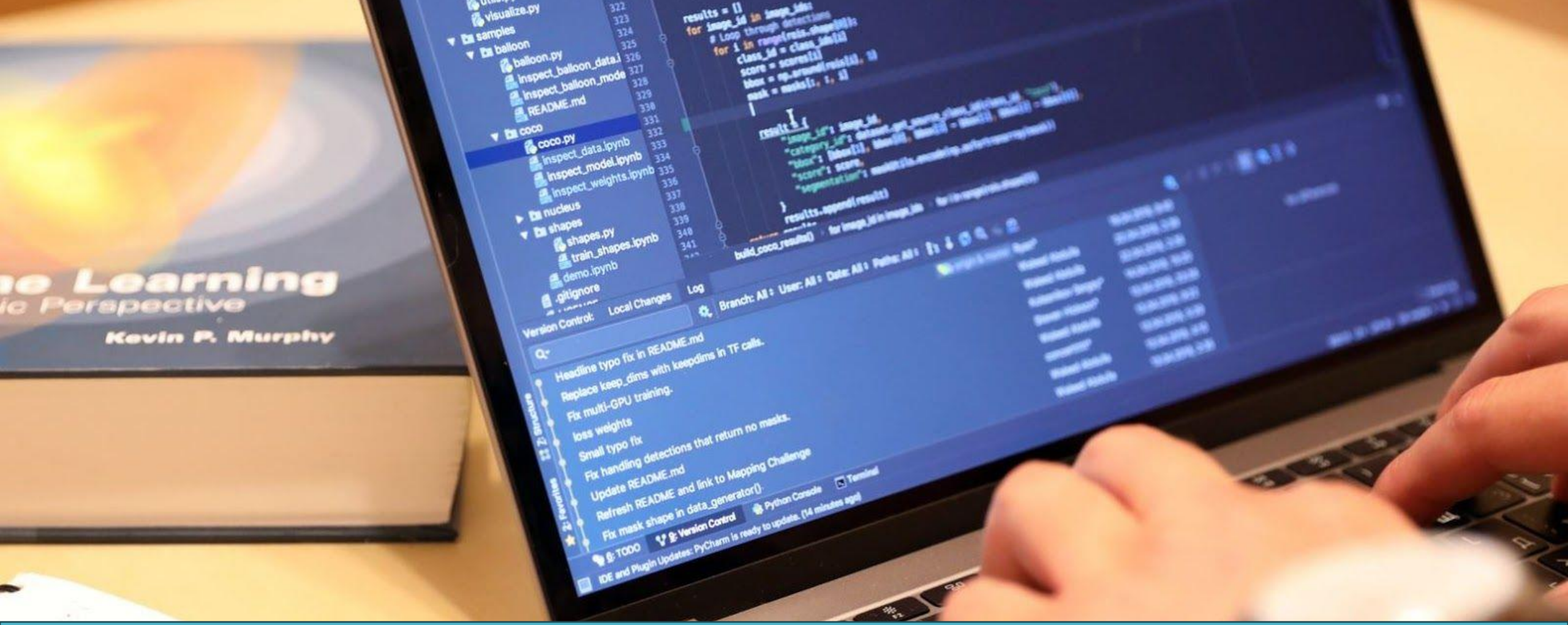
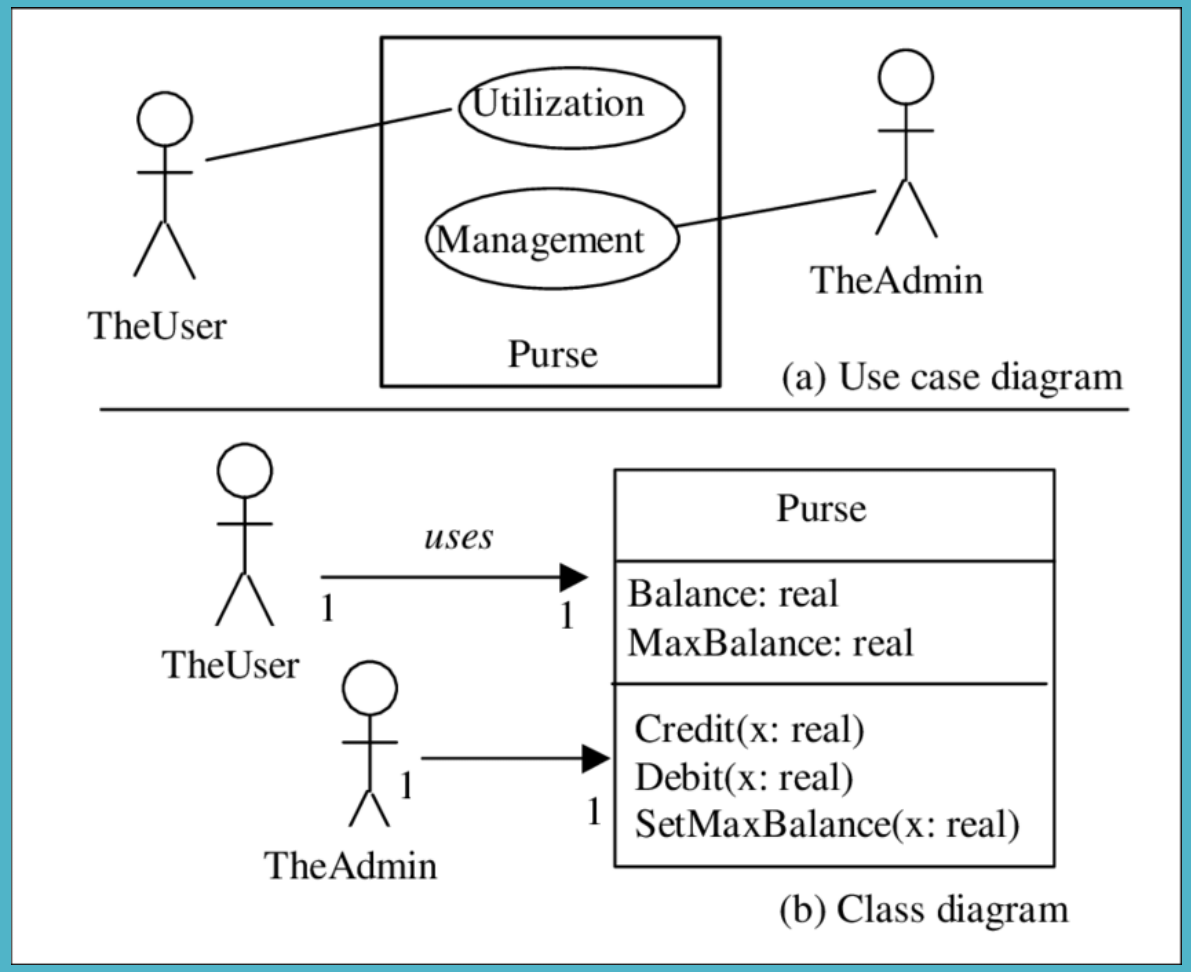


Тема 1: Введение в разработку ПО
Занятие 1.5. Диаграмма последовательности. Диаграмма классов. Разработка технического задания.

