Министерство науки и высшего образования Российской Федерации

Южно-Уральский государственный университет

Институт естественных и точных наук

Кафедра прикладной математики и программирования

004.4(07)

Д304

**ПРОЕКТИРОВАНИЕ КОРПОРАТИВНЫХ ИНФОРМАЦИОННЫХ СИСТЕМ В СРЕДЕ RADIXWARE**

Методические указания по выполнению курсового проекта

Челябинск

Издательский центр ЮУрГУ

2021

УДК 004.434(075.8)

Д304

Одобрено

учебно-методической комиссией

института естественных и точных наук

Рецензент

Н.В. Плотникова

|  |  |
| --- | --- |
| Д304 | **Проектирование корпоративных информационных систем в среде RadixWare:** методические указания по выполнению курсового проекта / сост. А.К. Демидов, Т.Ю. Оленчикова, М.Ю. Сартасова. – Челябинск: Издательский центр ЮУрГУ, 2021. – 51 с. |
|  | В методическом издании рассматриваются этапы разработки курсового проекта по дисциплине «Моделирование и проектирование  корпоративных информационных систем», приводятся критерии оценки, требования к оформлению и содержанию пояснительной записки и презентации.  Методическое издание содержит варианты тем для курсовых проектов, рекомендации по технологии разработки проекта с использованием среды разработки RadixWare, по организации совместной работы над проектом, описываются этапы разработки, примеры оформления результатов каждого этапа и критерии их оценки по балльно-рейтинговой системе ЮУрГУ. Рассмотрены порядок защиты курсовой работы, структура доклада и презентации. В приложениях приводятся примеры оформления частей пояснительной записки.  Методическое издание предназначено для магистрантов направления 01.04.02 «Прикладная математика и информатика». |

УДК 004.434(075.8)

# 1 ЭТАПЫ ВЫПОЛНЕНИЯ КУРСОВОГО ПРОЕКТА

Курсовой проект (КП) по дисциплине «Моделирование и проектирование корпоративных информационных систем» представляет собой учебно-практическое исследование, предназначенное для систематизации, углубления и закрепления знаний, полученных студентом в процессе изучения данной дисциплины в соответствии с учебным планом.

Цель курсового проекта – приобретение студентом практических навыков по разработке корпоративных информационных систем (КИС) в среде и по технологии проектирования RadixWare. Последовательно рассматриваются все этапы разработки КИС: формулирование требований к системе, создание инфраструктуры проекта, построение модели системы в среде RadixWare, тестирование, документирование, выпуск дистрибутива и развертывание системы на базе заказчика. Для реализации данной цели студент должен:

* уметь проводить системный анализ объекта автоматизации и на его основе формулировать требования к разрабатываемой КИС;
* владеть технологией командной разработки сложных систем, уметь разделять систему на модули, создавать инфраструктуру проекта c использованием системы управления версиями Subversion (SVN);
* владеть технологией проектирования баз данных (БД);
* знать основы обеспечения целостности и безопасности данных в многопользовательской среде, разграничения прав доступа пользователей;
* иметь навыки администрирования и ведения БД в СУБД ORACLE;
* уметь распределять задачи между сервером приложений и сервером баз данных;
* иметь навыки построения и документирования модели КИС по технологии RadixWare;
* уметь создавать экранные формы и отчеты;
* уметь программировать пользовательские функции;
* показать умение применять теоретические положения к решению практических задач, четко формулировать свои мысли и предложения.

Курсовой проект ориентирован на разработку корпоративной информационной системы трехзвенной архитектуры: клиент – сервер приложений – сервер баз данных, предназначенной для решения конкретных прикладных задач. Проект выполняется в среде RadixWare на языке программирования Java, в качестве СУБД используется Oracle RDBMS. Для успешного выполнения проекта под руководством преподавателя создается команда исполнителей и между ними распределяются задачи по реализации проекта.

Перечень этапов, распределение времени и баллов приведены в таблице 1.

Таблица 1 – Продолжительность и трудоемкость этапов

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Наименование этапа | Продолжительность | Часы | Баллы | Вес | Результат работы |
| Разработка технического задания | 1 неделя | 10 | 10 | 1 | Раздел 4 Техническое задание |
| Создание инфраструктуры и модулей проекта. Распределение задач между исполнителями | 2 недели | 20 | 10 | 2 | Раздел 5 .1 Архитектура проекта |
| Разработка БД | 2 недели | 20 | 10 | 2 | Раздел 5.3.1 Структура БД |
| Разработка системы, интерфейса пользова-теля, тестирование | 5 недель | 40 | 10 | 3 | Разделы 5.3.2–5.3.4 Руководство пользователя  Раздел 5.2 Программа и методика приемо-сдаточных испытаний |
| Выпуск дистрибутива | 1 неделя | 5 | 10 | 1 | Раздел 6 Пояснительная записка |
| Защита проекта | 1 неделя | 3 | 10 | 1 | Презентация к докладу |
| ВСЕГО | 12 недель | 100 |  | 100 |  |

Оценка за курсовой проект выставляется по числу набранных баллов согласно «Положению о балльно-рейтинговой системе ЮУрГУ» (БРС) [1].

# 2 ТЕМЫ КУРСОВЫХ ПРОЕКТОВ

Для каждой темы из списка приведен краткий перечень объектов, функций и отчетов, которые необходимо подготовить во время выполнения курсового проекта.

## Система отслеживания ошибок

Назначение системы:

Система отслеживания ошибок (Bug Traсking System, BTS) помогает разработчикам следить за ошибками. Когда пользователь или тестировщик замечает ошибку, необходимо собрать о ней максимальное количество доступной информации.

Как правило, BTS позволяет хранить информацию об ошибке в следующем виде:

* название системы (подсистемы) с ошибкой
* кто сообщил о проблеме;
* дата и время, когда была обнаружена проблема;
* серьёзность проблемы;
* описание неправильного поведения программы;
* кто занимается устранением проблемы;
* состояние ошибки.

BTS использует тот или иной вариант «жизненного цикла» ошибки, стадия которого определяется текущим состоянием, или статусом, в котором находится ошибка. Типичный жизненный цикл дефекта:

1. *новый* – дефект зарегистрирован тестировщиком;
2. *назначен* – назначен ответственный за исправление дефекта;
3. *разрешён* – дефект переходит обратно в сферу ответственности тестировщика. Как правило, статус *разрешен*, сопровождается резолюцией, например:

* *исправлено* (исправления включены в версию *такую-то*);
* *дубль* (повторяет дефект, уже находящийся в работе);
* *не исправлено*(работает в соответствии со спецификацией, имеет слишком низкий приоритет, исправление отложено до следующей версии и т.п.);
* *невоспроизводимо* (запрос дополнительной информации об условиях, в которых дефект проявляется).

1. далее тестировщик проводит проверку исправления, в зависимости от чего дефект либо снова переходит в статус назначен (если он описан как исправленный, но не исправлен), либо в статус закрыт, открыт повторно (дефект вновь найден в другой версии).

Система может предоставлять администратору возможность настроить, какие пользователи могут просматривать и редактировать ошибки в зависимости от их состояния, переводить их в другое состояние или удалять. В корпоративной среде BTS может использоваться для получения отчётов, показывающих продуктивность программистов при исправлении ошибок. Однако, часто такой подход не даёт достаточно точных результатов, потому что разные ошибки имеют различную степень серьёзности и сложности. При этом серьёзность проблемы не имеет прямого отношения к сложности устранения ошибки.

Система должна предоставлять отчеты о статистике исправления ошибок за определенный период времени, о состоянии системы на текущий момент времени. Уведомление пользователей системы о текущем состоянии задачи исправления ошибки по e-mail.

Пользователи системы:

* администратор – настройка функций и прав пользователей системы;
* менеджер – получение сообщений о багах и выдача заданий программистам на их устранение;
* программисты – устранение ошибок;
* пользователи – информирование о возникших ошибках.

Исходные данные:

1) Web-интерфейс для описания проблемы – для пользователя.

2) Desktop-интерфейс – для администратора, менеджера, программиста, тестировщика.

## Система учета оплаты услуг поставщиков электроэнергии

Назначение системы:

* автоматизация бизнес-процессов энергосбытовой деятельности в части расчетов с потребителями – юридическими и физическими лицами за потребленную электроэнергию;
* ведение базы данных полной истории взаимоотношений с клиентами;
* автоматизация работы с потребителями-должниками: начисление пени
* обеспечение доступа клиентов к учетным данным для оперативного контроля их достоверности, ввода показаний приборов учета контроль оплаты в личном кабинете абонента
* электронный документооборот с потребителями электроэнергии по e-mail
* формирование аналитической и сводной отчетности

Пользователи системы:

* администратор – настройка функций и прав пользователей системы;
* менеджер – внесение изменений в настройки базы данных: ввод пользователей, номеров счетчиков, тарифов, настройка WorkFlow-процессов (оплата, начисление, информирование);
* пользователи системы – передача показаний счетчиков, просмотр состояния счета.

Исходные данные:

1) Web-интерфейс пользователя (личный кабинет)

2) Desktop-интерфейс для администратора и менеджера

3) Обмен информацией о платежах и начислениях с системой Город через файлы[[1]](#footnote-2)

## Сервис доставки еды

Назначение системы:

Использование постоянного штата курьеров для небольших кафе слишком дорого, поэтому они могут воспользоваться внешним специализированным сервисом по доставке.

Задача ПО:

* автоматизация бизнес-процессов от момента принятия заказа до доставки клиенту, отслеживание этапов выполнения заказа, назначение маршрутов курьеров;
* формирование аналитической и сводной отчетности по курьерам и заказам на доставку.

Пользователи системы:

* администратор – настройка функций и прав пользователей системы;
* менеджер – управление курьерами;
* администраторы кафе — запросы на доставку, изменение состояния заказа;
* курьеры — получение маршрута, изменение состояния заказа;
* клиенты – отслеживание состояния заказа.

Исходные данные:

1) Web-интерфейс для курьеров

2) Desktop-интерфейс для администратора и менеджера

3) Интерфейс для получения информации о состоянии заказа через REST API или SOAP в мобильном приложении (само мобильное приложение не реализуется).

## Оптово-розничный магазин

Назначение системы.

Если оформление розничной сделки заключается в выдаче покупателю кассового чека, то для реализации товаров оптом продавцу необходимо оформить еще несколько документов. К ним относятся накладные и счета-фактуры. Отличия этих документов описаны в таблице 2. Кроме того, необходимо вести журналы учета полученных и выставленных счетов-фактур, книги покупок и продаж.

Пользователи системы:

* администратор – настройка функций и прав пользователей системы;
* менеджер по продажам – оформление документов: накладных и счетов-фактур, печать кассовых чеков при покупке в розницу;
* кладовщик – выдача товаров по накладным (информация о месте хранения);
* покупатели – получение информации о наличии товаров и ценах.

