

МИНОБРНАУКИ РОССИИ

Орский гуманитарно-технологический институт (филиал)  
федерального государственного бюджетного образовательного учреждения  
высшего образования «Оренбургский государственный университет»  
(Орский гуманитарно-технологический институт (филиал) ОГУ)

Кафедра программного обеспечения

Методические указания  
для обучающихся по освоению дисциплины

«Б.1.В.ОД.14 Разработка и применение прикладного программного  
обеспечения»

Уровень высшего образования

БАКАЛАВРИАТ

Направление подготовки

09.03.03 Прикладная информатика  
(код и наименование направления подготовки)

Прикладная информатика в экономике  
(наименование направленности (профиля) образовательной программы)

Тип образовательной программы

Программа академического бакалавриата

Квалификация

Бакалавр

Форма обучения

Очная

Год начала реализации программы (набора)

2014, 2015, 2016


г. Орск 2017

Методические указания для обучающихся по освоению дисциплины «Б.1.В.ОД.14 Разработка и применение прикладного программного обеспечения» предназначены для обучающихся очной и заочной форм обучения направления подготовки 09.03.03 Прикладная информатика, профиля «Прикладная информатика в экономике»

Составитель \_\_\_\_\_  \_\_\_\_\_ О.В. Подсобляева

Методические указания рассмотрены и одобрены на заседании кафедры программного обеспечения, протокол № 9 от «07» июня 2017 г.

Заведующий кафедрой программного обеспечения

\_\_\_\_\_  \_\_\_\_\_ Е.Е.Сурина

© Подсобляева О.В., 2017  
© Орский гуманитарно-технологический институт (филиал) ОГУ, 2017

## **1 Методические указания по проведению лекционных занятий**

Лекционные занятия в высшем учебном заведении являются основной формой организации учебного процесса и должны быть нацелены на выполнение ряда задач:

- ознакомить студентов со структурой дисциплины;
- изложить основной материал программы курса дисциплины;
- ознакомить с новейшими подходами и проблематикой в данной области;
- сформировать у студентов потребность к самостоятельной работе с учебной, нормативной и научной литературой.

Лекционное занятие представляет собой систематическое, последовательное, монологическое изложение преподавателем-лектором учебного материала, как правило, теоретического характера.

Цель лекции – организация целенаправленной познавательной деятельности студентов по овладению программным материалом учебной дисциплины.

Чтение курса лекций позволяет дать связанное, последовательное изложение материала в соответствии с новейшими данными науки, сообщить слушателям основное содержание предмета в целостном, систематизированном виде.

В ряде случаев лекция выполняет функцию основного источника информации, когда новые научные данные по той или иной теме не нашли отражения в учебниках.

Организационно-методической базой проведения лекционных занятий является рабочий учебный план направления подготовки. При подготовке лекционного материала преподаватель обязан руководствоваться учебными программами по дисциплинам кафедры, тематика и содержание лекционных занятий которых представлена в рабочих программах, учебно-методических комплексах.

При чтении лекций преподаватель имеет право самостоятельно выбирать формы и методы изложения материала, использовать различные технические средства обучения.

Рекомендации по работе студентов с конспектом лекций.

Изучение дисциплины студенту следует начинать с проработки рабочей программы, особое внимание, уделяя целям и задачам, структуре и содержанию курса.

При конспектировании лекций студентам необходимо излагать услышанный материал кратко, своими словами, обращая внимание, на логику изложения материала, аргументацию и приводимые примеры. Необходимо выделять важные места в своих записях. Если непонятны какие-либо моменты, необходимо записывать свои вопросы, постараться найти ответ на них самостоятельно. Если самостоятельно не удалось разобраться в материале, впоследствии необходимо либо на следующей лекции, либо на лабораторном занятии или консультации обратиться к ведущему преподавателю за разъяснениями.

Успешное освоение курса предполагает активное, творческое участие студента путем планомерной, повседневной работы. Лекционный материал следует просматривать в тот же день. Рекомендуемую дополнительную литературу следует прорабатывать после изучения данной темы по учебнику и материалам лекции.

Каждая тема имеет свои специфические термины и определения. Усвоение материала необходимо начинать с усвоения этих понятий. Если какое-либо понятие вызывает затруднения, необходимо посмотреть его суть и содержание в словаре (Интернете), выписать его значение в тетрадь для подготовки к занятиям.

При подготовке материала необходимо обращать внимание на точность определений, последовательность изучения материала, аргументацию, собственные примеры, анализ конкретных ситуаций. Каждую неделю рекомендуется отводить время для повторения пройденного материала, проверяя свои знания, умения и навыки по контрольным вопросам и тестам.

## 2 Методические указания по практическим работам

Изучение дисциплины «Разработка и применение прикладного программного обеспечения» предполагает посещение обучающимися не только лекций, но и лабораторных работ. Лабораторные работы со студентами предназначены для проверки усвоения ими теоретического материала дисциплины.

Основные цели лабораторных работ:

- закрепить основные положения дисциплины;
- проверить уровень усвоения и понимания студентами вопросов, рассмотренных на лекциях и самостоятельно изученных по учебной литературе;
- научить пользоваться нормативной и справочной литературой для получения необходимой информации о конкретных технологиях;
- оказать помощь в приобретении навыков расчета точностных характеристик;
- восполнить пробелы в пройденной теоретической части курса и оказать помощь в его усвоении.

Для контроля знаний, полученных в процессе освоения дисциплины на лабораторных занятиях обучающиеся выполняют задания реконструктивного уровня и комплексное практическое задание.