Цели занятия:

- Усвоить основные понятия UML.
- Понять, как использовать диаграммы UML для моделирования требований.



1. Диаграмма последовательности.

Диаграммы последовательности UML (Unified Modeling Language) являются одним из видов диаграмм, используемых для визуализации взаимодействия между объектами или компонентами в системе. Они позволяют показать последовательность сообщений и взаимодействий между различными участниками (объектами, компонентами, акторами) в рамках определенного сценария или функциональности.

Основной целью диаграммы последовательности является иллюстрация временного порядка выполнения операций и обмена сообщениями между объектами в системе. Она помогает визуально представить, как различные элементы взаимодействуют друг с другом и какой порядок действий следует.

Ключевые элементы, используемые в диаграммах последовательности:

1.Участники (Participants): Участники представляют объекты, компоненты или акторы, которые участвуют во взаимодействии. Они обычно представлены в виде прямоугольников с их именами сверху.

2.Линии жизни (Lifelines): Линии жизни представляют время существования участников во время выполнения сценария. Они обычно представлены вертикальными линиями, которые проходят через участников.

3.Сообщения (Messages): Сообщения показывают взаимодействие между участниками. Они указывают, какие операции или сообщения передаются между участниками. Сообщения могут быть синхронными (прямые стрелки), асинхронными (пунктирные стрелки) или возвращаемыми (стрелки с возвращаемым значением).

4.Фреймы (Frames): Фреймы используются для группировки связанных сообщений внутри диаграммы последовательности. Они помогают улучшить структуру и читаемость диаграммы.

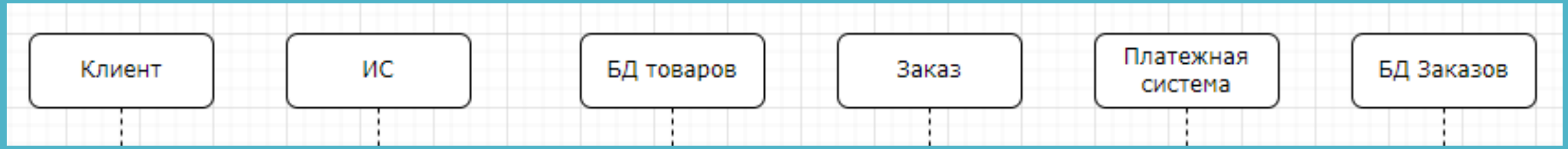
5.Операции и условия (Operations and Conditions): Диаграммы последовательности могут также содержать операции и условия, которые показывают логику выполнения и принятия решений в рамках сценария.

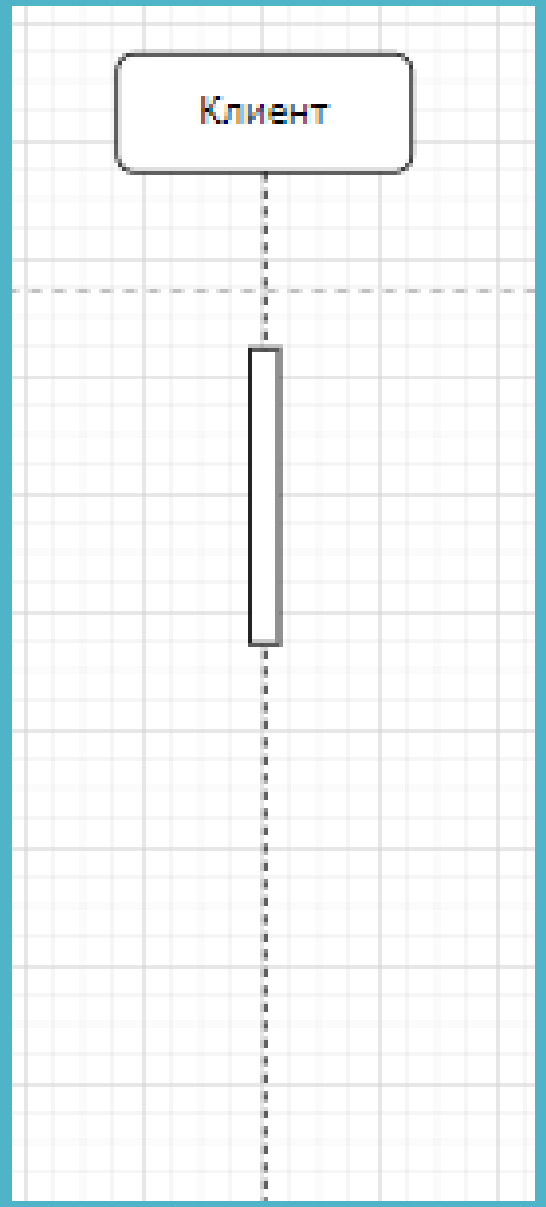
На горизонтальной оси показываются элементы, которые участвуют во взаимодействии. Располагаться они могут в любом порядке, но обычно объект, который участвует первым, расположен слева, за ним тот, что участвует вторым и т.д.

Вертикальная ось показывает время (прогресс), а точнее порядок выполнения.