Таблица 2 – Документы для оптовой торговли

|  |  |  |
| --- | --- | --- |
| Критерий для сравнения | Счет-фактура | Товарная накладная |
| Назначение | Документ нужен покупателю, чтобы принять к вычету НДС | Документ подтверждает факт передачи товара поставщиков покупателю |
| Когда оформляют | На оплату товаров, работ, услуг | Только при отгрузке товаров |
| Форма | Оформляют на унифицированном бланке из постановления Правительства от 26.12.2011 № 1137 | По унифицированной форме ТОРГ-12 |
| Дата выставления | Не позднее 5 календарных дней со дня получения предоплаты или отгрузки товаров | Составляют на дату отпуска товаров со склада |
| Количество экземпляров | Поставщик выписывает документ в одном экземпляре | Два экземпляра. Один – продавцу, второй – покупателю |
| Срок хранения | Не менее 4 лет | Не менее 5 лет |

Исходные данные:

1) Web-интерфейс для покупателей

2) Desktop-интерфейс для администратора и менеджеров, кладовщиков

3) Интерфейс с ККМ https://integration.atol.ru/api/

## Гостиница

Назначение системы:

Гостиница содержит номера, для которых указывается: класс номера, максимальное число проживающих, стоимость за номер по выходным и будням. Ведется учет гостей гостиницы (Ф.И.О., паспорт, дата заезда, предполагаемая дата отъезда, сумма предоплаты). Возможно бронирование номеров в гостинице.

Основная бизнес-функция: учет проживающих в номерах, сдача мест в гостинице гостям.

Отчеты: наличие свободных номеров на заданный диапазон дат в гостинице и их характеристика, формирование счета за проживание.

Пользователи системы:

* администратор гостиницы – настройка функций и прав пользователей системы, характеристика номеров и их стоимость, просмотр отчетов, например, статистика востребованности номеров;
* менеджер по работе с клиентами – оформление документов для гостей, печать кассовых чеков при выезде гостей;
* гости – получение информации о свободных номерах, их характеристиках и ценах.

Исходные данные:

1) Desktop-интерфейс для администратора

2) Web-интерфейс для гостей

## Отдел кадров

Предприятие имеет определенную номенклатуру должностей (код, наименование, тариф). Предприятие делится на подразделения, подразделения на отделы. Каждое подразделение или отдел имеют руководителя. Каждое подразделение или отдел имеют штатное расписание. Должности по штатному расписанию занимают сотрудники. Сотрудник имеет атрибуты (Ф.И.О., год рождения). Сотрудник имеет послужной список. Сотрудник имеет поощрения и взыскания. Сотрудник имеет некоторую историю получения образования и повышения квалификации.

Основная бизнес-функция: распределение сотрудников по подразделениям согласно штатному расписанию, ведение электронных трудовых книжек сотрудников.

Отчеты: послужной список сотрудника: должности, награды, поощрения, выговоры; штатное расписание подразделения на указанную дату с отметкой незанятых штатных единиц

Пользователи системы:

* администратор – настройка функций и прав пользователей системы;
* менеджер – учет сотрудников, отчеты, внесение записей в трудовую книжку;
* сотрудники – просмотр информации о свободных должностях, просмотр записей в своей трудовой книжке

Исходные данные:

1) Desktop-интерфейс для пользователя и администратора

2) Web-интерфейс для сотрудников

Студенты могут предложить свои темы КП, согласовав их с руководителем.

# 3 Технология проектирования КИС в RadixWare

RadixWare представляет собой технологию и CASE-средство для разработки масштабных проектов КИС в рамках трехуровневой архитектуры: сервер СУБД – сервер приложений – клиентские приложения. Основу приложений составляют SOA-сервисы (сервис-ориентированная архитектура).

Укрупненно процесс создания КИС можно представить в виде следующих стадий (ГОСТ 34.601-90):

* предпроектная;
* проектная;
* послепроектная.

В рамках данного курсового проектирования рассмотрим этапы создания КИС «с нуля» под конкретного заказчика. В этом случае при создании КИС разработка ведется по модификации водопадной, а точнее по V-модели (рисунок 1), жизненного цикла.

***Предпроектная стадия*** включает:

* аудит предприятия и формирование требований к КИС
* разработку концепции КИС
* разработку и утверждение технического задания на создание КИС.

Данная стадия не поддерживается платформой RadixWare и выполняется с применением других средств моделирования архитектуры ПО: SADT, UML, IDEF0, IDEF3, DFD и т.п. Результатом работы данной стадии является *техническое задание* на разработку КИС. Состав этого документа регламентируется стандартом ГОСТ 34.602 (существует аналогичный международный стандарт IEEE830-1998. Методика составления спецификаций требований к программному обеспечению).

В рамках данного курсового проекта мы предлагаем использовать язык UML [2]. Однако это не обязательно. Требования к разработке технического задания рассмотрены в п. 4.

Остальные 2 стадии проектирования полностью поддерживаются и выполняются в среде RadixWare, что обеспечивает высокую скорость и качество разработки ПО.

Отличительной особенностью среды RadixWare является модельно-ориентированный подход к разработке программного обеспечения. Модель платформы RadixWare является *единственной* формой представления проекта. Никаких промежуточных представлений, доступных разработчикам, не существует. Вся инфраструктура платформы RadixWare, фактически, является контейнером, в котором живет эта модель. Именно модель автоматически преобразуется в исполняемый код инфраструктурой RadixWare.

Результатом ***проектной стадии*** является технический проект (Solution Design), рабочий проект (готовый программный код плюс эксплуатационная документация), а также написанная и отлаженная на контрольных примерах система.

Разработка технического проекта традиционно разделяется на три этапа:

* разработка архитектуры ПО;
* эскизное проектирование;
* рабочее проектирование.

Поскольку в среде RadixWare фактически необходимо разработать только модель системы, то эти этапы объединяются в один процесс – ***проектирование и реализация*** ПО. В среде Radix мы разрабатываем модели узлов будущей системы, одновременно создавая документацию по принятым решениям. Модель может быть доработана, неудачные решения – изменены, что сразу отражается в реализации системы.

В частности, в модели КИС должны присутствовать узлы (подмодели):

* состав подсистем будущей системы, разделение запланированных функций системы между подсистемами (модулями), описание способов, моделей и протоколов взаимодействия будущей системы с внешними системами;
* модель структуры базы данных системы;
* состав рабочих мест информационной системы и набор доступных им функций;
* модели экранных форм будущей системы с перечнем полей и особенностями работы с их значениями;
* модели печатных документов будущей системы, с перечнем полей, макетом формы документа и особенностями их заполнения;
* специальные алгоритмы обработки информации, если таковые присутствуют в будущей системе. Последнее обычно требуется редко, так как ядро системы RadixWare содержит большой набор готовых решений.

После окончания этапа проектирования, система должна быть полностью готова к установке у заказчика, который к этому времени должен подготовить (приобрести и установить) необходимое аппаратное и программное (оговаривается в ТЗ), необходимые для работы системы. Система переходит в стадию ***опытной эксплуатации и сопровождения*.** Систему устанавливают у заказчика и производят приемочное тестирование и доводку разработанной системы. На этом же этапе производится разработка должностных инструкций по эксплуатации для персонала будущей информационной системы (в первую очередь руководства пользователей), а также обучение персонала заказчика работе с системой.

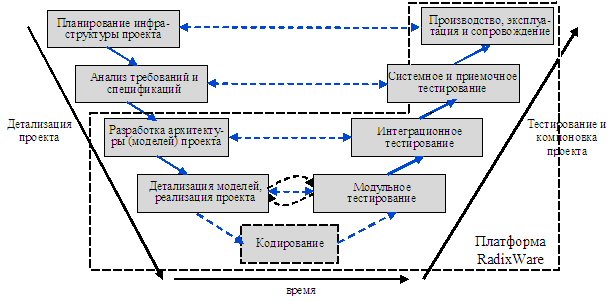


Рисунок 1 – V-model процесса разработки ИС

На рис. 1 выделены этапы модели ЖЦ ПО, поддержка которых встроена в технологию RadixWare. Левая ветвь V-модели представляет модель ИС, её развитие от черновых требований клиента, через определение концепции и до описания решения и полностью построенного образца. Правая ветвь V-модели представляет последовательность сборки системы и ее тестирования. Существует непосредственная зависимость между действиями на левой и правой ветвях V-модели. Это сделано специально. Методы проверки (тестирования), которые будут использоваться в правой ветви V-модели должны быть определены одновременно с разработкой требований на левой ветви, в противном случае могут быть созданы требования, которые не могут быть проверены.

Например, «быть дружественным к пользователю» является подходящим требованием, но его проверить невозможно. Вместо этого, требование, которое утверждает, что на экране компьютера может быть «не более пяти строк текста, набранных 14 шрифтом Arial» определяет один пользовательский критерий требования «быть дружественным к пользователю» в измеримых величинах.

Тестовые модули разрабатываются одновременно и согласуются с разработкой соответствующего компонента системы по горизонтальным линиям на рис. 1. При запуске системы на платформе RadixWare тестовые модули запускаются автоматически, гарантируя правильность функционирования и сборки приложения. Рекомендуемая полнота покрытия кода тестами – около 75%. Впрочем, это касается только вновь разрабатываемого кода. Компоненты ядра системы уже снабжены Unit-тестами.

При необходимости доработки системы возврат к предыдущим этапам осуществляется по горизонтальным линиям.

Платформа RadixWare предназначена для разработки приложений, использующих трехзвенную архитектуру (рисунок 2). Трехзвенная архитектура «живет» одновременно в трех местах: на сервере базы данных, на сервере приложений и в клиентской части. Соответственно модель ИС учитывает эту архитектуру.

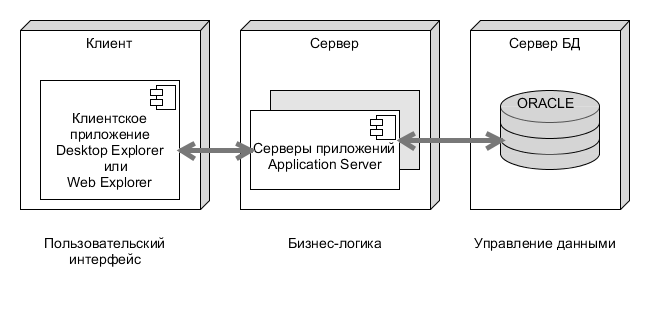


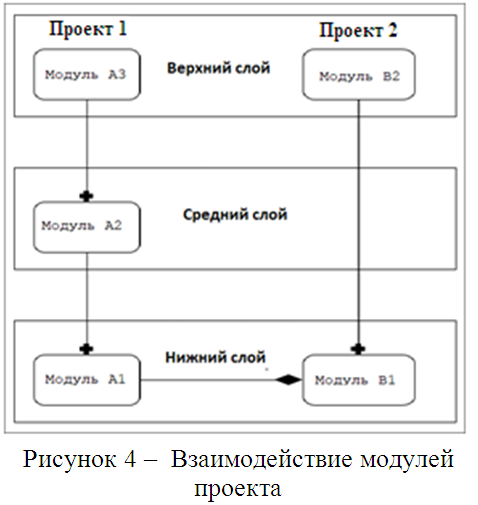
Рисунок 2 – Архитектура стандартного приложения RadixWare

Модель системы состоит из *слоев* (рисунок 3).

***Слой*** – это самостоятельная единица проекта, обладающая своей собственной функциональностью. Слоев в проекте может быть несколько, и, помимо прочего, они позволяют разграничить зоны ответственности различных команд разработчиков. Самым первым, *базовым*, слоем любого проекта является, собственно, система RadixWare (ветвь Radix на рис.3). Остальные слои проекта строятся на ее основе и *расширяют* функциональность базового слоя (ветвь Radinsk). Этот механизм подобен механизму *наследования* в объектном программировании. В этом плане слой Radix можно рассматривать как подобие класса Object в языке Java, а новые слои – как подобия дочерних классов. Количество слоев в проекте может быть сколь угодно большим, однако в отличие от классов, у данного слоя в данном проекте может быть *только один* наследник. Если от данного слоя требуется еще один наследник, это будет *другой* проект. Таким образом, слои в каждом конкретном проекте образуют не дерево, а *цепочку* наследования.

