

**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
КЫРГЫЗСКОЙ РЕСПУБЛИКИ
КЫРГЫЗСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
имени И. Раззакова
Институт электроники и телекоммуникаций
Кафедра «Информационные системы и технологии в телекоммуникациях»**

**Методические указания к выполнению лабораторных работ по
дисциплине «Архитектура информационных систем»**

Бишкек 2020

Рассмотрено на заседании кафедры «ИСТТ» Протокол № 4 от 05.12.19	Одобрено Учебно-методическим советом ИЭТ Протокол №2 от «__»____ 2019
--	--

УДК 621. 386 11

Составитель: Каримова Г.Т.

Визуальное проектирование. Методические указания к выполнению лабораторных работ по дисциплине архитектура информационных систем /КГТУ им. И. Раззакова; сост.: Г.Т. Каримова -Б.: ИЦ «Техник», 2019. - 36 страниц.

Целью настоящих методических указаний является оказание помощи студентам в освоении технологии проектирования архитектуры информационных систем с позиции объектно-ориентированного проектирования на языке UML. Пособие соответствует курсу Визуальное проектирование для направления 710200 «Информационные системы и технологии».

Предназначены для студентов специальности 710200 «Информационные системы и технологии»

Ил.: 3. Рис.: 14.

Рецензент доцент

кафедры «ПОКС» Стамкулова Г.К.

ОБЩИЕ УКАЗАНИЯ
к выполнению лабораторных работ по курсу
«Архитектура информационных систем»

Порядок выполнения работ

1. Ознакомиться с теоретическим материалом;
2. Выбрать тему (см. Приложение):
3. Провести исследование предметной области;
4. Построить UML диаграммы;
5. Провести количественную оценку для сравнения диаграмм;
6. Выполнить задания;
7. Ответить на контрольные вопросы;
8. Оформить отчёт.

Отчёт оформляется по каждой лабораторной работе и состоит из следующих разделов:

1. Тема лабораторной работы
2. Цель работы
3. Описание задания результаты работы интерпретация результатов
4. Ответ на контрольные вопросы

ЛАБОРАТОРНАЯ РАБОТА №1

ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ

Цель работы:

- изучение структуры, цели и задачи активируемой информационной системы, т.е. постановка задачи, так как процесс создания программ начинается с этапа изучения предметной области.

При автоматизации деятельности любого предприятия одним из первых и наиболее важных шагов является анализ этой деятельности. К под автоматизацией здесь понимается либо разработка корпоративной информационной системы, либо выбор таковой на рынке, её адаптация под специфику предприятия и последующее внедрение. Вы упомянутый выше анализ, в частности, входят:

- описание бизнес-процессов, происходящих на предприятии;
- выделения бизнес-процессов или их участков, подлежащих автоматизации.

Не сделав корректного описания бизнес-процессов, без смысла переходить к следующим стадиям анализа деятельности предприятия и тем более к его автоматизации.

Современное проектирование сложных информационных систем используют новые информационные технологии и программные средства поддержки системного инжиниринга – CASE технологии и средства.

В основе CASE технологий лежат соответствующие методы и методики, описывающие различные свойства систем, важное, например, с точки зрения их автоматизации, а также позволяющие количественно оценить параметры проектов. UML-диаграммы разрабатываются на основе высказываний, содержащихся в концептуальной схеме и информационной базе. Поэтому необходимо сформулировать высказывания концептуальной схемы, отразить в текстовой форме, а затем UML-диаграммы. Рекомендуется разработать как минимум диаграммы двух типов: диаграмма классов (структурная модель) и диаграмма последовательности (аспект поведения). Количественная оценка диаграмм.

Диаграмма классов должна содержать классы минимумом атрибутов. Диаграммы последовательности должны моделировать перечень ключевых функций системы (т.е. ответственностей), которые суть действия, замещающие в предметной области ручные действия, то есть автоматизирующие труд пользователей, а в терминах предыдущего раздела прикладные функциональные требования.

1.1. Описание системы

Данный курс практических работ ориентирован на изучение языка UML на примере диаграмм, создаваемых для проекта "Служба занятости в рамках ВУЗа" концептуальной основной объект на-ориентированного анализа и проектирования ПО(ООАП) является объектная модель. Большинство современных методов основан на использовании языка UML. Унифицированный язык моделирования UML (Unified Modeling Language) предоставляет собой язык для определения, представления, проектирования и документирования программных

систем, организационно-экономических систем, технических систем и других систем различной природы. UML содержит стандартный набор диаграмм и нотаций самых разнообразных видов.

Большинство существующих CASE - средств основано на методах структурного или объектно-ориентированного анализа и проектирования, использующих спецификации в виде диаграмм или текстовых для описания внешних требований, связей между моделями системы, динамики поведения системы и архитектуры программных средств.

Система "Служба занятости в рамках ВУЗа" предназначена для того, чтобы помочь студенту устроиться на работу уже в процессе его обучения в ВУЗе. Заявление представляет собой анкету. Система предлагает профессиональные (основные на изучаемых предметах), психологические тестирования, проводимые регулярно (раз в семестр (полгода)).

Особое внимание уделяется обучению студента, и по итогам успеваемости составляются экспертные оценки. На основе собранной информации составляется резюме, представляющие собой полную характеристику человека. Это резюме отсылается всем организациям, имеющим необходимые вакансии.

Основным назначением системы является автоматизация ввода и хранения отчетных данных по студентам, составления характеристик и резюме, поиск вакансий в фирмах. Система позволяет изменять, дополнять, вести поиск и просмотр информации о студентах, накладывать ограничения доступа к системе, хранить списки студентов, закончивших обучение, в виде архива, контролировать выдачу студенту заданий на курсовые работы и проекты, связывать ВУЗ с фирмами, заинтересованными в поиске сотрудников.

Также данная система может быть использована для составления отдельных списков групп, для печати зачетных ведомостей, для печати полной базы данных и для статистики.

Система состоит из четырех подсистем:

1. контроля успеваемости студентов;
2. профессиональных и психологических тестов;
3. обработки запросов, определения категорий полномочий пользователей;
4. экспертных оценок.

1.1.1. Подсистема контроля успеваемости студентов

Эта система является частью системы "Служба занятости в рамках ВУЗа", которая отвечает за статистическую отчетность по успеваемости отдельного студента, группы или всего факультета, а также за хранение и правильность её ввода.

Входными данными под темой являются: оценки, даты сдачи экзаменов, имена студентов, номера групп, факультет. На выходе под система выдает обработанные данные: средний балл по студенту, группе или факультету, процентное соотношение оценок у студентов в группе или на факультете, имена и количество стипендиатов в группе или на факультете.

Подсистема "Контроля успеваемости студентов" может функционировать отдельно от всей системы, что дает возможность установить и использовать её независимо, если это необходимо.

Подсистема "Контроль успеваемости студентов" включает следующие функции:

- ввод, вывод и редактирование информации по информационным объектам подсистемы;
- сохранение информации, поступившей от подсистемы контроля успеваемости студентов;
- расчет процентного соотношения оценок у студента в группе или на факультете и вывод его в виде таблиц, графиков и диаграмм;
- расчет среднего балла по студенту, группе или факультету;
- выявления сильнейших из слабейших студентов в группе или на факультете;
- расчет количества стипендиатов в группе или на факультете;
- проверку правильности ввода данных.

1.1.2. Подсистема профессиональных и психологических тестов пользователей

Работа модулей под системой

Монитор-вызывает модуль ввода и получает от него данные для выполнения дальнейших задач; обращается к модулю тестирования, передавая ему исходные данные; после выполнения своей задачи тестирование передаёт ему свои результаты; вызывает модуль ввода и получает от него конечный результат.

Ввод- получает запрос от монитора, передаёт ему запрашиваемые данные.

Вывод - вызывается монитором и получает от него результаты, выводя их на дисплей.

Тестирование-вызывается монитором, получает от него исходные данные и зависимости от них обращается к тому или иному модулю. Выполнив свои функции, модули посылают тестированию результаты действий, а тестирование уже передает полученные результаты монитору.

Проверка-получает исходные данные от тестирования и передаёт их своим подмодулям: регистрация тестируемых, определение тестов. Подмодули передают полученные результаты модулю проверки. Тестирование (профессиональное, психологическое, специальные) получает исходные данные от модуля тестирования не передаёт их своим подмодулям: профессиональное, психологическое, специальное тестирование. Подмодули передают полученные результаты.

Обработка-получает данные от тестирования, выполняет свои задачи и возвращает полученные результаты модулю тестирования.

Просмотр-получает данные от модуля тестирования, выполняет свои задачи и посылает полученные результаты модулю тестирования.

1.1.3. Подсистема обработки запросов, определение категории пользователей

Данная подсистема предназначена для определения категории, полномочий и обработки запросов пользователей службы занятости. В частности, она выполняет следующие функции:

- регистрации новых фирм;
- регистрацию новых студентов;
- определение прав доступа зарегистрированного пользователя;
- обработку запросов-приём регистрационных данных от фирм;
- приём регистрационных данных от студентов;
- приём регистрационных данных от обслуживающего персонала;
- составление резюме;
- запись данных в БД студентов;
- запись данных в БД фирм;
- запись данных в БД зарегистрированных пользователей.

Работа модулей подсистемы

Определение категории-модуль, определяющий категорию пользователя.

Определение полномочий-модуль, определяющие полномочия пользователя.

Выполнение запроса-модуль, я предназначенный для выполнения запросов пользователя.

Запись в БД зарегистрированных пользователей- модуль, предназначенный для работы с базой данных зарегистрированных пользователей.

Запись в БД студентов-модуль, предназначенный для работы с БД студентов.

Запись в БД фирмы-модуль, предназначенный для работы с БД фирм.

1.1.4. Подсистема экспертных оценок

Эта система предназначена для установки и просмотра экспертной оценки. Она дает краткую информацию преподавателю о студенте или группе. Студент может с помощью её но ориентироваться учёбе.

Кроме того, подсистема может дать представителю фирмы некоторое представление о студенте для рассмотрения его в качестве новых кадров.