Целью выполнения задания реконструктивного уровня и комплексного задания студентами является систематизация, закрепление и расширение теоретических знаний, полученных в ходе изучения дисциплины.

Ниже приводятся общие методические указания, которые относятся к занятиям по всем темам:

- в начале каждого лабораторного занятия необходимо сформулировать цель, поставить задачи;
- далее необходимо проверить знания студентами лекционного материала по теме занятий;
- в процессе занятия необходимо добиваться индивидуальной самостоятельной работы студентов;
- знания студентов периодически контролируются путем проведения текущей аттестации (рубежного контроля), сведения о результатах которой доводятся до студентов и подаются в деканат;
- время, выделенное на отдельные этапы занятий, указанное в рабочей программе, является ориентировочным; преподаватель может перераспределить его, но должна быть обеспечена проработка в полном объеме приведенного в рабочей программе материала;
- на первом занятии преподаватель должен ознакомить студентов с правилами поведения в лаборатории и провести инструктаж по охране труда и по пожарной безопасности на рабочем месте;
- преподаватель должен ознакомить студентов со всем объемом лабораторных работ и требованиями, изложенными выше;
- преподаватель уделяет внимание оценке активности работы студентов на занятиях, определению уровня их знаний на каждом занятии.

На лабораторных работах решаются задачи из всех разделов изучаемой дисциплины.

### Лабораторная работа 3

#### ДИАГРАММА КЛАССОВ

Цель работы: изучение элементов UML, присутствующих на диаграммах классов, и их расширений, получение навыков построения диаграмм классов.

#### 3.1. Теоретические сведения

Класс представляют собой описание совокупности объектов с общими атрибутами, операциями, отношениями и семантикой.

Графически класс изображается прямоугольником с несколькими полями, как показано на рис. 12. Обычно полей три. Первое поле это название класса. Второе – атрибуты (члены класса, которые в зависимости от языка программирования также могут называться полями или свойствами). В примере на рисунке класс «Класс1» имеет один атрибут «x» целого типа. Перед атрибутом можно указывать тип его видимости (другое название – модификатор доступа), в данном случае значок «-» означает, что атрибут имеет тип «protected».

Третье поле в изображении класса отведено под операции (член класса, которые также называются методами, функциями или процедурами). На рисунке это поле пока пустое, но показаны окна, добавляющие операцию в класс.

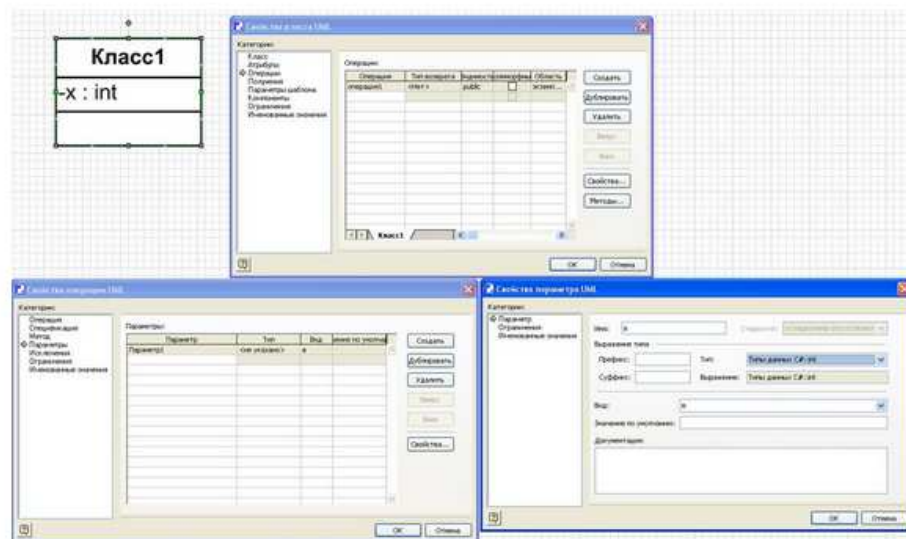


Рис. 12. Графическое представление класса UML и его свойств в MS Visio

Верхнее окно показывает свойства класс. Как видно в левом поле этого окна операции (как и атрибуты) являются свойством класса. В правом поле этого окна приведена таблица со списком операций, и кнопки для редактирования списка операций. В этой таблице указаны основные свойства операции, например, тип возврата.

Более подробно свойства операций можно установить, нажав кнопку «свойства» в верхнем окне. Появится окно свойств операции (на рисунке – ниже левое). В левом поле этого окна приведен список свойств. В примере выбрано свойство параметр операции (другим словами это аргумент функции класса). Основные свойства параметра приведены в таблице справа окна, и, аналогично свойствам операции, более подробно свойства параметров можно задать, нажав кнопку свойства (нижнее правое окно).

Подробное описание всех свойств классов заняло бы очень много времени, поэтому рекомендуется их изучать по мере использования, используя в качестве справочной информации книги по объектно-ориентированному программированию [1].

Совокупность всех классов в информационной системе образует так называемый *словарь* системы. Составление словаря – это наиболее важная и сложная часть проектирования объектно-ориентированной системы. Кроме того, эта часть проекта наиболее часто подвергается переделке. Существует много способов, методов и рекомендации по поводу того, как нужно составлять словарь системы. Мы рассмотрим лишь некоторые приемы по созданию диаграммы классов.

