МЕЖГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛОРУССКО-РОССИЙСКИЙ УНИВЕРСИТЕТ»

Кафедра «Автоматизированные системы управления»

ИНТЕГРИРОВАННЫЕ ИНФОРМАЦИОННЫЕ СИСТЕМЫ ПРЕДПРИЯТИЙ

Методические рекомендации к лабораторным работам для студентов направлений подготовки 09.03.01 «Информатика и вычислительная техника» и 09.03.04 «Программная инженерия» дневной формы обучения

Могилев 2023

УДК 658.012.011.56
ББК 65.050.2
И73

Рекомендовано к изданию
учебно-методическим отделом
Белорусско-Российского университета

Одобрено кафедрой «Автоматизированные системы управления» «17» октября 2023 г., протокол № 3

Составитель ст. преподаватель И. В. Акиншева

Рецензент канд. техн. наук, доц. В. В. Кутузов

Методические рекомендации содержат требования к выполнению лабораторных работ по дисциплине «Интегрированные информационные системы предприятий».

Учебное издание

ИНТЕГРИРОВАННЫЕ ИНФОРМАЦИОННЫЕ СИСТЕМЫ ПРЕДПРИЯТИЙ

Ответственный за выпуск	А. И. Якимов
Корректор	И. В. Голубцова
Компьютерная верстка	Е. В. Ковалевская

Подписано в печать . Формат 60×84/16. Бумага офсетная. Гарнитура Таймс.
Печать трафаретная. Усл. печ. л. . Уч.-изд. л. . Тираж 21 экз. Заказ №

Издатель и полиграфическое исполнение:
Межгосударственное образовательное учреждение высшего образования
«Белорусско-Российский университет».
Свидетельство о государственной регистрации издателя,
изготовителя, распространителя печатных изданий
№ 1/156 от 07.03.2019.
Пр-т Мира, 43, 212022, г. Могилев.

Содержание

Введение…….. 5
1 Лабораторная работа № 1. Построение структурной схемы предприятия ……..…….. 6
2 Лабораторная работа № 2. Построение функциональной модели бизнес-процессов предприятия …… 6
3 Лабораторная работа № 3. Язык SQL. Реинжиниринг бизнес-процессов предприятия при внедрении интегрированной информационной системы …… 8
4 Лабораторная работа № 4. Проектирование хранилища данных интегрированной информационной системы предприятия……... 9
5 Лабораторная работа № 5. Реализация хранилища данных интегрированной информационной системы ……..….... 10
6 Лабораторная работа № 6. Реализация рабочих мест ввода данных в интегрированную информационную систему предприятия ……... 11
7 Лабораторная работа № 7. Реализация бизнес-логики процессов интегрированной информационной системы предприятия …….. 12
8 Лабораторная работа № 8. Реализация взаимодействия бизнес-логики процессов интегрированной информационной системы с хранилищем данных …… 16
9 Лабораторная работа № 9. Проектирование пользовательского интерфейса интегрированной информационной системы предприятия ….. 17
10 Лабораторная работа № 10. Реализация взаимодействия пользовательского интерфейса с бизнес-логикой процессов интегрированной информационной системы предприятия …….. 18
11 Лабораторная работа № 11. Оптимизация пользовательского интерфейса интегрированной информационной системы предприятия ….. 19
12 Лабораторная работа № 12. Тестирование бизнес-логики разработанной интегрированной информационной системы предприятия ….. 20
13 Лабораторная работа № 13. Тестирование пользовательского интерфейса разработанной интегрированной информационной системы предприятия ……... 22
14 Лабораторная работа № 14. Построение модели бизнес-процесса планирования в разработанном модуле интегрированной информационной системы предприятия …….. 22
15 Лабораторная работа № 15. Получение исходных данных разработанной модели из хранилища данных интегрированной информационной системы предприятия …… 23
16 Лабораторная работа № 16. Анализ результатов моделирования

бизнес-процесса планирования в разработанном модуле интегрированной информационной системы предприятия ……. 24

17 Лабораторная работа № 17. Рациональный выбор параметров бизнес-процесса планирования в разработанном модуле интегрированной информационной системы предприятия ……. 27

18 Лабораторная работа № 18. Построение модели взаимодействия компонентов разработанного модуля интегрированной информационной системы предприятия ……..…….. 28

19 Лабораторная работа № 19. Построение модели размещения компонентов разработанного модуля интегрированной информационной системы предприятия …… 32

20 Лабораторная работа № 20. Разработка технической документации на модуль интегрированной информационной системы предприятия …….. 36

Список литературы……..…... 37

Приложение А. Варианты заданий для выполнения лабораторных работ……..…….. 38

Введение

Целью курса является обучение студентов основам создания автоматизированных систем управления предприятием.

Методические рекомендации предназначены для изучения современных технологий моделирования и реинжиниринга бизнес-процессов предприятий при внедрении интегрированных информационных систем, проектирования хранилища данных интегрированных информационных систем на базе современных СУБД, создания модулей интегрированных информационных систем, подключения разработанных модулей к существующим информационным системам предприятий.

В методических рекомендациях содержится информация к двадцати лабораторным работам.

При оформлении отчетов по лабораторным работам следует придерживаться следующих правил:

– отчёт может быть рукописным или отпечатанным на принтере;

– текст отчета оформляется в соответствии с ГОСТ 2.105–95;

– отчет содержит следующие разделы: титульный лист, цель и задачи работы, выполненные задания, выводы;

– отчет сдается на проверку скрепленным.

1 Лабораторная работа № 1. Построение структурной схемы предприятия

Порядок выполнения работы

1 Изучить виды организационных структур.

2 Провести анализ различных типов организационных структур предприятий.

3 Получить навыки построения организационных структур в различных офисных приложениях MS Office (инструментарий SmartArt), в MS Visio и системах моделирования CA ERwin Process Modeler, ARIS, Business Studio.

4 Провести сравнительный анализ процессов построения организационных структур в разных нотациях.

5 Построить структурную схему предприятия в соответствии с вариантом задания к лабораторной работе. Варианты заданий приведены в приложении А.

6 Оформить работу.

7 Осуществить защиту работы.

Контрольные вопросы

1 Что такое организационная структура предприятия? Для чего необходимо разрабатывать организационную структуру?

2 Какие существуют типы организационных структур? Перечислите их преимущества и недостатки.

3 Проведите сравнительный анализ линейно-функциональной структуры управления и проектной. В каких видах бизнеса используются эти структуры?

4 Проведите сравнительный анализ матричной и процессной структур управления. В каких видах бизнеса используются эти структуры?

5 Проведите сравнительный анализ процессов построения организационных структур в различных программных системах.

6 Каковы особенности построения организационных структур в системах моделирования CA ERwin Process Modeler, ARIS и Business Studio?

2 Лабораторная работа № 2. Построение функциональной модели бизнес-процессов предприятия

Порядок выполнения работы

1 Построить функциональную модель бизнес-процессов предприятия.

2 Оформить работу.

3 Осуществить защиту работы.

Ход выполнения работы

1 Запустите **BPwin** (кнопка Пуск /BPwin 

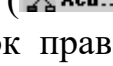
).

2 Если появляется диалог **ModelMart Connection Manager**, нажмите на кнопку **Cancel.**

3 Щелкните по кнопке  . Появляется диалоговое окно**I would like to**. Внесите в текстовое поле **Name** имя модели «Деятельность компании» и выберите **Type – Business Process (IDEF0)**. Нажмите кнопку **ОК**.

4 Откроется диалоговое окно **Properties for New Models** (Свойства новой модели). Введите в текстовое поле **Author** (Автор) имя автора модели, в текстовое поле **Author initials** – его инициалы. Нажмите последовательно кнопки **Apply** и **ОК**.

5 Автоматически создается незаполненная контекстная диаграмма.

6 Обратите внимание на кнопку  на панели инструментов. Эта кнопка включает и выключает инструмент просмотра и навигации **Model Explorer** (Браузер модели). **Model Explorer** имеет три вкладки – **Activities** (

), **Diagrams** (

) и **Objects** (

). Во вкладке **Activities** щелчок правой кнопкой по объекту в браузере модели позволяет выбрать опции редактирования его свойств. Откройте окно свойств модели и ознакомьтесь с ними.

7 Если непонятно, как выполнить то или иное действие, можно вызвать контекстную помощь (клавиша **F1)** или воспользоваться меню **Help**.

8 Перейдите в меню **Model/Model Properties**. Во вкладке **General** диалогового окна **Model Properties** в текстовое поле **Model name** следует внести имя модели «Деятельность компании», в текстовое поле **Project** – имя проекта «Модель деятельности компании» и, наконец, в текстовое поле **Time Frame** (Временной охват) – **AS-IS** (Как есть).

9 Во вкладке **Purpose** диалогового окна **Model Properties** в текстовое поле **Purpose** (цель) внесите данные о цели разработки модели «Моделировать текущие (AS-IS) бизнес-процессы компании», в текстовое поле **Viewpoint** (точка зрения) – «Директор».

10 Во вкладке **Definition** диалогового окна **Model Properties** в текстовое поле **Definition** (Определение) внесите «Это учебная модель, описывающая деятельность компании» и в текстовое поле **Scope** (охват) – «Общее управление бизнесом компании: исследование рынка, закупка компонентов, сборка, тестирование и продажа продуктов».

11 Перейдите на контекстную диаграмму и правой кнопкой мыши щелкните по прямоугольнику, представляющему в нотации **IDEF0** условное графическое обозначение работы. В контекстном меню выберите опцию **Name**. Во вкладке **Name** внесите имя «Деятельность компании».