Подсистема является как информационной средой, так и средой установки экспертной оценки.

1.2. Цели и задачи системы

Система будет обеспечивать хранение, выдачу и обновление информации системы дистанционного обучения студентов и системы "Служба занятости в рамках ВУЗа", а именно но:

- обеспечивать информационную поддержку системы дистанционного обучения студентов;
- представлять и получать накопленную информацию по конкретным объектам;
- представлять и получать информацию от подсистемы обработки запросов пользователей;
- представлять и получать информацию от подсистемы контроля студентов по изучаемым на кафедре дисциплинам;
- обеспечивать разграничение доступа к информационным ресурсам системы;
- обеспечивать мониторинг активных и пассивных пользователей и системных событий;
- обеспечивать пользователей возможностью информационного обмена;
- обеспечивать связь между фирмами и службой (ВУЗом);
- обеспечивать регулярное прохождение студентами профессиональных и психологических тестов;
- обеспечивать поиск данных по запросам.

Определение статуса клиента

- Клиент входит в подсистему идентифицированным номером. По номеру присваивается уровень доступа. Подсистема выдаёт меню работы соответствующее уровню доступа.

Обработка отчётного задания

- Эксперт: посылает запрос на предоставление информации о студенте (группе), результатах тестирования. Из общей БД поступает экспертная оценка.

- Студент: посылает запрос на предоставление информации об успеваемости, результаты тестирования. Делает запрос на просмотр экспертной оценки, которая поступает из общей БД.

- Представитель фирмы: делает запрос на просмотр экспертной оценки, которая поступает из общей БД.

Свертка: модуль, получающий из временной БД экспертные оценки , проводит их обработку и передает обработанные данные в общую БД.

Общая база данных содержит в себе:

- экспертные оценки (предоставляется под системой экспертных оценок),
- успеваемость студента (предоставляется подсистемой контроля студенческой успеваемости),
- успеваемость группы (предоставляются подсистемой контроля успеваемости),
- результаты профессионального теста (предоставляются подсистемой профессиональных психологических тестов),
- результаты психологического теста (предоставляются под системой профессиональных и психологических тестов).

Категории пользователей

При работе с системой на стадиях заполнения эксплуатации БД необходимо участие следующих категорий пользователей:

- администратора БД,
- группы экспертов.

Администратор системы осуществляет заполнение БД информацией, подготовленной учебной частью, деканатом или группа экспертов. Внесение изменений в БД системы осуществляется лишь администратором системы под руководством группы экспертов. Преподаватели и студенты являются внешними пользователями, работающими с системой в соответствии с ролями доступа в информационно-поисковом режиме.

Предоставляемые возможности пользователям системы:

студенту:

- ввод личных анкетных данных;
- просмотр экспертных оценок почетным заданиям и результатам тестов;
- прохождение психологических и профессиональных тестов;
- получение и сдача контрольных заданий;
- доступ к справочным материалам (данные из службы удаленного обучения, а именно методическое обеспечение);
- просмотр сообщений и внесении изменений в сообщения доски объявлений;
- поиск вакансии в БД по запросу;

эксперту преподавателю:

- предоставление экспертные оценки, а также изменение её;
- просмотр других оценок;
- просмотр программы курса и внесении изменений в неё;
- ввод контрольных заданий и назначение их студенту;
- контроль ответов на задания;
- доступ к интеллектуальным ресурсам;
- составление резюме (характеристик);

деканату (декану, зам. декана и т.д.):

- просмотр программы курса;
- просмотр динамики успеваемости курса, группы, отдельного студента;
- просмотр сводных таблиц и графиков;

-просмотр экспертных оценок и характеристик преподавателей;

администратору:

-определение прав доступа;

-ввод и корректировка системных данных;

-контроль работы системы;

-осуществление контроля защиты системы от несанкционированного доступа;

-изменение физической модели данных системы;

оператору:

-составление сводных таблиц и графиков;

-заполнение полей БД системы (ввод информации);

фирме:

-поиск в БД данных о студенте по запросу;

-просмотр резюме студентов.

Задания

1. Ознакомьтесь с теоретическим материалом
2. Выберите тему (см. Приложение)
3. Привести текстовое описание предметной области (бизнес-процесс)

Контрольные вопросы

1. Укажите цели системы.
2. Изобразите модульную структуру всей системы.
3. Перечислите основные информационные объекты системы служба занятости.
4. Опишите порядок занесения студентов БД.
5. Перечислите функциональные характеристики системы.
6. Какие данные поступают на вход подсистемы контроля успеваемости студентов?
7. Назовите категории пользователей системы и предоставляемые им возможности.
8. Каково назначение подсистемы обработки запросов, определения категории пользователей?

ЛАБОРАТОРНАЯ РАБОТА № 2

**Унифицированный язык моделирования и методика количественной оценки
диаграмм**

Цель работы:

-изучение методики количественной оценки диаграмм

2.1. Назначение языка UML

Унифицированный язык моделирования UML предоставляет собой язык для определения, представления, проектирования и документирования программных систем, организационно-экономических систем, технических систем и других систем различной природы. UML содержит стандартный набор диаграмм и нотации самых разнообразных видов.

Язык UML предназначен для решения следующих задач:

1. Предоставить в распоряжении пользователей легко воспринимаемый и выразительный язык визуального моделирования, специально предназначенные для разработки и документирования моделей сложных систем самого различного целевого назначения.
2. Снабдить исходные понятия языка UML возможностью расширения и специализации для более точного предоставления моделей систем в конкретной предметной области.

Хотя язык UML является формальным языком-спецификацией, формальность его описания отличается от синтаксиса как традиционных формально-логических языков, так и известных языков программирования. Язык UML как никакой другой может быть приспособлен для конкретных предметных областей. Это становится возможным по той причине, что в самом описании языка о NL заложен механизм разрешения базовых понятий, который является самостоятельным элементом языка и имеет собственное описание в форме правил разрешения.

В то же время крайне нежелательно переопределение базовых понятий языка по какой бы то ни было причине. Это может привести к неоднозначной интерпретации их семантики и возможной путанице.

1. Описание языка UML должно поддерживать такую спецификацию моделей, которая не зависит от конкретных языков программирования и инструментальных средств проектирования программных систем.
2. Описание языка UML должно включать в себя, семантический базис для понимания общих особенностей ООАП.
3. Площадь развитие рынка объектных инструментальных средств.
4. Интегрировать в себя новейшие и наилучшие достижения практике ООАП.

В структуре универсального языка моделирования выделяются две основные составляющие:

- Метамодель
- Правила построения диаграмм.

Основной интерес для проектировщика предоставляют правила построения UML диаграмм, основными разновидностями которых являются диаграммы;

- Прецедентов, или вариантов использования;
- Классов;
- Состояний;

- Активности, или деятельности;
- Взаимодействия, к которым относятся диаграммы последовательности и кооперация;
- Компонентов;
- Развертывания.

Именно диаграммы в нотации UML содержит удобным средством передачи информации об особенностях построения информационной системы между участниками проекта.

2.2. Количественная оценка диаграмм

Методика количественной оценки и сравнения диаграмм канал строится на присвоение элементом диаграмм оценок, зависящих от их информационной ценности, а также от вносимой ими в диаграмму дополнительной сложности. Ценность отдельных элементов меняется в зависимости от типа диаграмм, на которой они находятся.

Словарь языка UML включает два вида строительных блоков:

- сущности;
- отношения.

Сущности - это абстракции, являющиеся основными элементами модели. Отношения связывают различные сущности.

Количественную оценку диаграммы можно провести по следующей формуле:

$$S = \frac{\sum S_{об} + \sum S_{св}}{1 + Ч_{об} + \sqrt{T_{об} + T_{св}}} \quad (1)$$

где

S-оценка диаграммы;

S_{об}-оценка для элементов диаграммы;

S_{св}-оценки для связи на диаграмме;

Ч_{об}-число объектов на диаграмме;

T_{об}-число типов объектов на диаграмме;

T_{св}-число типов связей и на диаграмме;

Если диаграмма содержит большое число связей 1 типа например, модель БД, то число и тип связей можно не учитывать и формула расчёта приводится в виду:

$$S = \frac{\sum S_{об}}{1 + Ч_{об} + \sqrt{T_{об}}} \quad (2)$$

Если на диаграмме показаны атрибуты и операции классов, можно учесть их при расчёте, при этом оценка прибавляется к оценке соответствующего класса:

$$S = \frac{\sum On + \sum Amp}{0,3 * (On + Amp)} \quad (3)$$

где

$S_{кл}$ -оценка операции и атрибутов на диаграмме;

Оп-число операции у класса;

$A_{тр}$ -число атрибутов у класса;

При этом учитываются только атрибуты и операции, отображение на диаграмме.

Далее в табл. 2.1 и 2.2 приводятся оценки для различных типов элементов и связей.

Таблица 2.1

Основные элементы языка UML

Тип элемента	Оценка для элемента
Класс (class)	5
Интерфейс (interface)	4
Прецедент (use case)	2
Компонент (component)	4
Узел (node)	3
Процессор (processor)	2
Взаимодействие (interaction)	6
Пакет (package)	4
Состояние (state)	4
Примечание (note)	2

Таблица 2.2

Основные типы связей языка UML

Тип связи	Оценка для связи
Зависимость (dependency)	2
Ассоциация (association)	1
Агрегирование (aggregation)	2
Композиция (composition)	3
Обобщение (generalization)	3
Реализация (realization)	2

Остальные типы связей должны рассматриваться как ассоциации.

Недостатком диаграммы является как слишком низкая оценка (при этом диаграмма недостаточно информативна), так и слишком высокая оценка (при этом диаграмма обычно слишком сложна для понимания)

В табл. 2.3. приведены диапазоны оптимальных оценок для основных типов диаграмм.

