Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ

УНИВЕРСИТЕТ»

**Отчёт**

по предмету «Проектирование интернет систем»

Лабораторная работа №4

«Объектно-ориентированное моделирование. UML – диаграммы поведения»

Студент: Фамилия И. О.

ФИТ 4 курс 6 группа

Преподаватель: Якубенко К. Д.

Минск 2024

1. **Цель лабораторной работы**

Цель лабораторной работы по теме "Объектно-ориентированное моделирование. UML-диаграммы поведения" заключается в углубленном изучении методологии объектно-ориентированного моделирования (ООМ) с использованием UML (Unified Modeling Language) для создания и анализа диаграмм поведения программных систем. Объектно-ориентированное моделирование представляет собой ключевой подход в программировании, направленный на описание и разработку сложных систем в виде взаимодействующих объектов, что позволяет создавать устойчивую, гибкую и масштабируемую архитектуру программных решений. Работа с UML-диаграммами поведения помогает лучше понять функциональные требования к системе и то, как различные компоненты взаимодействуют между собой.

Одной из задач лабораторной работы является ознакомление с основными принципами объектно-ориентированного проектирования программного обеспечения, такими как инкапсуляция, наследование и полиморфизм, которые лежат в основе объектно-ориентированного подхода. Эти принципы позволяют более эффективно управлять сложностью разработки программных систем, что способствует созданию структурированных и понятных проектов. В ходе работы изучаются методы проектирования системы в виде объектов и их взаимосвязей, что играет важную роль в современном программировании.

В ходе лабораторной работы внимание уделяется моделированию динамики поведения объектов и их взаимодействия. Рассматривается, как объекты системы обмениваются сообщениями, изменяют свое состояние и реагируют на события. Описание динамических взаимодействий позволяет лучше понять, как поведение объектов изменяется во времени, что способствует более точному отражению требований к системе и ее адаптации к будущим изменениям.

Еще одной целью работы является овладение навыками анализа и корректировки UML-моделей для обеспечения их соответствия реальным требованиям информационных систем. UML позволяет анализировать как статическую, так и динамическую структуру системы, выявляя возможные ошибки на этапе проектирования. В процессе работы формируется понимание того, как выявлять узкие места в системе, проверять корректность поведения объектов и их взаимодействий, а также предлагать возможные улучшения и оптимизации.

Кроме того, выполнение лабораторной работы направлено на развитие практических навыков использования UML как универсального инструмента для документирования и анализа информационных систем. Осваиваются программные средства и инструменты, позволяющие автоматизировать процесс создания UML-диаграмм и интегрировать UML в процесс разработки программного обеспечения. Это необходимо как для проектирования новых систем, так и для анализа и поддержки существующих решений, что делает UML важным инструментом в разработке программного обеспечения.

1. **Описание функциональных требований**

Функциональные требования к проекту «Платформа для обучения и контроля знаний по математике» можно разделить по ролям: администратор, репетитор и ученик. Каждая роль имеет свои специфические задачи и доступ к различным функциям системы.

Функционально web-приложение должно:

* обеспечивать возможность авторизации и аутентификации;
* поддерживать роли администратора, репетитора и ученика;
* предоставлять администратору возможность управления пользователями;
* предоставлять администратору возможность управления контентом, включая учебные материалы, курсы, тесты;
* предоставлять репетитору возможность регистрации учеников;
* предоставлять репетитору возможность изменения информации об учениках;
* предоставлять возможность интеграции с Google календарем;
* предоставлять репетитору возможность управления учебными курсами и тестами;
* предоставлять репетитору возможность создания интерактивных математических моделей, включая 2D и 3D модели;
* предоставлять репетитору возможность отслеживания результатов учеников;
* предоставлять возможность ученикам просматривать учебные материалы;
* предоставлять возможность ученикам проходить учебные тесты;
* предоставлять возможность ученикам отслеживать прогресс в личном кабинете.

Функциональные требования платформы для обучения и контроля знаний по математике отражают ключевые аспекты работы системы, направленные на удовлетворение потребностей всех пользователей. Разделение на роли – преподаватель, ученик и администратор – позволяет гибко управлять процессом обучения, предоставляя каждому пользователю необходимый набор инструментов для выполнения их задач. Преподаватели получают мощные средства для создания курсов и тестов, ученики могут взаимодействовать с материалами и отслеживать свой прогресс, а администратор обеспечивает контроль за системой и её поддержкой. Такой подход обеспечивает целостность и функциональность платформы, делая её удобной и эффективной в использовании.