12 Во вкладке **Definition** диалогового окна **Activity Properties** в текстовое поле **Definition** (Определение) внесите «Текущие бизнес-процессы компании». Текстовое поле **Note** (Примечания) оставьте незаполненным.

13 Создайте **ICOM**-стрелки на контекстной диаграмме (таблица 1). Для этого выберите инструмент Arrow на панели инструментов, щелкните мышью на

границе источника стрелки, отпустите мышь, щелкните мышью на подсвеченной границе приемника стрелки. Свойства стрелок (Arrow Name, Arrow Definition) задайте в окне **Arrow Properties,** которое вызывается в контекстном меню.

Таблица 1 – Стрелки контекстной диаграммы

Название стрелки (Arrow Name)	Определение стрелки (Arrow Definition)	Тип стрелки (Arrow Type)
Звонки клиентов	Запросы информации, заказы, техподдержка и т. д.	Input
Правила и процедуры	Правила продаж, инструкции по сборке, процедуры тестирования, критерии производительности и т. д.	Control
Проданные продукты	Настольные и портативные компьютеры	Output
Бухгалтерская система	Оформление счетов, оплата счетов, работа с заказами	Mechanism

14 С помощью кнопки **T** внесите текст в поле диаграммы – точку зрения и цель.

15 Создайте отчет по модели. В меню **Tools/Reports/Model Report** задайте опции генерирования отчета (установите все галочки) и нажмите кнопку **Preview** (Предварительный просмотр).

Контрольные вопросы

1 Какие свойства имеет модель?

2 Какие вкладки имеет инструмент Model Explorer (Браузер модели) и что каждая из них отображает?

3 Назовите типы стрелок в диаграмме IDEF0.

4 Какие виды текста можно внести в поле диаграммы?

5 Назовите опции генерирования, которые включает в себя отчет Model Report.

3 Лабораторная работа № 3. Язык SQL. Реинжиниринг бизнес-процессов предприятия при внедрении интегрированной информационной системы

Порядок выполнения работы

1 По указанию преподавателя из предложенного перечня процессов предприятия выбрать один.

2 Разбить процесс на основные операции, описать их, установить взаимосвязи, исполнителей.

3 Выделить основные элементы процесса (входы, выходы, ресурсы и т. д.) и описать их.

4 Выделить основных участников процесса и описать их полномочия по участию в процессе.

5 Провести идентификацию процесса в различных видах классификаций и обосновать свои выводы.

6 Построить модель SADT выбранного процесса и приложить к рисунку поясняющую спецификацию в виде таблицы.

7 Оформить работу.

8 Осуществить защиту работы.

Контрольные вопросы

1 Дайте определение процесса и назовите основные группы процессов предприятия.

2 Назовите основные элементы процесса и дайте их определения.

3 Назовите классификационные типы процессов предприятия и определите их основные отличительные признаки.

4 Назовите основные принципы выделения процессов на предприятии.

5 Назовите основные правила описания процесса.

4 Лабораторная работа № 4. Проектирование хранилища данных интегрированной информационной системы предприятия

Порядок выполнения работы

1 Изучить методику моделирования при помощи ERwin.

2 Провести анализ данных для предметной области из лабораторной работы № 1:

- определить цели моделирования;
- определить сущности;
- определить связи и атрибуты сущностей;
- определить наличие альтернативных ключей.

3 Построить логическую информационную модель экономического или производственного процесса на основе проведенного выше анализа.

4 Построить физическую модель и сгенерировать схему БД.

5 Оформить работу.

6 Осуществить защиту работы.

Контрольные вопросы

1 Обоснуйте необходимость использования CASE-средств для моделирования экономических и производственных процессов.

2 Что представляет собой модель системы в нотации IDEF1X?

3 Назовите все возможные типы моделей, используемых при проектировании информационных систем.

4 Перечислите этапы экспертизы модели.

5 Какие виды связей существуют в модели, построенной с использованием ERwin?

6 Как проводится генерация схемы БД в ERwin?

5 Лабораторная работа № 5. Реализация хранилища данных интегрированной информационной системы

Порядок выполнения работы

1 Выбрать вариант реализации хранилища данных, спроектированного в лабораторной работе № 4.

К вариантам реализации информационных хранилищ относятся:

– виртуальное хранилище данных;
– витрины данных;
– глобальное хранилище данных;
– многоуровневая архитектура хранилища данных.

Виртуальное хранилище данных. В его основе – хранилище метаданных, которые описывают источники информации (БД транзакционных систем, внешние файлы и др.), SQL-запросы для их считывания и процедуры обработки и предоставления информации. Непосредственный доступ к последним обеспечивает ПО промежуточного слоя. В этом случае избыточность данных нулевая. Конечные пользователи фактически работают с транзакционными системами напрямую со всеми вытекающими отсюда плюсами (доступ к «живым» данным в реальном времени) и минусами (интенсивный сетевой трафик, снижение производительности OLTP-систем и реальная угроза их работоспособности вследствие неудачных действий пользователей-аналитиков).

Витрина данных (Data Mart) по своему исходному определению – это набор тематически связанных баз данных, которые содержат информацию, относящуюся к отдельным аспектам деятельности корпорации. По сути, витрина данных – это облегченный вариант хранилища данных, содержащий только тематически объединенные данные. Целевая база данных максимально приближена к конечному пользователю и может содержать тематически ориентированные агрегатные данные. Витрина данных, естественно, существенно меньше по объему, чем корпоративное хранилище данных, и для его реализации не требуется особо мощная вычислительная техника.

Глобальное хранилище данных. В последнее время все более популярной становится идея совместить концепции хранилища и витрины данных в одной реализации и использовать хранилище данных в качестве единственного источника интегрированных данных для всех витрин данных. Тогда естественной становится такая трехуровневая архитектура системы.

На первом уровне реализуется корпоративное хранилище данных на основе одной из развитых современных реляционных СУБД. Это хранилище интегрированных в основном детализированных данных. Реляционные СУБД обеспечивают эффективное хранение и управление данными очень большого объема, но не слишком хорошо соответствуют потребностям OLAP-систем, в частности, в связи с требованием многомерного представления данных.

На втором уровне поддерживаются витрины данных на основе многомерной системы управления базами данных (примером такой системы является Oracle Express Server). Такие СУБД почти идеально подходят для целей разработки OLAP-систем, но пока не позволяют хранить сверхбольшие объемы данных (предельный размер многомерной базы данных составляет 10…40 Гбайт). В данном случае это и не требуется, поскольку речь идет о витринах данных. Заметим, что витрина данных необязательно должна быть полностью сформирована. Она может содержать ссылки на хранилище данных и добирать оттуда информацию по мере поступления запросов. Конечно, это несколько увеличивает время отклика, но зато снимает проблему ограниченного объема многомерной базы данных.

На третьем уровне находятся клиентские рабочие места конечных пользователей, на которых устанавливаются средства оперативного анализа данных.

2 Реализовать хранилище данных.

3 Заполнить хранилище данных реальными данными.

4 Оформить работу.

5 Осуществить защиту работы.

Контрольные вопросы

1 Что такое хранилище данных?

2 Дайте сравнительную характеристику различных типов хранилищ данных.

3 Опишите основные методологические подходы к построению хранилища данных.

4 Что такое жизненный цикл хранилища данных?

5 Перечислите основные этапы разработки хранилища данных.

6 Лабораторная работа № 6. Реализация рабочих мест ввода данных в интегрированную информационную систему предприятия

Порядок выполнения работы

1 Выбрать вариант реализации рабочих мест.

2 Выбрать платформу реализации рабочих мест.

3 Реализовать рабочие места.

4 Оформить работу.

5 Осуществить защиту работы.

Контрольные вопросы

1 Что такое автоматизированное рабочее место (АРМ)?
2 Какие требования предъявляются к автоматизированным рабочим местам?
3 Назовите особенности технологии обработки данных на АРМ.
4 Опишите основные режимы работы пользователя на АРМ.
5 Опишите инструментальные средства АРМ.
6 Опишите способы ввода информации.

7 Лабораторная работа № 7. Реализация бизнес-логики процессов интегрированной информационной системы предприятия

Порядок выполнения работы

1 Выбрать способ организации бизнес-логики информационной системы.

Существует три основных способа организации бизнес-логики программной системы: функциональный, объектный и смешанный. Каждый из способов имеет свои преимущества и недостатки, и задача разработчика – выбрать способ, оптимальный для данного проекта.

Сценарий транзакции (функциональный подход).

Это простейший подход к описанию бизнес-логики. За термином «сценарий транзакции» кроется функция системы, реализующая определенную функцию прикладной логики. Например, расчет стоимости заказа, формируемого в системе.

При таком подходе программная система представляет собой набор функций, в котором каждая функция соответствует операции, которую приложение выполняет для пользователя. Это старая добрая процедурная модель, хорошо известная разработчикам. Проектируется иерархия функций, возможно повторное использование функций нижних уровней. Существуют давно и хорошо отработанные методологии, а также стандарты функционального моделирования, с помощью которых можно проектировать функциональную структуру программной системы в сложных случаях.

Несомненное достоинство подхода – простота реализации в программном коде. Однако у функционального подхода есть и определенные недостатки. С возрастанием сложности прикладной логики становится чрезвычайно сложно формировать структуру, не содержащую дублированных функций, а попытка переделать имеющуюся сложную структуру при изменении прикладной логики быстро приводит к тому, что функциональная структура выходит из-под контроля разработчиков. Все это в конечном счете осложняет развитие и поддержку системы.

Модель предметной области (объектный подход).