Таблица 2.3

Диапазоны оценок для диаграммы UML

Тип диаграммы	Диапазон оценок
Классов (class) – с атрибутами и оценками	5 - 5,5
Классов (class) – без атрибутов и операций	3 - 3,5
Компонентов (component)	3,5 - 4
Вариантов использования (use case)	2,5 - 3
Развертывания (deployment)	2 - 2,5
Последовательности (sequences)	3 - 3,5
Кооперативная (cooperative)	3,5 - 4
Пакетов (package)	3,5 - 4
Состояний (state)	2,5 - 3

Приведем пример оценки простой под диаграммой, показанной на рис. 2.1 по данной методике.

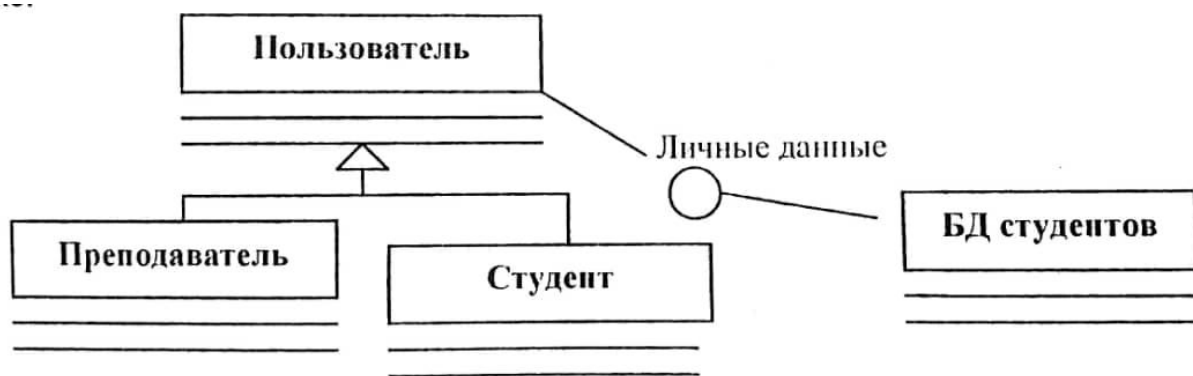


Рис. 2.1. Диаграмма

$$S = \frac{\sum S_{об} + \sum S_{св}}{1 + \sqrt{C_{об} + T_{св}}} = \frac{(20 + 4) + (3 + 2)}{1 + 5 + \sqrt{2 + 2}} = \frac{29}{8} = 3,63$$

Значение для диаграммы находится в допустимых пределах.

Задания

1. Построить две диаграммы классов, и указать для классов основные атрибуты и операции, указать тип связей.
2. Построить несколько диаграмм UML и провести количественную оценку.

Контрольные вопросы

1. Каково назначение языка UML?
2. Перечислите основные диаграммы UML?
3. Какие типы связей знаете?
4. Для чего используется методика количественной оценки диаграмм?

ЛАБОРАТОРНАЯ РАБОТА № 3

Создание диаграммы классов

Цель работы:

- изучение диаграммы классов,
- изучение их применение в процессе проектирования.

3.1. Диаграммы классов (class diagrams)

Диаграммы классов являются центральным звеном методологии объектно-ориентированных анализа и проектирования.

Диаграмма классов показывает классы и их отношения, тем самым представляя логических аспект проекта. Отдельная диаграмма классов представляет определенный ракурс структурой классов. На стадии анализа диаграммы классов используется, чтобы выделить общие роли и обязанности сущностей, обеспечивающих требуемое поведение структуры классов, я формирующих архитектуру системы.

Каждый класс должен иметь имя и имя каждого класса должно быть уникально в содержащим его проекте.

Диаграмма классов определяет типы объектов системы и различного рода статистические связи, которые существуют между ними. На диаграммах классов изображаются также атрибуты классов, операции и ограничения, которые накладываются на связи между объектами. Каждый класс на диаграмме выглядит в виде прямоугольника, разделенного на 3 части как показано на рис. 3.1. В первой содержится имя класса, во второй-его атрибуты. В последней части содержатся операции класса, отражающие его поведение (действия, выполняемые классом).

Атрибут. Атрибуты- это элемент информации, связанной с классом.

Так как атрибуты содержатся внутри класса, они скрыты от других классов. В связи с этим может понадобиться указать, какие классы имеют право читать и изменять атрибуты. Это свойство называется видимостью атрибута (attribute visibility). У атрибута можно определить 4 возможных значения этого параметра:

- *Public (общий, открытый)*. Это значение видимости предполагает, я что атрибут будет виден всеми остальными классами. Любой класс может просмотреть или изменить значение атрибута в соответствии с нотацией UML общему атрибуту предшествует знак "+".

- *Private (закрытый, секретный)*. Соответствующий атрибут не виден никаким другим классам. Чтобы его посмотреть необходимо попросить разрешения у класса, которому принадлежит атрибут. Закрытой атрибут обозначается знаком «-».

- *Protected (защищённый)*. Такой атрибут доступен только самому классу су и его потока. Нотация UML для защищённого атрибута-это знак "#".

- *Package or Implementation (пакетный)*. Предполагает, что данный атрибут является общим, но только в пределах его пакета. Этот тип видимости не обозначается никакими специальным значком.

В общем случае, атрибуты рекомендуется делать закрытыми или защищенными. Это позволяет лучше контролировать сам атрибут и код. С помощью закрытости или защищённости удаётся избежать ситуации, когда значение атрибута изменяется всеми классами системы. Вместе этого логика изменения атрибута будет заключена в том же классе, что и сам этот атрибут. Задаваемые параметры видимости повлияют на генерируемый код.

Приведем пример графического изображения конкретного класса на рис.3.1.

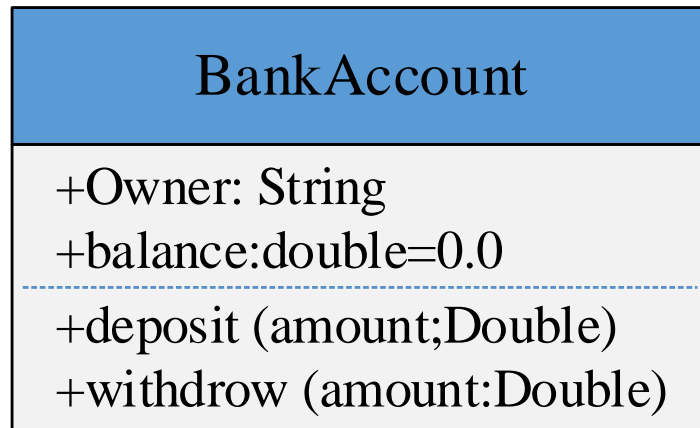


Рис.3.1. Графическое изображение класса «Банковский счет»

На примере банковской системы разработаем диаграмму классов, которая представлена в виде классов: Bank, Customer, BankAccount, ATM, ATM Transactions, CurrentAccount, SavingAccount и отношений между ними.

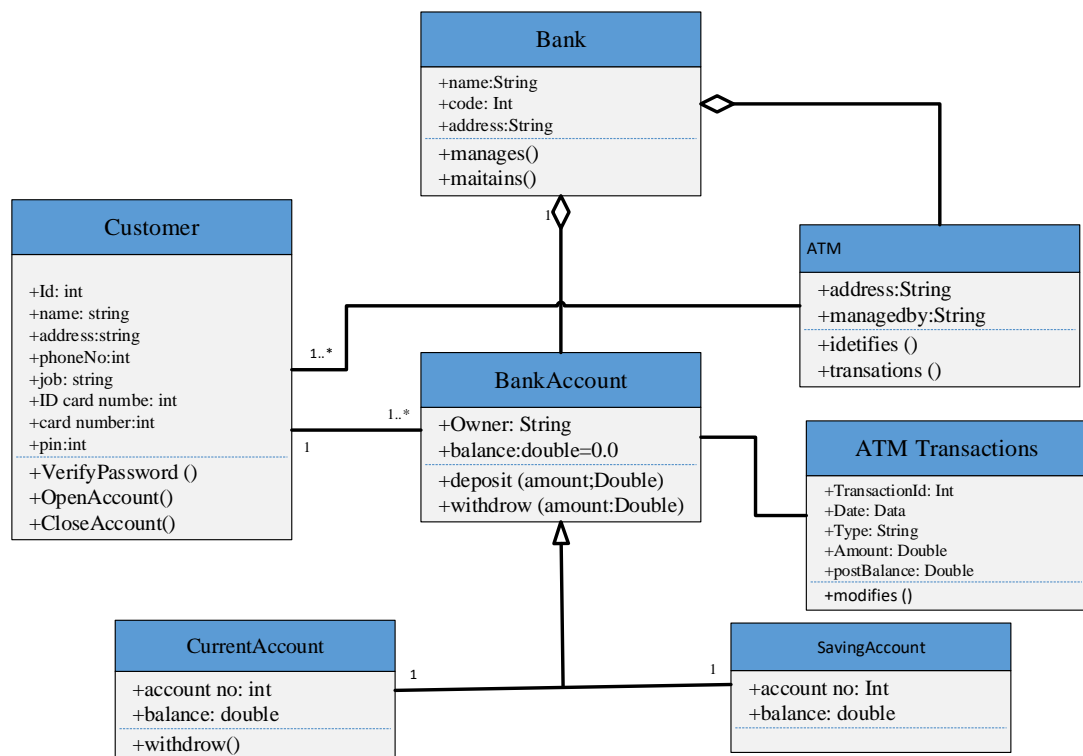


Рис.3.2. Графическое изображение диаграммы класса «Банковская система»

Операции. Операции реализуют связанные с классом поведение. Операция включает три части - имя, параметры и тип возвращаемого значения.

Параметры- это аргументы, получаемые операцией «на входе». Тип возвращаемого значения относится к результату действия операции. К на диаграмме классов можно показывать как имена операций, так и имена операции вместе с их параметрами и типом возвращаемого

значения. Чтобы уменьшить загроможденность диаграммы, полезно бывает на некоторых из них показывать только имена операций, а на других их полную сигнатуру.

Связи. Связь представляет собой семантическую взаимосвязь между классами. Она даёт классу возможность узнавать об атрибутах, операциях и связях другого класса. Иными словами, чтобы 1 класс мог послать сообщение другому на диаграмме последовательности или кооперативной диаграмме, между ними должно существовать связь.

