color 13434879

anchor\_loc 1

nlines 2

max\_width 260

justify 0

label "Добавить/редактировать")

icon\_style "Icon"

line\_color 3342489

fill\_color 13434879

quidu "587D0912018F"

width 356

height 152

autoResize TRUE)

(object StateView "Normal" "Добавить проект" @8

Parent\_View @1

location (1952, 928)

font (object Font

size 10

face "Arial"

charSet 204

bold FALSE

italics FALSE

underline FALSE

strike FALSE

color 0

default\_color TRUE)

label (object ItemLabel

Parent\_View @8

location (1952, 917)

fill\_color 13434879

anchor\_loc 1

nlines 2

max\_width 204

justify 0

label "Добавить проект")

icon\_style "Icon"

line\_color 3342489

fill\_color 13434879

quidu "587D0915011D"

autoResize TRUE)

(object StateView "Normal" "Просмотреть статистику" @9

Parent\_View @1

location (2368, 928)

font (object Font

size 10

face "Arial"

charSet 204

bold FALSE

italics FALSE

underline FALSE

strike FALSE

color 0

default\_color TRUE)

label (object ItemLabel

Parent\_View @9

location (2368, 908)

fill\_color 13434879

anchor\_loc 1

nlines 2

max\_width 302

justify 0

label "Просмотреть статистику")

icon\_style "Icon"

line\_color 3342489

fill\_color 13434879

quidu "587D0917035B"

width 398

height 152

autoResize TRUE)

(object StateView "Normal" "Валидировать данные" @10

Parent\_View @1

location (1488, 1344)

font (object Font

size 10

face "Arial"

charSet 204

bold FALSE

italics FALSE

underline FALSE

strike FALSE

color 0

default\_color TRUE)

label (object ItemLabel

Parent\_View @10

location (1488, 1324)

fill\_color 13434879

anchor\_loc 1

nlines 2

max\_width 282

justify 0

label "Валидировать данные")

icon\_style "Icon"

line\_color 3342489

fill\_color 13434879

quidu "587D09A7004B"

width 378

height 152

autoResize TRUE)

(object TransView "" @11

stereotype TRUE

line\_color 3342489

quidu "587D09ED01A1"

client @7

supplier @10

line\_style 0

x\_offset FALSE)

(object StateView "Normal" "Добавить пользователей к проекту" @12

Parent\_View @1

location (2048, 1344)

font (object Font

size 10

face "Arial"

charSet 204

bold FALSE

italics FALSE

underline FALSE

strike FALSE

color 0

default\_color TRUE)

label (object ItemLabel

Parent\_View @12

location (2048, 1324)

fill\_color 13434879

anchor\_loc 1

nlines 2

max\_width 496

justify 0

label "Добавить пользователей к проекту")

icon\_style "Icon"

line\_color 3342489

fill\_color 13434879

quidu "587D09AD03CA"

width 592

height 152

autoResize TRUE)

(object TransView "" @13

stereotype TRUE

line\_color 3342489

quidu "587D0A34031C"

client @8

supplier @12

line\_style 0

x\_offset FALSE)

(object StateView "Normal" "Запросить данные из БД" @14

Parent\_View @1

location (2656, 1344)

font (object Font

size 10

face "Arial"

charSet 204

bold FALSE

italics FALSE

underline FALSE

strike FALSE

color 0

default\_color TRUE)

label (object ItemLabel

Parent\_View @14

location (2656, 1324)

fill\_color 13434879

anchor\_loc 1

nlines 2

max\_width 286

justify 0

label "Запросить данные из БД")

icon\_style "Icon"

line\_color 3342489

fill\_color 13434879

quidu "587D09BA024F"

width 382

height 152

autoResize TRUE)

(object TransView "" @15

stereotype TRUE

line\_color 3342489

quidu "587D0A380316"

client @9

supplier @14

line\_style 0

x\_offset FALSE)

(object DecisionView "Данные корректны" @16

Parent\_View @1

location (1504, 1552)

font (object Font

size 10

face "Arial"

charSet 204

bold FALSE

italics FALSE

underline FALSE

strike FALSE

color 0

default\_color TRUE)

label (object ItemLabel

Parent\_View @16

location (1560, 1503)

nlines 2

max\_width 300

label "Данные корректны")

icon\_style "Icon"

line\_color 3342489

fill\_color 13434879

quidu "587D0A18036F"

autoResize TRUE)