Базовый слой любого проекта, а именно система RadixWare, состоит из четырех компонент, которые называются ***сегментами***:

* **Kernel** – сегмент ядра системы;
* **DDS** (Database Definition Segment) описывает модель структуры БД приложения и предназначен для сервера СУБД (ORACLE, Postgres).
* **ADS** (Application Definition Segment) описывает модель приложения, которая исполняется в среде Application Runtime Environment (ARTE), входящей в состав RadixWare Server, а также в RadixWare Explorer.
* **UDS** (User Definition Segment) содержит элементы, которые будут добавляться или изменяться под запросы пользователя в ходе эксплуатации и сопровождения приложения.

Разделение проекта по сегментам – это, в некотором смысле, разделение проекта по местам локализации соответствующего кода. Сегмент ядра системы – это набор классов, обеспечивающих функциональность системы RadixWare. Классы **kernel-сегмента** выполняются непосредственно в среде виртуальной Java-машины и обеспечивают функционирование всех остальных компонент системы. Сегмент ядра системы есть у слоя Radix, у других же слоев его наличие *не приветствуется*, поскольку он, в отличие от других сегментов, не содержит модели, тем самым сводя на нет одно из главных преимуществ Radix.

Все сущности, описанные в сегменте **DDS**, в готовом проекте будут располагаться на сервере базы данных.

В готовом проекте классы сегмента **ADS** располагаются либо на сервере приложения (RadixWare Server), либо на клиентской части (RadixWare Explorer и/или RadixWare WEB Presentation Server).

**Сегмент UDS** отличается от сегмента ADS набором классов, которые внутри него можно создавать (он более узкий, чем набор классов сегмента ADS). Основное его назначение – хранение сущностей, разработанных пользователями вашей системы.

Любой проект, в свою очередь, может послужить начальным (базовым, родительским) слоем для еще одного проекта (дочернего), расширяющего его новой функциональностью, и т. д. Поэтому первый этап жизненного цикла в RadixWare всегда начинается с установки всех слоев, которые потребуются для работы вновь создаваемой системы. После установки всех требуемых слоев выполняется собственно подготовка нового проекта.

Сущности каждого слоя могут использовать сущности нижележащего слоя. В дереве проекта, на рис. 3 дочерний слой (Radinsk) расположен ниже родительского (Radix). Сущности, определенные в дочерних слоях, могут переписывать сущности родительских слоев. Таким образом, дочерний слой может не только добавлять новую функциональность, но и изменять поведение родительских слоев (чего однако делать не рекомендуется).

***Модульность*** – это свойство системы, которая была разложена на внутренне слабо связанные между собой модули. Разделение программы на части (модули) до некоторой степени позволяет уменьшить ее сложность выполнять разработку параллельно. Внутри модульной программы создаются множества хорошо определенных и документированных интерфейсов. Каждый сегмент слоя содержит набор модулей.

Установка родительских слоев

2

Создание нового проекта

3

Разработка, тестирование модулей

4

Интеграционное тестирование

5

Выпуск релиза

6

Поставка и сопровождение системы у заказчика

8

Рисунок 5 – Реализация проекта в RadixWare

Создание инфраструктуры про

екта

Функциональное   
тестирование, создание дистрибутива

7

1

Модульная организация позволяет решать следующие задачи:

* разделение логически независимых частей;
* управление областью видимости: в область видимости каждого модуля входят только те модули, которые указаны в списке зависимостей;
* поддержка создания дистрибутивов: достаточно указать, какие модули необходимо включить в дистрибутив, и зависимые модули будут включены в него автоматически.

Систему модулей необходимо продумать при командной разработке проекта, поскольку в противном случае разработчики системы будут блокировать работу друг друга.

Приведем последовательность работ по проектированию и созданию ИС на платформе RadixWare. Глобально процесс создания ИС в RadixWare состоит из 8 этапов (рисунок 5). Что включает в себя процесс подготовки нового проекта?

Шаг 1. Подготовка репозитория проекта. Разработка КИС не удел одиночек, сделать подобного рода системы возможно, только работая в команде. А для командной работы, безусловно, необходим репозиторий проекта. Нужен он также и для хранения истории релизов. Результатом подготовки проекта к работе является создание внутри репозитория (SVN) необходимой структуры для хранения проекта. Кроме этого, стадия подготовки нового проекта включает в себя получение доступа к SQL-базе данных в качестве которой в настоящее время используются СУБД ORACLE или PostgreSQL.

Выполнение необходимых действий на этих этапах обеспечивается средством платформы RadixWare, называемым RadixWareManager. После этого проект переводится в режим разработки.

Шаги 2-5 отвечают за разработку проекта. Стадия разработки проекта включает в себя:

* создание слоя (проекта);
* планирование модулей;
* проектирование структуры базы данных: создание модуля DDS, захват модуля DDS, создание сущностей DDS (таблиц, индексов, триггеров, функций и хранимых процедур БД в соответствии с моделью предметной области), фиксация DDS в базу данных;
* создание в модуле ADS классов, отвечающих за взаимодействие с объектами БД;
* создание в модуле ADS экранных форм и отчетов;
* создание дерева навигации по проекту;
* интеграционное тестирование.

Поддержку процесса разработки всех этих компонент обеспечивает средство разработки, называемое RadixWareDesigner. Обратите внимание, что по результатам модульного тестирования и интеграционного тестирования может потребоваться вернуться назад и изменить модель ИС. Более подробно выполнение этих этапов описано в документации разработчика RadixWare RadixWare Programmer Guide/Quick Start Guide/Developing an Application on RadixWare Platform [3].

Шаг 6. Готовый (по мнению разработчиков) вариант проекта после окончания процесса разработки клонируется, нумеруется и переводится в состояние «заморозки». Делается это с помощью того же RadixWareDesigner и называется «Выпуск релиза».

Шаг 7. Затем релиз проекта переводится в состояние тестирования. Перевод релиза в это состояние выполняется с помощью средства разработки RadixWareManager. После успешного окончания приемочных испытаний релиз переводится в состояние готовности к выпуску. Это, фактически, означает завершение стадии рабочего проектирования в жизненном цикле системы. После успешных испытаний платформа RadixWare создает вариант поставки продукта (в пределе – для каждой инсталляции) – дистрибутив проекта со своим набором поставляемых модулей. Дистрибутивы могут быть выгружены в файл, заархивированы, подписаны и переданы заказчикам.

Шаг 8. Модули ИС, поставленные в виде дистрибутива, могут быть развернуты на полигоне заказчика. В жизненном цикле системы это соответствует началу опытной эксплуатации системы. Развертывание продукта выполняется путем загрузки готового дистрибутива в репозиторий заказчика и настройки его под конфигурацию. Делается такое развертывание средствами RadixWareManager и SVN. Одновременно с развертыванием продукта производится и развертывание у заказчика платформы RadixWare. Ошибки, выявленные на стадии эксплуатации системы, обрабатываются так же, как на стадии разработки, т.е. посредством исправления сопровождаемых версий системы и выпуска обновлений (шаги 3-8).

Далее рассматриваются требования к выполнению этапов курсового проекта. Пример задания на курсовой проект «Библиотека» приведен в приложении А.

# 4 техническое задание на разработку КИС

Техническое задание (ТЗ) составляется разработчиком (проектировщиком) на основе исходных данных, предоставленных заказчиком, содержит основные технические требования к создаваемой системе и служит основанием для проектирования. ТЗ содержит следующую информацию:

* классы пользователей и соответствующие диаграммы (алгоритмы) бизнес-процессов;
* модели (диаграммы) процессов прикладной деятельности и соответствующие перечни функциональных задач КИС;
* классы объектов предметной области и соответствующие диаграммы «сущность-связь», отражающие информационную модель этой предметной области;
* топология расположения подразделений и пользователей, обслуживаемых данной КИС;
* параметры защиты данных, информации и самой системы;
* сведения об очередности создания системы, о выделяемых ресурсах, директивных сроках проведения отдельных этапов работы
* сведения об организационных процедурах и мероприятиях по приемке этапов, защите проектной информации и т. д.

Техническое задание составляется в результате системного анализа объекта автоматизации, основных бизнес-процессов заказчика, информационной модели предметной области, интерфейсов с пользователями и другими системами.

После согласования с заказчиком ТЗ является основным нормативным документом для разработчика.

Примерный состав документа.

1. Введение

2. Термины и определения

3. Описание технических решений

3.1. Состав автоматизированных рабочих мест (АРМ) системы

3.2. Информационная модель системы

3.3. Организация поддержки основных технологических процессов в системе

3.4. Интерфейсы обмена данными с внешними системами и оборудованием

3.5. Требования к реализации АРМов системы

4. Порядок контроля и приемки системы

5. Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие

6. Требования к документированию

7. Приложения

Пример оформления ТЗ приведен в приложении Е.

Критерии оценки этапа для БРС:

|  |  |
| --- | --- |
| Содержание работы | Баллы |
| – приведен список объектов автоматизации и состав функций АРМов | 2 |
| – описаны основные бизнес-процессы; | 2 |
| – построена ER-диаграмма информационной модели системы; выявлены пользовательские ограничения целостности БД, роли и права доступа к информации БД; | 2 |
| – определены интерфейсы обмена данными с внешними системами и оборудованием; | 2 |
| – приведены эскизы форм и отчетов системы | 2 |
| *Итого,* максимум | 10 |

# 5 ПРОЕКТИРОВАНИЕ И РЕАЛИЗАЦИЯ СИСТЕМЫ

5.1 Архитектура проекта

Приступая к реализации проекта по ТЗ, необходимо правильно организовать совместную работу группы студентов над проектом. Рассмотрим основные моменты

Критерии успешности проекта.

Задача проекта – достижение конкретной бизнес-цели, при соблюдении ограничений «железного треугольника: функциональность – стоимость – время» Это означает, что ни один из углов треугольника не может быть изменен без оказания влияния на другие. Например, чтобы уменьшить время, потребуется увеличить стоимость и/или сократить функциональность. Успешный проект отвечает следующим критериям.

1. Выполнен в соответствие со спецификациями.

2. Выполнен в срок.

3. Выполнен в пределах бюджета (в учебном проекте – не учитывается).

4. Каждый участник команды уходил с работы в 18:00 с чувством успеха.

***5.1.1 Создание команды разработчиков, распределение ролей***

Приведем один из возможных способов организации команды проекта, наиболее подходящий, по нашему мнению, для реализуемого учебного проекта.

На первом этапе все студенты группы участвуют в «мозговом штурме» по анализу требований заказчика (составлению технического задания) и выработке концепции системы, что способствует пониманию цели всеми участниками проекта и повышению их активности. Результатом является техническое задание на проект, которое утверждается преподавателем.

Следующим этапом является организация команды исполнителей проекта

Организационная структура команды проекта определяет распределение ответственности и полномочий среди участников проекта, а также обязанностей в отношении отчетности. Основные роли участников проекта:

* 1. *Руководитель (менеджер) проекта*, который реально управляет выделенными на проект ресурсами. Он планирует работы, распределяет задачи среди исполнителей, контролирует сроки и результаты, несет полную ответственность за достижение целей проекта, при соблюдении ограничений.
  2. *Системный архитектор.* Разработка технической концепции системы. Принятие ключевых проектных решений относительно внутреннего устройства (слоев и модулей) программной системы и еѐ технических интерфейсов.
  3. Ответственный за *управление изменениями*, конфигурациями, за сборку и поставку программного продукта
  4. *Тестировщик*. Определение целей и стратегии тестирования. Тестирование продукта. Анализ и документирование результатов
  5. *Проектировщик.* Проектирование компонентов и подсистем в соответствие с общей архитектурой, разработка архитектурно значимых модулей.
  6. Проектировщик *базы данных*
  7. Проектировщик *интерфейса пользователя*
  8. *Разработчик*. Проектирование, реализация и отладка отдельных модулей системы.
  9. *Технический писатель.* Оформление технической документации

Студенты самостоятельно, с помощью преподавателя, распределяют роли участия в проекте. При этом необходимо учесть их уровень, психологические особенности, интересы, квалификацию.