Наиболее известным формальным методом создания словаря системы является метод CRC-карт или таблиц (Class-responsibility-collaboration, класс-ответственность-кооперация или сущность-связь-ответственность). Идея заключается в том, что при проектировании словаря основное внимание должно уделяться сущности класса, т.е. его функциям и отношениям с другими классами, а не деталям реализации, таким как конкретные атрибуты и операции. Поэтому для каждого класса заводиться карточка (или строка в таблице), на которой отображаются существенные детали класса: название класса, его ответственность (за что класс отвечает) и кооперация или связь (т.е. с какими классами он связан). В некоторых случаях в карточки заносятся также подклассы, суперклассы и автор этого класса.

Общая схема разработки диаграммы при этом не изменяется. Обычно имеется общее описание системы, например, в виде диаграммы прецедентов. Необходимо составить полный список всех возможных сущностей системы вместе с их назначением. Выделяя общие свойства все сущности и их ответственности группируются в относительно небольшое число классов. Для классов создаются CRC-карточки или таблицы, в которые заносятся основные свойства классов. Полученная таблица анализируется. Классы в таблице должны покрывать все сущности и ответственности, при этом они должны быть сбалансированы, т.е. иметь примерно одинаковый объем ответственности. В случае несбалансированности нагрузки классов нужно провести перераспределение обязанностей или реформирование классов.

Дополнительные возможности по балансировке нагрузки между сущностями предоставляет механизм наследования в объектно-ориентированном программировании. Если несколько сущностей имеют общие свойства или ответственность, и, в тоже время имеют и различные свойства, не позволяющие их удобно сгруппировать в один класс, то можно выделить суперкласс, включающий все общие свойства, а различия в свойствах сущностей реализовать в классах-наследниках. В этом случае в CRC-таблице имеет смысл завести колонки подклассов и суперклассов.

После получения сбалансированной CRC-таблицы классы можно расположить на диаграмме классов. Отношения между классами берутся из колонки «связь». В этом случае наиболее часто между классами используется отношения «ассоциация». Если есть, то подклассы и суперклассы связываются между собой отношением наследования.

На диаграмме классов для указания характера отношений между классами часто используются дополнения к отношению ассоциация. Примеры дополнения приведены на рис. 13.

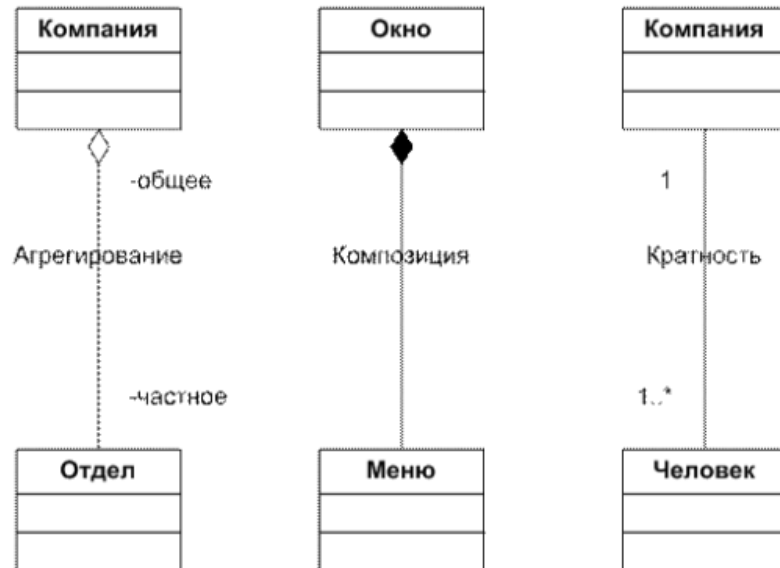


Рис. 13. Дополнения к отношению ассоциация на диаграмме классов

Агрегирование показывает отношения «часть/целое» между двумя классами, например, отдел является частью компании. Композиция показывает, что один класс входит в другой, например, в окна графического интерфейса обычно входит меню. В MS Visio это дополнение выставляется в свойствах отношения, в колонке агрегат. При моделировании баз данных указывается кратность ассоциации.

После составления словаря для каждого класса определяются атрибуты и операции. На этом этапе проектирования информационной системы не нужно детально выписывать все члены классов. Более точное описание класса удобно производить на последующих диаграммах, на которых рассматривается работа объектов этого класса.

### 3.2. Задание к работе

- 1) Сформировать словарь системы.
- 2) Составить таблицы «сущность-связь-ответственность».
- 3) По таблице сформировать классы.
- 4) Сбалансировать ответственность классов, перерисовать таблицу «сущность-связь-ответственность» для классов.
- 5) Нарисовать диаграмму классов.

## Лабораторная работа 4

### ДИАГРАММЫ ПОСЛЕДОВАТЕЛЬНОСТЕЙ

Цель работы: изучение элементов UML, присутствующих на диаграммах взаимодействий, и их расширений, получение навыков построения диаграммах взаимодействий.

#### 4.1. Теоретические сведения

В UML существует 5 видов диаграмм, которые отображают динамические свойства системы: последовательностей и коопераций (эти два типа называются диаграммами взаимодействий), деятельности, состояний и прецедентов. В нашем курсе рассмотрим диаграммы *последовательностей*.

Эти диаграммы описывают взаимодействия множества объектов, включая сообщения, которыми они обмениваются.

