Учреждение образования

«Белорусский государственный технологический университет»

Проектирование информационных систем

Лабораторная работа №5

**«ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ МОДЕЛИРОВАНИЕ.**

**СТРУКТУРНЫЕ ДИАГРАММЫ UML»**

Выполнил:

Минск 2021

Оглавление

[Цель 3](#_Toc89299281)

[Теоретические вопросы 3](#_Toc89299282)

[1 Постановка задачи 5](#_Toc89299283)

[2 Описание программных средств 7](#_Toc89299284)

[3 Описание практического задания 8](#_Toc89299285)

Цель: Изучение методологии объектно-ориентированного моделирования средствами UML. Ознакомление с основными принципами объектно-ориентированного проектирования программного обеспечения, получение навыков проектирования структуры информационной системы с применением UML.

Теоретические вопросы:

1. Перечислите структурные диаграммы, которые входят в UML 2.0.

Диаграмма классов, диаграмма компонентов, диаграмма композитной структуры, диаграмма кооперации, диаграмма развёртывания, диаграмма объектов, диаграмма пакетов.

1. Укажите назначение структурных диаграммы.

Диаграммы для визуализации, специфицирования, конструирования и документирования статических аспектов системы.

1. Опишите нотации, которые используются для построения Classes диаграмм.

Класс – это описание набора объектов с одинаковыми атрибутами, операциями.

Атрибуты (свойства) описывают содержимое класса.

Операции (методы) представляют собой некоторый сервис, предоставляемый каждым экземпляром класса.

Зависимость – семантически представляет собой связь между двумя элементами модели, в которой изменение одного элемента (независимого) может привести к изменению семантики другого элемента (зависимого).

Ассоциация – это структурная связь между элементами модели, которая описывает набор связей, существующих между объектами.

Агрегация – особая разновидность ассоциации, представляющая структурную связь целого с его частями. Как тип ассоциации, агрегация может быть именованной. Одно отношение агрегации не может включать более двух классов (контейнер и содержимое).

Наследование – связь, в которой специализированный элемент (потомок) строится по спецификациям обобщенного элемента (родителя).

Реализация – это семантическая связь между классами, когда один из них (поставщик) определяет соглашение, которого второй (клиент) обязан придерживаться.

1. Для чего применяются расширения диаграмм UML?

Для уточнения или специализации общих элементов метамодели при разработке конкретных элементов метамодели при разработке конкретных моделей UML

1. Что означают понятия «стереотип» и «тегированное значение» в контексте расширенных диаграмм?

Стереотипы являются одним из трех типов механизмов расширяемости в унифицированном языке моделирования. Они позволяют проектировщикам расширять словарь UML для создания новых элементов моделирования, получаемых из существующих, но имеющих определенные свойства, которые подходят для конкретной проблемы предметной области или для другого специализированного использования. Тегированное значение – часть стандартного UML-элемента, содержащая о нём дополнительную информацию.

# Постановка задачи

Система цифровизации учебного процесса. В приложении выделено 10 ролей: администратор, декан, лектор, преподаватель, куратор группы, методист, секретарь, студент, родитель студента, староста. Функциональные требования отображены в таблице 1.1.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Роли | Администратор | Декан | Лектор | Преподаватель | Секретарь | Методист | Студент | Староста | Родитель студента |
| Функционал |
| Просмотр расписания | + | + | + | + | + | + | + |  | + |
| Добавление, изменение, удаление расписания | + | + |  |  | + |  |  |  |  |
| Просмотр успеваемости | + | + | + | + | + |  | + |  | + |
| Добавление, изменение, удаление успеваемости | + | + | + | + |  |  |  |  |  |
| Просмотр посещения | + | + | + | + | + |  | + |  | + |
| Добавление, изменение, удаление посещения | + |  | + | + |  |  |  | + |  |
| Просмотр объявлений | + | + | + | + | + |  | + |  | + |
| Изменение статуса объявления | + | + | + | + | + |  | + |  | + |
| Добавление, изменение, удаление объявления | + | + | + | + | + |  |  |  |  |
| Просмотр заявлений | + | + |  |  | + |  | + |  |  |
| Добавление, изменение, удаление заявлений | + |  |  |  |  |  | + |  |  |
| Подтверждение/отклонение заявлений | + | + |  |  | + |  |  |  |  |
| Прикрепление комментариев к заявлению | + | + |  |  | + |  | + |  |  |
| Обновление информации о пользователе | + | + |  |  | + |  |  |  |  |
| Добавление, изменение, удаление учебного плана |  |  |  |  |  | + |  |  |  |
| Просмотр учебного плана | + | + | + | + | + | + | + |  | + |
| Добавление комментария к занятию | + | + | + | + |  |  |  |  |  |
| Получение уведомлений | + | + | + | + | + |  | + |  | + |
| Просмотр личного кабинета | + | + | + | + | + |  | + |  |  |

