Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное

учреждение высшего образования

«Воронежский государственный лесотехнический университет

имени Г.Ф.Морозова»

Кафедра Вычислительной техники и информационных систем

(название кафедры)

**Пояснительная записка**

**к Расчётно-графической работе(РГР)**

(вид работы)

«Проектирование архитектуры информационной системы»

(тема)

09.03.02Информационные системы и технологии

(код и наименование направления подготовки)

По дисциплине:Архитектура информационных систем

**Вариант 9**

Студент группы ИС2–191–ОБ \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Величко В.А.

(номер группы) (подпись) (инициалы и фамилия)

Руководитель к.т.н.доцент \_\_\_\_\_\_\_\_\_\_\_\_\_Ачкасов А.В.

(ученая степень,ученоезвание)(подпись)(инициалыифамилия)

Воронеж 2020г.

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ ЛЕСОТЕХНИЧЕСКИЙ  
УНИВЕРСИТЕТ ИМЕНИ Г.Ф.МОРОЗОВА

Кафедра Вычислительной техники и информационных систем

ЗАДАНИЕ

на расчетно-графическую работу по дисциплине

«Информационные системы: этапы развития и введения

в специальность»

Студенту 2 курса гр.ИС2-191-ОБ Величко В.А.

(Фамилия И.О.)

1. ТЕМА РАБОТЫ

«Правила языка UML. Общие механизмы языка UML».

1. ПОСТАНОВКА ЗАДАЧИ

Для выполнения необходимо для своей предметной области разработать и описать:

- первоначальная постановка задачи: диаграмму прецедентов, диаграмму деятельности, диаграмму классов;

- развитие постановки задачи: диаграмму прецедентов, диаграмму деятельности, диаграмму классов.

Вариант№9

Задание выдано«\_\_»\_\_\_\_\_\_\_2020\_\_г.

Срок выполнения«\_\_»\_\_\_\_\_\_2020\_\_г.

Задание выдал, к.т.н.,доц.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_А.В.Ачкасов

Зав.кафедрой, д.т.н.,проф.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_В.К.Зольников

ВВЕДЕНИЕ

В современном обществе своевременная обработка информации способствует совершенствованию организации производства, оперативному и долгосрочному планированию, прогнозированию и анализу хозяйственной деятельности.

Для правильного координирования процессов протекающих в моделированной системе управления необходимо создать структуру, т.е. упорядочить процессы. Моделирование работы информационной системы особенно важно на первых этапах её создания. Так как исправление допущенных на этом этапе ошибок обходится наиболее дорого, то и польза на этапе анализа задачи и разработки логической модели её решения значительна

1. **Теоретическая часть**
   1. **Характеристика организации**

Разрабатываемая модель информационной системы реализует работу ломбарда, где осуществляется выдача наличных под залог, покупка и продажа товара

Ломбард работает с различного рода клиентами, осуществляя услуги по выдачи денег под залог, а также покупка и продажа различного рода товара. Работа с самими клиентами организована следующим образом: клиент, пришедший в ломбард, оставляет свой товар у приёмщика товара, например, телефон, телевизор. После этого сотрудник оценивает себестоимость товара и даёт клиенту деньги. Наряду с этим, обсуждается срок залога, по истечении которого, клиент обязуется вернуть деньги

База данных должна в полной мере осуществлять все задуманные процессы: регистрация приходящих клиентов, запись даты выдачи денег под залог, поиск товара и поиск клиентов.

Необходимо, чтобы база данных по мере внесения новых записей в систему вовремя обновлялась, включая в себя следующую информацию:

1. Информация, которая будет характеризовать товары: категория товаров, клиент, описание товара, дата сдачи, дата возврата, сумма, комиссионные;

2. Информация, характеризующая поступающих клиентов – фамилия, имя, отчество, номер паспорта, серия паспорта, дата выдачи паспорта;

3. Информация о продажах и покупках товара;

* 1. **Понятие архитектуры информационной системы**

Архитектура – это совокупность существенных решений об организации ИС. Обычно в понятие архитектуры входят решения об основных аппаратных и программных составляющих системы, их функциональном назначении и ор-ганизации связей между ними.

Выбор архитектуры ИС влияет на следующие характеристики:

1. Производительность ИС – количество работ, выполняемых в ИС за единицу времени.

2. Время реакции системы на запросы пользователя (время отклика си-стемы)

3. Надёжность – способность к безотказному функционированию в тече-ние определенного периода времени.

Локальные ИС, которые располагаются целиком на одном компьютере и предназначены для работы только одного пользователя, сейчас встречаются крайне редко. В дальнейшем речь пойдет о распределенных ИС, которые функционируют в сети и предназначены для многопользовательской (коллек-тивной) работы.