(object TransView "" @17

stereotype TRUE

line\_color 3342489

quidu "587D0A1C0137"

client @10

supplier @16

line\_style 0

x\_offset FALSE)

(object TransView "" @18

stereotype TRUE

line\_color 3342489

quidu "587D0A2A037A"

client @16

supplier @7

vertices (list Points

(1429, 1552)

(1040, 1553)

(1037, 922)

(1374, 925))

line\_style 0

x\_offset FALSE)

(object StateView "Normal" "Сформировать графики" @19

Parent\_View @1

location (2656, 1568)

font (object Font

size 10

face "Arial"

charSet 204

bold FALSE

italics FALSE

underline FALSE

strike FALSE

color 0

default\_color TRUE)

label (object ItemLabel

Parent\_View @19

location (2656, 1548)

fill\_color 13434879

anchor\_loc 1

nlines 2

max\_width 288

justify 0

label "Сформировать графики")

icon\_style "Icon"

line\_color 3342489

fill\_color 13434879

quidu "587D0A430075"

width 384

height 152

autoResize TRUE)

(object TransView "" @20

stereotype TRUE

line\_color 3342489

quidu "587D0A5203D7"

client @14

supplier @19

line\_style 0

x\_offset FALSE)

(object StateView "Normal" "Добавить/обновить" @21

Parent\_View @1

location (1488, 1712)

font (object Font

size 10

face "Arial"

charSet 204

bold FALSE

italics FALSE

underline FALSE

strike FALSE

color 0

default\_color TRUE)

label (object ItemLabel

Parent\_View @21

location (1488, 1701)

fill\_color 13434879

anchor\_loc 1

nlines 2

max\_width 204

justify 0

label "Добавить/обновить")

icon\_style "Icon"

line\_color 3342489

fill\_color 13434879

quidu "587D0A580360"

autoResize TRUE)

(object TransView "" @22

stereotype TRUE

line\_color 3342489

quidu "587D0A660240"

client @16

supplier @21

line\_style 0

x\_offset FALSE)

(object StateView "Normal" "Отобразить результат" @23

Parent\_View @1

location (1952, 1936)

font (object Font

size 10

face "Arial"

charSet 204

bold FALSE

italics FALSE

underline FALSE

strike FALSE

color 0

default\_color TRUE)

label (object ItemLabel

Parent\_View @23

location (1952, 1916)

fill\_color 13434879

anchor\_loc 1

nlines 2

max\_width 298

justify 0

label "Отобразить результат")

icon\_style "Icon"

line\_color 3342489

fill\_color 13434879

quidu "587D0A9201CB"

width 394

height 152

autoResize TRUE)

(object StateView "EndState" "$UNNAMED$4" @24

Parent\_View @1

location (1920, 2064)

font (object Font

size 10

face "Arial"

charSet 204

bold FALSE

italics FALSE

underline FALSE

strike FALSE

color 0

default\_color TRUE)

label (object ItemLabel

Parent\_View @24

location (1974, 2022)

nlines 2

max\_width 600

label "")

icon\_style "Icon"

line\_color 3342489

quidu "587D0AA901C8"

autoResize TRUE)

(object SynchronizationView "$UNNAMED$2" @25

location (1952, 688)

font (object Font

size 10

face "Arial"

charSet 204

bold FALSE

italics FALSE

underline FALSE

strike FALSE

color 0

default\_color TRUE)

label (object ItemLabel

Parent\_View @25

location (2114, 617)

nlines 2

max\_width 300

label "")

icon\_style "Icon"

line\_color 3342489

quidu "587D090D0224"

autoResize TRUE

sync\_flow\_direction 1

sync\_is\_horizontal TRUE)

(object TransView "" @26

stereotype TRUE

line\_color 3342489

quidu "587D092201F2"

client @25

supplier @7

line\_style 0

x\_offset FALSE)

(object TransView "" @27

stereotype TRUE

line\_color 3342489

quidu "587D09310339"

client @25

supplier @8

line\_style 0

x\_offset FALSE)

(object TransView "" @28

stereotype TRUE

line\_color 3342489

quidu "587D09350065"

client @25

supplier @9

line\_style 0

x\_offset FALSE)