1. **Описание программных средств**

Draw.io (или diagrams.net) — это мощный и удобный инструмент для создания разнообразных диаграмм и схем, который завоевал популярность благодаря своей функциональности и бесплатной основе. Одной из главных особенностей платформы является простота в использовании, что делает её доступной как для новичков, так и для профессионалов. Благодаря интуитивно понятному интерфейсу, пользователи могут быстро создавать различные типы диаграмм, такие как блок-схемы, диаграммы UML, сетевые диаграммы, карты процессов, модели IDEF0 и организационные структуры.

Инструмент также предлагает обширную библиотеку готовых шаблонов и элементов для создания визуализаций. Пользователи могут выбирать из множества стандартных фигур и иконок, что значительно ускоряет процесс построения диаграмм.

Платформа поддерживает множество форматов для экспорта, включая PNG, JPG, SVG и PDF, что делает ее универсальной для различных задач, таких как представление проектов, печать или размещение в электронных документах. Более того, для будущего редактирования диаграммы можно сохранить в собственном формате XML, что позволяет пользователям возвращаться к проектам в любое время для их доработки. Draw.io также интегрируется с популярными облачными сервисами, такими как Google Drive, Dropbox и OneDrive, что упрощает процесс хранения, редактирования и обмена диаграммами, а также обеспечивает доступ к проектам с любого устройства и совместную работу в реальном времени.

Для командной работы draw.io предлагает инструменты для совместного редактирования диаграмм, позволяя нескольким пользователям одновременно вносить изменения в схемы и делиться комментариями. Дополнительным плюсом является возможность работы в оффлайн-режиме через настольное приложение, что позволяет создавать и редактировать схемы даже без доступа к интернету.

Платформа также предлагает интеграцию с различными корпоративными системами и популярными платформами для управления проектами, такими как Atlassian Confluence и Jira. Это позволяет использовать draw.io для визуализации данных и процессов в рамках более широких решений для управления проектами. Для пользователей, которые уже работают с другими инструментами для диаграмм, draw.io предлагает импорт и экспорт данных, что делает его совместимым с такими сервисами, как Lucidchart и Microsoft Visio, позволяя легко переходить между различными инструментами.

Таким образом, draw.io сочетает в себе простоту использования, гибкость работы с различными форматами и возможность интеграции с облачными сервисами, что делает его одним из самых популярных инструментов для создания диаграмм и визуализации данных. Платформа является идеальным решением как для индивидуальных пользователей, так и для команд, работающих над совместными проектами, предлагая все необходимые инструменты для эффективной работы.

1. **Описание практического задания**

В рамках данного практического задания необходимо разработать проектируемую подсистему для образовательной платформы. Подсистема предназначена для управления учебными курсами, тестами и взаимодействия пользователей с этими компонентами. В платформе выделяются несколько типов пользователей (роли), каждый из которых имеет определённые права и обязанности. Для реализации системы важно учесть права доступа, иерархию пользователей и их взаимодействие с ключевыми функциями системы.

Основной целью является создание удобного интерфейса для администраторов, преподавателей и студентов, где каждый пользователь сможет выполнять свои задачи в соответствии с ролью. Взаимодействие между пользователями и системой должно быть логичным и безопасным, с учётом прав доступа и функциональных обязанностей каждого пользователя.

В проектируемой системе образовательной платформы выделяются три ключевые роли: администратор, репетитор и ученик.

В системе образовательной платформы роли администратора, репетитора и ученика имеют четко разграниченные функции и права, направленные на эффективное управление учебным процессом.

Администратор (admin) — это пользователь, который управляет всей платформой. Его основная роль заключается в контроле и поддержании функционирования системы. Администратор занимается созданием учетных записей для других пользователей, распределяет роли, управляет правами доступа, создаёт и редактирует учебные курсы и тесты, а также следит за тем, чтобы образовательный процесс на платформе проходил бесперебойно. Помимо этого, администратор может просматривать статистику успеваемости всех учеников, обеспечивать поддержку пользователям и решать технические вопросы.