Обычно база данных целиком хранится в одном узле сети, поддержива-ется одним сервером и доступна для всех пользователей локальной сети, назы-ваемых клиентами. Такая база данных называется централизованной. Распре-деленные базы данных, в которых БД распределена по нескольким узлам сети, обычно используются в организациях, содержащих территориально удален-ные подразделения.

Сервер, как правило, — самый мощный и самый надежный компьютер. Он обязательно подключается через источник бесперебойного питания, в нем предусматриваются системы двойного или даже тройного дублирования. В за-висимости от распределения функций обработки данных между сервером и клиентами различают две основных архитектуры – «файл-сервер» и «клиент-сервер». Возможны разновидности этих двух вариантов.

* 1. **Общие механизмы языка UML**

Строительство упрощается и ведется более эффективно, если придерживаться некоторых соглашений. Следуя определенным архитектурным образцам, можно оформить здание в викторианском или французском стиле. Тот же принцип применим и в отношении UML. Работу с этим языком существенно облегчает последовательное использование общих механизмов, перечисленных ниже:

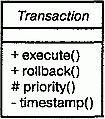
* спецификации (Specifications);
* дополнения (Adornments);
* принятые деления (Common divisions);
* механизмы расширения (Extensibility mechanisms).

UML - это не просто графический язык. За каждой частью его системы графической нотации стоит спецификация, содержащая текстовое представление синтаксиса и семантики соответствующего строительного блока. Например, пиктограмме класса соответствует спецификация, полностью описывающая его атрибуты, операции (включая полные сигнатуры) и поведение, хотя визуально пиктограмма порой отражает только малую часть этой совокупности. Более того, может существовать другое представление этого класса, отражающее совершенно иные его аспекты, но тем не менее соответствующее все той же спецификации. С помощью графической нотации UML вы визуализируете систему, с помощью спецификаций UML - описываете ее детали. Таким образом, допустимо строить модель инкрементно, то есть пошаговым образом - сначала нарисовать диаграмму, а потом добавить семантику в спецификацию модели, или наоборот - начать со спецификации (возможно, применив обратное проектирование к существующей системе), а потом на ее основе создавать диаграммы.

Спецификации UML создают семантический задний план, который полностью включает в себя составные части всех моделей системы, согласованные между собой. Таким образом, диаграммы UML можно считать визуальными проекциями на этот задний план, при этом каждая из них раскрывает один из значимых аспектов системы.

Почти каждый из элементов UML имеет соответствующее ему уникальное графическое обозначение, которое дает визуальное представление о самых важных аспектах этого элемента. Например, обозначение класса специально придумано так, чтобы его было легко рисовать, поскольку классы - наиболее употребительный элемент при моделировании объектно-ориентированных систем. Нотация класса содержит самые важные его характеристики: имя, атрибуты и операции.

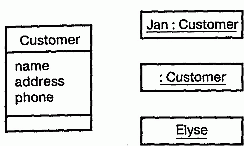
Спецификация класса может содержать и другие детали, например видимость атрибутов и операций или указание на то, что класс является абстрактным. Многие такие детали можно визуализировать в виде графических или текстовых дополнений к стандартному прямоугольнику, служащему изображением класса. Так, на рис. 1 показан класс, в обозначение которого включены сведения о том, что он абстрактный и содержит две открытые, одну защищенную и одну закрытую операцию.

  
Рис. 1Дополнения

Каждый элемент нотации UML содержит базовый для него символ, к которому можно добавлять разнообразные специфичные для него дополнения.

Принятые деления. При моделировании объектно-ориентированных систем реальность членится с учетом по крайней мере двух подходов.

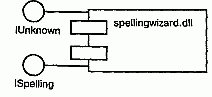
Прежде всего, существует разделение на классы и объекты. Класс - это абстракция, объект - конкретная материализация этой абстракции. В языке UML можно моделировать и классы, и объекты, как показано на рис. 2.

  
Рис. 2 Классы и объекты

На этом рисунке показан один класс Customer (Клиент) и три объекта: Jan (явно определенный как объект данного класса), :Customer (анонимный объект класса Customer) и Elyse (спецификация которого относит его к классу Customer, хотя это и не выражено явно).

Практически все строительные блоки UML характеризуются дихотомией "класс/объект". Так, имеются прецеденты и экземпляры прецедентов, компоненты и экземпляры компонентов, узлы и экземпляры узлов и т.д. В графическом представлении для объекта принято использовать тот же символ, что и для его класса, а название объекта подчеркивать.