Часто несколько ролей приходится совмещать одному исполнителю. Возможны следующие совмещения ролей:

* руководитель проекта + системный архитектор;
* системный архитектор + разработчик;
* проектировщик + проектировщик базы данных + разработчик;
* проектировщик интерфейса пользователя +проектировщик тестов (+ технический писатель)
* разработчик + технический писатель.

Крайне нежелательно совмещать следующие роли:

* разработчик + руководитель проекта;
* разработчик + проектировщик интерфейсов пользователя;
* разработчик + тестировщик.

При распределении ролей следует равномерно распределить нагрузку по работе над проектом в соответствии с п.5.1.2

***5.1.2 Планирование проекта***

Уточнение содержания и состава работ.

Сложные программные системы разрабатываются через анализ и декомпозицию задач на боле простые подзадачи, которые, в свою очередь, могут быть разделены на еще боле простые подзадачи и так далее. Получается некоторая иерархическая структура работ (ИСР) – дерево, в корне которого находится проект c его требованиями к характеристикам, концепция проекта, а на листьях элементарные задачи или работы, которые надо выполнить, чтобы завершить проект в условиях заданных ограничений. ИСР обеспечивает выявление всех работ, необходимых для достижения целей проекта. ИСР делит проект на подпроекты, пакеты работ, подпакеты. Выполнять декомпозицию работ можно разными способами.

Среда RadixWare задает определенный порядок декомпозиции. На верхнем уровне декомпозиции нашего проекта должны находиться продукты проекта (слои), а на следующем уровне – компоненты, из которых эти продукты состоят сегменты DDS, ADS, UDS. Сегменты состоят из модулей – единиц компиляции, внутри которых создаются элементы системы нижнего уровня, например: таблицы, связи, последовательности – в DDS, а также сущности (Entity Class), деревья навигации, роли, перечисления, отчеты – в ADS.

Декомпозиция на модули и создание необходимой инфраструктуры проекта – задача *системного архитектора*. Модули системы должны быть слабо связаны, чтобы разработчики не мешали друг другу при совместной работе над проектом, захватывая одни и те же ресурсы. На данном этапе выявляется состав и трудоемкость работ по проекту.

Базовое расписание проекта.

После определения трудоемкости работ необходимо определить график их выполнения и общие сроки реализации проекта – составить расписание работ по проекту. Базовое расписание – утвержденный план-график с указанными временными фазами проекта, контрольными точками и элементами иерархической структуры работ. За составление и контроль выполнения графика работ отвечает *руководитель проекта*.

График работ может быть наиболее наглядно представлен *диаграммой Ганта*. В этой диаграмме плановые операции или элементы иерархической структуры работ перечислены с левой стороны, даты отображаются сверху, а длительность операций показана горизонтальными полосками от даты начала до даты завершения. Мы предлагаем дополнить эту диаграмму указанием исполнителей работ и статусом. Базовое расписание это, как правило, элемент контракта с заказчиком. Контрольные точки должны служить точками анализа состояния проекта и принятия решения о необходимости уменьшения количества реализуемых функций. При этом вносятся изменения в ТЗ и в базовое расписание проекта. Контрольные точки должны зримо демонстрировать статус проекта. Контрольная точка «Проектирование завершено» – плохо. Наиболее эффективный подход – метод последовательных поставок: контрольная точка «Завершено тестирование требований 1, 3, 5, 7».

Как правило, в любом проекте существует хотя бы один критический путь, но их может быть несколько. *Критический путь проекта* – самая длинная цепочка работ в проекте. Увеличение длительности любой работы в этой цепочке приводит к увеличению длительности всего проекта. Критический путь может меняться во время исполнения проекта. При исполнении проекта руководитель должен обращать внимание на исполнение задач на критическом пути в первую очередь и следить за появлением других критических путей. Практическая рекомендация: на критическом пути должны стоять работы с нежесткими связями, которые всегда можно перепланировать, если возникает угроза срыва сроков.

Однако мы не можем на этом остановиться, поскольку должны еще учесть ограничение по ресурсам – число разработчиков и возможность доступа к серверу и базе данных. Следовательно, мы должны выполнить *выравнивание ресурсов*. Поскольку одним из критериев успеха проекта является его минимальная длительность, то если мы не хотим ее увеличивать, мы должны выявить критический путь в проекте и не сдвигать работы, которые на нем находятся. Остальные работы распределяются межу исполнителями с учетом их загруженности и возможности распараллеливания работ. В нашем проекте состав и длительность работ определяются архитектором системы, а график работ и назначение исполнителей – руководителем проекта. Подробнее о планировании проекта см. в [4, C.58–66].

***5.1.3 Создание и подготовка проекта для разработки***

Создание инфраструктуры проекта выполняется в программе RadixWareManager и состоит из этапов:

* создание папки проекта в SVN;
* создание проекта в RadixWareManager:
* настройка параметров базы данных и SVN для проекта;
* выгрузка проекта для разработки и конфигурирование командных файлов.

Процесс создания проекта по шагам описан в [5].

Пример отчета по данному этапу приведен в приложении Ж.

Критерии оценки этапа для БРС:

|  |  |
| --- | --- |
| Содержание работы | Баллы |
| – распределены роли между членами команды разработчиков | 1 |
| – распределены работы над проектом в соответствии с ролями | 2 |
| – разработан план-график работ | 2 |
| – создан проект в SVN | 2 |
| – создана база данных ORACLE | 2 |
| – созданы модули системы | 1 |
| *Итого,* максимум | 10 |

**5.2 Программа и методика приемо-сдаточных испытаний**

Прежде чем выпускать программный продукт в эксплуатацию, заказчик должен убедиться в возможности использования разработанного решения, в его безопасности, надёжности, корректном функционировании и проверить другие характеристики, описанные в требованиях ТЗ к продукту. Приемо-сдаточные испытания (ПСИ) особенно актуальны для проектов, на которых выпуск некачественного продукта может привести к значительным финансовым потерям. Как правило именно к таким продуктам относятся КИС.

Цели проведения ПСИ:

* подтвердить соответствие продукта заявленным требованиям.
* выявить ошибки, допущенные при разработке.
* подтвердить возможность внедрения ПО в опытную и промышленную эксплуатацию.

Документ «Программа и методика приемо-сдаточных испытаний» разрабатывается параллельно с разработкой системы и должен быть подготовлен до начала этапа сдачи работ заказчику. Написанием данного документа обычно занимаются специально обученные люди, например, технические писатели, и функциональные тестировщики. Согласно межгосударственному стандарту ГОСТ 19.301-79, документ «Программа и методика испытаний» должен иметь следующую структуру:

Раздел «**Объект испытаний**» – название продукта, область его применения.

Раздел «**Цель испытаний**» – цели ПСИ, которые мы обозначили выше.

Раздел «**Требования к программе**» – требования, подлежащие проверке во время испытаний и перечисленные в «Техническом задании» на программу. Обратите внимание, все требования, которые указываются ТЗ, должны быть продемонстрированы во время испытаний (или хотя бы запланированы).

Раздел «**Требования к программной документации**» – состав программной документации, предъявляемой на испытании, а также специальные требования к документации, если они заданы в ТЗ.

Раздел «**Средства и порядок испытаний**» – перечень технических и программных средств, используемых во время испытаний, а также порядок проведения испытаний. Рекомендуем максимально подробно описать все средства, которые вы планируете использовать для демонстрации. Базы данных, программы для чтения содержимого очередей, консоли, логи, автотесты – всё это должно быть зафиксировано в данном разделе, иначе засчитано и допущено к демонстрации не будет.

Раздел «**Методы испытаний**» – указывается, как именно проводится демонстрация, какие методы используются для проверки реализации того или иного требования и ожидаемый результат по каждой проверке.

Пример оформления документа «Программа и методика приемо-сдаточных испытаний» приведен в приложении Л.

**5.3 Реализация системы**

Реализация системы выполняется в программе RadixWareDesigner и заключается в последовательном создании необходимых компонентов системы, их тестировании. Эти работы выполняются разработчиками, ответственными за них согласно плана работ.

***5.3.1 Разработка базы данных***

Структура базы данных задается в сегменте DDS. Если БД не слишком сложная, то структуру можно задать в одном модуле. В сложных ИС могут быть десятки таблиц, поэтому стоит поделить БД на несколько модулей, чтобы упростить обзор структуры БД и обеспечить некоторую независимость изменений при разработке. Исходная информация для проектирования берется из ER-диаграммы и описания информационной модели, прописанных в ТЗ. Так как ORACLE – это объектно-реляционная СУБД, то существуют некоторые особенности преобразования ER-диаграммы в схему базы данных:

1) сущности информационной модели становятся таблицами БД;

2) в каждой таблице создаются первичные ключи;

свойства (атрибуты) сущностей образуют столбцы в таблицах БД, при этом 3) необходимо определиться с размещением составных и многозначных атрибутов информационной модели в таблице, здесь возможны решения:

* создать столбец для составных типов данных (записей) [6];
* создать дополнительную внешнюю таблицу – справочник;

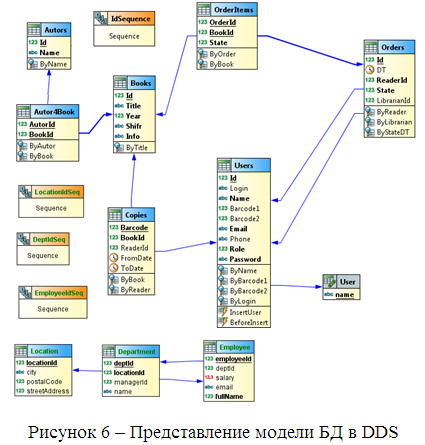
1. создаются индексы – внешние ключи, реализующие связи между таблицами типа «один-ко-многим»;
2. допустимы связи «многие-ко-многим», как показано на рисунке 6 (таблицы с двунаправленными стрелками).

В остальном проектирование БД ничем не отличается от проектирования реляционных баз данных: создаются индексы и триггеры в соответствии с информационной моделью предметной области, назначаются роли пользователям БД и их права, создаются функции, хранимые процедуры, представления для доступа к данным.

Таблицы БД сразу размещаются на диаграмме, что позволяет использовать RadixWareDesigner как инструмент для анализа и проектирования (рис. 6). Анализ созданной БД заключается в проверке непротиворечивости и полноты модели:

* модель не должна содержать *повторяющихся атрибутов*, так как избыточность данных приводит к аномалиям вставки, удаления и обновления. Например, атрибуты «Возраст» и «Дата рождения» не могут одновременно присутствовать в БД даже в разных таблицах. Поэтому столбец «Возраст» необходимо сделать вычислимым [7];
* чтобы убедиться *в полноте и непротиворечивости модели* полезно на схеме базы данных выделить таблицы и связи (граф), участвующие в реализации как минимум основных бизнес-процессов и удостовериться, что вся необходимая информация однозначна и доступна.

Создание и сопровождение БД – это задача *разработчика базы данных*. Следует позаботиться о том, чтобы никто другой из членов команды проекта не имел прав на изменение таблиц и связей между ними.