(object TransView "" @29

stereotype TRUE

line\_color 3342489

quidu "587D094000D0"

client @4

supplier @25

line\_style 0

x\_offset FALSE)

(object SynchronizationView "$UNNAMED$3" @30

location (1968, 1840)

font (object Font

size 10

face "Arial"

charSet 204

bold FALSE

italics FALSE

underline FALSE

strike FALSE

color 0

default\_color TRUE)

label (object ItemLabel

Parent\_View @30

location (2130, 1769)

nlines 2

max\_width 300

label "")

icon\_style "Icon"

line\_color 3342489

quidu "587D0A760274"

autoResize TRUE

sync\_flow\_direction 1

sync\_is\_horizontal TRUE)

(object TransView "" @31

stereotype TRUE

line\_color 3342489

quidu "587D0A7B00D6"

client @21

supplier @30

line\_style 0

x\_offset FALSE)

(object TransView "" @32

stereotype TRUE

line\_color 3342489

quidu "587D0A7F0082"

client @12

supplier @30

line\_style 0

x\_offset FALSE)

(object TransView "" @33

stereotype TRUE

line\_color 3342489

quidu "587D0A8302DC"

client @19

supplier @30

line\_style 0

x\_offset FALSE)

(object TransView "" @34

stereotype TRUE

line\_color 3342489

quidu "587D0AA30386"

client @30

supplier @23

line\_style 0

x\_offset FALSE)

(object TransView "" @35

stereotype TRUE

line\_color 3342489

quidu "587D0AAD001C"

client @23

supplier @24

line\_style 0

x\_offset FALSE)))))

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Скопцов А. П. Локальная сеть в школе и малом офисе: создание и поддержка своими силами. – М.: [КУДИЦ-Пресс](http://www.knigalub.ru/publisher/141189.html), 2000. – 240 с.
2. Гук М. Аппаратные средства локальных сетей. – Спб.: Питер, 2000. – 544 c.
3. Капустин Н.М., Кузнецов П.М., Схиртладзе А.Г. Автоматизация производственных процессов в машиностроении. Учебник для втузов. – М.: Высшая школа, 2004. – 415 c.
4. [Минаев](http://www.knigalub.ru/author/127979.html) И. Я. 100% самоучитель. Локальная сеть своими руками. – М.: [Технолоджи-3000](http://www.knigalub.ru/publisher/136066.html), 2003. – 320 с.
5. VitoJ. AmatoCiscoNetworkingEssentials, Volume 2. – М.:Вильямс, 2002. – 464 с.
6. Единое окно образовательный ресурс [Электронный pecypc] Режим доступа:

http: // window.edu.ru., свободный .

1. Вишневский В.М. Теоретические основы проектирования компьютерных сетей. – М.: Техносфера, 2003. – 324 с.
2. Столингс В. Современные компьютерные сети. 2-е изд. – М.: Высшая школа, 2005. – 784 с.
3. Книга: Энциклопедия. – М.: Большая российская энциклопедия, 1999. – 800 с.
4. Ермишин П. Г. Основы экономической теории. – Спб.: Питер, 2004. – 340 с.
5. Каталог оборудованя cisco [Электронный pecypc] Режим доступа:

http: //[www.cisco.com](http://www.cisco.com), свободный.

1. Юрлов Ф.Ф. Технико-экономическая эффективность сложных радиоэлектронных систем. — М.: Сов. радио, 1980. — 280 с.
2. Юрлов Ф.Ф., Плеханова А.Ф., Маркитанов М.Ю. Выбор эффективных решений в экономике. — Н. Новгород: Изд-во НГТУ, 2004. — 115 с.
3. Сайт microsoft.com.
4. Гофман В., Хоменко А. Работа с базами данных в Delphi – Спб.: БХВ-Петербург, 2000. – 643 с.
5. Бобровский С.И. Delphi 7: Учебный курс. – Спб.: Питер, 2005. – 736 с
6. Jeff Cogswell LEARN DELPHITM 2 DATABASE PROGRAMMING TODAY!. – IDG Books Worldwide, Inc., 1996. – 689 с.
7. Фёдоров А. Г. Delphi 3.0 для всех.–М.: КомпьютерПресс, 1998.–360 с.