Этот подход наиболее сложен в реализации, но и наиболее продуктивен. Объектная модель предметной области разрабатывается, по крайней мере, для основных понятий. Например, для интернет-магазина могут быть созданы объекты «клиент», «товар», «корзина», «заказ» и т. д. Вместо того чтобы помещать бизнес-логику расчета стоимости в одну или несколько процедур, для каждого объекта реализуется функциональность исходя из зоны ответственности этого объекта. Например, объект «заказ» содержит алгоритм вычисления стоимости заказа на основе цен входящих в него товаров, при этом алгоритмы определения цен на товары реализуются отдельно, в объектах «товар». Если для разных товаров применяются разные алгоритмы вычисления цены, это достаточно легко реализовать в модели за счет создания разных типов товаров.

Достоинством модели предметной области традиционно признается гибкость. Объектный подход предоставляет в распоряжение разработчика множество способов моделирования отношений и распределения ответственности между объектами, что позволяет настраивать бизнес-логику в очень широких пределах. Кроме того, важной особенностью объектного подхода является то, что объекты могут близко соответствовать понятиям прикладной области, что в принципе позволяет разработчику и заказчику «говорить на одном языке».

Помимо этого, подавляющее большинство типовых решений – шаблонов проектирования – создано именно для применения в объектной модели. Шаблоны позволяют разрабатывать объектные структуры с заданными свойствами. В частности, создатели шаблонов уделяют значительное внимание последствиям применения шаблонов и возможностям внесения изменений в объектные структуры, предлагаемые шаблонами.

Однако чтобы применять объектный подход необходимо выработать определенный, своеобразный стиль мышления. Новичкам приходится затрачивать много времени даже на то, чтобы разобраться в имеющейся объектной структуре (например, в шаблоне), не говоря уже о самостоятельном проектировании объектной структуры. Не всем удается преодолеть барьер освоения этой технологии; многие разработчики, даже работая на объектно ориентированном языке и применяя объектные библиотеки, фактически используют процедурный стиль программирования.

Другая особенность модели предметной области – необходимость создания сложных программных механизмов взаимодействия объектных структур с реляционными базами данных. Разработка такого механизма обычно обходится дорого (хотя на этот счет тоже имеются типовые решения).

Модуль таблицы (смешанный подход).

Перечисленные выше методы организации бизнес-логики представляют собой два «крайних» случая. Третий случай – смешанный, гибридный подход, сочетающий в себе определенные достоинства (и недостатки) функционального и объектного подходов.

Типовое решение «модуль таблицы» предусматривает, как и в модели предметной области, отдельные объекты для товаров, заказов и т. д. Однако,

в отличие от «настоящей» модели предметной области, в модуле таблицы для работы со всеми (к примеру) заказами, содержащимися в базе данных, применяется только один объект. Именно этот единственный объект содержит логику обработки заказов, а чтобы работать с отдельным заказом, следует указывать его уникальный идентификатор.

Недостатком данного подхода является сложность разработки и сложность восприятия кода. С точки зрения прозрачности, читабельности кода модуль таблицы заметно проигрывает «настоящей» модели предметной области (конечно, предполагается, что разработчик в каждом случае использует максимум возможностей сделать код простым и понятным). При несложной логике предметной области модуль таблицы проигрывает в простоте реализации и функциональному подходу.

Основные преимущества модуля таблицы – простота взаимодействия с базой данных и гибкость структурирования бизнес-логики по сравнению с функциональным подходом. Возможно также применение шаблонов проектирования. Однако по сравнению с моделью предметной области, гибкость и возможности применения шаблонов в модуле таблицы существенно ограничены.

Целесообразность применения модуля таблицы также зависит от уровня поддержки структуры множества записей в конкретной среде разработки (включая удобство отображения множества записей на базу данных и на элементы графического интерфейса). Развитая поддержка набора данных (DataSet) в среде Microsoft.Net во многих случаях оправдывает применение этой платформы разработки.

Особенности подходов.

Перечислим основные особенности рассмотренных типовых решений, позволяющие сравнивать эти решения между собой и выбирать подход, соответствующий задаче.

Сценарий транзакции (функциональный подход): предельно прост, плохо работает при сложной логике, затрудняет развитие системы.

Модель предметной области (объектный подход): наиболее сложный и затратный, позволяет реализовать сложную бизнес-логику, дает возможность «цивилизованно» развивать систему; для простых систем подход целесообразен, только если уже «обкатан» разработчиком.

Модуль таблицы (смешанный подход): обеспечивает определенное сочетание простоты и гибкости, однако гибкость ограничена, а эффективность подхода зависит от среды разработки.

На основе рассмотренных особенностей можно сформулировать несколько критериев и предложить упрощенную схему выбора подхода на основе качественной оценки критериев.

Качественные критерии и схема выбора.

Особенности типовых решений показывают, что решение по выбору подхода можно принять на основе качественной оценки следующих критериев:

– опыт команды разработчиков;

– стабильность бизнес-логики (ожидаемая стабильность функциональных требований);

– сложность бизнес-логики (функциональная сложность задач, решаемых программной системой);

– среда разработки (удобство работы с множеством записей).

Схема принятия решения на основе этих критериев приведена на рисунке 1.

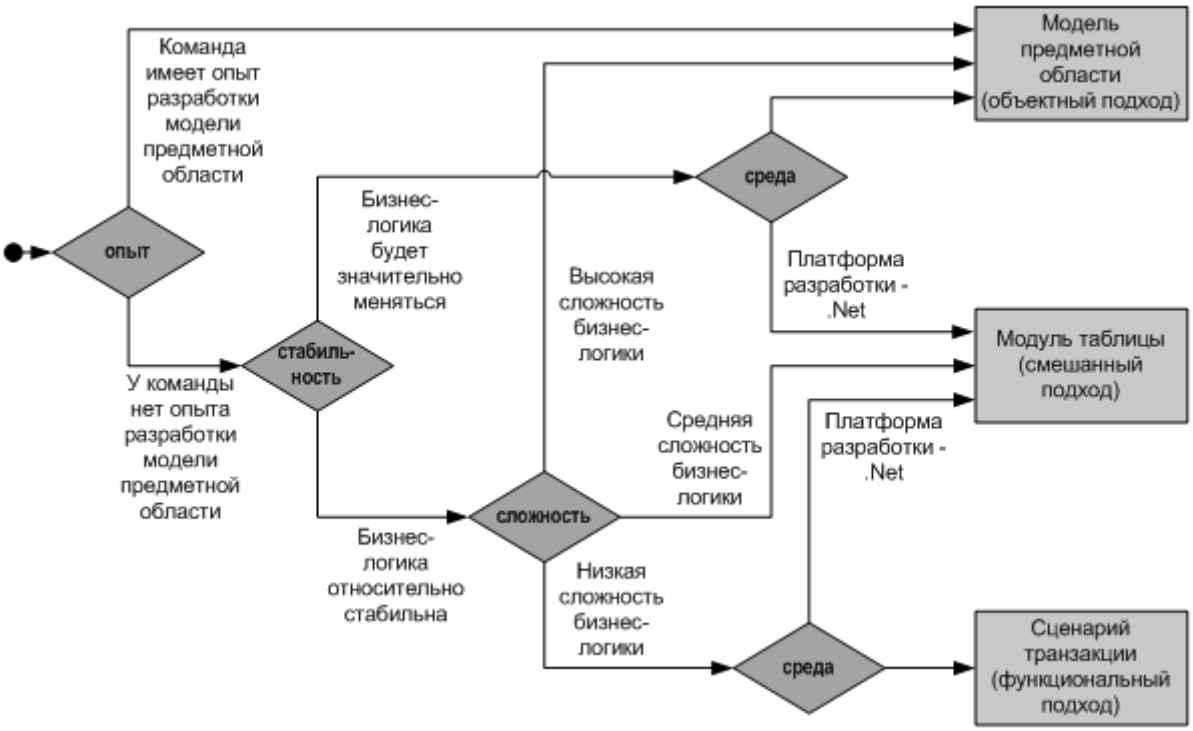


Рисунок 1 – Схема принятия решения

Упрощенная схема выбора типового решения для реализации бизнес-логики.

На схеме хорошо видно, что в некоторых случаях выбор сделать очень легко. Например, когда приложение простое, изменений не предвидится, объектная модель является для разработчика слишком сложной, можно применить простой функциональный подход. С другой стороны, команда разработчиков, хорошо владеющих объектным подходом, будет строить объектную модель предметной области во всех случаях.

Пограничные ситуации требуют оценки стабильности и сложности разработки. На предлагаемом уровне это не должно вызвать затруднений.

Данная методика предназначена в первую очередь для оперативного «прокручивания» в голове с целью избежать грубых ошибок при выборе программного решения. Подобная систематизация также представляется полезной как первый шаг программиста к сознательному и ответственному проектированию приложений в противоположность «проектированию на лету».

В реальных проектах, как обычно, все обстоит сложнее. Например, вопрос организации бизнес-логики может теряться на фоне массы других вопросов

разработки архитектуры. Однако это не значит, что решение нужно пускать на самотек. Приведенная схема показывает, что можно предельно упростить принятие решения, не теряя при этом возможность выявить «подводные камни».

2 Реализовать бизнес-логику процессов интегрированной информационной системы предприятия.

3 Оформить работу.

4 Осуществить защиту работы.

Контрольные вопросы

1 Опишите функциональный способ организации бизнес-логики программной системы.

2 Опишите смешанный способ организации бизнес-логики программной системы.

3 Опишите объектный способ организации бизнес-логики программной системы.

4 Опишите достоинства и недостатки основных способов организации бизнес-логики программной системы.

8 Лабораторная работа № 8. Реализация взаимодействия бизнес-логики процессов интегрированной информационной системы с хранилищем данных

Порядок выполнения работы

1 Разработать уровень обработки данных информационной системы.