В отличие от диаграммы классов, на которой изображаются абстрактные элементы в виде классов, на диаграмме последовательностей используются конкретные экземпляры классов – *объекты*. Объекты отображаются прямоугольником без полей. Для того чтобы подчеркнуть, что это экземпляр абстрактной сущности название объекта подчеркивается. При необходимости через двоеточие после названия можно указать сущность (класс) экземпляром которой является этот объект. Отметим, что объект может быть экземпляром не только класса, но и других абстракций, например, актера. Обратите внимание, что при указании в качестве классификатора актера изменится графическое обозначение объекта (рис. 14).



Рис. 14. Графическое обозначение объекта UML.

На диаграмме последовательностей у объекта может присутствовать *линия жизни*, на которой отмечаются происходящие с объектом события. Линия жизни отображается пунктирной линией (рис.).

Между собой объекты могут связаны связями. *Связь* – это экземпляр отношения ассоциация, и имеет такое же графическое обозначение, что и ассоциация. Они применяются, обычно, на диаграмме коопераций.

На диаграмме последовательностей объекты обмениваются сообщениями. *Сообщение* – это спецификация передачи данных от одного объекта другому, который предполагает какое-то ответное действие. Графически сообщение обозначается сплошной линией со стрелкой.



Часто операция вызывает какую-либо операцию в объекте. Очевидно, что класс, экземпляром которого является объект, должен иметь такую операцию. Привязка сообщения к операции класса объекта выполняется в свойствах сообщения (рис. 15).

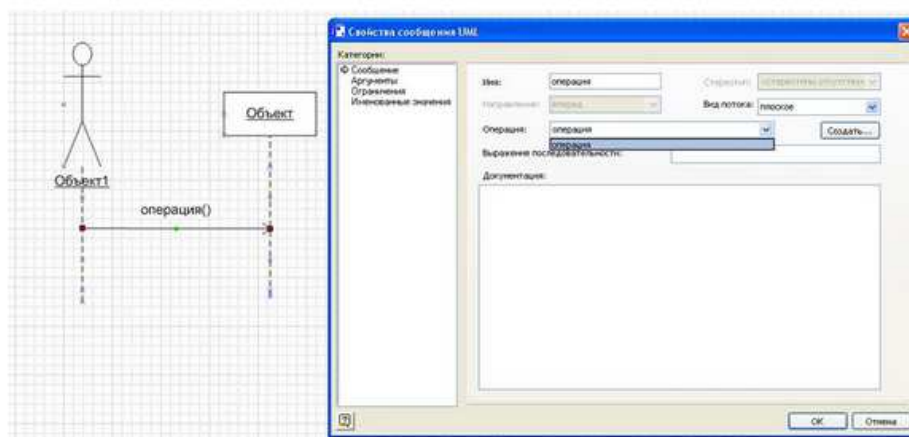


Рис. 15. Сообщение UML и его свойства

При построении диаграммы классов обычно определяются только основные свойства сущностей, а такие детали, как операции удобно создавать при построении диаграммы последовательности, для чего в свойствах сообщения UML есть кнопка создания операции.

Диаграммы последовательностей, как и другие диаграммы для отображения динамических свойств системы, могут быть выполнены в контексте многих сущностей UML. Они могут описывать поведение системы в целом, подсистемы, класса или операции класса и др. К сожалению, Visio не достаточно гибка в плане поддержки раскрытия содержания отдельных элементов с помощью других диаграмм. Например, кликнув правой кнопкой мыши по классу можно обнаружить, что для его описания можно создать лишь диаграммы классов, состояний и деятельности. Поэтому возможность привязать диаграмму последовательностей к элементу, который она реализует, средствами Visio не возможно, эту связь нужно подразумевать.

В рамках нашего курса диаграммы последовательностей мы будем делать в контексте прецедентов с диаграммы прецедентов, реализуя те функции, которые должна выполнять наша система.

Как говорилось выше, при построении динамических диаграмм используется уже разработанная структура информационной системы. Для диаграммы последовательностей не нужно придумывать объекты, а достаточно определить, экземпляры каких классов участвуют в этом действии.

Определив необходимые объекты (как экземпляры классов, так и экземпляры актеров), вторым этапом построения диаграмм последовательностей определяются сообщения, пересылаемые между актерами. Фактически определяется последовательность шагов, для выполнения нужного действия.

## 4.2. Задание к работе

- 1) Выбрать три наиболее значимых прецедента на диаграмме прецедентов, для каждого из них создать диаграмму взаимодействий.
- 2) Для каждой диаграммы определить объекты, участвующие во взаимодействии, расположить их на диаграммах.
- 3) Определить сообщения между объектами на диаграммах, создайте необходимые операции для классов.

## Лабораторная работа 5

### АНАЛИЗ ДИАГРАММ ВЗАИМОДЕЙСТВИЙ И КЛАССОВ

Цель работы: получение навыков сопоставления различных точек зрения на информационную систему, разработки нескольких диаграмм, рассматривающие различные аспекты информационной системы, проведение рефакторинга системы.

#### 5.1. Теоретические сведения

Существует много различных методов проектирования, которые акцентируются на различных аспектах информационной системы. Так, функциональный анализ разбивает систему на отдельные функции, а объектно-ориентированный анализ на объекты. Каждый из подходов имеет свои преимущества, но все они не лишены и недостатков – не основным чертам уделяется меньшее внимание.

В объектно-ориентированном анализе в первую очередь строится структурная схема. При ее построении динамические аспекты поведения системы не учитываются, однако они могут оказать существенное влияние.

Например, при проектировании структуры системы составления расписания мы можем выделить сущности «студенты», «преподаватели» и «аудиторий». Однако начав расписывать диаграммы последовательностей для прецедентов «ввод данных», «выбор свободной пары» или другие, мы обнаружим, что над этими сущностями осуществляются одинаковые действия, например, просматривается их расписание. При этом можно значительно упростить диаграммы последовательностей, введя абстрактную сущность «участник пары» и сделав имеющиеся сущности наследниками.