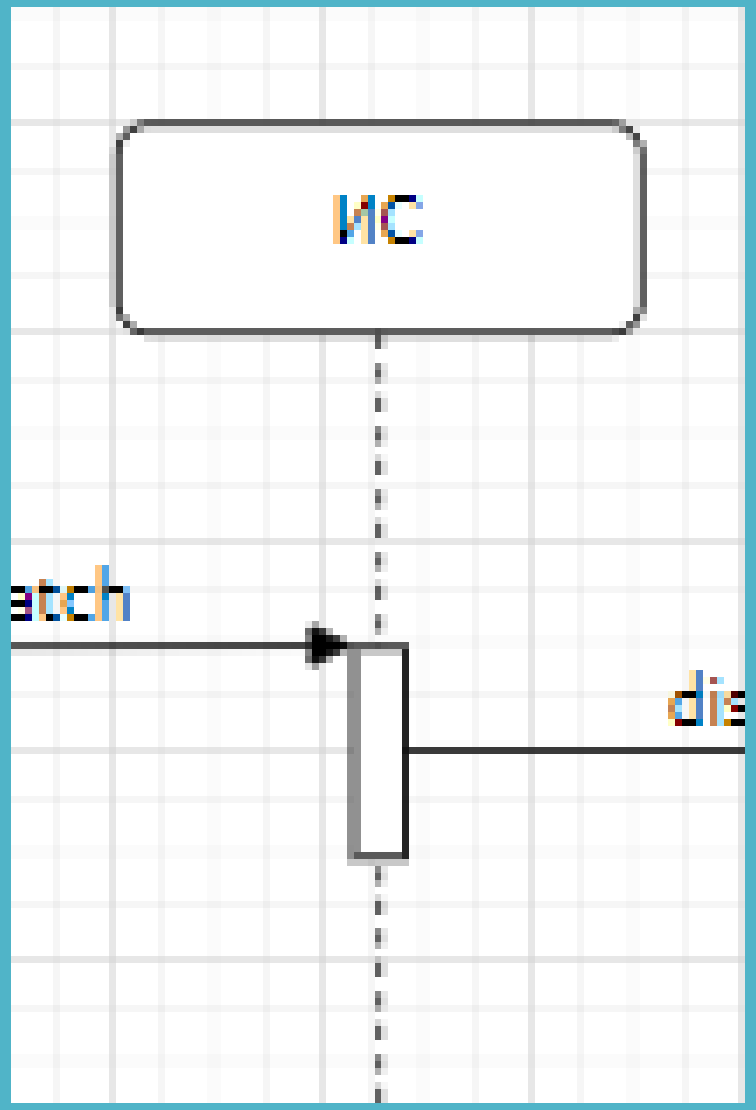
Актер – тип роли, которую выполняет экземпляр, взаимодействующий с субъектом.

Объекты, участвующие во взаимодействии.

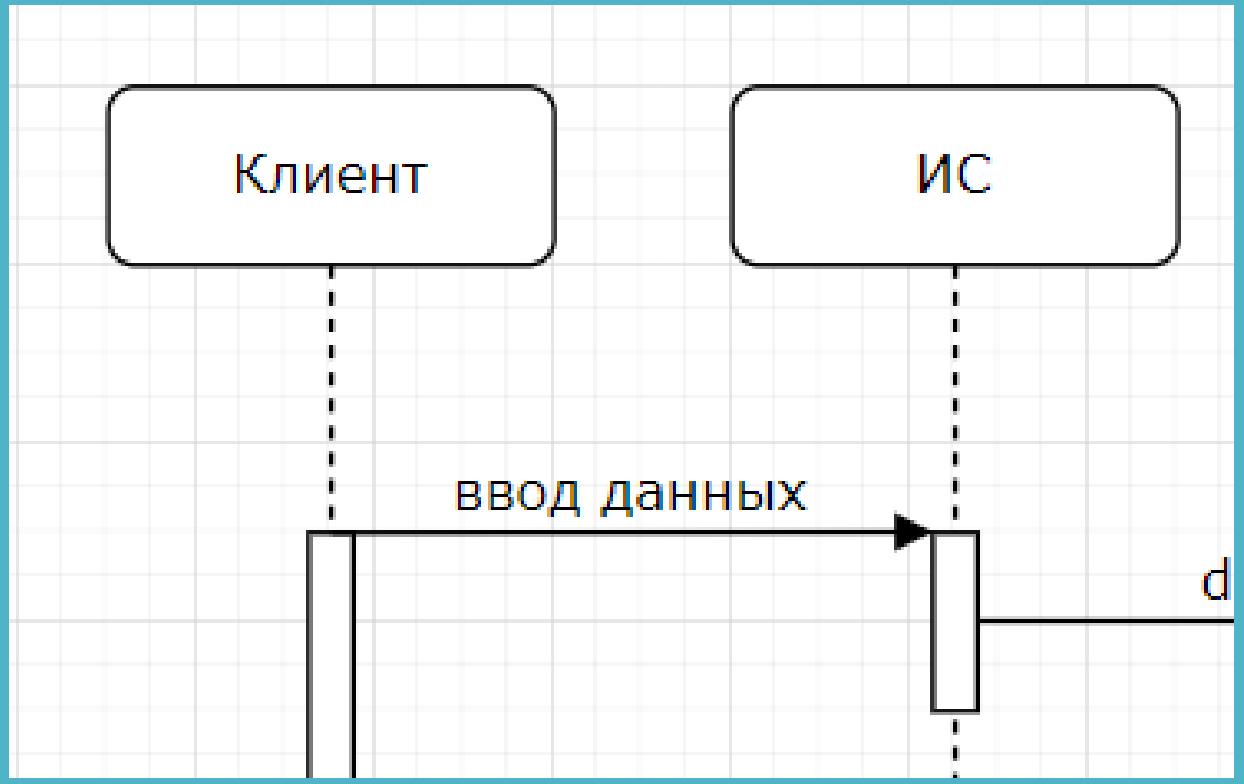




Линия жизни – отображает течение времени.