Существуют четыре типа связей, которые могут быть установлены между классами: ассоциации, в зависимости, агрегации и обобщения.

Ассоциация (association) - это семантическая связь между классами. Их рисуют на диаграмме классов в виде обыкновенной линии. Ассоциации могут быть направленными или однонаправленными на языке UML направленные ассоциации рисуют в виде простой линии без стрелок или со стрелками с обеих её сторон. На однонаправленной ассоциации изображают только одну стрелку, показывающую её направление. Направлению ассоциации можно определить, изучая диаграммы последовательности и корпоративные диаграммы. Если все сообщения на них отправляются только одним классом и принимаются только другим классом, но не наоборот, между этими классами имеет место однонаправленная связь. Если хотя бы одно сообщение отправляется в обратную сторону, ассоциация должна быть двунаправленной.

Ассоциации могут быть рефлексам. Рефлексная ассоциация предполагает, что один экземпляр класса взаимодействует с другими экземплярами этого же класса.

Связи зависимости (dependency) также отражают связь между классами, но они всегда однонаправленные и показывают, что один класс зависит от определений, сделанных в другом. Зависимости изображают в виде стрелки и проведенной пунктирной линией.

При генерации кода для этих классов к ним не будут добавляться новые атрибуты. Однако, будут созданы спецификации для языка операторы, необходимые для поддержки связи. Например, на языке C++ войдут необходимые операторы `#include`.

Агрегации (aggregations) представляет собой более тесную форму ассоциации. Агрегация - это связь между целым и его частью. Агрегации визуализируют в виде линии с ромбиком у класса, являющегося целым.

В дополнение к простой агрегации UML вводит более сильную разновидность агрегации, называемую *композицией*. Согласно композиции, объект часть может принадлежать только единственному целому, и, кроме того, как правило, жизненный цикл частей совпадает с циклом целого: они живут и умирают вместе с ним. Любое удаление целого распространяется на его части.

С помощью *обобщений (generalization)* показывает связи наследования между двумя классами. Большинство объектно-ориентированных языков непосредственно поддерживает концепцию наследования. Она позволяет одному классу наследовать все атрибуты, операции и связи другого. На языке UML связи наследования называют обобщениями и изображают в виде стрелок от класса-потомка к классу-предку.

Помимо наследуемых, каждый подкласс имеет свои собственные уникальные атрибуты, операции и связи.

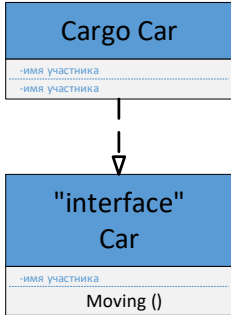
Множественность(multiplicity) показывает, сколько экземпляров одного класса взаимодействует с помощью этой связи с одним экземпляром другого класса в данный момент времени. Её индикаторы устанавливаются на обоих концах линий связи.

Связи можно уточнить с помощью *имен связей* или ролевых имен. Имя связи-это обычно глагол или глагольная фраза, описывающая, зачем она нужна .

Имена у связи определять не обязательно. Обычно это делают, если причина создания связи не очевидна. Имя показывают около линии соответствующие связи.

Таблица 1. Виды отношений и роль в диаграмме классов

Графическое изображение отношения	Название отношения	Описание	Пример использования
	Ассоциация	Ассоциация читается как «..имеет..».	<p>Ассоциация показывает, что Окно имеет курсор</p>
	Обобщения	Обобщения обычно читаются, как «..является..»	<p>Кошка характеризует базовый класс Животное</p>
	Агрегация	Ассоциация читается как «..обладает..».	<p>Окно «обладает» Прямоугольником</p>
	Композиция	Ассоциация читается как «..является частью..».	<p>Панель Названия «является частью» Окна</p>
	Навигации	Используется при перемещении из одного класса в другой	<p>Ассоциация м/у Window и Cursor, показывающая о невозможности навигации (X) от Cursor к Window</p>
	Зависимость	Зависимость читается как «..использует..»	

			Зависимость класса Window от WindowClosingEvent
----->	Детализация	Детализация читается как «...реализует...»	 <p>Класс <i>Грузовая машина</i> должен представить метод, реализующий операцию <i>Движение</i>, унаследованную от интерфейса <i>Машина</i></p>

Кратность – определяет, на сколько экземпляров может ссылаться данный атрибут. Имеются следующие виды кратности

- 1 (Заказ может представить только один клиент)
- 0..1 (Корпоративный клиент может иметь, а может не иметь единственного торгового представителя)
- * (Клиент не обязан размещать заказ, и количество заказов не ограничено. Он может разместить ноль или более заказов)

Имеет формат:

<нижняя граница>..<верхняя граница>

Верхняя и нижняя граница указывают минимальное и максимальное количество объектов, участвующих в ассоциации. Если для верхней границы стоит символ ‘*’, то это значит, что она бесконечна.

Ролевые имена применяют в связях ассоциации или агрегации вместо имен для описания того, зачем эти связи нужны. Ролевые имена-это обычно имена существительные или основанные на них фразы, их показывают на диаграмме рядом с классом, играющим соответствующую роль. Как правило, пользуются или ролевым именем, я или именем связи, но не обоими сразу. Как и имена связей, проливы и имена не обязательны, их дают, только если цель связи не очевидна.

3.2. Количественная оценка

На рис. 3.1 и 3.2 приведены диаграммы классов модели под системы «Служба занятости в рамках ВУЗа» системы «Дистанционное обучение».

Эти диаграммы реализуют один и тот же фрагмент под системы «Служба занятости в рамках ВУЗа», но первая из них более полно реализует принципы объектно-ориентированного подхода.

Найдём численную оценку для каждой из диаграмм.

Диаграмма 1

Проведем расчет оценки атрибутов и операции для классов «Работодатель», «БД студентов» и «Студент».

«Работодатель»:

$$S = \frac{\sqrt{On} + \sqrt{Amp}}{0,3 * (On + Amp)} = \frac{\sqrt{2} + \sqrt{0}}{0,3 * (2 + 0)} = 2,36$$

«БД студентов»:

$$S = \frac{\sqrt{On} + \sqrt{Amp}}{0,3 * (On + Amp)} = \frac{\sqrt{2} + \sqrt{4}}{0,3 * (2 + 4)} = 1,89$$

«Студент»:

$$S = \frac{\sqrt{On} + \sqrt{Amp}}{0,3 * (On + Amp)} = \frac{\sqrt{0} + \sqrt{1}}{0,3 * (0 + 1)} = 3,33$$

Рассчитаем полное значение для диаграмм:

$$S = \frac{\sum S_{об} + \sum S_{св}}{1 + \sqrt{1 + 4}} = \frac{(30 + 2,36 + 1,89 + 3,33) + (1 + 1 + 2 + 3 + 2)}{1 + 6 + \sqrt{1 + 4}} = \frac{37,58 + 9}{9,24} = 5,1$$

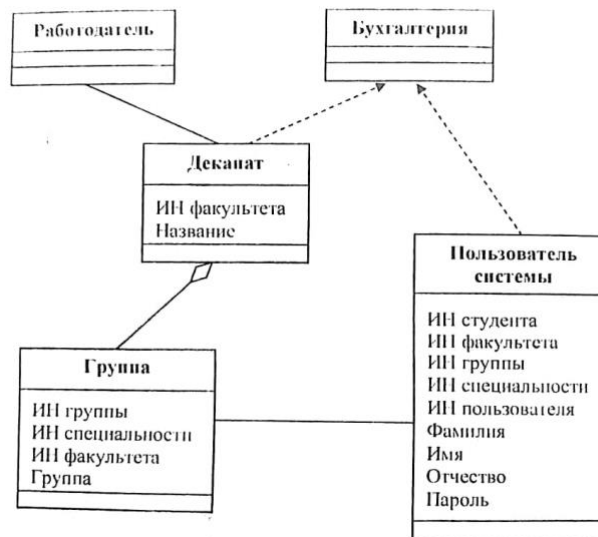


Рис. 3.2. Диаграмма 2

Диаграмма 2

Проведем расчет оценки атрибутов и операций для классов «Деканат», «Группа» и «Пользователь системы».

Для класса «Деканат» получаем:

$$S = \frac{\sqrt{On} + \sqrt{Amp}}{0,3 * (On + Amp)} = \frac{\sqrt{0} + \sqrt{2}}{0,3 * (0 + 2)} = 2,36$$

Для класса «Группа»:

$$S = \frac{\sqrt{On} + \sqrt{Amp}}{0,3 * (On + Amp)} = \frac{\sqrt{0} + \sqrt{4}}{0,3 * (0 + 4)} = 1,66$$

Для класса «Пользователь системы»:

$$S = \frac{\sqrt{On} + \sqrt{Amp}}{0,3 * (On + Amp)} = \frac{\sqrt{0} + \sqrt{9}}{0,3 * (0 + 9)} = 1,11$$

Рассчитаем полное значение для диаграммы:

$$S = \frac{\sum S_{об} + \sum S_{св}}{1 + \sqrt{Ch_{об} + T_{св}}} = \frac{(25 + 2,36 + 1,66 + 1,11) + (1 + 1 + 2 + 2)}{1 + 5 + \sqrt{1 + 3}} = \frac{30,13}{8} \approx 4,77$$

В результате оценка для диаграммы 1 попадает в середину оптимального диапазона для диаграмм классов, оценка для диаграммы 2 оказывается ниже оптимального диапазона.

Такой результат можно объяснить следующими причинами:

1. Диаграмма 2 содержит излишне детализированный класс пользователь системы, тогда как в диаграмме 1 он упрощенка с помощью построения иерархии классов .
2. Класс деканат на диаграмме 2 берёт на себя слишком много функций, следствием чего является избыток связи.
3. Класс бухгалтерия на диаграмме 2 не относится напрямую к фрагменту, смоделированному на диаграмме, т.е. усложняет модель , не внося при этом полезной информации .