Хотя в рассмотренном примере «участник пары» мог появиться уже на этапе построения диаграммы классов, тем не менее, выявление общих свойства классов при построении диаграммы последовательностей встречается довольно часто. В общем случае, при построении любых диаграмм могут открываться свойства системы, которые желательно учесть на ранее созданных диаграммах.

Изменение структуры системы (без изменения ее внешних свойств) называется *рефакторингом*. Основная цель рефакторинга – упростить систему, сделать ее более понятной. В общем случае рефакторинг может проводиться даже на стадии программирования, когда часть системы уже реализована.

Изменение уже частично спроектированной системы требует дополнительной работы. Не всегда результат оправдывает затраты на эту работу. Однако, во многих методиках проектирования и разработки информационных систем рекомендуется проводить рефакторинг как можно чаще и раньше, во всех случаях когда есть возможность упростить систему, поскольку это позволит значительно сэкономить на последующих стадиях – программирование, тестирование, отладка, внедрение.

В текущей лабораторной работе нужно попытаться упростить структурную схему (диаграмму классов) после построения диаграмм взаимодействий (последовательностей).

Некоторые динамические свойства системы отображаются на структурной схеме средствами Visio – в классах появятся операции, введенные на диаграммах последовательностей. Если количество операций в различных классах сильно отличается, то это может говорить о несбалансированности нагрузки на классы. Также нужно обратить внимание на частоту использования объектов на диаграмме последовательностей – если объекты какого-либо класса используются слишком часто, то, возможно, класс имеет слишком много обязанностей.

Проанализировав все диаграммы можно попытаться перераспределить обязанности между классами (или создать/сократить какие-либо классы). Главная цель и критерий необходимости всех изменений – упрощение системы, т.е. улучшение понимания диаграмм их читателем, уменьшение количество связей между элементами, приведение количество элементов к удобному для восприятия числу (от 3 до 5).

#### 5.2. Задание к работе

- 1) Проанализировать диаграммы взаимодействий, выявить функционально недогруженные или перегруженные, имеющие общие свойства объекты и классы.
- 2) Провести перераспределение обязанностей между классами, переделать таблицы «сущность-связь-ответственность».
- 3) Переделать диаграммы классов и взаимодействий в соответствии с внесенными изменениями

## Лабораторная работа 6

### ДИАГРАММЫ КОМПОНЕНТОВ И РАЗВЕРТЫВАНИЯ

Цель работы: изучение элементов UML, присутствующих на диаграммах компонентов и развертывания, и их расширений, получение навыков построения диаграмм компонентов и развертывания.

#### 6.1. Теоретические сведения

Диаграммы компонентов (Component diagram) – это один из двух видов диаграмм, применяемых при моделировании физических аспектов объектно-ориентированной системы (второй вид – диаграммы развертывания). Она показывает набор структурных компонентов и отношения между ними.

Диаграммы компонентов показывают, как выглядит модель на физическом уровне.

Компонент – это физическая заменяемая часть системы, совместимая с одним набором интерфейсов и обеспечивающая реализацию какого-либо другого.

Компоненты используются для моделирования физических сущностей размещенных в узле: исполняемых модулей, библиотек, таблиц, файлов и документов. Обычно компонент представляет собой физическую упаковку логических элементов, таких как классы, интерфейсы и кооперации.

Диаграммы компонентов могут также содержать пакеты, которые используются для группирования элементов модели в крупные блоки, и интерфейсы, которые определяют сервис (набор услуг), предоставляемый компонентом.

Графически элементы диаграммы компонентов в UML изображаются так, как показано на рис. 16.

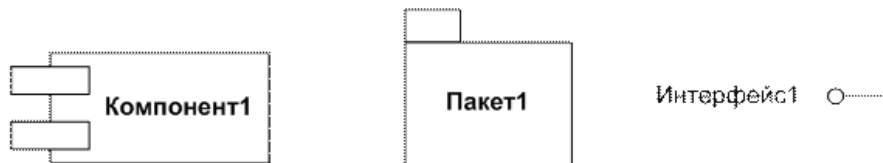


Рис. 16. Элементы диаграммы компонентов

Между отдельными компонентами изображают зависимости (рис. 17). Так, например, изображенный ниже фрагмент диаграммы компонентов представляет информацию о том, что компонент с именем Control зависит от импортируемого интерфейса IDialog, который, в свою очередь, реализуется компонентом с именем DataBase .

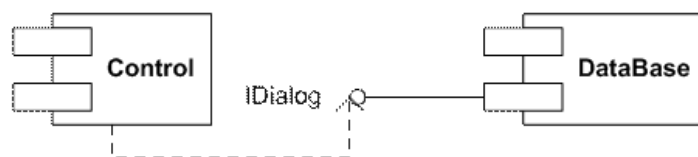


Рис. 17. Фрагмент диаграммы компонентов

Диаграммы развертывания, или применения (Deployment diagram), показывают конфигурацию узлов, где производится обработка информации, и то, какие компоненты размещены на каждом узле.

Диаграммы развертывания обычно включают в себя узлы и отношения.

Узлы используются для моделирования топологии аппаратных средств, на которых исполняется система. Как правило, узел – это процессор или устройство, на котором могут быть развернуты компоненты. Принадлежность компонентов к узлам в MS Visio наглядно не изображается, а определяется в свойствах узла, как показано на рис. 18.