Для каждой таблицы необходимо задать поля, индексы и триггеры с помощью редактора (рисунок 7). Для целей документирования рекомендуется задать описание для таблицы в целом, каждого поля и триггера.

Подробнее создание слоя и определение структуры БД описано в [8].

После определения структуры БД необходимо сгенерировать документацию с помощью контекстного меню сегмента DDS. Документация формируется в виде HTML-документа. Пример оформления отчета по данному этапу приведен в приложении И.

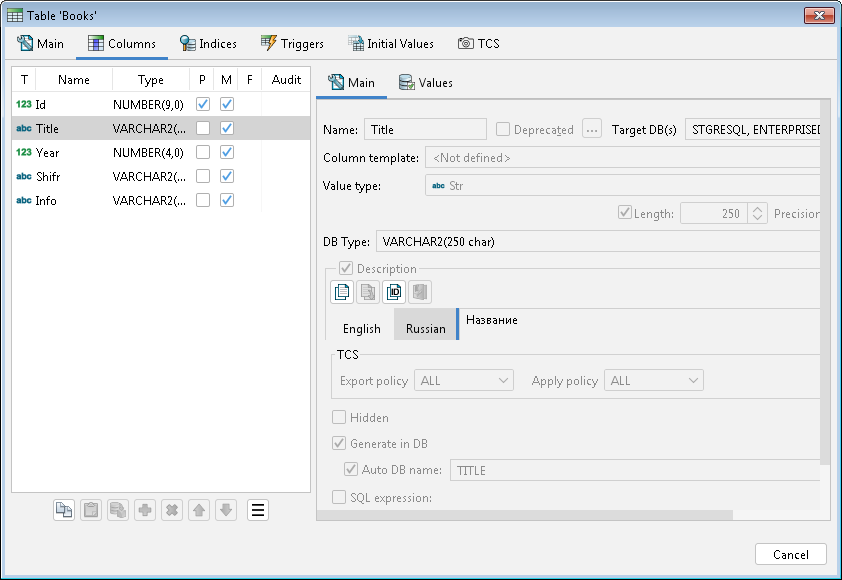


Рисунок 7 – Редактор таблиц

Критерии оценки этапа для БРС:

|  |  |
| --- | --- |
| Содержание работы | Баллы |
| – построена модель предметной области в DDS; | 4 |
| – выполнена проверка соответствия БД 3 НФ и выше | 1 |
| – созданы объекты обеспечения целостности БД на стороне сервера (триггеры, ограничения, роли) | 3 |
| *–* создана документация по сущностям базы данных | 2 |
| *Итого,* максимум | 10 |

***5.3.2 Создание Entity Class и определение интерфейса пользователя***

Для всех таблиц из DDS нужно определить Entity Class. Оптимально делать разделить систему на модули, которые будут соответствовать основным сущностям, используемым в деревьях навигации. Вспомогательные элементы можно поместить в модули к наиболее близким им основным сущностям. Отдельный модуль можно выделить для деревьев навигации всех АРМ.

Определения Entity Class являются ключевыми при разработке ИС. Они содержат элементы, связанные с визуальным представлением сущностей из БД, и могут использоваться в программе для выполнения следующих действий:

* создание сущности;
* загрузку сущности из БД по ее первичному ключу;
* сохранение сущности в БД;
* удаление сущности из БД;
* копирование сущности.

Большинство остальных классов (кроме Report Class и Algorithm Class, имеющих свою свиту вспомогательных классов, и узкоспециализированных типа Test Class) служат для организации взаимодействия с Entity Class.

Entity Class содержит определения свойств (Property), внутренних вспомогательных классов, представления для выбора (Selector Presentation) и редактирования (Editor Presentation), команды (Command), фильтры (Filter) и сортировки (Sorting). Добавление модулей и разработка Entity Class описана в [9]. Создание экранных форм для выбора и редактирования описана в [8]. Создание деревьев навигации описана в [10].

***5.2.3 Создание и публикация отчетов***

Создание отчета выполняется с помощью элемента Report. Разработка и публикация отчета описана в [11]. Создание контекста для контекстных отчетов требует несколько вспомогательных классов и доменов.

***5.2.4 Руководство пользователя***

Руководство пользователя оформляется после завершения тестирования интерфейса пользователя.

Руководство пользователя должно содержать описание дерева навигации для каждого АРМ, скриншоты экранных форм для поиска, выбора и редактирования основных сущностей КИС, особенности взаимодействия и ограничения. Пример оформления руководства пользователя показан в приложении К.

Критерии оценки этапа для БРС:

|  |  |
| --- | --- |
| Содержание работы | Баллы |
| – созданы формы для ввода и редактирования данных; | 2 |
| – реализованы селекторы для просмотра, фильтрации и выбора данных | 2 |
| – реализованы деревья навигации для соответствующих АРМ-ов | 2 |
| *–* реализованы отчеты для соответствующих АРМ-ов | 2 |
| – создание руководства пользователя | 2 |
| *Итого,* максимум | 10 |

# 6 СТРУКТУРА И СОДЕРЖАНИЕ ПОЯСНИТЕЛЬНОЙ ЗАПИСКИ

Окончательное содержание отчета по курсовому проекту определяется самим студентом, но отчет должен содержать обязательные разделы, приведенные ниже обязательные разделы.

* 1. Задание на курсовой проект ( приложение А)
  2. Пояснительная записка к курсовому проекту

*Титульный лист* (приложение Б)

*Аннотация* (приложение В)

*Оглавление* (приложение Г)

*Введение* (приложение Д)

*1. Техническое задание* (приложение Е)

*2. Планирование проекта* (приложение Ж)

*3. Структура базы данных* (приложение И)

*4. Руководство пользователя* (приложение К)

*5. Программа и методика приемо-сдаточных* *испытаний* (приложение Л)

*Заключение* (приложение М)

*Библиографический список* (приложение Н)

*Приложения (по необходимости)*

* 1. Дистрибутив разработанной системы

В приложения можно включать вспомогательный материал, необходимый для изложения результатов курсовой работы в пояснительной записке, например:

* таблицы вспомогательных цифровых данных;
* иллюстрации вспомогательного характера;
* экраны интерфейса;
* схемы алгоритмов и т.д.

Текст отчета должен быть напечатан на одной стороне стандартного листа белой бумаги формата 210х297 мм (А4) в редакторе MS W*ord*. Шрифт – *Times New Roman*, размер – 14 pt, интервал одинарный, выравнивание «по ширине». Размеры полей: слева и справа – 2,5 см, сверху – 2 см, снизу – 3 см. Абзацный отступ – 1,25 см. Нумерация страниц – внизу, посередине листа. На титульном листе номер не ставится.

К отчету прикладывается дистрибутив разработанной системы.

Критерии оценки этапа для БРС:

|  |  |
| --- | --- |
| Содержание работы | Баллы |
| – отчет содержит все необходимые разделы | 6 |
| – нет ошибок в оформлении | 2 |
| – нет речевых и грамматических ошибок | 2 |
| *Итого,* максимум | 10 |

7 ЗАЩИТА КУРСОВОГО ПРОЕКТА

Защита курсового проекта проходит в открытой форме. На защите могут присутствовать студенты и преподаватели. Оценка по результатам защиты выставляется комиссией из преподавателей, назначаемых заведующим кафедрой.

Для защиты курсового проекта студентом должны быть представлены:

* файл-дистрибутив проекта системы;
* отчет в электронном и печатном виде;
* презентация доклада (должна быть загружена в электронный курс, если защита КП происходит online).

План презентации:

*1 Титульный лист*

*2 Постановка задачи* – 1 слайд

*3 Схема основного бизнес-процесса* – 1 слайд

*3 Особенности реализации (за что себя хочется похвалить)* – 1-3 слайда

*6 Пример использования: скриншоты интерфейса* – 1-3 слайда

*7 Заключение* – 1 слайд

Защита включает в себя доклад студента с демонстрацией презентации перед комиссией в течение 7 минут, и последующие ответы на вопросы (например, «почему было реализовано именно таким образом», «имело ли смысл предусмотреть в программе такие-то функции» и т.п.).

Критерии оценки этапа для БРС:

|  |  |
| --- | --- |
| Содержание работы | Баллы |
| В устном докладе студент показывает знания методологии проектирования ИС, свободно оперирует терминами применительно к рассматриваемой задаче | 6 |
| Подготовлена презентация | 2 |
| Студент может правильно ответить на вопросы членов комиссии | 2 |
| *Итого,* максимум | 10 |

# БИБЛИОГРАФИЧЕСКИЙ СПИСОК

* + - 1. Приказ №179 от 25.05.2019г. «Положение о балльно-рейтиноговой системе результатов учебной деятельности обучающегося» [Электронный ресурс]. – URL: https://nte.susu.ru/wordpress/wp-content/uploads/2019/06/ Polozhenie.pdf (дата обращения 02.11.2021)
      2. Новиков, Ф.А. Моделирование на UML. Теория, практика, видеокурс. / Ф.А. Новиков, Д.Ю. Иванов. – СПб.: Профессиональная литература, Наука и Техника, 2010. – 640 с. (+2DVD)
      3. Руководство программиста по программному обеспечению RadixWare/Краткое руководство по началу работы/Разработка приложения на платформе RadixWare [Электронный ресурс]. – URL: http://wiki.radixware.org/index.php?title=RadixWare\_Programmer\_Guide/Quick\_Start\_Guide/Developing\_an\_Application\_on\_RadixWare\_Platform#Starting\_RadixWare\_Designer (дата обращения 28.11.2021)
      4. Архипенков, С.В. Лекции по управлению программными проектами / С.В. Архипенков. – Москва, 2009. – 128 с.
      5. Создание проекта [Электронный ресурс]. – URL: https://ipc.susu.ru /p40521/ (дата обращения 28.11.2021)
      6. Oracle PL/SQL учебник [Электронный ресурс]. – URL: https://oracleplsql.ru (дата обращения 28.11.2021)
      7. Кузнецов, С. Д. Базы данных: учебник для студ. учреждений высшего проф. образования / С. Д. Кузнецов. – М.: Издательский центр «Академия», 2012. – 496 с.
      8. Создание слоя и определение структуры БД. [Электронный ресурс]. – URL: https://ipc.susu.ru/p40522/ (дата обращения 28.11.2021)
      9. Добавление модулей и Entity классов в ADS [Электронный ресурс]. – URL: https://ipc.susu.ru/p40523/ (дата обращения 28.11.2021)
      10. Создание экранных форм для выбора и редактирования [Электронный ресурс]. – URL: https://ipc.susu.ru/p40524/ (дата обращения 28.11.2021)
      11. Создание дерева навигации. Компиляция и запуск программы [Электронный ресурс]. – URL: https://ipc.susu.ru/p40525/ (дата обращения 28.11.2021)
      12. Создание и публикация отчетов [Электронный ресурс]. – URL: https://ipc.susu.ru/p40526/ (дата обращения 28.11.2021)

# ПРИЛОЖЕНИЯ

# ПРИЛОЖЕНИЕ А

# Пример задания на курсовой проект

Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования «Южно-Уральский государственный университет

(национальный исследовательский университет)»  
Институт естественных и точных наук

Кафедра прикладной математики и программирования

Направление подготовки Прикладная математика и информатика

УТВЕРЖДАЮ

Заведующий кафедрой прикладной математики и программирования

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_А.А.Замышляева

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2022 г.