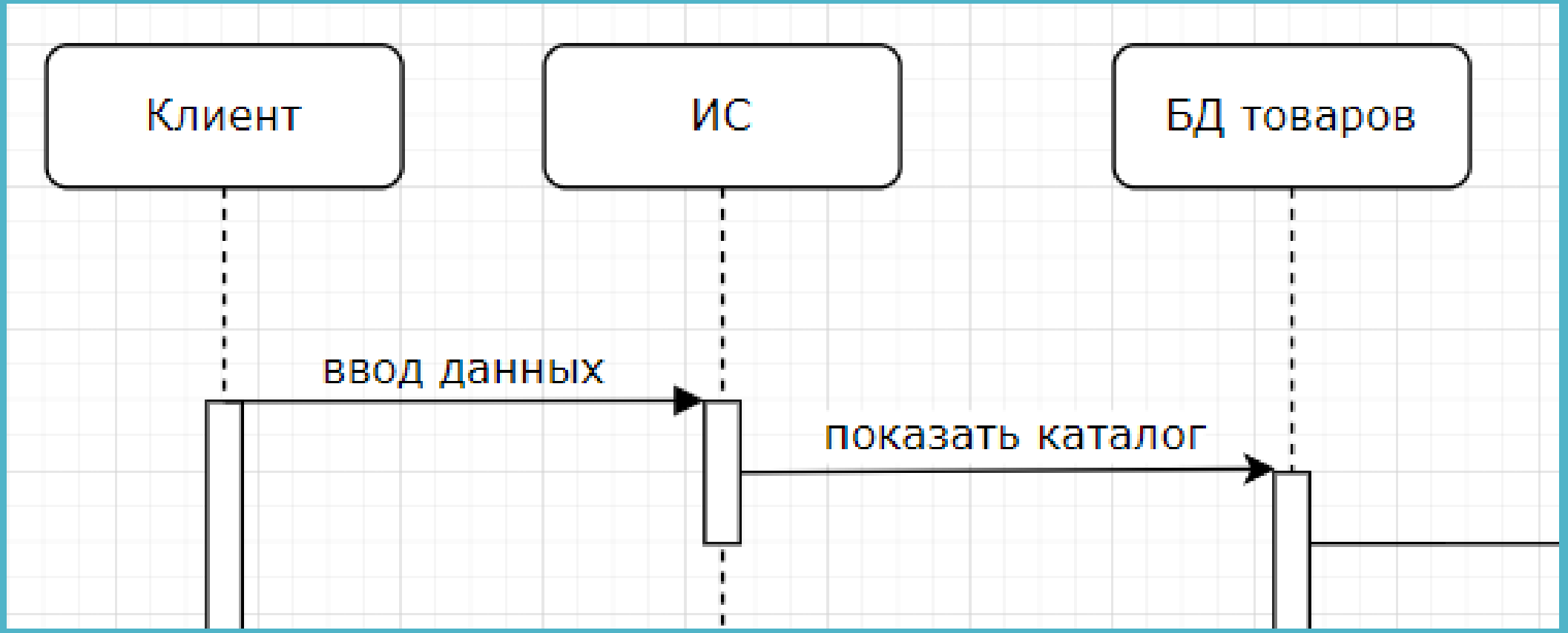


Активация – тонкий прямоугольник на линии жизни, показывает время, в течении которого элемент выполняет операцию.

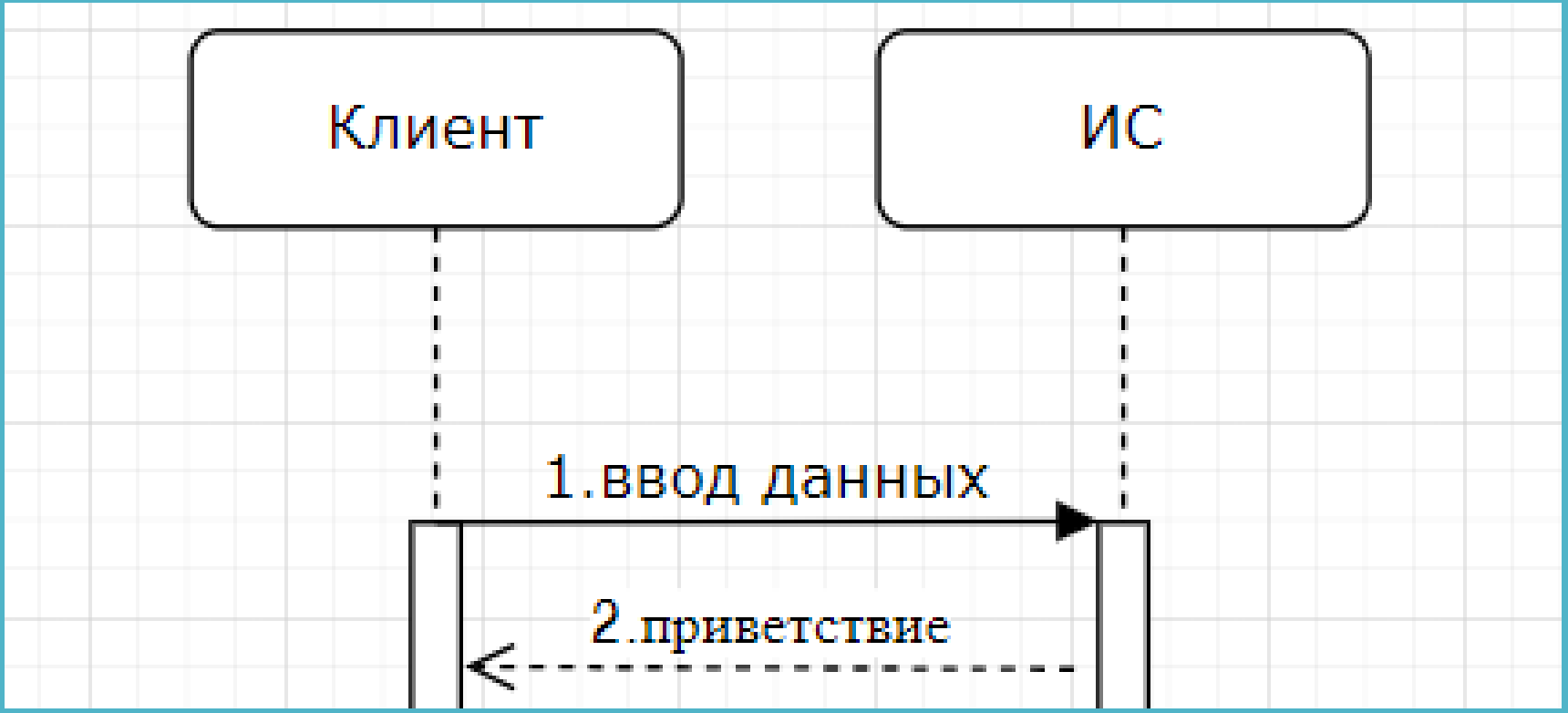


Вызов (сообщение) – используется для вызова процедур, выполнения операций. Начало стрелки всегда соприкасается с линией жизни того объекта, который инициирует сообщение. Кончик жизни соприкасается с линией жизни объекта, принимающего сообщение.