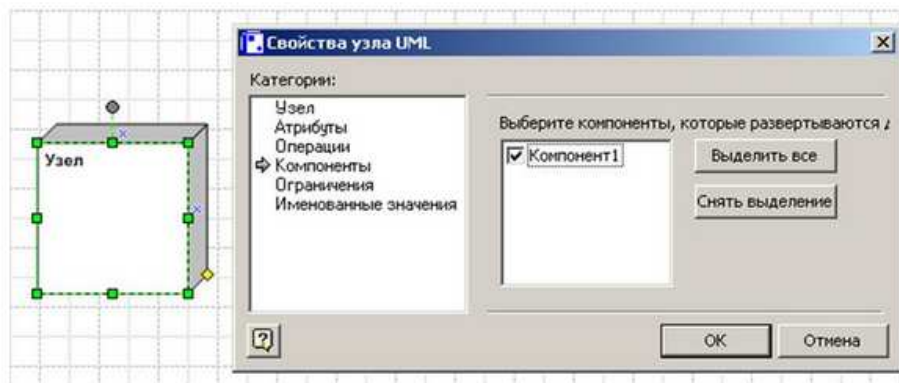


Рис. 18. Графическое представление узла UML и его свойств

Во многих отношениях узлы подобны компонентам. Те и другие наделены именами, могут быть участниками отношений зависимости, обобщения и ассоциации, бывают вложенными, могут иметь экземпляры и принимать участие во взаимодействиях. Однако между ними есть и существенные различия:

- 1) Компоненты принимают участие в исполнении системы; узлы – это сущности, которые исполняют компоненты.
- 2) Компоненты представляют физическую упаковку логических элементов; узлы представляют средства физического развертывания компонентов.

Узлы можно организовывать, группируя их в пакеты, точно так же, как это делается с компонентами.

Кроме собственно изображений узлов на диаграмме развертывания указываются отношения между ними (рис. 19). В качестве отношений выступают физические соединения между узлами и зависимости между узлами и компонентами, изображения которых тоже могут присутствовать на диаграммах развертывания.

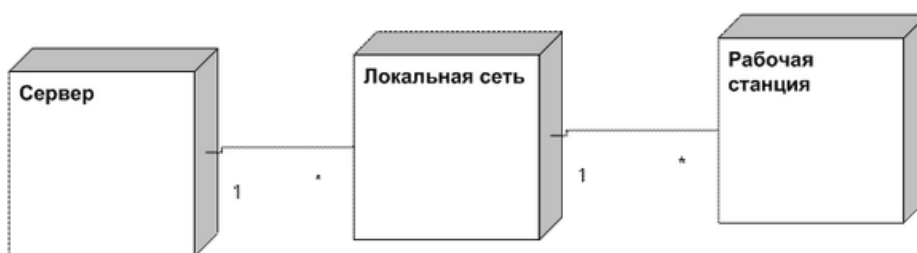


Рис. 19. Фрагмент диаграммы развертывания с соединениями между узлами

## 6.2. Задание к работе

- 3) Проанализировать диаграммы классов, выделить логические компоненты информационной системы, нарисовать диаграмму компонентов.
- 4) Проанализировать диаграммы классов и прецедентов, выделить физические компоненты информационной системы, нарисовать диаграмму развертывания.

## ВАРИАНТЫ ЗАДАНИЙ

### 1. Программное обеспечение банкомата

Обзор: банкомат по карте позволяет снимать наличные со счета по и/или печатать справку об остатке на счете.

### 2. Информационная система библиотеки

Обзор: информационная система библиотеки позволяет искать книги в своем каталоге, учитывать выдачу книг на руки и возврат книг, а также позволяет добавлять книги в фонд и списывать их.

### 3. Информационная система поликлиники

Обзор: информационная система поликлиники позволяет ставить и снимать больных с учета, записывать больных на прием к врачам, учитывать факт приема, а также позволяет вести историю болезни (медицинскую карту) больного.

### 4. Информационная система деканата

Обзор: информационная система деканата позволяет принимать и отчислять студентов, вести учет успеваемости по итогам сессии, переводить студентов из группы в группу и с курса на курс.

### 5. Система мгновенного обмена сообщениями

Обзор: система позволяет регистрировать и аннулировать абонентов, позволяет абонентам подключаться и отключаться от системы, и позволяет подключенным абонентам обмениваться текстовыми сообщениями в реальном времени.

### 6. Информационная система склада

Обзор: информационная система склада позволяет учитывать поступление, уход и списание товаров со склада, а также определять место хранения товаров на складе.

### 7. Система учета рабочего времени

Обзор: Система учета рабочего времени позволяет руководителям выдавать задания и отслеживать ход их выполнения, а исполнителям - вести учет рабочего времени, затраченного на выполнение каждого задания.

### 8. Информационная система жилищного агентства

Обзор: информационная система жилищного агентства позволяет квартиросъемщикам подобрать и снять жилье, а владельцам жилья - предложить и сдать жилье.

### 9. Информационная система технической экспертизы

Обзор: информационная система технической экспертизы позволяет соискателям грантов подавать заявки, независимым экспертам оценивать заявки, а держателям фонда принимать решение о выдаче грантов по результатам экспертизы заявок.

### 10. Система продажи билетов на футбол

Обзор: система продажи билетов позволяет покупать и сдавать билеты и абонементы на матчи, проходящие на одном стадионе с нумерованными местами через несколько одновременно работающих касс.