ЗАДАНИЕ

на курсовой проект студентов

\_\_\_\_\_\_\_\_\_Иванова Д.С., Петрова А.К., Осиповой Е.В\_\_\_\_\_ Группа \_\_ЕТ-222\_\_

1. Дисциплина \_\_\_\_\_Моделирование и проектирование КИС\_

2. Тема работы \_\_\_\_Информационная система «Библиотека» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_ 2.1. Назначение системы:

* получение информации о наличии книг в библиотеке;
* заказ книг для получения в читальном зале или абонементе;
* выдача и возврат книг.

2.2. Пользователи системы:

* Администратор – настройка функций и прав пользователей системы;
* Библиотекарь – выдача и возврат книг;
* Студент – поиск и заказ книг

2.3. Предметная область:

Каждый экземпляр книги имеет уникальный штрих код, студент может получать книги в библиотеке по студенческому билету или зачетной книжке, в которых также есть уникальный штрих-код.

2.4. Исходные данные:

1. Web-интерфейс для студентов и библиотекарей для выполнения заказов

2. Desktop-интерфейс для администратора и библиотекарей

3. Сканер штрих-кодов подключен в режиме клавиатуры

4. SMS-информирование о готовности заказа   
5. Отправка e-mail о не сданных вовремя книгах

3. Срок сдачи студентом законченной работы *25 мая 2022 г.*

4. Перечень вопросов, подлежащих разработке:

* разработка технического задания на проектирование
* создание инфраструктуры проекта,

Окончание приложения А

* создание базы данных на сервере СУБД ORACLE, обеспечение непротиворечивости и разграничения прав пользователей
* разработка системы в среде RadixWare, отладка модулей, выпуск релиза
* оформление программной документации (руководство пользователя, листинг кода) и отчета по курсовой работе.

5. Календарный план

|  |  |  |
| --- | --- | --- |
| Наименование разделов  курсовой работы | Срок выполнения  разделов работы | Отметка руководителя  о выполнении |
| Разработка технического задания | 14.02.2022-20.02.2022 |  |
| Создание инфраструктуры и модулей проекта. Распределение задач между исполнителями | 22.02.2022-05.03.2022 |  |
| Разработка модели БД | 06.03.2022-19.03.2022 |  |
| Разработка модулей системы,  интерфейса пользователя, программной документации. | 21.03.2022-24.04.2022 |  |
| Тестирование и сборка модулей, выпуск релиза, оформление отчета | 25.04.2022-30.04.2022 |  |
| Защита курсового проекта | 02.05.2022-10.05.2022 |  |

Руководитель работы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Т.Ю.Оленчикова

(подпись)

Студенты \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Д.С.Иванов\_\_\_\_\_\_\_

(подпись)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /А.К.Петров\_\_\_\_\_\_\_

(подпись)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Е.В.Осипова\_\_\_\_\_\_\_

(подпись)

# ПРИЛОЖЕНИЕ Б

# ПРИМЕР ТитульнОГО ЛИСТА

Министерство образования и науки Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

«Южно-Уральский государственный университет

(национальный исследовательский университет)»

Институт естественных и точных наук

Кафедра прикладной математики и программирования

|  |  |
| --- | --- |
|  |  |

Информационная система «Библиотека»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К КУРСОВОМУ ПРОЕКТУ

по дисциплине Моделирование и проектирование корпоративных информационных систем

ЮУрГУ–01.04.02.2022.023 ПЗ КП

|  |  |
| --- | --- |
|  | Руководитель работы,  \_\_\_\_\_\_\_\_\_\_\_\_\_\_/ФИО  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_2022 г. |
|  | Авторы работы  Студенты группы ЕТ-222  \_\_\_\_\_\_\_\_\_\_\_\_\_/ ФИО  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_2022 г.  \_\_\_\_\_\_\_\_\_\_\_\_\_/ ФИО  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_2022 г.  \_\_\_\_\_\_\_\_\_\_\_\_\_/ ФИО  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_2022 г. |

Челябинск 2022

1. Приложение В

ПРИМЕР ОФОРМЛЕНИЯ АННОТАЦИИ

АННОТАЦИЯ

Иванов Д.С., Петров А.К., Осипова Е.В. Информационная система «Библиотека». – Челябинск: ЮУрГУ, ЕТ-222, 2022. – 19 с., 12 ил., библиографический список – 6 наим.

В пояснительной записке по курсовому проекту описывается разработка информационной системы «Библиотека» с помощью RadixWare. Работа содержит результаты этапов выполнения проекта. Приведено планирование проекта, распределение ролей и задач между участниками команды проекта, результаты работы над проектом в среде RadixWare.

Проект выполнен полностью в соответствии с техническим заданием в заданные сроки.

1. Приложение Г

ПРИМЕР ОФОРМЛЕНИЯ ОГЛАВЛЕНИЯ

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ 1

1 ТЕХНИЧЕСКОЕ ЗАДАНИЕ 2

2 ПЛАНИРОВАНИЕ ПРОЕКТА 3

3 Структура БАЗЫ ДАННЫХ 4

1. РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ 5
2. ПРОГРАММА И МЕТОДИКА ПРИЕМО-СДАТОЧНЫХ ИСПЫТАНИЙ 6

ЗАКЛЮЧЕНИЕ 7

БИБЛИОГРАФИЧЕСКИЙ СПИСОК 8

1. Приложение Д

ПРИМЕР ОФОРМЛЕНИЯ ВВЕДЕНИЯ

ВВЕДЕНИЕ

**Актуальность темы**. Сегодня корпоративные информационные системы (КИС) стали главной движущей силой научно-технической революции и развития современной мировой экономики. Правильно выбранная и внедренная КИС улучшает управляемость предприятия и повышает результативность его работы. Создание КИС для автоматизации разных видов деятельности компаний стало актуальным практически для всех. В связи с этим растет спрос на системы и технологии автоматизации проектирования КИС. Данный курсовой проект посвящен разработке КИС с применением современного CASE средства – среды проектирования RadixWare, разработанного компанией «Компас-Плюс».

**Цель работы** – разработать корпоративную информационную систему для задачи «тема курсового проекта», освоить этапы проектирования КИС с применением современных технологий.

**Задачи работы**:

– научиться проводить системный анализ объекта автоматизации и на его основе формулировать требования к разрабатываемой КИС;

– получить навыки командной разработки сложных систем;

– уметь проектировать базы данных (БД) в СУБД Oracle, знать основы администрирования БД, приемы обеспечения целостности и безопасности данных в многопользовательской среде.

– получить навыки проектирования распределенных приложений.

**Объект работы** – КИС «название проекта»

**Предмет работы** – технология проектирования КИС в среде RadixWare [3].

**Результаты работы** – полученные навыки эффективного проектирования сложных информационных систем можно в дальнейшем использовать в производственной деятельности или для создания научно-исследовательских систем сбора и обработки данных.

ПРИЛОЖЕНИЕ Е

ПРИМЕР ОФОРМЛЕНИЯ ТЗ

# 1 ТЕХНИЧЕСКОЕ ЗАДАНИЕ

# 1.1 Введение

Данный документ содержит описания технических решений, принятых в процессе проработки технического задания на проектирование автоматизированной системы «Библиотека».

Документ состоит из следующих разделов:

* состав автоматизированных рабочих мест системы – список автоматизированных рабочих мест системы, запланированный для реализации, с кратким описанием их функций
* структура базы данных системы – описание информационной модели системы
* организация поддержки основного технологических процесса в системе – заказ и выдача книг
* описание интерфейсы обмена данными с внешним оборудованием – сканером штрих-кодов.

# 1.2 Термины и определения

|  |  |
| --- | --- |
| **Термин** | **Определение** |
| АРМ | Автоматизированное рабочее место |

# 1.3 Описание технических решений

# **1.3.1 Состав автоматизированных рабочих мест системы**

В системе необходимо предусмотреть создание следующих автоматизированных рабочих мест.

Таблица 1 – Список автоматизированных рабочих мест системы

|  |  |  |
| --- | --- | --- |
| **Наименование АРМ** | **Пользователь** | **Состав функций АРМа** |
| АРМ библиотекаря | Библиотекарь | 1. Добавление и редактирование книг в БД  2. Подтверждение выдачи книг  3. Подтверждение возврата книг  4. Статистическая информация о количестве выданных книг и просроченных книгах |
| Продолжение приложения Е | | |
| АРМ исполнителя заказа | Библиотекарь | 1. Просмотр списка заказов и выбор на выполнение  2. Подтверждение формирования заказа |
| АРМ читателя | Студент | 1. Формирования заявки на получение книг  2. Поиск книг в БД и определение их наличия |
| АРМ администратора системы | Системный администратор | 1. Настройки и конфигурации системы |

Для поддержки разграничения доступа к АРМам в системе должны быть предусмотрены роли:

* «Библиотекари»;
* «Студенты»;
* «Администратор»;

Каждая роль (или группа) должна быть назначена на функции АРМов в соответствии с таблицей 1.

**1.3.2 Информационная модель системы**

Основные классы объектов предметной области приводятся в таблице 2, связи между ними показаны на рисунке 1.

Таблица 2 – Информационная модель

|  |  |
| --- | --- |
| Наименование | Назначение и свойства |
| Книга | Информация о книге: авторы. название, год издания, шифр места хранения, краткое описание, количество экземпляров и экземпляров в наличии |
| Экземпляр книги | Уникальный номер (штрих-код), текущее состояние (в наличии или кому-то выдана, когда должна быть возвращена) |
| Заказ | Кто сделал заказ, список книг, состояние обработки заказа |
| Пользователь | Штрих-коды документов, e-mail, номер телефона для SMS-информирования, роль |

Продолжение приложения Е

## Рисунок 1 – ER-диаграмма предметной области

Пользователь

Заказ

Книга

Экземпляр книги

1

\*

0..\*

1..\*

1

\*

\*

\*

содержит

читает

## 1.3.3 Организация поддержки основного технологического процесса в системе

Основной технологический процесс в библиотеке – это формирование заказа на книги и их выдача.

1. Студент ищет книги, имеющиеся в наличии на данный момент в библиотеке, и добавляет их в заказ.
2. Когда заказ будет сформирован, студент отправляет его на выполнение. Перед выполнением проверяется наличие книг с учетом уже выполняемых заказов.
3. Библиотекарь выбирает один из поступивших заказов и собирает книги по нему.
4. Когда книги в заказе будут собраны и доставлены на стойку регистрации, происходит SMS-информирование студента.
5. При выдаче заказа производится сканирование штрих-кода идентифицирующего документа студента и штрих-кодов книг.

**1.3.4 Организация обмена данными со сканером штрих-кодов**

Сканер штрих-кодов подключен как клавиатура, перед считанным номером посылается код клавиши F7, после кода — код клавиши Enter.

При выдаче книги библиотекарь сканирует штрих-код студенческого билета или зачетки, затем штрих-коды книг. При возврате – штрих-код библиотеки, затем штрих-коды возвращенных книг.

**1.3.5 Требования к реализации АРМов системы**

Доступ к экранным формам соответствующих АРМов должен быть поддержан только в контексте соответствующего процесса. Для АРМов администратора системы необходимо обеспечить также прямой доступ к данных для любой формы.

Продолжение приложения Е

**1.4. Порядок контроля и приемки системы**

**1.4.1 Виды и объем испытаний системы**

Система подвергается испытаниям следующих видов:

1. Предварительные испытания.

2. Опытная эксплуатация.

3. Приемочные испытания.

Состав, объем и методы предварительных испытаний системы определяются документом «Программа и методика испытаний», разрабатываемым на стадии «Рабочая документация».