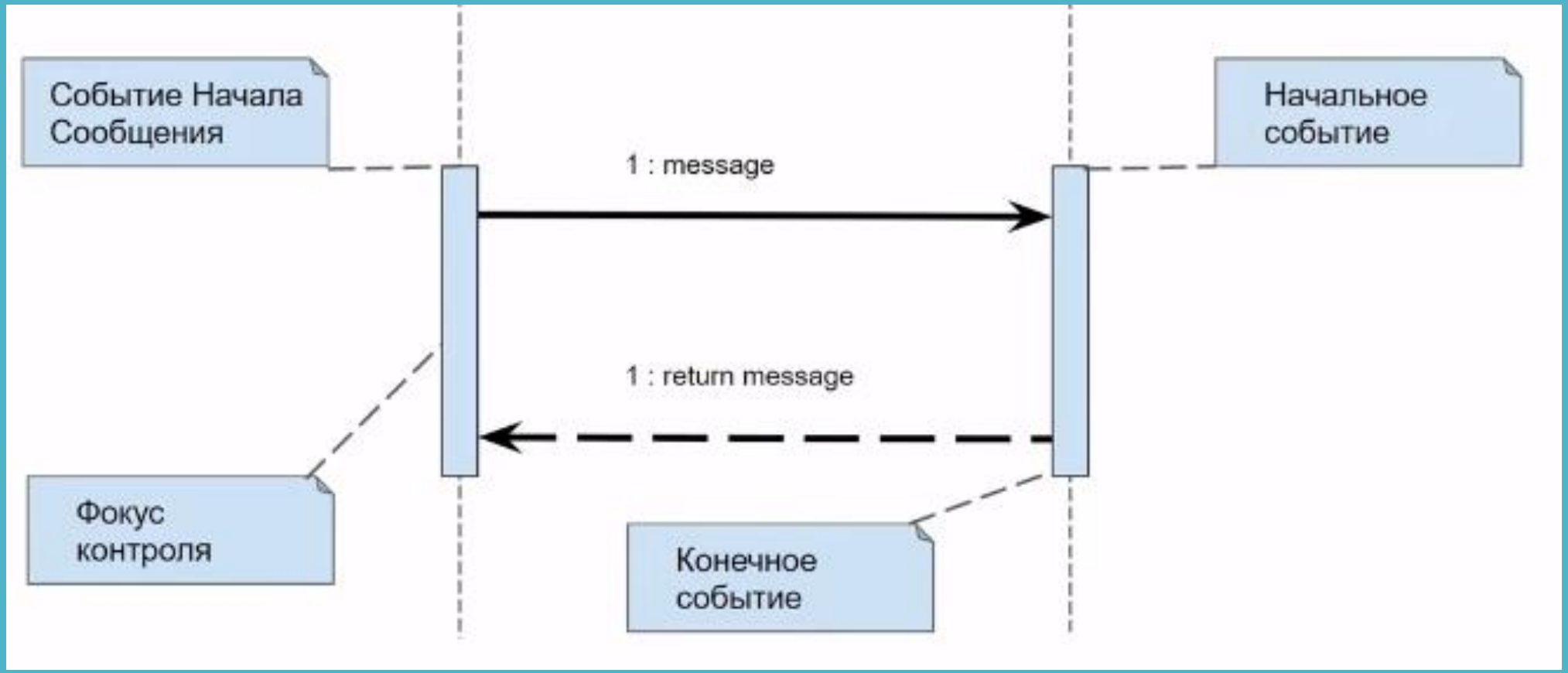
Сообщение может быть синхронным (стрелка закрашена) и асинхронным (обычная стрелка).

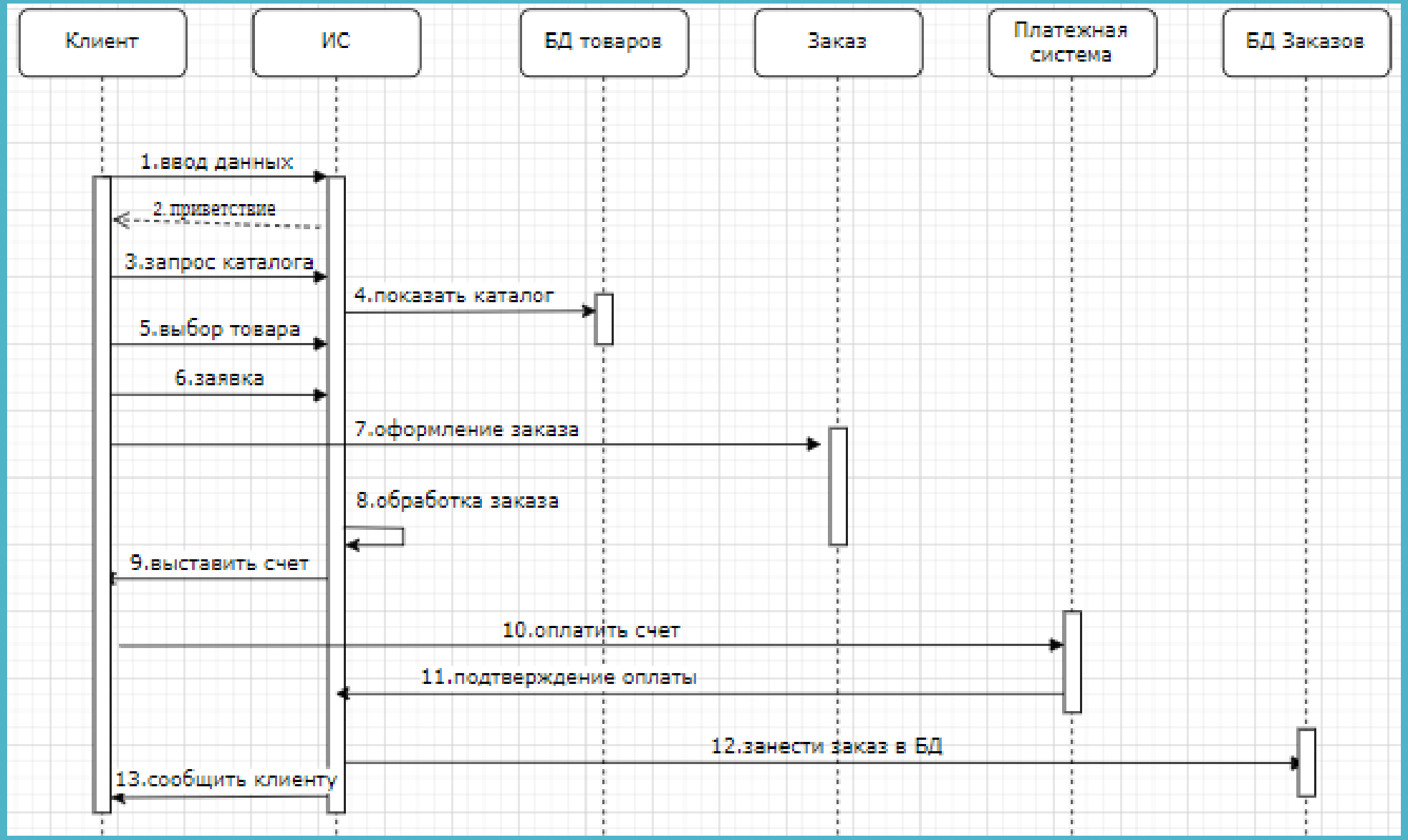


Ответное сообщение (возврат) – показывает передачу сообщения вызывающей стороне.



Фокус управления (фокус контроля) – любая точка взаимодействия, где что-то происходит. Высокий, тонкий прямоугольник на линии жизни. Период, в течение которого элемент выполняет операцию.