### 11. Текстовый редактор

Обзор: текстовый редактор позволяет создавать, редактировать и печатать текстовые файлы. При отображении файлов специальных форматов поддерживается подсветка ключевых слов.

### 12. Система автоматического тестирования

Обзор: Система позволяет автоматически запускать тесты, отслеживать результаты их выполнения и выдавать отчеты.

### 13. Электронная доска объявлений

Обзор: информационная система позволяет размещать и удалять объявления о продаже различных товаров.

### 14. Игра Монополия

Обзор: Игра человека против машины

## 4 Методические указания по итоговому контролю

Итоговый контроль знаний по дисциплине «Разработка и применение прикладного программного обеспечения» проводится в форме экзамена. Для подготовки к итоговому контролю знаний по дисциплине «Разработка и применение прикладного программного обеспечения» обучающиеся используют перечень вопросов, приведенный в фонде оценочных средств. Экзамен проводится в устной форме. В экзаменационный билет включен один теоретический вопрос. На подготовку студенту отводится 20-25 минут. На дифференцированном зачете ответы обучающегося оцениваются с учетом их полноты, правильности и аргументированности с учетом шкалы оценивания.

Оценка «отлично» выставляется студенту, если он глубоко и прочно усвоил программный материал, исчерпывающе, последовательно, четко и логически его излагает, умеет тесно увязывать теорию с практикой, свободно справляется с вопросами и другими видами применения знаний, причем не затрудняется с ответом при видоизменении заданий, использует в ответе профессиональные термины, правильно обосновывает принятое решение.

Оценка «хорошо» выставляется студенту, если он твердо знает материал, грамотно и по существу излагает его, не допуская существенных неточностей в ответе на вопрос, правильно применяет теоретические положения при решении практических вопросов, владеет необходимыми навыками и приемами их выполнения.

Оценка «удовлетворительно» выставляется студенту, если он имеет знания только основного материала, но не усвоил его деталей, допускает неточности, недостаточно правильные формулировки, нарушения логической последовательности в изложении программного материала.

Оценка «неудовлетворительно» выставляется студенту за отсутствие знаний по дисциплине, представления по вопросу, непонимание материала по дисциплине, наличие коммуникативных «барьеров» в общении, отсутствие ответа на предложенный вопрос.

## 5 Учебно-методическое обеспечение дисциплины

### 5.1 Основная литература

1. Золотов, С.Ю. Проектирование информационных систем : учебное пособие / С.Ю. Золотов ; Министерство образования и науки Российской Федерации, Томский Государственный Университет Систем Управления и Радиоэлектроники (ТУСУР). - Томск : Эль Контент, 2013. - 88 с. : табл., схем. - ISBN 978-5-4332-0083-8 ; То же [Электронный ресурс]. - URL: <http://biblioclub.ru/index.php?page=book&id=208706>, коэффициент книгообеспеченности 1

2. Стасышин, В.М. Проектирование информационных систем и баз данных : учебное пособие / В.М. Стасышин. - Новосибирск : НГТУ, 2012. - 100 с. - ISBN 978-5-7782-2121-5 ; То же [Электронный ресурс]. - URL: <http://biblioclub.ru/index.php?page=book&id=228774>, коэффициент книгообеспеченности 0,9

### 5.2 Дополнительная литература

1. Проектирование информационных систем [Текст] : учебное пособие / Г. Н. Исаев.- 2-е изд., стер. - Москва : Омега - Л, 2015. - 424 с. - (Высшее техническое образование) - ISBN 978-5-370-03507-4, коэффициент книгообеспеченности 1

2. Абрамов, Г.В. Проектирование информационных систем : учебное пособие / Г.В. Абрамов, И.Е. Медведкова, Л.А. Коробова. - Воронеж : Воронежский

государственный университет инженерных технологий, 2012. - 172 с. : ил.,табл., схем. - ISBN 978-5-89448-953-7 ; То же [Электронный ресурс]. - URL: [//biblioclub.ru/index.php?page=book&id=141626](http://biblioclub.ru/index.php?page=book&id=141626), коэффициент книгообеспеченности 1.

3. Проектирование информационных систем [Текст] : учебное пособие / Г. Н. Исаев.- 2-е изд., стер. - Москва : Омега - Л, 2015. - 424 с. - (Высшее техническое образование) - ISBN 978-5-370-03507-4. аб. СПО-16; ч/з N1-1, коэффициент книгообеспеченности 1.

4. Заика, А.А. Разработка прикладных решений для платформы 1С:Предприятие 8.2 в режиме "Управляемое приложение" / А.А. Заика. - 2-е изд., испр. - М. : Национальный Открытый Университет «ИНТУИТ», 2016. - 239 с. : ил. ; То же [Электронный ресурс]. - URL: [//biblioclub.ru/index.php?page=book&id=429019](http://biblioclub.ru/index.php?page=book&id=429019), коэффициент книгообеспеченности 1.