Уровень обработки данных объединяет части, реализующие бизнес-логику приложения, и является посредником между уровнем представления данных и уровнем их хранения. Через него проходят все данные и претерпевают в нем изменения, обусловленные решаемой задачей. К функциям этого уровня относятся следующие:

– обработка потоков данных в соответствии с бизнес-правилами;

– взаимодействие с уровнем представления данных для получения запросов и возвращения ответов;

– взаимодействие с уровнем хранения данных для передачи запросов и получения ответов.

Чаще всего уровень обработки данных отождествляют с промежуточным ПО распределенного приложения. Такая ситуация в полной мере верна для «идеальной» системы и лишь отчасти для реальных приложений. Что касается последних, промежуточное ПО для них содержит большую долю правил обработки данных, но часть из них реализована в серверах SQL в виде хранимых процедур или триггеров, а часть включена в состав клиентского ПО.

Такое «размывание» бизнес-логики оправданно, т. к. позволяет упростить часть процедур обработки данных. Возьмем классический пример выписки заказа. В него могут быть включены наименования только тех продуктов, которые имеются на складе. Следовательно, при добавлении в заказ некоторого наименования и определения его количества соответствующее число должно быть вычтено из остатка этого наименования на складе. Очевидно, что лучше всего реализовать эту логику средствами сервера БД – хранимой процедурой или триггером.

2 Оформить работу.

3 Осуществить защиту работы.

Контрольные вопросы

1 Сформулируйте основные требования к современным приложениям масштаба предприятия.

2 Опишите свойства распределенных вычислительных систем.

3 Опишите трехуровневую архитектуру распределенных приложений.

9 Лабораторная работа № 9. Проектирование пользовательского интерфейса интегрированной информационной системы предприятия

Порядок выполнения работы

1 Разработать прототип интерфейса windows-приложения в соответствии с основными принципами проектирования интерфейса.

2 Оформить отчет о проделанной работе, включающий в себя пример использования каждого из шести основных принципов проектирования интерфейса.

3 Осуществить защиту работы.

Контрольные вопросы

1 В чем заключается структурный принцип? Каким образом он был использован в интерфейсе разработанной программы?

2 В чем заключается принцип простоты? Каким образом он был использован в интерфейсе разработанной программы?

3 В чем заключается принцип видимости? Каким образом он был использован в интерфейсе разработанной программы?

4 В чем заключается принцип обратной связи? Каким образом он был использован в интерфейсе разработанной программы?

5 В чем заключается принцип толерантности? Каким образом он был использован в интерфейсе разработанной программы?

6 Каким образом производится обработка событий для элементов интерфейса windows-приложения?

7 Каким образом следует проверять ошибки во введенных пользователем данных и каким образом сообщать о них?

10 Лабораторная работа № 10. Реализация взаимодействия пользовательского интерфейса с бизнес-логикой процессов интегрированной информационной системы предприятия

Порядок выполнения работы

1 Реализовать прототип интерфейса windows-приложения (веб-приложения) в соответствии с основными принципами проектирования пользовательского интерфейса. При этом должны быть реализованы все возможные проверки данных, вводимых пользователем. Функциональность приложений должна быть достаточна для выполнения всех выделенных бизнес-процессов из лабораторной работы № 2.

2 Оформить отчет о проделанной работе, отражающий пошаговое выполнение всех бизнес-процессов с помощью разработанных интерфейсов.

3 Осуществить защиту работы

Контрольные вопросы

1 В чем заключаются основные отличия веб-интерфейса от интерфейса windows-приложения?

2 Какими преимуществами обладает веб-интерфейс в сравнении с интерфейсом windows-приложения?

3 Какими недостатками обладает веб-интерфейс в сравнении с интерфейсом windows-приложения?

4 В каких случаях целесообразно применять веб-интерфейс? Какие элементы интерфейса могут использоваться при построении веб-интерфейса?

5 Отличаются ли эти элементы веб-интерфейса от соответствующих элементов windows-приложения?

6 Каким образом производится обработка событий для элементов веб-интерфейса?

7 Какую роль играет HTML в построении веб-интерфейса? Каким образом производится проверка вводимых пользователем данных в веб-приложении? В чем заключаются отличия данного способа проверки от проверки данных в windows-приложении?

11 Лабораторная работа № 11. Оптимизация пользовательского интерфейса интегрированной информационной системы предприятия

Порядок выполнения работы

1 Оптимизировать интерфейс windows-приложения (веб-приложения) в соответствии с основными принципами проектирования пользовательского интерфейса.

В лабораторной работе необходимо улучшить пользовательский интерфейс, добавив, например, дополнительные проверки вводимых данных, улучшив шаблоны печатаемых документов и т. д.

При модификации пользовательского интерфейса руководствуйтесь следующими рекомендациями:

– используйте шаблон Separated Presentation, такой как MVP, для отделения компоновки пользовательского интерфейса от его обработки. С помощью шаблонов обеспечьте единообразие внешнего вида и взаимодействия со всеми окнами UI и единый внешний вид и стиль взаимодействия для всех элементов UI, чтобы обеспечить максимальное удобство доступа и простоту использования. Избегайте слишком сложных компоновок;

– используйте основанные на применении форм элементы управления для задач сбора данных, механизм ввода на базе документов для ввода данных в более свободной форме, таких как текстовые или графические документы; или подход с применением мастеров для более упорядоченных или управляемых рабочим процессом задач сбора данных;

– избегайте применения жестко кодированных строк и внешних ресурсов для текстовых данных или данных компоновки (например, для поддержки языков с написанием справа налево), особенно если приложение будет подлежать локализации;

– учтите удобство и простоту доступа. При продумывании стратегии ввода следует подумать о пользователях с ограниченными возможностями; например, реализовать ПО для преобразования текста в речь для слепых пользователей или увеличить текст и изображение для пользователей с проблемами зрения. По возможности, поддерживайте сценарии работы только через клавиатуру для пользователей, которые не могут работать с координатно-указательными устройствами;

– принимайте во внимание наличие экранов разных размеров и с разными разрешениями, существование разных типов устройств и разных типов ввода, таких как мобильные устройства, сенсорные экраны и устройства с возможностью рукописного ввода. Например, для сенсорных экранов обычно используются большие кнопки с большими расстояниями между ними, чем для обычных UI, предназначенных для ввода только с помощью мыши или клавиатуры. Для

создания компоновки веб-приложения пользуйтесь каскадными таблицами стилей (Cascading Style Sheets, CSS). Это обеспечит более высокую производительность формирования визуального представления и удобство обслуживания.

2 Оформить отчет.

3 Осуществить защиту работы.

Контрольные вопросы

1 В чем заключаются основные отличия веб-интерфейса от интерфейса windows-приложения?

2 Какими преимуществами обладает веб-интерфейс в сравнении с интерфейсом windows-приложения?

3 Какими недостатками обладает веб-интерфейс в сравнении с интерфейсом windows-приложения?

4 В каких случаях целесообразно применять веб-интерфейс? Какие элементы интерфейса могут использоваться при построении веб-интерфейса?

5 Отличаются ли эти элементы веб-интерфейса от соответствующих элементов windows-приложения?

6 Каким образом производится обработка событий для элементов веб-интерфейса?

7 Какую роль играет HTML в построении веб-интерфейса? Каким образом производится проверка вводимых пользователем данных в веб-приложении? В чем заключаются отличия данного способа проверки от проверки данных в windows-приложении?

8 Назовите основные типы объектов интерфейсов прямого манипулирования. Охарактеризуйте каждый из них.

9 В каких случаях предпочтительно использовать элементы анимации?

12 Лабораторная работа № 12. Тестирование бизнес-логики разработанной интегрированной информационной системы предприятия

Порядок выполнения работы

1 Разработать рабочую тестовую документацию.

Создание тестовой документации значительно улучшает качество продукта за счет более тесного сотрудничества, уточнения деталей при разработке плана тестирования и документации. После завершения тестирования наличие тестовой документации позволяет проверить, насколько успешно были проведены все этапы тестирования.

Существует несколько разновидностей рабочей тестовой документации:

1) Check List;

2) Acceptance Sheet;
3) Test Survey;
4) Test Cases.

Check List – высокоуровневый список проверок, набор правил и критериев, по которым проводится тестирование приложения. Описывает основные проверки для типовой функциональности.

Acceptance Sheet – документ, который содержит подробный перечень всех модулей и функций приложения, а также результаты всех тестов данных функций. Как правило, содержит статистику по наиболее важным показателям каждой сборки, определяющим ее качество.

Test Survey – документ, который содержит подробный перечень всех модулей и функций приложения, конкретные проверки для них, а также результаты всех тестов. В некоторых случаях для проверок может быть указан ожидаемый результат. Как правило, содержит статистику по наиболее важным показателям каждой сборки, определяющим ее качество.

Test Cases (набор тест-кейсов) – набор входных значений, предусловий выполнения, ожидаемых результатов и постусловий выполнения, разработанный для определенной цели или тестового условия, таких как выполнение определенного пути программы, или же для проверки соответствия определенному требованию.

Основной фактор выбора тестовой документации – сложность бизнес-логики проекта. Кроме того, определяющими факторами могут быть сроки проекта, размер команды и объем проекта.

На одном проекте могут комбинироваться несколько типов тестовой документации. Например, для всего проекта составлен Test Survey, но для наиболее сложных частей составлены тест-кейсы.

2 Оформить отчет.
3 Осуществить защиту работы.

Контрольные вопросы

1 Какие существуют разновидности рабочей тестовой документации?
2 Check List: что описывает и когда используют?
3 Acceptance Sheet: что описывает и когда используют?
4 Test Survey: что описывает и когда используют?
5 Test Cases: что описывает и когда используют?