Задания

1. Выделите основные классы объектов в проектируемой системе.
 2. Построить диаграмму классов, в общем виде демонстрирующую архитектуру системы.
 3. Построить одну-две диаграммы классов, где-то лидирующие отдельные подсистемы.
- Указать для классов основные атрибуты и операции, указать вид и направление ассоциаций.

Контрольные вопросы

1. Каково назначение диаграмм классов?
2. Для чего используется диаграмма классов на стадии анализа?
3. Для чего используется диаграмма классов на стадии проектирования?
4. Назовите основные компоненты диаграммы классов.
5. Назовите основные типы статистических связей между классами.
6. Что представляет собой ассоциация?
7. В чём смысл множественности ассоциаций?
8. В чём отличие атрибутов от ассоциаций?

9. Что такое признак видимости?
10. Что представляет собой операции класса?
11. В чём смысл обобщение?
12. Каково назначение ограничений на диаграммах классов?

ЛАБОРАТОРНАЯ РАБОТА №4

Создание диаграммы вариантов использования

Цель работы:

- изучение диаграммы вариантов использования,
- изучение и их применение в процессе постановки задачи.

4.1. Диаграммы вариантов использования (use-case diagrams)

Диаграммы вариантов использования (use-case), иногда называемые диаграммами прецедентов это одна из моделей формализации процесса постановки целей и задач проекта. Вариант использования представляет собой типичная взаимодействие пользователя и проектируемой системы. Варианты использования характеризуется рядом свойств:

- вариант использования охватывает некоторую очевидную для пользователей функции;
- вариант использования может быть как небольшим, так и достаточно крупным;
- вариант использования решают некоторую дискретную задачу пользователя;

В простейшем случае варианты использования создаётся в процессе обсуждения с пользователями тех вещей, которые они хотели бы получить от системы. При этом каждой отдельной функции, которую они хотели бы реализовать, присваивается некоторые имя и записывается её краткое текстовое описание.

Это всё, что необходимо в фазе анализа. Знание некоторых деталей может потребоваться, если предполагается, что данный вариант использования содержит важные архитектурные ответвления. Большинство вариантов использования может быть детализировано вовремя конкретный итерации в процессе проектирования.

Основными элементами диаграммы вариантов использования является действующие лица, варианты использования и отношения между ними.

Действующее лицо-это роль, которую пользователь играет по отношению к системе.

На рис. 4.1 присутствует 3 действующих лица: «Работодатель», «БД студентов» и «Студент». Говоря о действующих лицах важно видеть в них роли, они конкретных людей или наименование работ. Действующие лица вовсе не обязаны быть людьми, несмотря на то что на

диаграммах вариантов использования они изображаются в виде стилизованных человеческих фигурок. действующее лицо может также быть внешней системой, которая необходима некоторая информация от нашей системы. На диаграмме присутствуют два варианта использования: «Изменить личную информацию» и «Запросить данные студента». Все варианты использования так или иначе связаны с внешними требованиями к функциональности системы. варианты использования всегда следует анализировать вместе с действующими лицами системы, определяя при этом реальные задачи пользователей и рассматривая альтернативные способы решения этих задач.

Действующие лица могут играть различные роли по отношению к вариантам использования. Они могут применять его результаты или сами непосредственно в нём участвовать.

Хорошим источником для идентификации вариантов использования служат внешние события. Для этого необходимо перечислить все происходящее во внешнем мире события, на которые система должна реагировать.

Какое-либо конкретное событие может повлечь за собой реакцию системы, не требующую вмешательства пользователей, или, наоборот, вызвать чисто пользовательскую реакцию. Идентификация событий, на которые необходимо реагировать, поможет идентифицировать варианты использования.

4.2.Пример описания диаграммы вариантов использования для ПО.



Рис 4.1 Концептуальная модель для модели банкомата

Таблица 1. Сценарий вариант использования снятие наличных по кредитной карточке

Вариант использования	Снятие наличных по кредитной карточке
Актеры	Клиент, Банк
Цель	Получение денег наличной суммой
Краткое описание	Клиент запрашивает требуемую сумму, банкомат обеспечивает доступ к счету клиента, банкомат выдает клиенту наличные
Тип	Базовый
Ссылки на другие варианты использования	Включает в себя проверка пин-кода КК идентифицировать КК

Таблица 2. Типичный ход событий сценарий выполнения “ВИ”, снятие наличных по КК

Действие клиента		Ответ системы	
1.	Клиент вставляет кредитную карточку в устройство чтения банкомата	2.	Банкомат проверяет кредитную карточку
		2.1.	Кредитная карта недействительна
4.	Клиент выбирает язык	3,	Банкомат предлагает выбрать языковые параметры: английский, русский, кыргызский языки.

6.	Клиент вводит пин-код	5,	Банкомат предлагает ввести пин-код
7.1.	Клиент вводит неправильный пин-код	7.	Банкомат обрабатывает запрос
9.	Клиент выбирает необходимую ему опцию	8.	Банкомат отображает меню с опциями: 1. Снятие наличных; 2. Запрос баланса;
9.1.	Клиент вводит снятие наличных со своего счета	10.	Банкомат предлагает выбрать денежную единицу в \$, €, сомах
11.	Клиент выбирает единицу	12.	Система банкомата отправляет запрос в банк и выясняет текущее состояние счета клиента
14.	Клиент выбирает/вводит требуемую сумму	13.	Банкомат предлагает снять наличные купюрами в размере 200, 500, 1000, 2000, 5000, 10000 или другая сумма
14.1.	Требуемая сумма превышает лимит на банковской карте клиента	15.	Банк проверяет запрос
16.	Клиент получает денежную сумму	17.	Банкомат предлагает «Продолжить» работу с картой
18.	Клиент отклоняет запрос/подтверждает	19.	Банкомат выдает чек
21.	Клиент получает свою кредитную карту	20.	Банкомат завершает обработку карты и отдает ее клиенту

Таблица 3. Исключение сценария выполнения, ВИ, снятие наличных по КК

Действия актеров	Отклик системы
Исключение 1. Кредитная карточка недействительна	Банкомат отображает информацию о неверно вставленной информации
	Банкомат возвращает клиенту его КК
Клиент получает свою КК	
Исключение 2. Клиент вводит неверный пин-код	Банкомат отображает информацию о неверном пин-коде
Клиент вводит новый пин-код	
Исключение 3. Требуемая сумма превышает сумму на счете клиента	Банкомат отображает информацию о превышении кредита
Клиент вводит новую требуемую сумму	

Дополнить данный сценарий, проверить баланс, вводишь пин-код – нажимаешь на enter, у видишь 3 вида денежных единиц,

4.3. Количественная оценка диаграмм

На рис. 4.1 и 4.2 приведены две диаграммы вариантов использования, описывающие одну из функций, подсистема «Служба занятости в рамках ВУЗа» системы «Дистанционное обучение». Отличия этих диаграмм в том, что первое из них более подробно описывает процесс взаимодействия пользователя и системы.

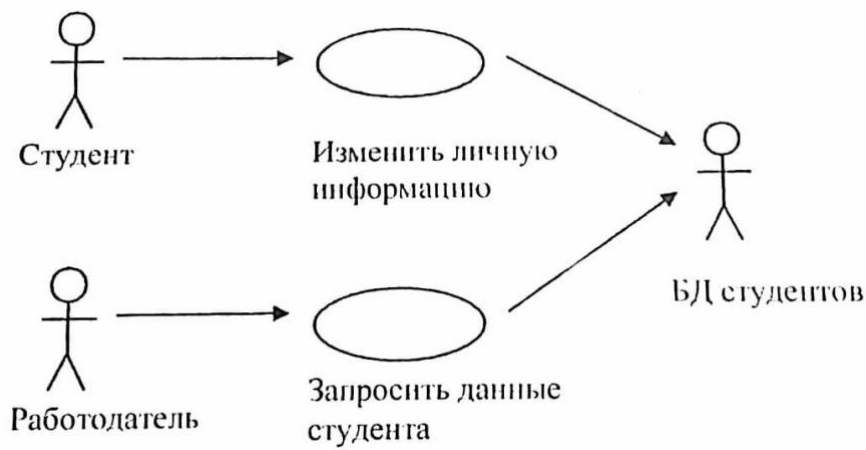


Рис. 4.2. Диаграмма 1

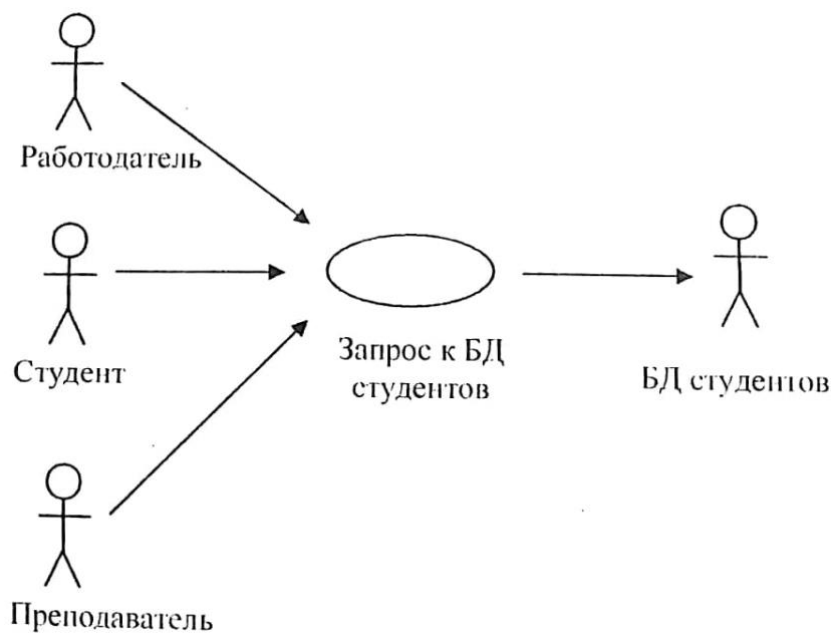


Рис. 4.3. Диаграмма 2

Найдём численную оценку для каждой из диаграмм.