Репетитор (tutor) — это педагогический пользователь платформы, который отвечает за взаимодействие с учениками и проведение учебного процесса. Репетитор создает и редактирует учебные материалы, подготавливает задания, курирует выполнение тестов, проводит онлайн-занятия и оценивает успеваемость учеников. Он следит за прогрессом каждого ученика, дает обратную связь и помогает с разбором трудных тем. Репетитор имеет доступ к курсам и материалам, которые назначены ученикам, и может вносить в них изменения для улучшения процесса обучения.

Ученик (student) — это пользователь, который проходит обучение на платформе. Его основная задача — изучение материалов, выполнение заданий и прохождение тестов. Ученик может просматривать назначенные ему курсы, выполнять задания и получать обратную связь от репетитора. Его успех и прогресс отслеживаются на протяжении всего учебного процесса, и результаты фиксируются для анализа успеваемости.

Гость (guest) — это неавторизованный пользователь, который может зарегистрироваться в роли ученика и авторизоваться.

Диаграмма прецедентов представлена на рисунке 4.1.

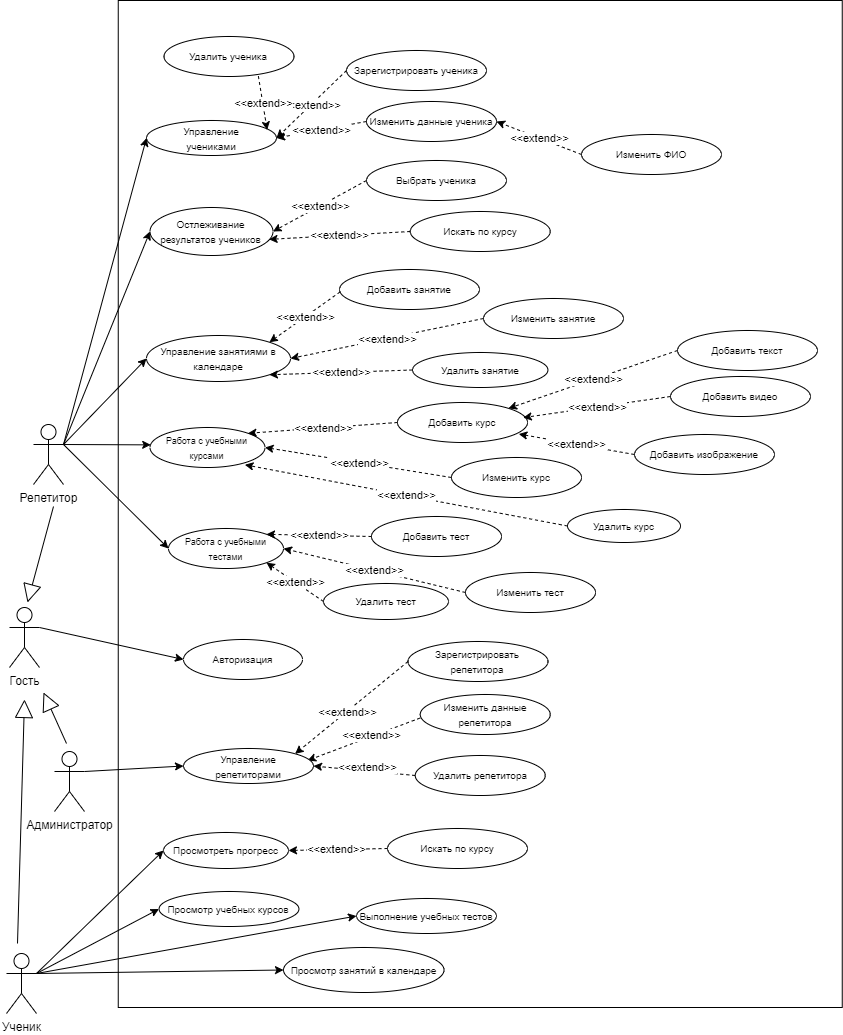


Рисунок 4.1 – Диаграмма прецедентов

В ходе лабораторной работы было уделено внимание знакомству с различными видами UML-диаграмм, такими как диаграммы классов, последовательности, активности, прецедентов и другие. Работа с CASE-средствами, такими как Rational Rose, Enterprise Architect или Visual Paradigm, позволила увидеть, как современные инструменты могут автоматизировать и структурировать процесс проектирования. CASE-средства ускоряют создание диаграмм, помогают избежать ошибок на этапе моделирования и упрощают документирование сложных систем. Эти инструменты позволяют вносить изменения в проект, пересчитывать зависимости и проверять корректность моделей, что особенно важно при разработке масштабных и многосоставных программных систем.