13 Лабораторная работа № 13. Тестирование пользовательского интерфейса разработанной интегрированной информационной системы предприятия

Порядок выполнения работы

1 Спроектировать тесты для оценки удобства применения разработанного интерфейса.

2 Выполнить спроектированные тесты с привлечением сторонних пользователей.

3 Проанализировать полученные результаты.

4 Оформить отчет. Отчет должен содержать таблицу результатов выполнения тестов (таблица 2).

Таблица 2 – Результаты выполнения тестов

Краткое описание тестируемого действия	Ожидаемый результат	Фактический результат

5 Осуществить защиту работы.

Контрольные вопросы

1 Что такое тестирование для оценки удобства применения?

2 Какие требования предъявляются к тестированию?

3 Какие государственные стандарты регламентируют проведение тестирования?

14 Лабораторная работа № 14. Построение модели бизнес-процесса планирования в разработанном модуле интегрированной информационной системы предприятия

Порядок выполнения работы

1 Согласно номеру своего варианта получить исходные данные.

2 В соответствии с правилами построения сетевых графиков и на основе исходных данных варианта построить сетевую модель.

3 В соответствии с методиками сетевого планирования:

– рассчитать и отобразить на сетевом графике временные параметры событий: ранний и поздний срок свершения события, резерв события;

– рассчитать и представить в таблице временные параметры работ: время раннего и позднего начала работ; время раннего и позднего окончания работ; полный и свободный резервы работ.

4 Оформить отчет.

5 Осуществить защиту работы.

Контрольные вопросы

1 Дайте определение понятию «события». Назовите виды событий. Приведите примеры событий.

2 Приведите обозначение событий на графике.

3 Перечислите временные параметры событий. Опишите алгоритм расчета временных параметров событий.

4 Дайте определение понятию «работа».

5 Классифицируйте работы с приведением соответствующих практических примеров.

6 Приведите обозначение работ на графике.

7 Назовите временные параметры работ и правила их нахождения.

8 Приведите правила построения сетевых графиков.

9 Дайте определение пути в сетевом графике, назовите виды путей.

10 Дайте определение понятию «критический путь». Каковы правила нахождения критического пути?

11 Какова взаимосвязь полного и свободного резервов работы?

12 Как можно найти критический путь в сетевой модели без непосредственного суммирования длительностей работ?

15 Лабораторная работа № 15. Получение исходных данных разработанной модели из хранилища данных интегрированной информационной системы предприятия

Порядок выполнения работы

1 Разработать уровень представления данных.

Уровень представления данных – единственный доступный конечному пользователю. Этот уровень моделирует клиентские рабочие места распределенного приложения и соответствующее ПО. Возможности клиентского рабочего места в первую очередь определяются возможностями операционной системы. В зависимости от типа пользовательского интерфейса клиентское ПО делится на две группы: клиенты, использующие возможности ГИП (например, Windows), и веб-клиенты. Но в любом случае клиентское приложение должно обеспечивать выполнение следующих функций:

– получение данных;

– представление данных для просмотра пользователем;

– редактирование данных;

– проверка корректности введенных данных;

– сохранение сделанных изменений;

– обработка исключительных ситуаций и отображение информации об ошибках для пользователя.

Все бизнес-правила желательно сконцентрировать на уровне обработки данных, но на практике это не всегда удается. Тогда говорят о двух типах клиентского ПО. «Тонкий» клиент содержит минимальный набор бизнес-правил, «толстый» реализует значительную долю логики приложения. В первом случае распределенное приложение существенно легче отлаживать, модернизировать и расширять, во втором – можно минимизировать расходы на создание и поддержание уровня управления данными, т. к. часть операций может выполняться на стороне клиента, а на долю промежуточного ПО ложится только передача данных.

2 Оформить отчет.

3 Осуществить защиту работы.

Контрольные вопросы

1 Перечислите функции клиентского приложения.

2 Опишите назначение уровня представления данных.

3 Опишите разницу между «тонким» и «толстым» клиентом.

16 Лабораторная работа № 16. Анализ результатов моделирования бизнес-процесса планирования в разработанном модуле интегрированной информационной системы предприятия

Порядок выполнения работы

1 Произвести имитационное моделирование бизнес-процесса, декомпозированного в лабораторной работе № 2. Предложить шаги по оптимизации данного бизнес-процесса. Рассчитать необходимое количество сотрудников в должности.

Получить отчеты «ФСА процесса», «Использование материального ресурса», отчет по результатам имитации.

Имитационное моделирование – метод исследования, основанный на том, что изучаемая система заменяется имитирующей. С имитирующей системой проводят эксперименты (не прибегая к экспериментам на реальном объекте) и в результате получают информацию об изучаемой системе. Метод позволяет имитировать выполнение модели бизнес-процессов так, как оно происходило бы в действительности. В результате можно оценить время выполнения как одного процесса, так и заданного их множества, и среднюю частоту повторений подпроцессов в рамках процесса.

Функционально-стоимостный анализ используется для операционно-ориентированного расчета себестоимости продукта (услуги). В основе ФСА лежит положение о том, что для производства продукта (услуги) необходимо выполнить ряд действий, каждое из которых требует определенных ресурсов. Расходы на выполнение каждого действия рассчитываются путем переноса стоимости ресурсов на стоимость действия. Сумма расходов на выполнение каждого действия с определенными поправками и будет составлять себестоимость продукта (услуги).

В Business Studio имитационное моделирование и функционально-стоимостный анализ используются параллельно для расчета времени выполнения и стоимости процессов. Функционально-стоимостный анализ позволяет рассчитать себестоимость продукции (услуги) через перенос затрат на стоимость выполняемых процессов пропорционально драйверам ресурсов. За драйвер временных ресурсов принимается время, затрачиваемое ресурсом на выполнение того или иного процесса (действия, функции). За драйвер материальных ресурсов принимается количество повторений процесса. Время выполнения и количество повторений процесса определяется посредством имитацион ного моделирования.

Описание методики имитационного моделирования.

Анализ деятельности компании с помощью методики имитационного моделирования осуществляется в три этапа:

1) разрабатывается модель бизнес-процессов компании либо диаграмма отдельного исследуемого бизнес-процесса;

2) для недекомпозированных процессов, входящих в исследуемые бизнес-процессы, заполняются следующие параметры: «Время выполнения процесса», «Время ожидания процесса». Для подпроцессов процесса в нотации 1DKF0 заполняется также параметр «Частота в рамках вышележащего процесса»;

3) проводится имитация для всей модели бизнес-процессов либо для одного исследуемого процесса и в результате определяется время, которое затрачивается на выполнение процессов.

При имитации бизнес-процесса в нотации 1DEF0 для определения времени выполнения процесса система суммирует продолжительности подпроцессов с учетом частоты их повторений в рамках бизнес-процесса.

Ход выполнения процессов в нотациях «Процедура», «Процесс», EPC в общем случае носит вероятностный характер, поэтому продолжительность процесса в общем случае является случайной величиной.

Правила расчета времени для процессов нотаций «Процедура», «Процесс», EPC

Последовательный блок.

При последовательном выполнении действий или функций их продолжительность суммируется и включается в общее время выполнения процесса.

Параллельный блок.

При параллельном выполнении веток процесса последовательно выполняются действия или функции всех веток, но в общую продолжительность процесса включается продолжительность той ветки, время выполнения которой наибольшее.

Блок с условиями.

В тех случаях, когда действия «Процедуры», «Процесса» выполняются в зависимости от какого-то условия, для обозначения условия используется специальный элемент «Решение». Стрелкам «Связь предшествования», исходящим из этого элемента, задается вероятность перехода к следующим действиям.

В тех случаях, когда функции EPC выполняются в зависимости от какого-то условия, для обозначения условия используются операторы or и xor. Событиям, следующим за этими операторами, задается вероятность перехода к следующим функциям.

При имитации «Процесса», как только система достигает одного из указанных операторов, она каждый раз в соответствии с заданной вероятностью принимает решение, какой путь выбрать.

При имитации процессов с условиями суммируется время выполнения пройденных системой действий или функций, и таким образом рассчитывается время выполнения всего «Процесса».

Описание методики ФСА.

Стоимость процесса определяется в результате проведения функционально-стоимостного анализа в пять этапов:

1) разрабатывается модель бизнес-процессов компании либо диаграмма отдельного исследуемого бизнес-процесса;

2) для недекомпозированных процессов, входящих в исследуемые бизнес-процессы, заполняются параметры «Время выполнения процесса», «Время ожидания процесса». Для подпроцессов процесса нотации IDEF0 заполняется также параметр «Частота в рамках вышележащего процесса»;

3) заполняются стоимостные параметры тех ресурсов, которые будут использованы при выполнении процессов. Ресурсы могут быть временными (стоимость использования зависит от времени выполнения процесса) и материальными (стоимость зависит от количества повторений процесса);

4) на каждый бизнес-процесс назначаются временные и материальные ресурсы, используемые при его выполнении;

5) проводится имитация для всей модели бизнес-процессов либо для одного исследуемого процесса, в результате определяется стоимость процессов.

Для «Процесса» в нотации IDEF0 в общей стоимости «Процесса» учитывается стоимость каждого подпроцесса, умноженная на частоту его выполнения в рамках «Процесса».

Для процесса в нотациях «Процедура», «Процесс», EPC стоимость процесса определяется как сумма стоимостей всех выполненных действий/функций.

Стоимость ресурсов переносится на стоимость «Процесса» пропорционально драйверам ресурсов. За драйвер временных ресурсов принимается время выполнения «Процесса». За драйвер материальных ресурсов принимается количество повторений «Процесса».