Состав, объем и методы опытной эксплуатации системы определяются документом «Программа опытной эксплуатации», разрабатываемым на стадии «Ввод в действие».

Состав, объем и методы приемочных испытаний системы определяются документом «Программа и методика приемо-сдаточных испытаний», разрабатываемым на стадии «Ввод в действие» с учетом результатов проведения предварительных испытаний и опытной эксплуатации.

**1.4.2 Требования к приемке работ по стадиям**

Требования к приемке работ по стадиям приведены в таблице.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Стадия испытаний** | **Участники испытаний** | **Место и срок проведения** | **Порядок согласования документации** | **Статус приемочной комиссии** |
| Предварительные испытания | Организации Заказчика и Разработчика | На территории Заказчика, с dd.mm.yyyy по dd.mm.yyyy | Проведение предварительных испытаний.  Фиксирование выявленных неполадок в Протоколе испытаний.  Устранение выявленных неполадок  Проверка устранения выявленных неполадок.  Принятие решения о возможности передачи АИС в опытную эксплуатацию.  Составление и подписание Акта приёмки АИС в опытную эксплуатацию. | Экспертная группа |
| Продолжение приложения Е | | | | |
| Опытная эксплуатация | Организации Заказчика и Разработчика | На территории Заказчика, с dd.mm.yyyy по dd.mm.yyyy | Проведение опытной эксплуатации.  Фиксирование выявленных неполадок в Протоколе испытаний.  Устранение выявленных неполадок.  Проверка устранения выявленных неполадок.  Принятие решения о готовности АИС к приемочным испытаниям.  Составление и подписание Акта о завершении опытной эксплуатации АИС. | Группа тестирования |
| Приемочные испытания | Организации Заказчика и Разработчика | На территории Заказчика, с dd.mm.yyyy по dd.mm.yyyy | Проведение приемочных испытаний.  Фиксирование выявленных неполадок в Протоколе испытаний.  Устранение выявленных неполадок.  Проверка устранения выявленных неполадок.  Принятие решения о возможности передачи АИС в промышленную эксплуатацию.  Составление и подписание Акта о завершении приемочных испытаний и передаче АИС в промышленную эксплуатацию.  Оформление Акта завершения работ. |  |

Продолжение приложения Е

# 1.5. Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие

Для создания условий функционирования АИС, при которых гарантируется соответствие создаваемой системы требованиям, содержащимся в настоящем техническом задании, и возможность эффективного её использования, в организации Заказчика должен быть проведен комплекс мероприятий.

**1.5.1. Технические мероприятия**

Силами Заказчика в срок до начала этапа «Разработка рабочей документации. Адаптация программ» должны быть выполнены следующие работы:

- осуществлена подготовка помещения для размещения АТК системы в соответствии с требованиями, приведенными в настоящем техническом задании;

- осуществлена закупка и установка необходимого АТК;

- организовано необходимое сетевое взаимодействие.

**1.5.2. Организационные мероприятия**  
Силами Заказчика в срок до начала этапа работ «Разработка рабочей документации. Адаптация программ» должны быть решены организационные вопросы по взаимодействию с системами-источниками данных. К данным организационным вопросам относятся:

- организация доступа к базам данных источников;

- определение регламента информирования об изменениях структур систем-источников;

- выделение ответственных специалистов со стороны Заказчика для взаимодействия с проектной командой по вопросам взаимодействия с системами-источниками данных.

**1.5.3. Изменения в информационном обеспечении**  
Для организации информационного обеспечения системы должен быть разработан и утвержден регламент подготовки и публикации данных из систем-источников.

Перечень регламентов может быть изменен на стадии «Разработка рабочей документации. Адаптация программ».

Окончание приложения Е

# 1.6. Требования к документированию

|  |  |
| --- | --- |
| **Этап** | **Документ** |
| Разработка рабочей документации. Адаптация программ | [Руководство пользователя](https://www.prj-exp.ru/patterns/pattern_user_guide.php) |
| [Программа](https://www.prj-exp.ru/patterns/pattern_program_of_test.php) и [методика испытаний](https://www.prj-exp.ru/patterns/pattern_methods_of_test.php) |
| Спецификация |
| Текст программ на машинном носителе |
| [Акт приёмки в опытную эксплуатацию](https://www.prj-exp.ru/patterns/pattern_act_of_trial_operation.php) |
| [Протокол испытаний](https://www.prj-exp.ru/patterns/pattern_report_of_test.php) |
| Акт приемки системы в промышленную эксплуатацию |

ПРИЛОЖЕНИЕ Ж

ПРИМЕР ПЛАНИРОВАНИЯ ПРОЕКТА

2 ПЛАНИРОВАНИЕ ПРОЕКТА

2.1 Распределение ролей

Таблица 1 – Распределение ролей участников проекта

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ФИО | Руководитель  проекта | Системный архитектор | Управление изменениями | Тестировщик | Проектировщик | Проектировщик базы данных | Проектировщик интерфейса пользователя | Разработчик | Технический писатель |
| Иванов Д.С. | + |  | + | + |  |  |  | + |  |
| Петров А.К. |  | + |  |  | + | + |  | + |  |
| Осипова Е.В. |  |  |  |  |  |  | + | + | + |

2.2 Состав работ

Таблица 2 – Содержание и сроки работ над проектом

|  |  |  |  |
| --- | --- | --- | --- |
| № | Этапы работ | Недели | Исполнители |
| 1 | Анализ требований заказчика, составление ТЗ | 1 | Иванов А.К.  Петров А.К.  Осипова Е.В. |
| 2 | Уточнение содержания и состава работ | 1 | Петров А.К. |
| 3 | Составление плана выполнения проекта | 1 | Иванов Д.С. |
| 4 | Создание проекта в SVN и загрузка слоев в RadixWareDesigner | 1 | Петров А.К. |
| 5 | Анализ предметной области, разработка базы данных | 2 | Петров А.К. |
| 6 | Отслеживание плана выполнения работ, внесение изменений в план или ТЗ в случае угрозы срыва работ | 10  \*0,2 | Иванов Д.С. |
| 7 | Внесение изменений БД при необходимости | 8  \*0,1 | Петров А.К. |
| 8 | Разработка модуля считывания информации со штрих-кода | 2 | Петров А.К. |
| 9 | Создание эскизов экранных форм и отчетов для всех АРМов | 3 | Осипова Е.В. |
| А. | Разработка АРМ библиотекаря |  |  |
|  | А1.Создание Entity классов и интерфейса  пользователя | 1 | Осипова Е.В. |
|  | А2. Создание и отладка экранных форм для выбора и редактирования | 1 | Осипова Е.В. |
|  | А3. Создание дерева навигации | 1 | Осипова Е.В. |
|  | А4. Создание отчетов | 1 | Осипова Е.В. |
|  | Продолжение приложения Ж | | |
| Б. | Разработка АРМ исполнителя заказа | | |
|  | Б1.Создание Entity классов и интерфейса  пользователя | 1 | Петров А.К. |
|  | Б2. Создание и отладка экранных форм для выбора и редактирования | 1 | Петров А.К. |
|  | Б3. Создание дерева навигации | 1 | Петров А.К. |
| В. | Разработка АРМ читателя | | |
|  | В1.Создание Entity классов и интерфейса  пользователя | 1 | Осипова Е.В. |
|  | В2. Создание и отладка экранных форм для выбора и редактирования | 1 | Осипова Е.В. |
|  | В3. Создание дерева навигации | 1 | Осипова Е.В. |
| Г. | Разработка АРМ администратора системы | | |
|  | Г1.Создание Entity классов и интерфейса  пользователя | 1 | Иванов Д.С. |
|  | Г2. Создание и отладка экранных форм для выбора и редактирования | 1 | Иванов Д.С. |
|  | Г3. Создание дерева навигации | 1 | Иванов Д.С. |
| 10 | Оформление руководства пользователя | 2 | Осипова Е.В. |
| 11 | Сборка и функциональное тестирование системы. Выпуск дистрибутива | 2 | Иванов А.К. |
| 12 | Сборка отчета по КП, создание презентации, защита КП | 1 | Иванов А.К.  Петров А.К.  Осипова Е.В. |

2.3 Граф зависимостей проекта

Рисунок 2 – Граф зависимостей этапов проекта

1

начало

2

4

5

6

7

8

9

11

3

10

12

окончание

А3

А1

А2

А4

Б1

Б2

Б3

В1

В2

В3

Г1

Г2

Г3

Окончание приложения Ж

2.4 Диаграмма Ганта

Оптимизированный по методу критического пути план выполнения работ приведен на рисунке 3. Красным цветом на плане показаны работы критического пути. Серым цветом – запас по времени, не нарушающий общих сроков выполнения проекта.

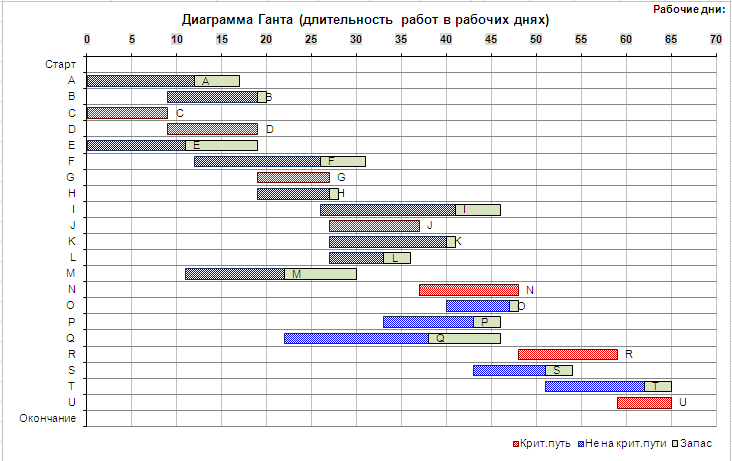


Рисунок 3 – Базовое расписание проекта

ПРИЛОЖЕНИЕ И

ПРИМЕР ОФОРМЛЕНИЯ СТРУКТУРЫ БАЗЫ ДАННЫХ

1. СТРУКТУРА БАЗЫ ДАННЫХ

**3.1 Модуль LibraryData**

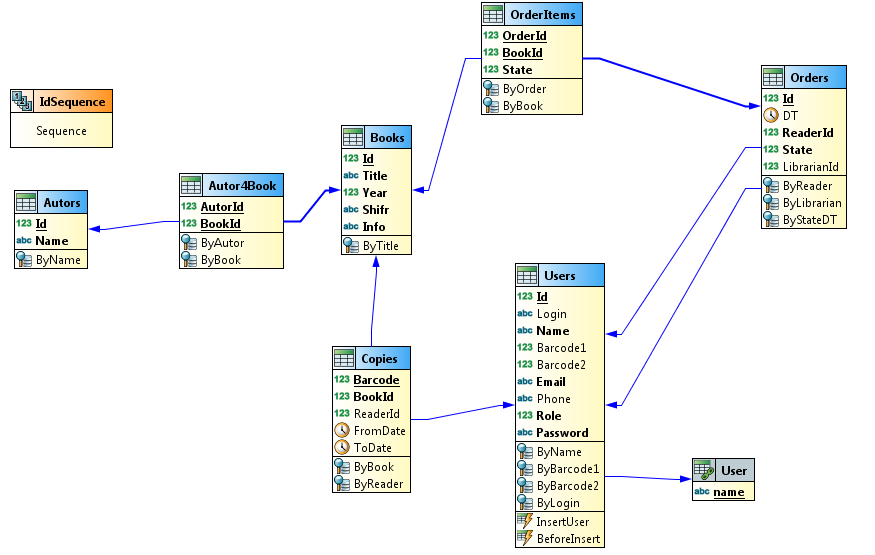
****

Рисунок 5 – Схема модели базы данных

3.1.1 **Таблица Books**

**Описание**

Информация о книгах

**Поля**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Имя | Имя в БД | Тип | Тип в БД | PK | Not Null | FK | Генератор | Описание |
| Id | ID | Int | NUMBER(9,0) | ✔ | ✔ |  | [IdSequence](file:///C:\Work\ru\my.library\dds\LibraryData\sqnLR3J4NXJQZHGHG5B4DTBRHZY4I.html) | Номер книги |
| Title | TITLE | Str | VARCHAR2(250 char) |  | ✔ |  | - | Название |
| Year | YEAR | Int | NUMBER(4,0) |  | ✔ |  | - | Год издания |
| Shifr | SHIFR | Str | VARCHAR2(20 char) |  | ✔ |  | - | Шифр книги |
| Info | INFO | Str | VARCHAR2(1000 char) |  | ✔ |  | - | Краткая информация |