Таблица 1.1 – Функциональные требования

# Описание программных средств

Построение моделей выполнялось в программной среде Microsoft Visio 2016.

Версия: 2016.

Дата публикации: 22.02.2016.

Разработчик: Microsoft.

Адрес загрузки:

<https://www.microsoft.com/ru-RU/download/details.aspx?id=51188>

Поддерживаемая операционная система: Windows 10, Windows 7 Service Pack 1, Windows 8.1.

Для создания диаграммы поведения используется схема «Класс UML». В ней представлены 7 типов фигур (занятие, член, интерфейс, перечисление, пакет (развернутый), пакет (свернутый), заметка) и 6 видов соединителей (ассоциация, наследование, агрегат, композиция, зависимость, реализация интерфейса).

# Описание практического задания

Выделим следующие классы в системе:

* Mark;
* Lesson;
* Student;
* MarkRepository;
* MarkController.

Между классом Mark и Lesson связь композиция. Один Lesson может содержать от 0 до неопределенного количества Mark. При удалении Lesson класс Mark не может существовать отдельно.

Между классом Mark и Student связь композиция. Один Student может содержать от 0 до неопределенного количества Mark. При удалении Student класс Mark не может существовать отдельно.

Между классом Mark и MarkRepository зависимость. Изменение класса Mark может повлечь за собой изменение MarkRepository.

Между классом MarkRepository и MarkController зависимость. Изменение класса MarkRepository может повлечь за собой изменение MarkController.

В ходе выполнения работы была разработана UML диаграмма классов (рис. 3.1), определены классы и взаимоотношения между ними, указана кратность.

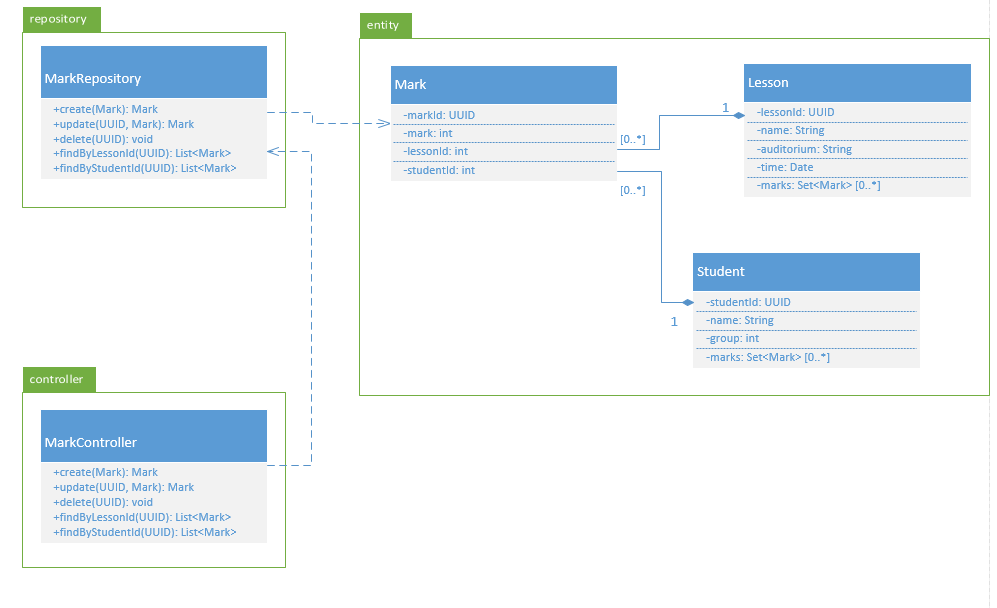


Рисунок 3.1 – Диаграмма классов управления успеваемостью