### **5.3 Периодические издания**

1. Журнал «Вестник компьютерных и информационных технологий»
2. Журнал «Информационные технологии и вычислительные системы»
3. Журнал «Стандарты и качество»
4. Журнал «Прикладная информатика»

### **5.4 Интернет-ресурсы**

#### **5.4.1 Современные профессиональные базы данных и информационные справочные системы:**

1. Информационная система «Единое окно доступа к образовательным ресурсам»- <http://window.edu.ru/>
2. КиберЛенинка - <https://cyberleninka.ru/>
3. Университетская информационная система Россия– [uisrussia.msu.ru](http://uisrussia.msu.ru)
4. Бесплатная база данных ГОСТ– <https://docplan.ru/>

#### **5.4.2 Тематически профессиональные базы данных и информационные справочные системы:**

1. Портал искусственного интеллекта – [AIPortal](#)
2. Web-технологии – [Web-технологии](#)
3. Электронная библиотека Института прикладной математики им. М.В. Келдыша – [Электронная библиотека публикаций Института прикладной математики им. М.В. Келдыша РАН](#)

#### **5.4.3 Электронные библиотечные системы**

1. ЭБС «Университетская библиотека онлайн» – <http://www.biblioclub.ru/>
2. ЭБС Znanium.com – <https://znanium.com/>

#### **5.4.4 Дополнительные Интернет-ресурсы**

1. <https://www.ixbt.com> - Интернет-издание о компьютерной технике, информационных технологиях и программных продуктах. На сайте публикуются новости IT, статьи с обзорами и тестами компьютерных комплектующих и программного обеспечения.

2. <http://www.intuit.ru> – ИНТУИТ – Национальный открытый университет.
3. <http://cppstudio.com/> - Основы программирования на языках Си и C++.
4. <https://www.anti-malware.ru/> - Информационно-аналитический центр, посвященный информационной безопасности.
5. <https://developer.mozilla.org/ru/docs/Tools> — Открытые уроки по веб-технологиям и инструментам разработчика.
6. <https://frontender.info> – Электронный журнал по фронтенд-разработке
7. <https://docs.oracle.com/en/java/> - Документация по языку Java.
8. [http://citforum.ru/SE/project/arkhipenkov\\_lectures](http://citforum.ru/SE/project/arkhipenkov_lectures) – Лекции по управлению программными проектами автор А. Архипенков
9. <http://1c.ru/> - сайт фирмы разработчика серии программ "1С:Предприятие", предназначенных для автоматизации управления и учета на предприятиях различных отраслей, видов деятельности и типов финансирования.

### **5.5 Программное обеспечение, профессиональные базы данных и информационные справочные системы современных информационных технологий**

Тип программного обеспечения	Наименование	Схема лицензирования, режим доступа
Операционная система	Microsoft Windows	Подписка Enrollment for Education Solutions (EES) по государственному контракту: № 2К/17 от 02.06.2017 г.
Текстовый редактор	Notepad++	Свободное ПО, <a href="https://notepad-plus-plus.org/">https://notepad-plus-plus.org/</a>
Интернет-браузер	Google Chrome	Бесплатное ПО, <a href="http://www.google.com/intl/ru/policies/terms/">http://www.google.com/intl/ru/policies/terms/</a>
Векторный графический редактор, редактор диаграмм и блок-схем	Microsoft Visio Standard 2007	Сертификат MicrosoftOpenLicense № 46284547 от 18.12.2009 г., академическая лицензия на рабочее место
Интегрированная среда разработки программного обеспечения	Microsoft Visual Studio Professional 2008	Сертификат MicrosoftOpenLicense № 46284547 от 18.12.2009 г., академическая лицензия на рабочее место
	Embarcadero RAD Studio 2010 Professional	Образовательная лицензия по государственному контракту № 32/09 от 17.12.2009 г., сетевой конкурентный доступ
	Turbo Pascal 7.0 for DOS	Образовательная лицензия по государственному контракту № 34/10 от 10.12.2010 г., лицензия на рабочее место
	Borland C++ 3.1 for DOS	Образовательная лицензия по государственному контракту № 34/10 от 10.12.2010 г., лицензия на рабочее место
	Dev-C++	Свободное ПО, <a href="http://www.gnu.org/licenses/gpl.html">http://www.gnu.org/licenses/gpl.html</a>

### **6 Материально-техническое обеспечение дисциплины**

Учебные аудитории для проведения занятий лекционного типа, семинарского типа, для проведения групповых и индивидуальных консультаций, текущего контроля и



промежуточной аттестации. Для проведения лабораторных и практических работ используются компьютерный класс (ауд. № 4-113, 4-116, 4-117), оборудованный средствами оргтехники, программным обеспечением, персональными компьютерами, объединенными в сеть с выходом в Интернет.

Аудитории оснащены комплектами ученической мебели, техническими средствами обучения, служащими для представления учебной информации большой аудитории.

Помещения для самостоятельной работы обучающихся оснащены компьютерной техникой, подключенной к сети «Интернет», и обеспечением доступа в электронную информационно-образовательную среду Орского гуманитарно-технологического института (филиала) ОГУ (ауд. № 4-307).

Наименование помещения	Материальное-техническое обеспечение
Учебные аудитории: - для проведения занятий лекционного типа, семинарского типа, - для групповых и индивидуальных консультаций; - для текущего контроля и промежуточной аттестации	Учебная мебель, классная доска, мультимедийное оборудование (проектор, экран, ноутбук с выходом в сеть «Интернет»)
Компьютерные классы № 4-113, 4-116, 4-117	Учебная мебель, компьютеры (29) с выходом в сеть «Интернет», проектор, экран, лицензионное программное обеспечение
Помещение для самостоятельной работы обучающихся, для курсового проектирования (выполнения курсовых работ)	Учебная мебель, компьютеры (3) с выходом в сеть «Интернет» и обеспечением доступа в электронную информационно-образовательную среду Орского гуманитарно-технологического института (филиала) ОГУ, программное обеспечение

Для проведения занятий лекционного типа используются следующие наборы демонстрационного оборудования и учебно-наглядные пособия:

- презентации к ку