Стоимость временных ресурсов переносится на стоимость «Процесса» путем умножения времени выполнения «Процесса» на стоимость единицы используемого временного ресурса, например, на стоимость часа работы сотрудника.

Стоимость материальных ресурсов переносится на стоимость «Процесса» путем умножения заданной стоимости материального ресурса на количество повторений «Процесса».

2 Оформить отчет.

3 Осуществить защиту работы.

Контрольные вопросы

1 Раскройте понятия «имитационное моделирование», «функционально-стоимостный анализ».

2 Для чего предназначены имитационное моделирование и функционально-стоимостный анализ?

3 Этапы имитационного моделирования.

4 Раскройте понятия «последовательный блок», «параллельный блок», «блок с условиями».

5 Основные этапы ФСА.

6 Расчет процесса в нотации IDEF0.

7 Расчет стоимости в ФСА, взаимосвязи переноса стоимости.

8 Задание вероятностей, событий, имитаций.

9 Изменение параметров: стоимость процесса, единица измерения стоимости.

10 Смысл использования имитации процесса.

11 Последовательность оптимизации.

12 Этапы расчета штатного расписания.

17 Лабораторная работа № 17. Рациональный выбор параметров бизнес-процесса планирования в разработанном модуле интегрированной информационной системы предприятия

Порядок выполнения работы

1 Выбрать два бизнес-процесса, декомпозированных в лабораторной работе № 2, удовлетворяющих следующей ситуации.

Пусть имеется некоторый бизнес-процесс «А». У этого бизнес-процесса есть владелец, отвечающий за его результативность и эффективность. Процесс выполняется по определенной технологии. Представим, что некоторый документ (или продукт), формируемый по ходу выполнения процесса «А», должен

быть изменен (проверен, доработан и т. п.) в процессе «Б» (функция «Х»). В данном случае бизнес-процесс «Б» фактически является субподрядчиком (или поставщиком процесса «А») бизнес-процесса «А».

Если процессы «А» и «Б» находятся в разных функциональных подразделениях (владельцы процессов «А» и «Б» подчиняются руководителям разных функциональных структур), то часто возникают проблемы при взаимодействии между процессами: срыв сроков предоставления документов, ошибки и несоответствия в оформлении и т. п. В результате увеличивается время выполнения работ по процессу «А», снижается качество его продуктов и т. п.

2 Рассмотреть вопрос перераспределения функций между процессами.

3 Оформить отчет.

4 Осуществить защиту работы.

Контрольные вопросы

1 Понятие и сущность системы стратегического управления.

2 Каким образом происходит интеграция системы стратегического управления с системой управления бизнес-процессами?

3 Структура системы целей, показателей и критериев.

4 Каким образом формируется карта стратегий?

5 Показатели, которые необходимо учитывать при разработке финансовых, рыночных целей, целей по улучшению бизнес-процессов.

6 Типичные ошибки при формировании карты стратегии.

7 Входы и выходы системы стратегического управления.

18 Лабораторная работа № 18. Построение модели взаимодействия компонентов разработанного модуля интегрированной информационной системы предприятия

Порядок выполнения работы

1 Разработать UML-диаграммы взаимодействия.

Диаграмма деятельности – это технология, позволяющая описывать логику процедур, бизнес-процессы и потоки работ. Во многих случаях они напоминают блок-схемы, но принципиальная разница между *диаграммами деятельности* и нотацией *блок-схем* заключается в том, что первые поддерживают параллельные процессы.

Диаграмма деятельности подвергалась самым большим изменениям при смене версий языка UML, поэтому неудивительно, что она были снова изменена и существенно расширена в UML 2. В UML1 диаграмма деятельности рассматривалась как особый случай диаграмм состояний. Это вызвало немало трудностей у специалистов, моделирующих потоки работ, для которых хорошо подходят диаграммы деятельности. В UML 2 это ограничение было ликвидировано.

На рисунке 2 показан пример простой диаграммы деятельности. Выполнение начинается с начального узла (*initial node*), затем выполняется операция *Receive Order* (Принять заказ). Затем идет ветвление (*fork*), которое имеет один входной поток и несколько выходных параллельных потоков.

Из рисунка 2 видно, что операции *Fill Order* (Заполнить заявку), *Send Invoice* (Послать счет) и следующие за ними выполняются параллельно. По существу, в данном случае это означает, что последовательность операций не имеет значения. Можно заполнить заявку, послать счет, доставить товар (Delivery), а затем получить оплату (*Receive Payment*); или можно послать счет, получить оплату, заполнить заявку, а затем доставить товар.

Можно также выполнять операции поочередно. Берем со склада первую позицию заказа, печатаем счет, берем вторую позицию заказа, кладем счет в конверт и т. д. Или можно что-то делать одновременно: печатать счет одной рукой, а другой рукой брать что-нибудь со склада. Согласно диаграмме, любая из этих последовательностей действий допустима.

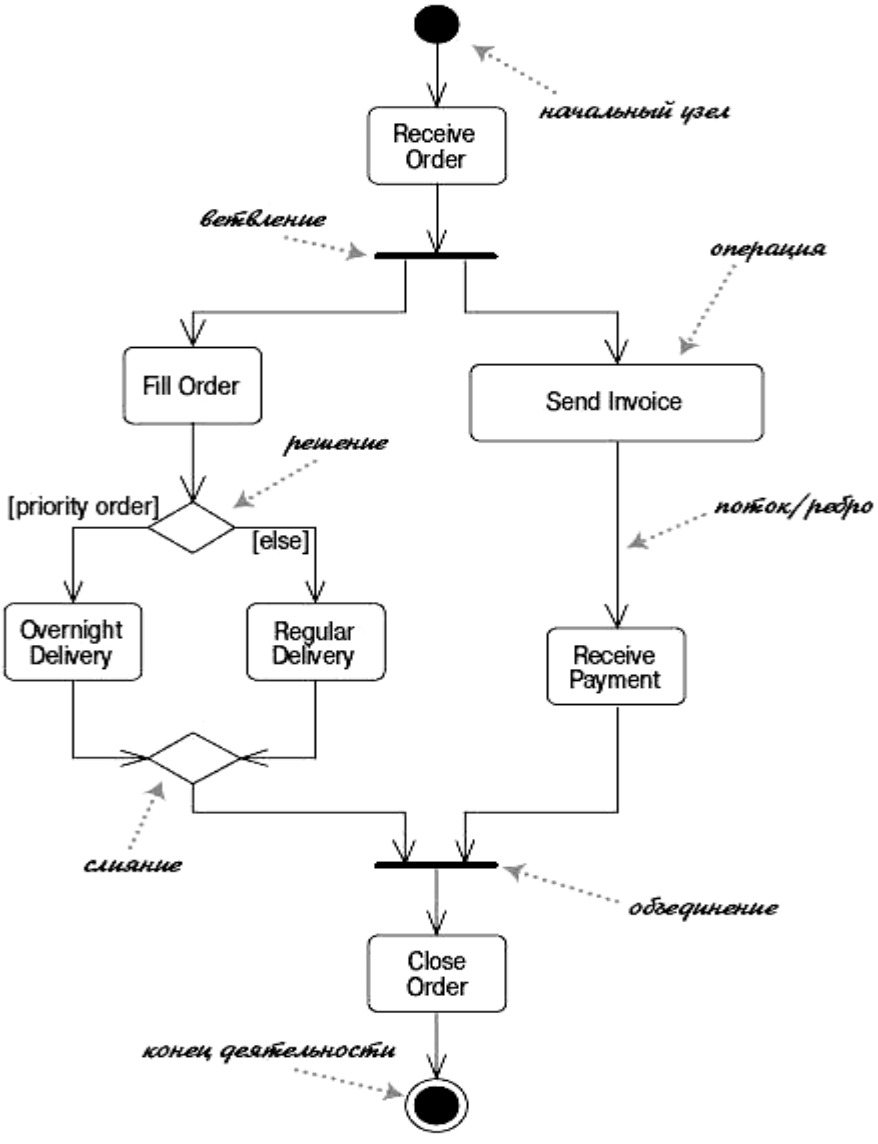


Рисунок 2 – Простая диаграмма деятельности

Диаграмма деятельности позволяет любому, кто выполняет данный процесс, выбирать порядок действий. Другими словами, диаграмма только устанавливает правила обязательной последовательности действий, которым нужно следовать. Это важно для моделирования бизнес-процессов, поскольку эти процессы часто выполняются параллельно. Такие диаграммы также полезны при разработке параллельных алгоритмов, в которых независимые потоки могут выполнять работу параллельно.

При наличии параллелизма необходима синхронизация. Заказ не закрываем, пока он не оплачен и не доставлен. Это показывается с помощью объединения (join) перед операцией *Close Order* (Закрыть заказ). Исходящий из объединения поток выполняется только в том случае, когда все входящие потоки достигли объединения. Поэтому можно закрыть заказ, только когда принята оплата и заказ доставлен.

Заметьте, что узлы на диаграмме деятельности называются операциями (actions), а не активностями (activities). Строго говоря, деятельность относится к последовательности действий, поэтому диаграмма представляет деятельность, состоящую из операций.

Условное поведение схематически обозначается с помощью решений (decisions) и слияний (merges)**.** Решение, которое в UML 1 называлось ветвью, имеет один входящий поток и несколько защищенных выходных потоков. Каждый выходной поток имеет защиту – условное выражение, помещенное в квадратные скобки. Каждый раз при достижении решения выбирается только один из выходных потоков, поэтому защиты должны быть взаимно исключающими. Применение [else] в качестве защиты означает, что поток [else] используется в том случае, когда другие защиты данного решения принимают ложное значение.

На рисунке 2 решение располагается после операции заполнения заявки. Если заказ срочный, то он выполняется в течение суток (*Overnight Delivery*); в противном случае производится обычная доставка (*Regular Delivery*).