**Индексы**

|  |  |  |  |
| --- | --- | --- | --- |
| Имя | Имя в БД | Поля | Опции |
| PrimaryKey | PK\_BOOKS | [Id](file:///C:\Work\ru\my.library\dds\LibraryData\tblH5Z7JAUNW5HZ5ASGJEWDJBI6OQ.html#tblH5Z7JAUNW5HZ5ASGJEWDJBI6OQ-col6WD6MADFZJGXVLOS3GBBIMOFZI) | Rely |
| ByTitle | IDX\_BOOKS\_BYTITLE | [Title](file:///C:\Work\ru\my.library\dds\LibraryData\tblH5Z7JAUNW5HZ5ASGJEWDJBI6OQ.html#tblH5Z7JAUNW5HZ5ASGJEWDJBI6OQ-colOZLPW2YSI5ASJEDFP3PERR7MHY) |  |

**Триггеры**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Имя | Имя в БД | Время обработки | Тип | Событие | Описание |
| InsertBook | TAIR\_BOOKS | After | None | Insert | - |
| BeforeInsert | TBIR\_BOOKS | Before | None | Insert | - |

ПРИЛОЖЕНИЕ К

ПРИМЕР ОФОРМЛЕНИЯ РУКОВОДСТВА ПОЛЬЗОВАТЕЛЯ

1. РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

**4.1 АРМ администратора**

Дерево навигации АРМ администратора содержит следующие ветки:

* “Вся система”, которая представляет собой дерево для управления платформой RadixWare;
* “Пользователи”, которая позволяет создавать пользователей системы и указывать их права.
* “Книги”, которая позволяет просматривать текущее состояние списка книг и экземпляров
* “Заказы”, которая позволяет просматривать список выполняющихся и выполненных заказов

Работа с пользователями показана на рисунке 6. Эта ветка частично дублирует функциональность ветки “Пользователи” в поддереве “Вся система”, но здесь нужно указывать только проблемно-ориентированные поля, остальные значения будут установлены по умолчанию. В качестве идентификатора пользователя используется штрих-код зачетки или читательского билета. Также администратор задает начальный пароль для пользователя.

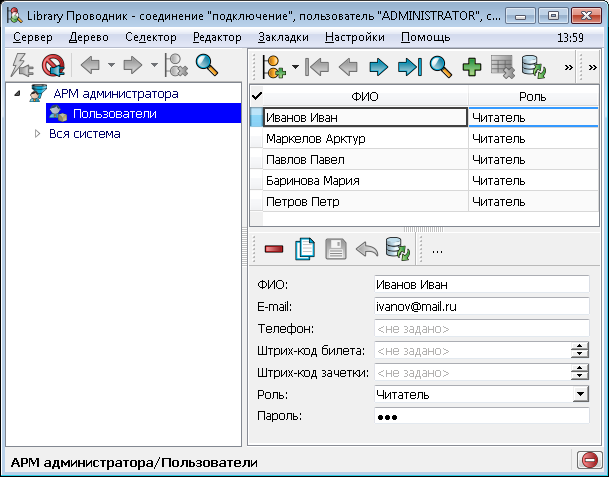


Рисунок 6 – Ветка Пользователи АРМ администратора

Работа с ветками “Книги” и “Заказы” совпадает с АРМ библиотекаря и будет рассмотрена в соответствующем разделе руководства.

1. ПРИЛОЖЕНИЕ Л

ПРИМЕР ОФОРМЛЕНИЯ ПРОГРАММЫ И МЕТОДИКИ  
ПРИЕМО-СДАТОЧНЫХ ИСПЫТАНИЙ

1. ПРОГРАММА И МЕТОДИКА ПРИЕМО-СДАТОЧНЫХ   
   ИСПЫТАНИЙ

**5.1 Объект испытаний**

Объектом приемо-сдаточных испытания (ПСИ) является корпоративно- информационная систему «Библиотека», предназначенный для автоматизации процесса формирования заказа и выдачи книг читателям.

**5.2 Цель испытаний**

Целью проведения ПСИ является:

* проверка соответствия документации КИС «Библиотека» согласованной в ТЗ рабочей документации;
* проверка корректности функционирования комплекса со сканером штрих-кодов и SMS-сервисом;
* проверка соответствия функционирования программного КИС «Библиотека» заявленным в ТЗ требованиям.

Этот документ определяет цели, масштабы, процедуры, подробное содержание и график приемо-сдаточных испытаний

**5.3 Требования к программе**

КИС должна обеспечивать функционирование следующих рабочих мест:

* АРМ читателя – Desktop-приложение с функциями поиска книг в БД, просмотра их наличия в библиотеке, формирования заказа
* АРМ библиотекаря – Desktop-приложение с возможностью добавления и редактирования книг в БД, фиксация выдачи и возврата книг, получения отчетов о количестве выданных и просроченных книг;
* АРМ исполнителя заказа – Android-приложение c возможностью просмотра не выполненных заказов, информирования читателя об исполнении заказа;
* АРМ администратора системы – Desktop-приложение для настройки и конфигурирования системы

**5.4. Требования к программной документации**

Следующие документы являются основой ведения ПСИ

1. Руководство пользователя по работе с КИС «Библиотека»

Документация, которая входит в комплект поставки, должна соответствовать спецификации КИС «Библиотека»

Окончание приложения Л

**5.5 Средства и порядок испытаний**

В состав технических средств входит

* сервер CPU 2,5+ГГц, ОЗУ 32+ГБ, ЖД 500ГБ
* компьютер CPU 2,5+ГГц, ОЗУ 16+ГБ, ЖД 500ГБ
* планшет ОЗУ 4+ГБ, встроенная память 16ГБ

Программное обеспечение для сервера: ОС Windows 10, Subversion 1.10, Java SE Development Kit 8, Oracle Database Express Edition 18c.

Программное обеспечение для клиентского компьютера: ОС Windows 10, TortoiseSVN 1.10, Java SE Development Kit 8, драйвер OJDBC для Oracle, RadixWare Manager, RadixWare Platform.

Программное обеспечение для планшета: ОС Android 10.

В БД загружена тестовые данные.

Запуск соответствующего АРМ для испытаний производится в соответствии с руководством пользователя.

**5.6 Методы испытаний**

**5.6.1 Проверка АРМ читателя**

Данная процедура проводится с целью подтверждения корректности функционирования АРМ читателя.

Проверка АРМ читателя проводится на тестовых данных, хранящихся в базе данных.

Выбираем ветку Книги, в появившемся окне задаем поиск по автору “Кнут”. Добавляем Том 1 в заказы. Выполняем поиск по части названия “Искусство” и добавляем Том 2 в заказы. Просматриваем список заказанных книг и формируем заказ. Заказ должен поменять статус на “В обработке”

Тест пройден успешно, если удалось сформировать заказ из двух книг.

**5.6.2 Проверка АРМ библиотекаря**

…аналогично…

1. Приложение М

ПРИМЕР ОФОРМЛЕНИЯ ЗАКЛЮЧЕНИЯ

ЗАКЛЮЧЕНИЕ

В ходе выполнения курсового проекта был проведен системный анализ требований к проекту и составлено техническое задание, спроектирована база данных Oracle, спроектировано распределенное приложение по архитектуре «клиент – сервер приложений – сервер базы данных», пользовательские формы и отчеты. Все компоненты системы протестированы и отлажены. Таким образом, цель работы была достигнута, задачи – решены. Создано работоспособное приложение, предназначенное для учета поступления и выдачи литературы в библиотеке.

В результате работы получены навыки совместной разработки КИС в среде RadixWare и оформления документации по проекту.

1. Приложение Н

ПРИМЕР ОФОРМЛЕНИЯ БИБЛИОГРАФИЧЕСКОГО СПИСКА

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

* + - 1. Архипенков, С.В. Лекции по управлению программными проектами / С.В. Архипенков. – Москва, 2009. – 128 с.
      2. Создание проекта [Электронный ресурс]. – URL: https://ipc.susu.ru /p40521/ (дата обращения 28.01.2022)
      3. Oracle PL/SQL учебник [Электронный ресурс]. – URL: https://oracleplsql.ru (дата обращения 28.01.2022)
      4. Создание слоя и определение структуры БД. [Электронный ресурс]. – URL: <https://ipc.susu.ru/p40522/> (дата обращения 28.01.2022)
      5. Добавление модулей и Entity классов в ADS [Электронный ресурс]. – URL: <https://ipc.susu.ru/p40523/> (дата обращения 28.01.2022)
      6. Создание экранных форм для выбора и редактирования [Электронный ресурс]. – URL: <https://ipc.susu.ru/p40524/> (дата обращения 28.11.2021)
      7. Создание дерева навигации. Компиляция и запуск программы [Электронный ресурс]. – URL: <https://ipc.susu.ru/p40525/> (дата обращения 28.01.2022)
      8. Создание и публикация отчетов [Электронный ресурс]. – URL: <https://ipc.susu.ru/p40526/> (дата обращения 28.11.2021)

ОГЛАВЛЕНИЕ

[1 Этапы выполнения курсового проекта 3](#_Toc90395278)

[2 Темы курсовых проектов 4](#_Toc90395279)

[3 Технология проектирования КИС в RadixWare 9](#_Toc90395280)

[4 Техническое задание на разработку КИС 16](#_Toc90395281)

[5 Проектирование и реализация системы 17](#_Toc90395282)

[6 Структура и содержание пояснительной записки 25](#_Toc90395283)

[Библиографический список 27](#_Toc90395284)

[Приложения](#_Toc90395285)

[Приложение А Пример задания на курсовой проект 29](#_Toc90395286)

[Приложение Б Пример титульного листа 31](#_Toc90395288)

[Приложение В Пример оформления аннотации 32](#_Toc90395290)

[Приложение Г Пример оформления оглавления 33](#_Toc90395291)

[Приложение Д Пример оформления введения 34](#_Toc90395292)

[Приложение Е Пример оформления ТЗ 35](#_Toc90395292)

[Приложение Ж Пример планирования проекта 42](#_Toc90395292)

[Приложение И Пример оформления структуры базы данных 45](#_Toc90395292)

[Приложение К Пример оформления руководства пользователя](#_Toc90395292) 46

[Приложение Л Пример оформления программы и методики приемо-сдаточных испытаний 47](#_Toc90395300)

[Приложение М Пример оформления заключения 49](#_Toc90395301)

[Приложение Н Пример оформления библиографического списка 50](#_Toc90395302)

1. Форматы файлов системы Город: <https://help.gkh365.ru/formula/instructions/447/> [↑](#footnote-ref-2)