Диаграмма 1

$$S = \frac{\sum S_{об} + \sum S_{св}}{1 + \chi_{об} + \sqrt{T_{об} + T_{св}}} = \frac{17 + 4}{1 + 5 + \sqrt{3}} \approx 2,72$$

Диаграмма 2

$$S = \frac{\sum S_{об} + \sum S_{св}}{1 + \chi_{об} + \sqrt{T_{об} + T_{св}}} = \frac{21 + 4}{1 + 5 + \sqrt{3}} \approx 3,23$$

1. Определить основные функции системы.
2. Для нескольких функций выделить действующие лица, участвующие в них или использующие их результаты.
3. Для каждой выбранной функции постройте диаграмму вариантов использования.

Контрольные вопросы

1. В чём смысл варианта использования?
2. Каково назначение диаграмм вариантов использования?
3. Назовите основные свойства вариантов использования.
4. Назовите основные компоненты диаграммы вариантов использования.
5. Что такое действующее лицо?
6. Какую роль могут играть действующие лица по отношению к варианту использования?
7. Каким образом анализ внешних событий позволяет определить варианты использования системы?

ЛАБОРАТОРНАЯ РАБОТА № 5

Диаграммы взаимодействия

Цель работы:

- изучение диаграммы взаимодействия,
- изучение их применения в процессе проектирования.

5.1. Диаграммы взаимодействия (interaction diagrams)

Диаграммы взаимодействия являются моделями, описывающими поведение взаимодействующих групп объектов.

Как правило, диаграмма взаимодействия охватывает поведение только одного варианта использования. На такой диаграмме отображаются ряд объектов и те сообщения, которыми они обмениваются между собой в рамках одного варианта использования.

Существуют два вида диаграмм взаимодействия: диаграммы последовательности (sequence diagrams) и кооперативные диаграммы (collaboration diagrams).

5.2. Диаграммы последовательности

На диаграмме последовательности объект изображается в виде прямоугольника на вершине пунктирной вертикальной линии (рис. 5.1, 5.2).

Эта вертикальная линия называется линией жизни (lifeline) объекта. Она представляет собой фрагмент жизненного цикла объекта в процессе взаимодействия.

Каждое сообщение представляется в виде стрелки между линиями жизни двух объектов. Сообщения появляются в том порядке, как они показано на диаграмме «сверху вниз». Каждое сообщение может быть помечена именем, при желании можно показать также аргументы и некоторую управляющую информацию. Также можно показать самоделегирование - сообщение, я которая объект посылает самому себе, при этом стрелка сообщения указывает на ту же самую линию жизни.

Изю всей возможной управляющей информации два её вида имеют существенное назначение. Во-первых, это условие, показывающие, в каком случае посылается сообщение. Сообщение посылается только при выполнении данного условия. Другой полезный управляющий маркер-это маркер итерации, показывающие что сообщение посылается много раз для множества объектов-адресатов (например, *обновить) .

Активация - прямоугольнике на линиях жизни-показывают, когда метод становится активным (во время его выполнения либо при ожидании результата выполнения какой-либо процедуры). Используя механизм активации, можно более четко показывать смысл самоделегирования. Без них довольно трудно определить, где же выполняются следующие после самоделегирования вызовы - вызывающие методе или в вызываем.

Активации вносят и ясность в этот вопрос.

5.3. Кооперативная диаграммы

Вторым видом диаграммой взаимодействия является кооперативная диаграмма (рис. 5.3, 5.4).

На кооперативной диаграмме экземпляры объектов показаны в виде диаграммы. Линия между ними обозначают сообщение, обмен которыми осуществляется в рамках данного варианта использования.

Каждый вид диаграмм взаимодействия имеет свои преимущества, выбор обычно осуществляется исходя из предпочтений разработчика. На диаграммах последовательностей делается акцент именно на последовательность сообщений, при этом легче наблюдать порядок, в котором происходят различные события. В случае корпоративных тигром можно использовать пространственное расположение объектов для того, чтобы показать их статистическое взаимодействие.

Одним из главных свойств любой диаграммы взаимодействия является её простота. Посмотрев на диаграмму, можно легко увидеть все сообщения. Однако при попытке изобразить нечто более сложное, чем единственный последовательный процесс без множества условных переходов или циклов, данный подход может не сработать.

Для отображения условного поведения на диаграммах взаимодействия существует два подхода. Один из них состоит в использовании отдельных диаграмм для каждого сценария. Второй заключается в том, что сообщения сопровождаются условиями, показывающими поведение объектов

5.4. Количественная оценка диаграмм

На рисунке 5.1 и 5.2 приведены диаграммы последовательности модели под системы «Служба занятости», показывающие взаимодействие двух классов модели: студент и студентов. На рисунке 5.3 и 5.4 тоже взаимодействие показано с помощью кооперативных диаграмм.

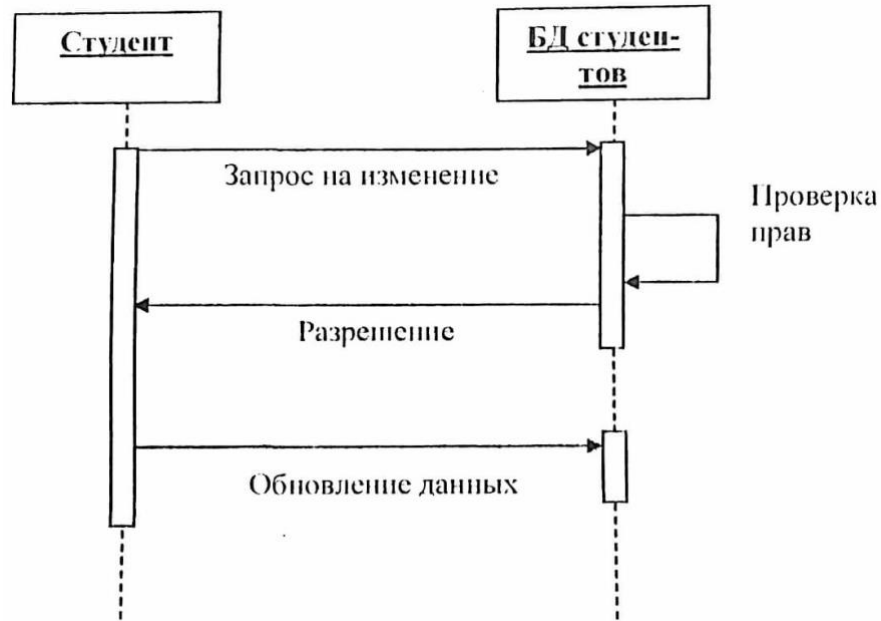


Рис. 5.1. Диаграмма 1

Найдём численную оценку для каждой из диаграмм.

Диаграмма 1

Так как на диаграмме последовательности связи отсутствуют, проведем расчет по сокращённой формуле:

$$S = \frac{\sum S_{об}}{1 + \mathcal{C}_{об} + \sqrt{T_{об}}} = \frac{34}{1 + 6 + \sqrt{2}} \approx 4,04$$

$$S = \frac{\sum S_{об} + \sum S_{св}}{1 + \mathcal{C}_{об} + \sqrt{T_{об} + T_{св}}} = \frac{22}{1 + 4 + \sqrt{2}} \approx 3,43$$

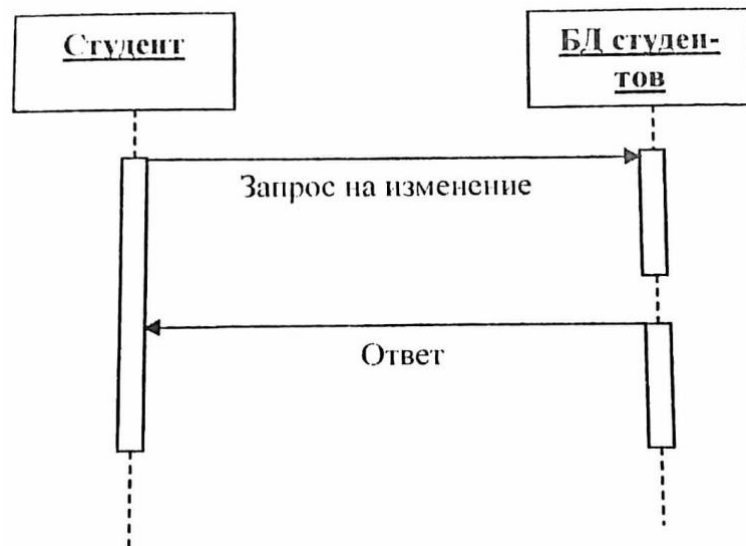


Рис. 5.2. Диаграмма 2

$$S = \frac{\sum S_{об} + \sum S_{св}}{1 + \sqrt{1 + T_{об} + T_{св}}} = \frac{15 + 3}{1 + 3 + \sqrt{1}} \approx 3,6$$

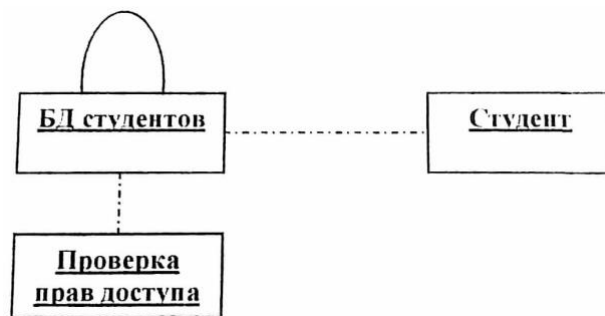


Рис. 5.3. Диаграмма 3

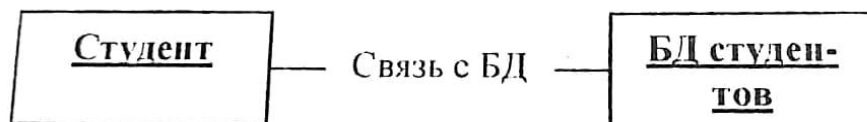


Рис. 5.4. Диаграмма 4

$$S = \frac{\sum S_{об} + \sum S_{св}}{1 + \sqrt{1 + T_{об} + T_{св}}} = \frac{10 + 1}{1 + 2 + 1} \approx 2,75$$

Задания

1. Выбрать в моделируемой системе варианты использования, для которого будут строиться диаграммы взаимодействия.
2. Постройте для выбранного варианта использования диаграммы последовательности

3. Построить для того же варианта использования кооперативную диаграмму.
4. Сформулировать достоинства и недостатки каждого вида диаграмм при моделировании данного варианта использования.