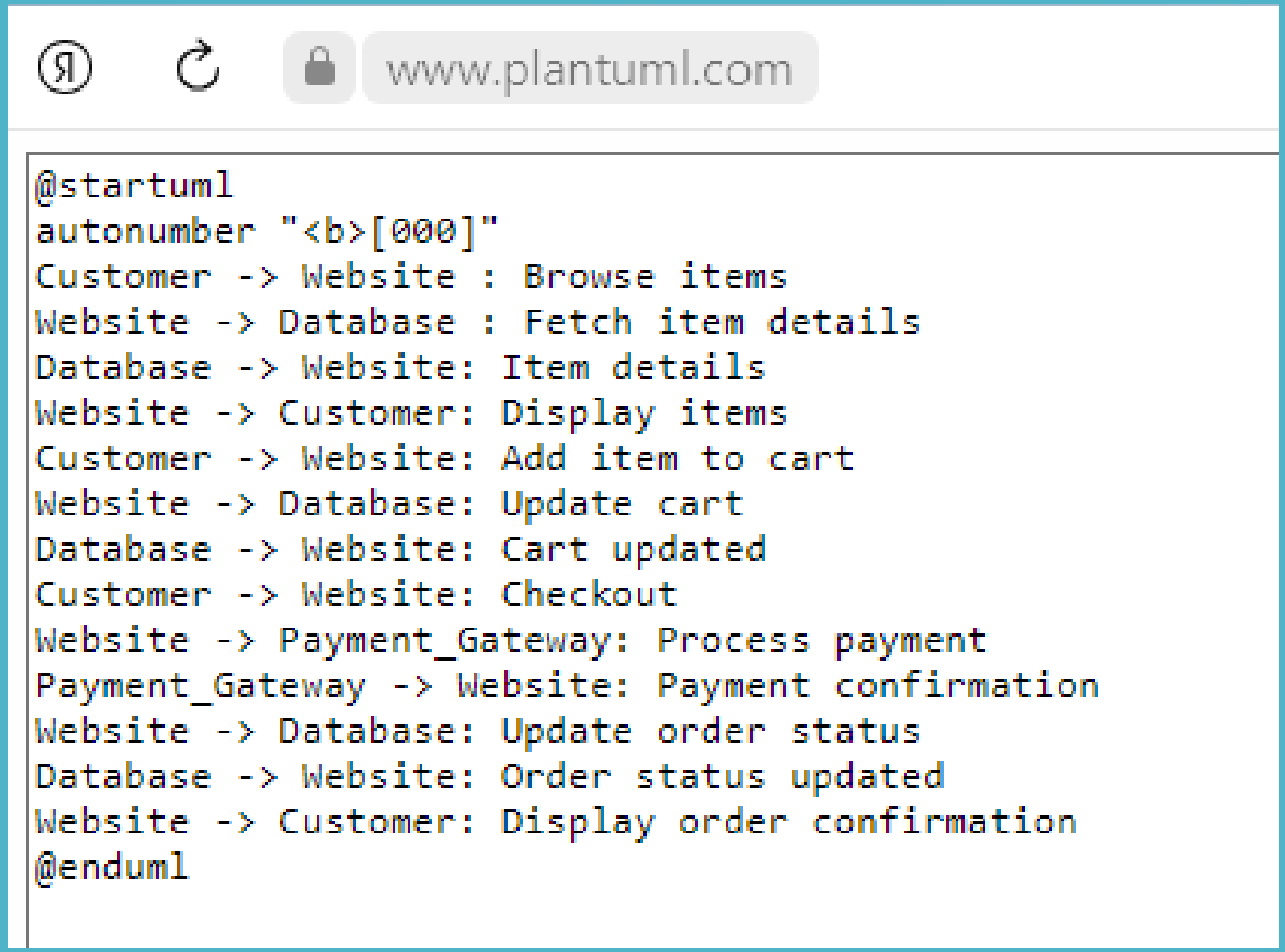


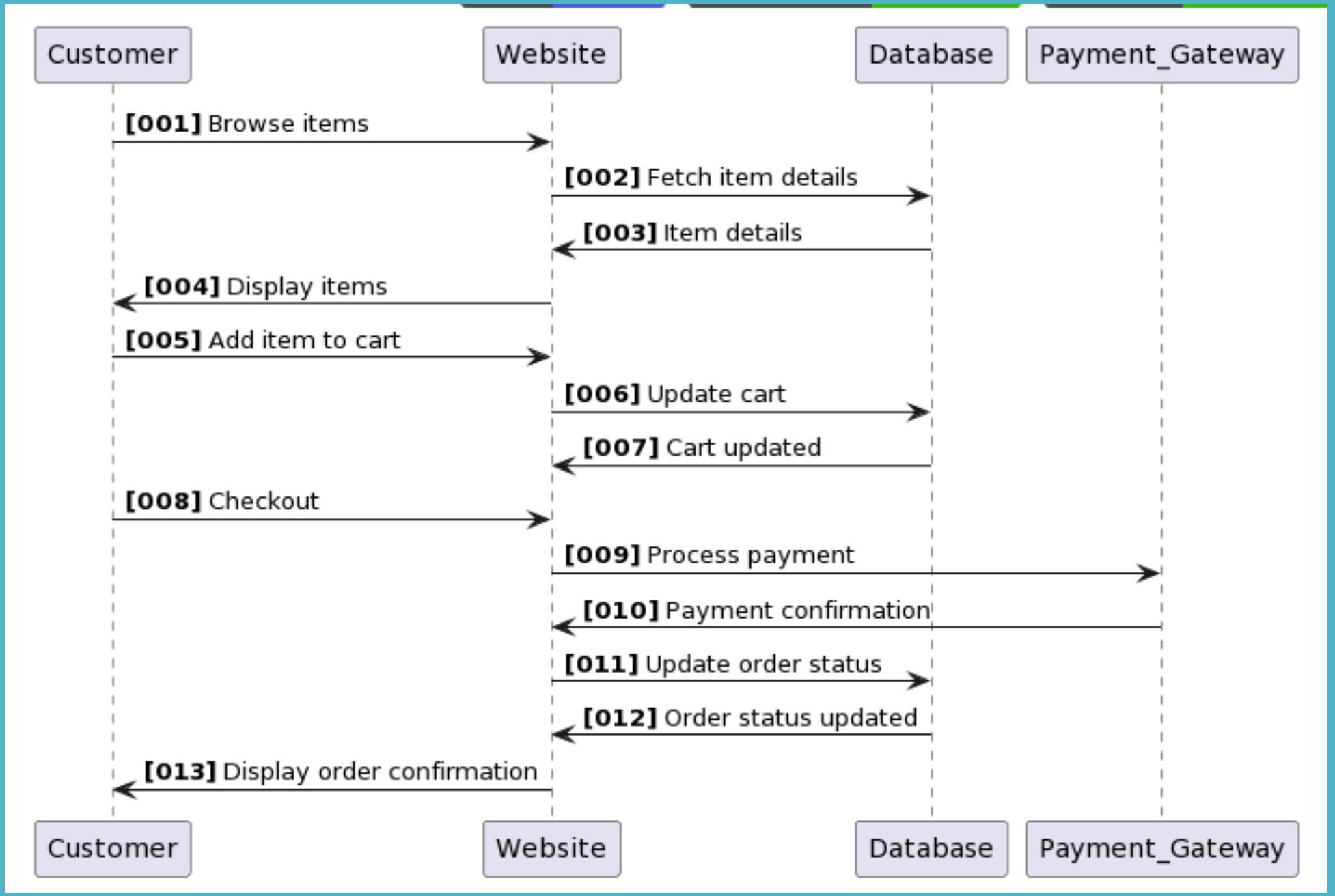


Пример диаграммы последовательности для интернет-магазина.