Слияние (merge) имеет несколько входных потоков и один выходной. Слияние обозначает завершение условного поведения, которое было начато решением. На рассматриваемой диаграмме каждая операция имеет один входящий в нее поток и один выходящий. В UML1 подразумевалось, что несколько входящих потоков имеют слияние. Другими словами, операция выполнялась, если запускался любой поток. В UML2 это было изменено, так что вместо слияния предполагается объединение; таким образом, операция выполняется, только если все потоки пройдены. Поэтому рекомендуем применять операции с единственным входным потоком и единственным выходным, а также явно показывать все объединения и слияния; это избавит от ошибки.

Декомпозиция операции в диаграмме деятельности.

Операции могут быть разбиты на вложенные деятельности (**subactivities**). Можно взять алгоритм доставки, показанный на рисунке 2, и определить его как самостоятельную деятельность (рисунок 3), а затем вызвать его как операцию (рисунок 4).

Операции могут быть реализованы или как вложенные деятельности или как методы классов. Вложенную деятельность можно обозначить с помощью символа «грабли».

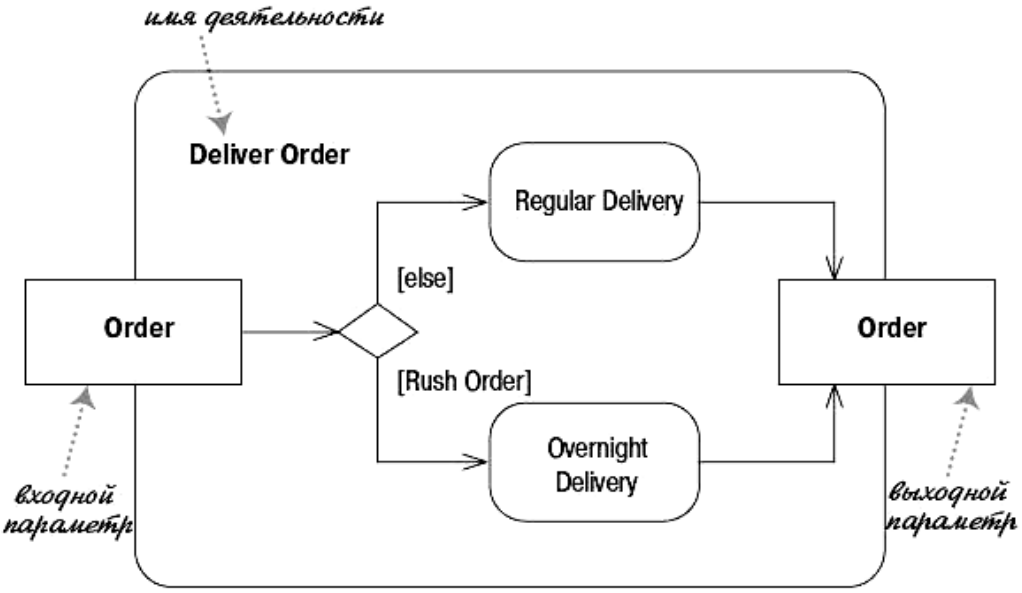


Рисунок 3 – Дополнительная диаграмма деятельности

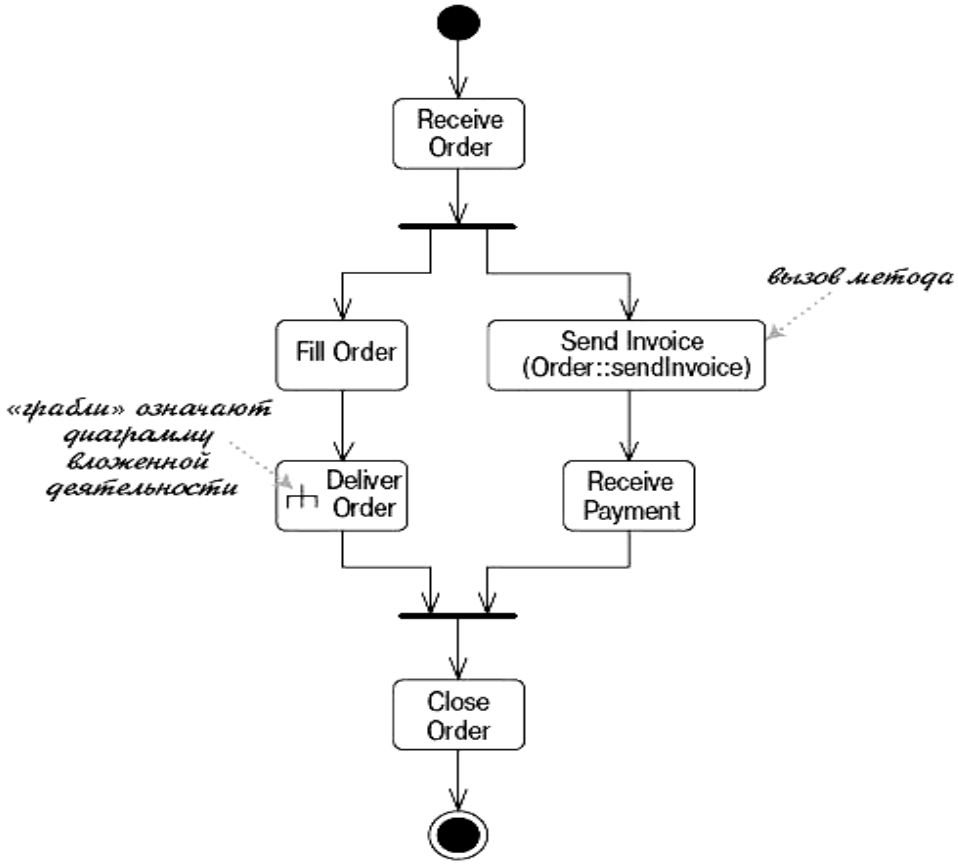


Рисунок 4 – Модифицированная диаграмма деятельности

Вызов метода отображается с помощью синтаксиса имя-класса::имя-метода. Можно также вставить в символ операции фрагмент кода, если поведение представлено не единственным вызовом метода.

2 Оформить отчет.

3 Осуществить защиту работы.

Контрольные вопросы

1 Дайте определение понятию «диаграмма деятельности».

2 Опишите назначение диаграммы деятельности.

3 Дайте определение понятиям «состояние деятельности» и «состояние действия». Графическое изображение состояния.

4 Приведите пример ветвления и параллельных потоков управления процессами на диаграмме деятельности.

5 Какие переходы используются на диаграмме деятельности?

6 Что представляет собой дорожка на диаграмме деятельности?

7 Как графически изображаются объекты на диаграмме деятельности?

19 Лабораторная работа № 19. Построение модели размещения компонентов разработанного модуля интегрированной информационной системы предприятия

Порядок выполнения работы

1 Разработать UML-диаграмму размещения.

Для создания конкретной физической системы необходимо реализовать все элементы логического представления в конкретные материальные сущности. Для описания таких реальных сущностей предназначен другой аспект модельного представления, а именно – физическое представление модели. В контексте языка UML это означает совокупность связанных физических сущностей, включая программное и аппаратное обеспечение, а также персонал, которые организованы для выполнения специальных задач. Физическая система – реально существующий прототип модели системы.

Полный проект программной системы представляет собой совокупность моделей логического и физического представлений, которые должны быть согласованы между собой. В языке UML для физического представления моделей систем используются так называемые диаграммы реализации, которые включают в себя две отдельные канонические диаграммы: диаграмму компонентов и диаграмму развертывания.

Диаграмма компонентов (component diagram), в отличие от ранее рассмотренных диаграмм, описывает особенности физического представления системы. Диаграмма компонентов позволяет определить архитектуру разрабатываемой системы, установив зависимости между программными компонентами,

в роли которых может выступать исходный, бинарный и исполняемый код. Во многих средах разработки модуль или компонент соответствует файлу. Пунктирные стрелки, соединяющие модули, показывают отношения взаимозависимости, аналогичные тем, которые имеют место при компиляции исходных текстов программ. Основными графическими элементами диаграммы компонентов являются компоненты, интерфейсы и зависимости между ними.

В разработке диаграмм компонентов участвуют как системные аналитики и архитекторы, так и программисты. Диаграмма компонентов обеспечивает согласованный переход от логического представления к конкретной реализации проекта в форме программного кода. Одни компоненты могут существовать только на этапе компиляции программного кода, другие – на этапе его исполнения. Диаграмма компонентов отражает общие зависимости между компонентами, рассматривая последние в качестве отношений между ними.

Для представления физических сущностей в языке UML применяется специальный термин – компонент.

Компонент (component) – физически существующая часть системы, которая обеспечивает реализацию классов и отношений, а также функционального поведения моделируемой программной системы.

Компонент предназначен для представления физической организации ассоциированных с ним элементов модели. Дополнительно компонент может иметь текстовый стереотип и помеченные значения, а некоторые компоненты – собственное графическое представление. Компонентом может быть исполняемый код отдельного модуля, командные файлы или файлы, содержащие интерпретируемые скрипты.

Компонент служит для общего обозначения элементов физического представления модели и может реализовывать некоторый набор интерфейсов. Для графического представления компонента используется специальный символ – прямоугольник со вставленными слева двумя более мелкими прямоугольниками (рисунок 5). Внутри объемлющего прямоугольника записывается имя компонента и, возможно, дополнительная информация. Этот символ является базовым обозначением компонента в языке UML.