Контрольные вопросы

1. Каково назначение диаграмм взаимодействия?
2. Как относятся между собой диаграмма вариантов использования и диаграммы взаимодействия?
3. Назовите два вида диаграмм взаимодействия.
4. Что такое «Жизненная линия» на диаграмме последовательности?
5. Как на диаграмме последовательности предоставляются сообщения?
6. Что такое самоделегирование?
7. Что показывает активация объекта?
8. В чём отличие корпоративных диаграмм от диаграммы взаимодействия?
9. Каковы преимущества и недостатки каждого вида взаимодействия?
10. Как отображается условное поведение на диаграммах взаимодействия?

ЛАБОРАТОРНАЯ РАБОТА №6

Диаграммы состояний

Цель работы:

-изучение диаграммы состояния и изучение их применение в процессе проектирования.

6.1. Диаграммы состояний (state diagrams).

Диаграмма состояний является хорошо известным средством описания поведения систем. Они определяют все возможные состояния, в которых может находиться конкретный объект, а также процесс смены состояний объекта в результате влияния некоторых событий.

На диаграмме имеются два специальных состояний-начальная (start) и конечное(stop). Начальное состояние выделено чёрной точкой, оно соответствует состоянию объекта непосредственно перед его уничтожением. На диаграмме состояний может быть одно и только одно начальное состояние. В тоже время, может быть столько конечных состояний, сколько вам нужно, или их может не быть вообще. Когда объект находится в каком-то конкретном состоянии, могут выполняться различные процессы. В нашем примере при превышении кредита клиенту

посылается соответствующее сообщение. Процессы, происходящие, когда объект находится в определенном состоянии, называется действиями (actions). Состояние можно связывать в данные пяти типов: деятельность, входное действие, я событие и история события.

Деятельность (activity) называется поведение, реализуемое объектом, пока он находится в данном состоянии. Деятельность-это переживаемая поведение. Оно может выполняться до своего завершения, пока объект находится в данном состоянии, или может быть прервано на переходом объекта в другое состояние. Деятельность изображают внутри самого состояния, ей должно предшествовать словно *do*(делать) и двоеточие

Входным действием (entry action) называется поведением, которое выполняется, когда объект переходит в данное состояние. Таким образом, данное состояние осуществляется не после того , как объект перешёл в это состояние, а, скорее, как часть этого перехода. В отличие от деятельности, входное действие рассматривается как не прерываемая. Входное действие также показывают внутри состояния, ему предшествует слово *entry*(вход) и двоеточие.

Выходное действие (*exit action*) подобно входному. Однако, оно осуществляется как составная часть процесса выхода из данного состояния. Она является частью процесса такого перехода. Как и входное и выходное действие является непрерываемым.

Выходное действие изображают внутри состояния, ему предшествует слово *exit* (выход) и двоеточие.

Поведение объекта во время деятельности, при входных и выходных действиях может включать отправку события другому объекту. В этом случае описанию деятельности, входного действия или выходного действия предшествует знак « ^ ».

Диаграммы состояния хорошо использовать для описания поведения некоторого объекта в нескольких различных вариантах использования.

Они не слишком пригодны для описания поведения ряда взаимодействующих объектов.

Рекомендуется строить строить диаграммы состояний только для тех классов, поведение которых влияет на общее поведение системы, например для классов пользовательского интерфейса и управляющих объектов.

6.2. Количественная оценка диаграмм

На рис. 6.1 и 6.2. приведены диаграммы состояния экземпляра класса «Студент». Эти диаграммы показывают состояния экземпляра в ходе взаимодействия объекта класса «Студент» с БД студентов. Первая диаграмма расписывается состояния объекта подробно, а вторая показывает только общее состояние взаимодействия с БД.

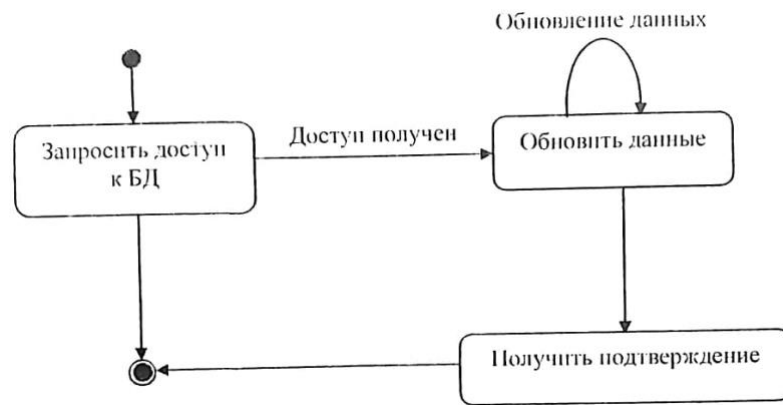


Рис.6.1 Диаграмма 1



Рис.6.2 Диаграмма 2

Найдём численную оценку для каждой из диаграммы.

Диаграмма 1

Так как на диаграмме состояния связи отсутствуют, проведем расчет по сокращённой формуле :

$$S = \frac{\sum S_{об}}{1 + \chi_{об} + \sqrt{T_{об}}} = \frac{20}{1 + 5 + 1} \approx 2,86$$

$$S = \frac{\sum S_{об}}{1 + \chi_{об} + \sqrt{T_{об}}} = \frac{12}{1 + 3 + 1} \approx 2,4$$

Полученный результат объясняется наличием недостаточно детализированного состояния на диаграмме 2.

Задания:

1. Выбрать моделируемой системе классы, для объектов которых будут строиться диаграммы состояний.

2. Построить для каждого выбранного класса диаграмму состояний, характеризующую поведение его объектов в нескольких вариантах использования.

Контрольные вопросы

1. Каково назначение диаграмм состояния?
2. Как отображаются действия и деятельности на диаграммах состояния?
3. Что такое условный переход и как он описывается на диаграмме?
4. Какие особые состояния объекта отображаются на диаграмме?
5. Каковы преимущества и недостатки диаграмм состояния?

ЛАБОРАТОРНАЯ РАБОТА № 7

Диаграмма пакетов, компонентов и размещения

Цель работы:

- изучение диаграммы пакетов: диаграммы компонентов и диаграммы размещения,
- изучение их применения в процессе проектирования.

7.1. Диаграммы пакетов (package diagrams)

Один из важнейших вопросов методологии создания программного обеспечения-как разбить большую систему на небольшие подсистемы? Именно с этой точки зрения изменения, связанные с переходом от структурного подхода к объектно-ориентированному, являются наиболее заметными. Одна из идей заключается группировке классов компоненты более высокого уровня. В UML такой механизм группировки носит название пакетов (package).

Диаграммы пакетов является диаграмма, содержащая пакеты классов и зависимости между ними. Строго говоря, пакеты и зависимости являются элементами диаграммы классов, т.е. диаграмма пакетов-это всего лишь форма диаграммы классов. Однако на практике причины построения таких диаграмм различны.

Зависимость между двумя элементами имеет место в том случае, если изменения в определении одного элемента могут повлечь за собой изменение в другом что касается классов, то причина зависимостей могут быть самыми разными: один класс посылает сообщение другому; один класс включает часть данных другого класса; один класс ссылается на другой как на параметр операции. Если класс меняет свой интерфейс, то любое сообщение, которое он посылает, может стать неправильным.

В идеальном случае только изменения в интерфейсе класса должна взаимодействовать на другие классы. Искусство проектирования больших систем включает в себя минимизацию зависимостей, я которая снижает взаимодействие изменений и требует меньше условий на их внесение.

Зависимость между двумя пакетами существует в том случае, если имеется какая-либо зависимость между любыми двумя классами в пакетах.

Пакеты являются жизненно необходимым средством для больших проектов. Их следует использовать в тех случаях, когда диаграмма классов, охватывающая всю систему в целом и размещенная на единственном листе бумаги формата А4, становится трудночитаемый.

Букеты не дают ответа на вопрос, каким образом может уменьшить количество зависимостей в разрабатываемой системе, однако они помогают выделить эти зависимости. Сведения к минимуму количества зависимостей позволяет снизить связанность компонентов системы. Но эвристический к этому процессу далек от идеала.

Пакеты особенно полезны при тестировании. Каждый пакет при тестировании может содержать один или несколько тестовых классов, с помощью которых проверяется поведение пакета.

7.2. Диаграммы компонентов (component diagrams)

Компоненты на диаграмме компонентов предоставляет собой физические модули программного кода. Обычно они в точности соответствуют пакетом на диаграмме пакетов; таким образом, диаграмма компонентов отражает выполнение каждого пакета в системе.

Зависимости между компонентами должна совпадать в зависимостями между пакетами. Эти зависимости показывают, каким образом одни компоненты взаимодействуют с другими. Направление данной зависимости показывает уровень осведомленности о коммуникации. Диаграммы компонентов показывают как выглядит модель на физическом уровне. На них изображены компоненты программного обеспечения и связи между ними. При этом на такой диаграмме выделяют два типа компонентов: исполняемые компоненты и библиотеки кода. Каждый класс модели (или подсистема) преобразуется в компонент исходного кода. После создания они сразу добавляются в диаграмме компонентов. Между отдельными компонентами изображают в зависимости, соответствующие зависимостям на этапе компиляции или выполнения диаграммы.

Диаграммы компонентов применяются теми участниками проекта, кто отвечает за компиляцию системы. Из неё видно, в каком порядке надо компилировать компоненты, а также какие исполняемые компоненты будут созданы системой. На такой диаграмме показано соответствие классов реализованным компонентом. Она нужна там, где начинается генерация кода.

7.3. Диаграмма размещения (deployment diagrams)

Диаграмма размещения отражают физические взаимосвязи между программами и аппаратными компонентами системы. Она является хорошим средством для того, чтобы показать маршруты перемещения объектов и компонентов в распределенной системе.