Инструменты для построения диаграмм последовательности UML:

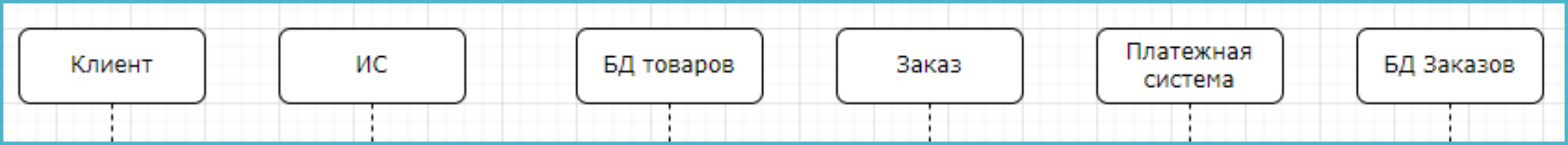
- Microsoft Visio
- draw.io
- Lucidchart:
- Enterprise Architect
- Visual Paradigm
- Astah
- yUML
- plantuml.com


www.plantuml.com
@startuml
autonumber "<b>[000]"
Customer -> Website : Browse items
Website -> Database : Fetch item details
Database -> Website: Item details
Website -> Customer: Display items
Customer -> Website: Add item to cart
Website -> Database: Update cart
Database -> Website: Cart updated
Customer -> Website: Checkout
Website -> Payment_Gateway: Process payment
Payment_Gateway -> Website: Payment confirmation
Website -> Database: Update order status
Database -> Website: Order status updated
Website -> Customer: Display order confirmation
@enduml

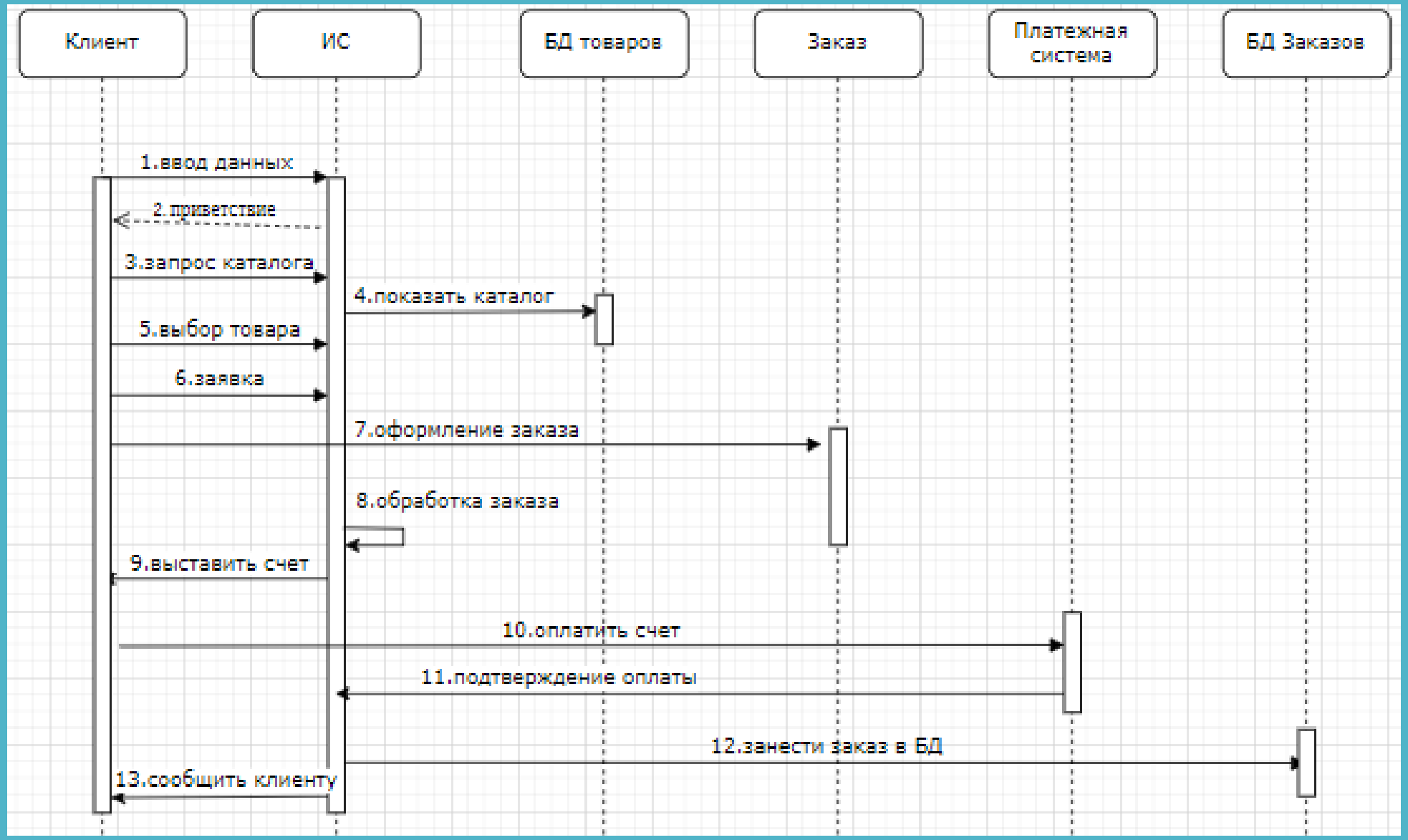

Customer
Website
Database
Payment_Gateway
[001] Browse items
[002] Fetch item details
[003] Item details
[004] Display items
[005] Add item to cart
[006] Update cart
[007] Cart updated
[008] Checkout
[009] Process payment
[010] Payment confirmation
[011] Update order status
[012] Order status updated
[013] Display order confirmation
Customer
Website
Database
Payment_Gateway

Порядок построения диаграммы последовательности

1. Располагаем элементы, участвующие во взаимодействии на горизонтальной оси, сверху, в порядке, в котором они действуют:



2. На вертикальной оси располагаем вызовы(сообщения), по порядку, сверху вниз.



2. Диаграмма классов.

Диаграмма классов UML - это структурная диаграмма, которая показывает классы, их атрибуты, методы и отношения между классами в системе.

