Государственное бюджетное профессиональное образовательное учреждение Республики Марий Эл

«ЙОШКАР-ОЛИНСКИЙ ТЕХНОЛОГИЧЕСКИЙ КОЛЛЕДЖ»

|  |
| --- |
| Специальность: 09.02.07 Информационные системы и программирование  Группа: А-31  Дисциплина: Технология разработки ПО |

ОТЧЕТ ПО

«Диаграмма прецедентов»

Руководитель: Пинешкин Ю.С

Выполнил(а): студентки А-31

Николаева С.В, Мочалова К.Е.

Йошкар-Ола

2025

**Предметная область**

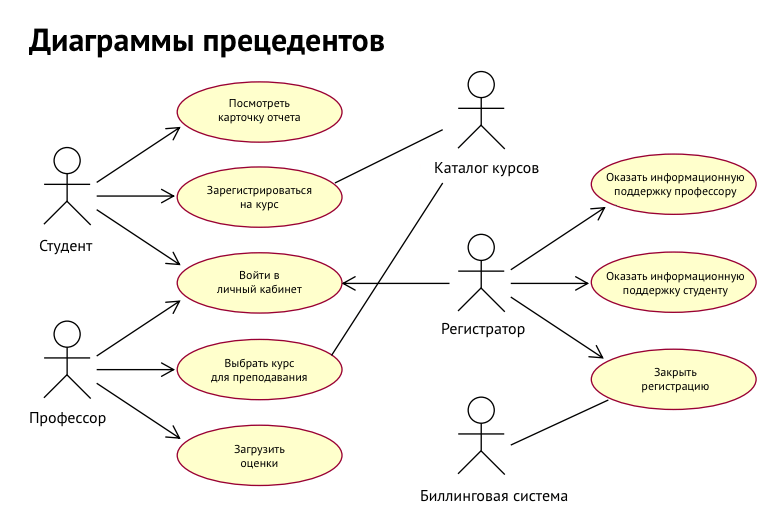
**Диаграмма прецедентов** (диаграмма вариантов использования) — это инструмент, который позволяет изобразить графически варианты использования системы и её пользователей. С его помощью можно описать основные действующие элементы системы и процессы, которые в ней протекают.

**Предметная область** — это часть реального мира, данные и особенности которой будут отражены в разрабатываемом программном решении. Например, в качестве предметной области можно выбрать бухгалтерию какого-либо предприятия, отдел кадров, банк, магазин и т. д.

**Основное назначение диаграммы** — описание функциональности и поведения, которое позволяет заказчику, конечному пользователю и разработчику совместно обсуждать проектируемую или существующую систему.

**Некоторые элементы диаграммы прецедентов**:

* **Рамки системы** — прямоугольник с названием в верхней части и эллипсами (прецедентами) внутри.
* **Актор** — стилизованный человечек, который обозначает набор ролей пользователя, взаимодействующего с некоторой сущностью (системой, подсистемой, классом).
* **Прецедент** — эллипс с надписью, который обозначает выполняемые системой действия, приводящие к наблюдаемым акторами результатам.

Пример диаграммы прецедентов



Пример моей диаграммы прецедентов

**Вывод:** Диаграмма прецедентов (диаграмма вариантов использования) применяется для описания функциональности и поведения системы. Она изображает, как пользователь взаимодействует с системой, объясняет, кому какие функции доступны и как при этом отрабатывает система.

Некоторые области применения диаграммы прецедентов:

* Определение сценариев использования системы. Это помогает выявить потенциальные проблемы взаимодействия между пользователями и системой.
* Проверка полноты требований к системе. Диаграмма позволяет идентифицировать пробелы и пропуски в функциональности, которые могут быть упущены при других методах анализа требований.
* Выявление других видов информации, необходимой для работы над проектом. Например, ограничений, атрибутов качества, требований к пользовательскому интерфейсу и внутренних правил работы в предметной области.
* Формирование подробной документации по продукту. На основе диаграммы прецедентов составляется полная и подробная документация проекта, которую можно удобно разложить для работы всей команды.

|  |  |
| --- | --- |
| **Плюсы** | **Минусы** |
| * Понимание поведения системы. Диаграммы помогают заинтересованным сторонам, в том числе не имеющим технических знаний, понять общую работу системы. * Прозрачность согласования работ с заказчиком. Каждый use case несёт конечную бизнес-ценность, понятную заказчику. * Проверка полноты требований к системе. Диаграммы позволяют идентифицировать пробелы и пропуски в функциональности, которые могут быть упущены при других методах анализа требований. * Выявление другой информации, необходимой для работы над проектом. Например, ограничений, атрибутов качества, требований к пользовательскому интерфейсу и внутренних правил работы в предметной области. * Экономия времени и ресурсов. Глядя на схему, разработчик понимает, что определённые элементы интерфейса и кода можно использовать повторно. | * Неэффективность в некоторых приложениях. Use case не будет эффективен, если основную роль в приложении играют внутренние правила взаимодействия объектов, а не действия пользователей. * Не покрытие нефункциональных требований. Например, скорости работы системы, безопасности, масштабируемости. * Громоздкость и сложность понимания. Для больших проектов use case может стать слишком громоздким и трудным для понимания. * Сложность поддержания актуальности. Из-за изменений в требованиях или системе часто приходится пересматривать и переписывать use case — это занимает много времени и сил. * Зависимость от опыта автора. Неопытный аналитик может упустить важные детали или неправильно понять требования — это приведёт к созданию некорректных сценариев. |