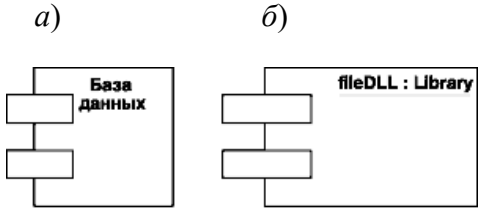


Рисунок 5 – Графическое изображение компонента

Графическое изображение компонента ведет свое происхождение от обозначения модуля программы, применявшегося некоторое время для отображения особенностей инкапсуляции данных и процедур.

Модуль (module) – часть программной системы, требующая памяти для своего хранения и процессора для исполнения.

В этом случае верхний маленький прямоугольник концептуально ассоциировался с данными, которые реализует этот компонент (иногда он изображается в форме овала). Нижний маленький прямоугольник ассоциировался с операциями или методами, реализуемыми компонентом. В простых случаях имена данных и методов записывались явно в маленьких прямоугольниках, однако в языке UML они не указываются.

Имя компонента подчиняется общим правилам именования элементов модели в языке UML и может состоять из любого числа букв, цифр и знаков препинания. Отдельный компонент может быть представлен на уровне типа или экземпляра. И хотя его графическое изображение в обоих случаях одинаково, правила записи имени компонента несколько отличаются.

Если компонент представляется на уровне типа, то записывается только имя типа с заглавной буквы в форме <Имя типа>. Если же компонент представляется на уровне экземпляра, то его имя записывается в форме <имя компонента>:<Имя типа>. При этом вся строка имени подчеркивается.

В отдельных случаях к простому имени компонента может быть добавлена информация об имени объемлющего пакета и о конкретной версии реализации данного компонента. Необходимо заметить, что в этом случае номер версии записывается как помеченное значение в фигурных скобках. В других случаях символ компонента может быть разделен на секции, чтобы явно указать имена реализованных в нем классов или интерфейсов. Такое обозначение компонента называется *расширенным*.

Поскольку компонент как элемент модели может иметь различную физическую реализацию, иногда его изображают в форме специального графического символа, иллюстрирующего конкретные особенности реализации. Строго говоря, эти дополнительные обозначения не специфицированы в нотации языка UML. Однако, удовлетворяя общим механизмам расширения языка UML, они упрощают понимание диаграммы компонентов, существенно повышая наглядность графического представления.

Для более наглядного изображения компонентов были предложены и стали общепринятыми следующие графические стереотипы:

– стереотипы для компонентов развертывания, которые обеспечивают непосредственное выполнение системой своих функций. Такими компонентами могут быть динамически подключаемые библиотеки, веб-страницы на языке разметки гипертекста и файлы справки;

– стереотипы для компонентов в форме рабочих продуктов. Как правило, это файлы с исходными текстами программ.

Эти элементы иногда называют **артефактами**, подчеркивая при этом их законченное информационное содержание, зависящее от конкретной технологии реализации соответствующих компонентов. Более того, разработчики могут для этой цели использовать самостоятельные обозначения, поскольку в языке UML нет строгой нотации для графического представления артефактов.

Другой способ спецификации различных видов компонентов – указание текстового стереотипа компонента перед его именем. В языке UML для компонентов определены следующие стереотипы:

– «file» (файл) – определяет наиболее общую разновидность компонента, который представляется в виде произвольного физического файла;

– «executable» (исполнимый) – определяет разновидность компонента-файла, который является исполнимым файлом и может выполняться на компьютерной платформе;

– «document» (документ) – определяет разновидность компонента-файла, который представляется в форме документа произвольного содержания, не являющегося исполнимым файлом или файлом с исходным текстом программы;

– «library» (библиотека) – определяет разновидность компонента-файла, который представляется в форме динамической или статической библиотеки;

– «source» (источник) – определяет разновидность компонента-файла, представляющего собой файл с исходным текстом программы, который после компиляции может быть преобразован в исполнимый файл;

– «table» (таблица) – определяет разновидность компонента, который представляется в форме таблицы базы данных.

Следующим графическим элементом диаграммы компонентов являются интерфейсы. В общем случае интерфейс графически изображается окружностью, которая соединяется с компонентом отрезком линии без стрелок. При этом имя интерфейса, которое рекомендуется начинать с заглавной буквы «I», записывается рядом с окружностью. Семантически линия означает реализацию интерфейса, а наличие интерфейсов у компонента означает, что данный компонент реализует соответствующий набор интерфейсов.

Кроме того, интерфейс на диаграмме компонентов может быть изображен в виде прямоугольника класса со стереотипом «interface» и секцией поддерживаемых операций. Как правило, этот вариант обозначения используется для представления внутренней структуры интерфейса.

При разработке программных систем интерфейсы обеспечивают не только совместимость различных версий, но и возможность вносить существенные изменения в одни части программы, не изменяя других. Характер применения интерфейсов отдельными компонентами может отличаться.

Различают два способа связи интерфейса и компонента. Если компонент реализует некоторый интерфейс, то такой интерфейс называют экспортируемым или *поддерживаемым*, поскольку этот компонент предоставляет его в качестве сервиса другим компонентам. Если же компонент использует некоторый интерфейс, который реализуется другим компонентом, то такой интерфейс для первого компонента называется *импортируемым*. Особенность импортируемого интерфейса состоит в том, что на диаграмме компонентов это отношение изображается с помощью зависимости.

Отношение зависимости служит для представления факта наличия специальной формы связи между двумя элементами модели, когда изменение одного элемента модели оказывает влияние или приводит к изменению другого элемента модели. Отношение зависимости на диаграмме компонентов изображается пунктирной линией со стрелкой, направленной от клиента или зависимого элемента к источнику или независимому элементу модели.

Зависимости могут отражать связи отдельных файлов программной сис-

темы на этапе компиляции и генерации объектного кода. В других случаях зависимость может указывать на наличие в независимом компоненте описаний классов, которые используются в зависимом компоненте для создания соответствующих объектов. Применительно к диаграмме компонентов зависимости могут связывать компоненты и импортируемые этим компонентом интерфейсы, а также различные виды компонентов между собой. В этом случае рисуют стрелку от компонента-клиента к импортируемому интерфейсу. Наличие такой стрелки означает, что компонент не реализует соответствующий интерфейс, а использует его в процессе своего выполнения. При этом на этой же диаграмме может присутствовать и другой компонент, который реализует этот интерфейс. **Отношение реализации** интерфейса обозначается на диаграмме компонентов обычной линией без стрелки.

2 Оформить отчет.

3 Осуществить защиту работы.

Контрольные вопросы

1 Дайте определение понятию «диаграмма компонентов» и опишите назначение диаграммы компонентов.

2 Дайте определение понятиям «компонент», «модуль». Графическое изображение компонента и модуля.

3 Какие стереотипы определены для компонентов?

4 Что представляет собой интерфейс на диаграмме компонентов? Назовите способы связи компонента и интерфейса.

5 Какие Вы знаете способы связи между компонентами? Приведите примеры.

20 Лабораторная работа № 20. Разработка технической документации на модуль интегрированной информационной системы предприятия

Порядок выполнения работы

1 Разработать документ «Описание программы» в соответствии с ГОСТ 19.402–2000.

2 Разработать руководство оператора в соответствии с ГОСТ 19.505–79.

3 Осуществить защиту работы.

Контрольные вопросы

1 Каково основное назначение ГОСТ 19.402–2000?

2 Какие основные разделы содержит ГОСТ 19.402–2000?

3 Каково основное назначение ГОСТ 19.505–79?

4 Какие основные разделы содержит ГОСТ 19.505–79?

Список литературы

1 **Варфоломеева, А. О.** Информационные системы предприятия : учебное пособие / А. О. Варфоломеева. – Москва : ИНФРА-М, 2016. – 283 с.

2 **Голицына**, **О. Л.** Информационные системы : учебное пособие / О. Л. Голицына. – Москва : ФОРУМ ; ИНФРА-М, 2014. – 448 с.

3 **Пирогов, В. Ю.** Информационные системы и базы данных: организация и проектирование: учебное пособие / В. Ю. Пирогов. – Санкт-Петербург : БХВ-Петербург, 2009. – 528 с.

4 Информационные системы и технологии управления : учебник для вузов / Под ред. Г. А. Титоренко. – 3-е изд., перераб. и доп. – Москва : ЮНИТИ, 2010. – 591 с.

Приложение А
(обязательное)

Варианты заданий для выполнения лабораторных работ

Вариант задания выбирается по сумме трех последних цифр зачетной книжки.

1 Автострахование.

2 Агентство по сдаче автомобилей в аренду.

3 Аренда коньков, роликов, велосипедов, лыж.

4 Аэропорт – пассажирское расписание и перевозки.

5 Банковская система вкладов (физических и юридических лиц) .

6 Банковская система кредитования (физических и юридических лиц).

7 Биллинг сотовой компании.

8 Ветеринарная лечебница.

9 Клуб обучения танцам.

10 Магазин косметики.

11 Машиностроительное предприятие: система по разработке и модификации изделий (ведение архива, стандартов и пр.).

12 Нефтеперерабатывающая компания.

13 Парикмахерская.

14 Поставка вин.

15 Приемная комиссия вуза.

16 Производство мебели (прием индивидуальных и типовых заказов и изготовление).

17 Рекламное агентство.

18 Риелторская компания: аренда, продажа первичного и вторичного жилья.

19 Санаторий.

20 Система управления проектом для IT-компании.

21 Складская логистика.

22 Спа-салон (услуги, обслуживающий персонал и пр.).

23 Страховая компания.

24 Такси.

25 Транспортная логистика.

26 Туристическое агентство (путешествия за рубеж).

27 Туристическое агентство (путешествия по России).

28 Учет оборудования на крупном промышленном предприятии.

29 Филармония.

30 Электронный проездной.