UML стал стандартом в сфере проектирования программного обеспечения благодаря своей гибкости и четкой структурированности. При использовании UML дизайнеры и аналитики могут представить архитектуру системы визуально, упростив коммуникацию между командой разработки и заказчиком. Использование UML помогает в разработке понятных и легко поддерживаемых моделей, что снижает вероятность недоразумений на стадии реализации и тестирования. Этот опыт показал, насколько важны методы UML для формализации требований и упрощения обсуждения функционала.

Лабораторная работа подчеркнула важность документирования решений на разных этапах разработки. Используя UML-диаграммы, можно фиксировать и отслеживать структуру и поведение системы, что помогает при передаче проекта новым разработчикам или при масштабировании. Грамотно оформленные диаграммы являются ценным артефактом в процессе разработки и технической поддержки, а также способствуют поддержанию проекта в актуальном состоянии в течение его жизненного цикла.

Разработка диаграммы прецедентов (use case diagram) была важной частью работы, так как она помогла сконцентрироваться на пользовательских сценариях и лучше понять, как взаимодействовать с системой. Создание диаграммы прецедентов позволяет определить основные функции системы и выделить ключевые роли пользователей. Этот процесс помогает формировать функциональные требования, ориентируясь на реальные потребности пользователей, что критично для успеха проекта. Такая ориентация на пользователей способствует адаптации системы к их ожиданиям и делает разработку более гибкой и целенаправленной.

Построение UML-диаграмм, включая диаграммы классов и последовательностей, способствовало выявлению функциональных требований и их взаимосвязи с архитектурными решениями. Эти диаграммы позволяют рассмотреть детали внутренней логики системы, определить ключевые объекты и взаимосвязи между ними. Подход к анализу функциональных требований на основе UML помогает разработчику лучше понять структуру системы, выявить потенциальные проблемы и заранее найти способы их решения.

1. **Ответы на вопросы**

1. Дайте описание понятиям Unified process (UP) и UML.

Unified Process (UP) — это методология разработки программного обеспечения, которая предполагает поэтапное выполнение работ через фазы: инициация, проработка, построение и внедрение. Каждый цикл направлен на улучшение системы. UML (Unified Modeling Language) — это стандартный язык визуального моделирования, используемый для описания как структуры, так и поведения программных систем, что упрощает их проектирование, анализ и документирование.

2. Перечислите основные диаграммы UML 2.0.

UML 2.0 включает множество диаграмм, каждая из которых предназначена для описания различных аспектов системы, таких как её структура, поведение и взаимодействие. Среди них: диаграммы классов, объектов, вариантов использования, последовательностей, состояний, и другие, каждая из которых играет роль в решении конкретных задач моделирования системы.

3. Назовите CASE-средства, поддерживающие создание UML диаграмм.

Для создания UML-диаграмм используются различные CASE-инструменты, такие как Enterprise Architect, IBM Rational Rose, Visual Paradigm и другие. Эти средства автоматизируют процесс проектирования, поддерживая моделирование и документирование программного обеспечения.

4. Укажите назначение диаграммы вариантов использования.

Диаграмма вариантов использования (Use Case Diagram) предназначена для отображения взаимодействия пользователей с системой. Она помогает определить функциональные требования, описывая задачи, которые пользователи могут выполнять в системе, и способы их достижения посредством взаимодействий с системой.

5. Опишите нотации, которые используются для построения Use-Case диаграммы.

Для создания диаграммы вариантов использования применяются следующие нотации: акторы (представляющие пользователей или внешние системы) изображаются в виде человечков, а варианты использования — овальными фигурами. Связи между ними показываются линиями взаимодействия. Дополнительно используются нотации для обобщения акторов, включения обязательных действий и расширения для альтернативных сценариев.