Каждый узел на диаграмме размещения представляет с собой некоторые тип вычислительного устройства-в большинстве случаев часть аппаратуры. Это аппаратура может быть простым устройством или датчиком, а может быть и большим компьютером. Подразделению администрирования сети необходимо знать, на каких компьютерах будут размещаться различные компоненты системы. по мере распространения распределённых систем важность данных диаграмм возрастает.

7.4. Количественная оценка диаграмм.

Проводить сравнение диаграмма компонентов, компонентов и размещения в общем случае бессмысленно, так как эти диаграммы не существуют сами по себе, а являются интерпретацией некоторой диаграммой класса, для которой и уместно проводить сравнение с другими диаграмме классов.

Диаграмма пакетов содержат один тип элементов-пакет и один тип связи-зависимость, поэтому численная оценка для диаграммы пакетов не столь важна, как для диаграммы классов.

На рисунке. 7.1 изображена диаграмма пакетов под системой «Служба занятости в рамках ВУЗа» системы «Дистанционное обучение».

Диаграмма компонентов и размещения строится используется на этапы реализации и сопровождения, когда базовая архитектура системы уже обычно определена, поэтому они однозначно получаются из диаграммы классов и для них достаточно привести по одному примеру.

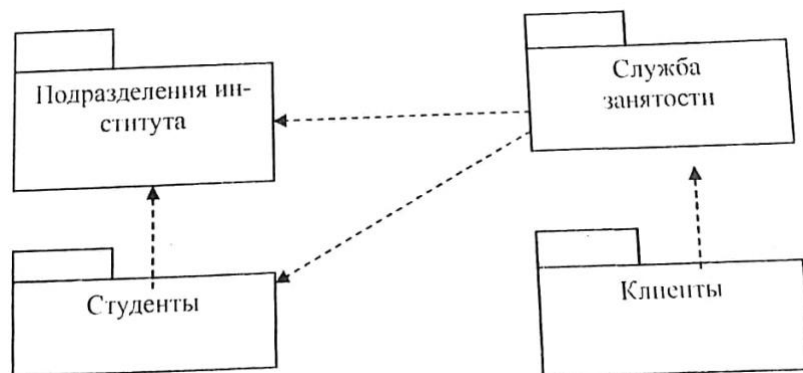


Рис. 7.1. Диаграмма 1

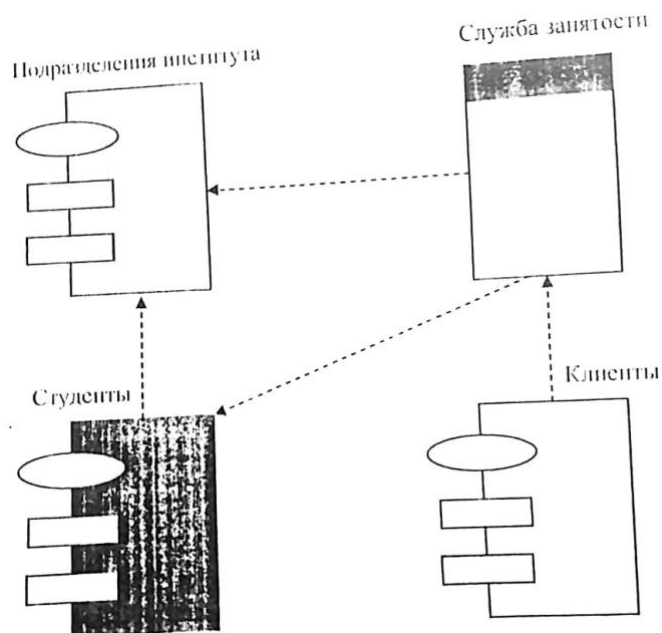


Рис. 7.2. Диаграмма 2

На рис. 7.2 изображена диаграмма компонентов, построенное на основе диаграммы пакетов, изображённой на рис. 7.1. На рис. 7.3 изображена диаграмма размещения подсистемы «Служба занятости в рамках ВУЗа».

Оценка для данной диаграммы компонентов равна:

$$S = \frac{\sum S_{об} + \sum S_{св}}{1 + \sqrt{C_{об} + T_{об} + T_{св}}} = \frac{16 + 8}{1 + 4 + 2} \approx 3,74$$

т.е. равна оценки для соответствующей диаграммы пакетов. Оценка для диаграммы размещения равна:

$$S = \frac{\sum S_{об} + \sum S_{св}}{1 + \sqrt{C_{об} + T_{об} + T_{св}}} = \frac{12 + 4}{1 + 5 + 2} = 2$$

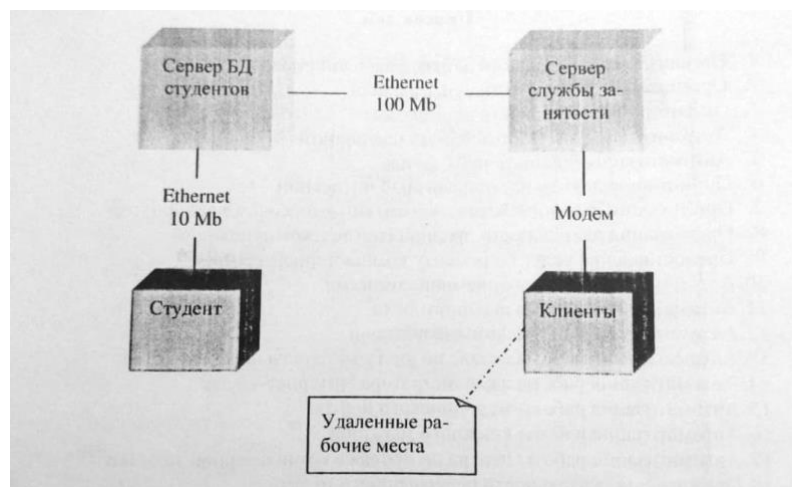


Рис. 7.3 Диаграмма 3

Задания

1. Построить для моделируемой системы общую диаграмму пакетов, отметить на ней пакеты с необходимыми системами библиотеками, отобразить зависимости между пакетами.
2. Построить для данной системы диаграмму компонентов, соответствующую построенной диаграмме пакетов, системные пакеты изобразить в виде спецификации пакетов.
3. Построить для проектируемой системы несколько вариантов диаграммы размещения (для архитектуры «клиент-сервер», трехуровневой архитектуры и т.д.), обосновать каждый вариант, предложите наиболее оптимальной.

Контрольные вопросы

1. Какую проблему проектирования призвана решить диаграмма пакетов?
2. В чём отличие диаграммы пакетов UML диаграммы классов?
3. В чём смысл зависимости между элементами диаграммы пакетов?

4. Что такое интерфейс класса?
5. По какому признакам классы группируются в пакеты?
6. Какие виды элементов модели представлены на диаграмме компонентов?
7. Как связаны между собой диаграмма пакетов и диаграмма компонентов?
8. Что показывает диаграмма размещения?
9. Какие сущности отображаются на диаграммах размещения?
10. В каких случаях необходимо применение диаграммы размещения?

Список тем

1. Организация деятельности автозаправочной станции
2. Организация деятельности супермаркета
3. Организация деятельности автошколы
4. Автоматизация документооборота предприятия
5. Автоматизация библиотечного фонда
6. Организация деятельности налоговой инспекции
7. Организация взаиморасчетов с клиентами в торговом предприятии
8. Организация деятельности предприятий телекоммуникаций
9. Предоставление услуг по ремонту компьютерной техники
10. Автоматизация работы приемной комиссии
11. Автоматизация работы швейного цеха

12. Автоматизация работы фото-лаборатории
13. Автоматизация работы склада по учету материальных ценностей
14. Автоматизация работы администратора Интернет-клуба
15. Автоматизация работы медицинского центра
16. Автоматизация работы книжного магазина
17. Автоматизация работы центра по продаже компьютерной техники
18. Организация деятельности переговорного пункта
19. Организация деятельности авиакомпании
20. Автоматизация работы склада
21. Автоматизация работы обменного пункта
22. Автоматизация работы отдела кадров
23. Автоматизация работы ГАИ
24. Автоматизация работы библиотеки
25. Автоматизация работы парикмахерской
26. Организация деятельности таможенной инспекции

Можно предложить свою тему.

Литература

Основная литература:

1. Избачков Ю.С., Петров В.Н. Информационные системы. Учебник для ВУЗов. - СПб.: Питер, 2016. -656 с.: ил.
2. Проектирование информационных систем: курс лекций. Учебное пособие для студентов вузов, обучающихся по специальностям в области информационных технологий/ В.И. Грекул, Г.И. Денищенко, Н.Л. Коровкина -М.: Интернет-университет информационных технологий, 2015. 304 с.
3. Калянов Г.Н. CASE: Структурный системный анализ (автоматизация и применение),- М.: Лари, 2012. -242 с.
4. Маклаков С.В. CASE средства разработки информационных систем. М.: Диалог МИФИ, 1999. -2014 с.
5. Ойхман Е.Г., Попов Т.В. Реинжиниринг бизнеса: Реинжиниринг организации и информационные технологии. 3 м.: Финансы и статистика, 2017. - 336 с.

6. Горин С.В, Тандоев А.Ю. Применение CASE-средства Erwin 2.0 для информационного моделирования в системах обработки и данных/ «СУБД», 2015, №3.

Дополнительная литература:

1. Марка Д.А., Мак.Гоуэн К. Методология структурного анализа и проектирования. М.: «МетаТехнология», 2013.
2. Новоженев Ю.В. Объектно-ориентированные технологии разработки сложных программных систем. М., 2016.
3. Панащук С.А. Разработка информационных систем с использованием CASE системы Silverrun в /СУБД , 2015, №3.
4. Горчинская О.Ю. Designer 2000 - новое поколение CASE продуктов фирмы ORACLE./СУБД, 2013, №3.
5. Маклаков С.В. BPwin ERwin. CASE - средства разработки информационных систем. М.: Диалог -МИФИ, 2001. -256 с.