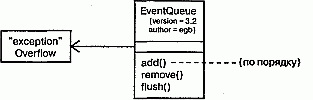
Еще одним вариантом членения является деление на интерфейс и его реализацию. Интерфейс декларирует контракт, а реализация представляет конкретное воплощение этого контракта и обязуется точно следовать объявленной семантике интерфейса. UML позволяет моделировать обе эти категории, интерфейсы и их реализации, как показано на рис. 3: в данном случае один компонент spellingwizard.dll реализует два интерфейса lUnknown и ISpelling. Почти все строительные блоки UML характеризуются дихотомией "интерфейс/реализация". Например, прецеденты реализуются кооперациями, а операции - методами.

  
Рис. 3 Интерфейсы и реализации

Механизмы расширения. UML - это стандартный язык для разработки "чертежей" программного обеспечения, но ни один замкнутый язык не в состоянии охватить нюансы всех возможных моделей в различных предметных областях. Поэтому UML является открытым языком, то есть допускает контролируемые расширения. Механизмы расширения UML включают:

* стереотипы;
* помеченные значения;
* ограничения.

Стереотип (Stereotype) расширяет словарь UML, позволяя на основе существующих блоков языка создавать новые, специфичные для решения конкретной проблемы. Например, работая с такими языками программирования, как Java или C++, часто приходится моделировать исключения (Exceptions) - они являются обыкновенными классами, хотя и рассматриваются особым образом. Обычно требуется, чтобы исключения можно было возбуждать и перехватывать, и ничего больше. Если пометить исключения соответствующим стереотипом, то с ними можно будет обращаться как с обычными строительными блоками языка; на рис. 4 это продемонстрировано на примере класса Overflow.

  
Рис. 4 Механизмы расширения

Помеченное значение (Tagged value) расширяет свойства строительных блоков UML, позволяя включать новую информацию в спецификацию элемента. Скажем, если вы работаете над "коробочным" продуктом и выпускаете много его версий, то зачастую необходимо отслеживать версию и автора какой-нибудь важной абстракции. Ни версия, ни автор не являются первичными концепциями UML, но их можно добавить к любому блоку, такому, например, как класс, задавая для него новые помеченные значения. На рис. 4 показано, как это можно сделать, на примере класса EventQueue.

Ограничения (Constraints) расширяют семантику строительных блоков UML, позволяя определять новые или изменять существующие правила. Вы можете, например, ограничить класс EventQueue так, чтобы все события добавлялись в очередь по порядку. На рис. 4 показано, как можно определить ограничение, которое явно постулирует это правило для операции add.

Совместно эти три механизма расширения языка позволяют модифицировать UML в соответствии с потребностями вашего проекта. Кроме того, они дают возможность адаптировать UML к новым технологиям разработки программного обеспечения, например к вероятному появлению более мощных языков распределенного программирования. С помощью механизмов расширения можно создавать новые строительные блоки, модифицировать существующие и даже изменять их семантику. Не забывайте, однако, о чувстве меры: за расширениями важно не потерять главную цель UML - возможность обмена информацией.

* 1. **Правила языка UML**

Строительные блоки UML нельзя произвольно объединять друг с другом. Как и любой другой язык, UML характеризуется набором правил, определяющих, как должна выглядеть хорошо оформленная модель, то есть семантически самосогласованная и находящаяся в гармонии со всеми моделями, которые с нею связаны.

В языке UML имеются семантические правила, позволяющие корректно и однозначно определять:

имена, которые можно давать сущностям, отношениям и диаграммам;

область действия (контекст, в котором имя имеет некоторое значение);

видимость (когда имена видимы и могут использоваться другими элементами);

целостность (как элементы должны правильно и согласованно соотноситься друг с другом);

выполнение (что значит выполнить или имитировать некоторую динамическую модель).

Модели, создаваемые в процессе разработки программных систем, эволюционируют со временем и могут неоднозначно рассматриваться разными участниками проекта в разное время. По этой причине создаются не только хорошо оформленные модели, но и такие, которые:

содержат скрытые элементы (ряд элементов не показывают, чтобы упростить восприятие);

неполные (отдельные элементы пропущены);

несогласованные (целостность модели не гарантируется).

Появление не слишком хорошо оформленных моделей неизбежно в процессе разработки, пока не все детали системы прояснились в полной мере. Правила языка UML побуждают - хотя не требуют - в ходе работы над моделью решать наиболее важные вопросы анализа, проектирования и реализации, в результате чего модель со временем становится хорошо оформленной.

1. **Практическая часть**

Самая первая диаграмма предметной области – диаграмма прецедентов. Первоначальная постановка задачи диаграммы прецедентов представлена на рисунке 1 и состоит из: системы выдачи ссуды под залог (подсистема), агента и клиент (актёры). "Агент" и " Клиент" являются внешними актёрами в системе выдачи ссуды под залог. Агент и Клиент связаны связью с «Выдача денег под залог» связью «один ко многим», так как один агент может выдать много ссуд и один клиент может заложить множество ценностей, но одина ссуда не может быть у многих Агентов или Клиентов.