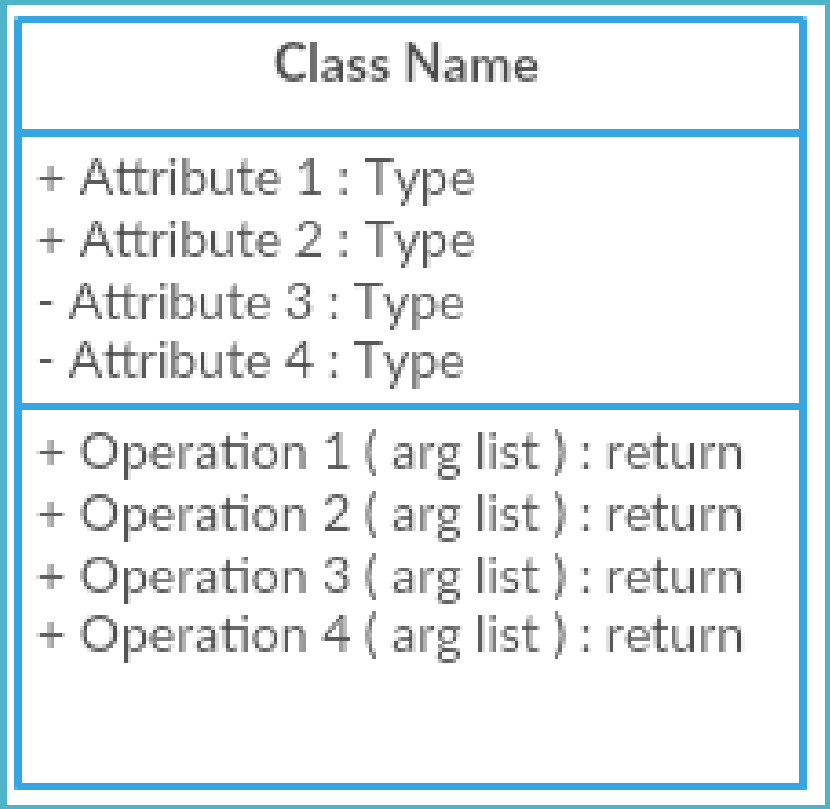
Она является важным инструментом для визуализации структуры программного продукта и его компонентов.

В основе любой объектно-ориентированной системы лежит этап проектирования структуры классов.

Основные элементы диаграммы классов:

1.Класс (Class):

- Классы представляют сущности или объекты, которые имеют общие атрибуты и методы.
- Класс обычно имеет имя, которое записывается в верхней части прямоугольника, представляющего класс.
- Пример: "Книга", "Студент", "Заказ".



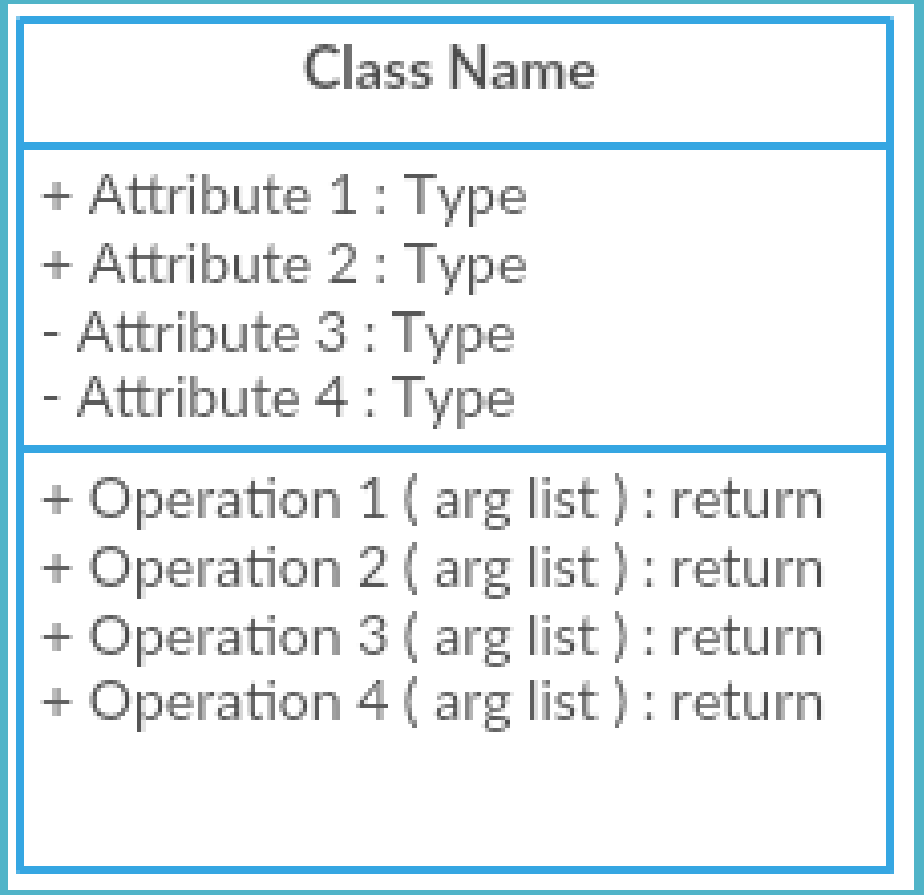
Основные элементы диаграммы классов:

2.Атрибут (Attribute):

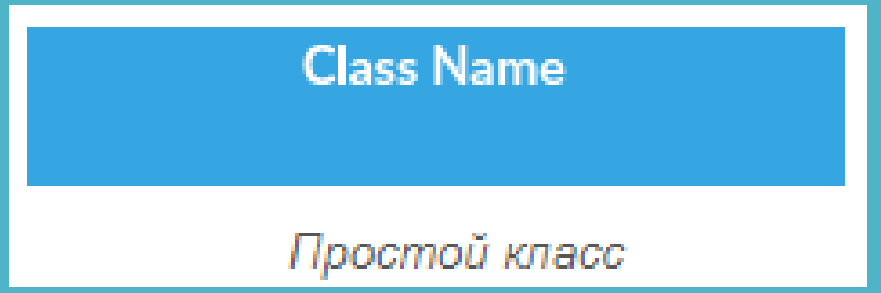
1. Атрибуты представляют характеристики или данные, которые принадлежат классу.
2. Они записываются в секции средней части прямоугольника класса.
3. Пример: "название", "автор", "год выпуска" для класса "Книга".

3.Метод (Method):

1. Методы представляют операции или функции, которые класс может выполнять.
2. Они также записываются в секции средней части прямоугольника класса.
3. Пример: "выдатьКнигу()", "расчетСреднегоБалла()" для класса "Студент".



Основные элементы диаграммы классов:



Последние два отсека являются необязательными. Нотация класса без последних двух отделений называется простым классом и содержит только имя класса.

Основные элементы диаграммы классов:

4.Отношения между классами:

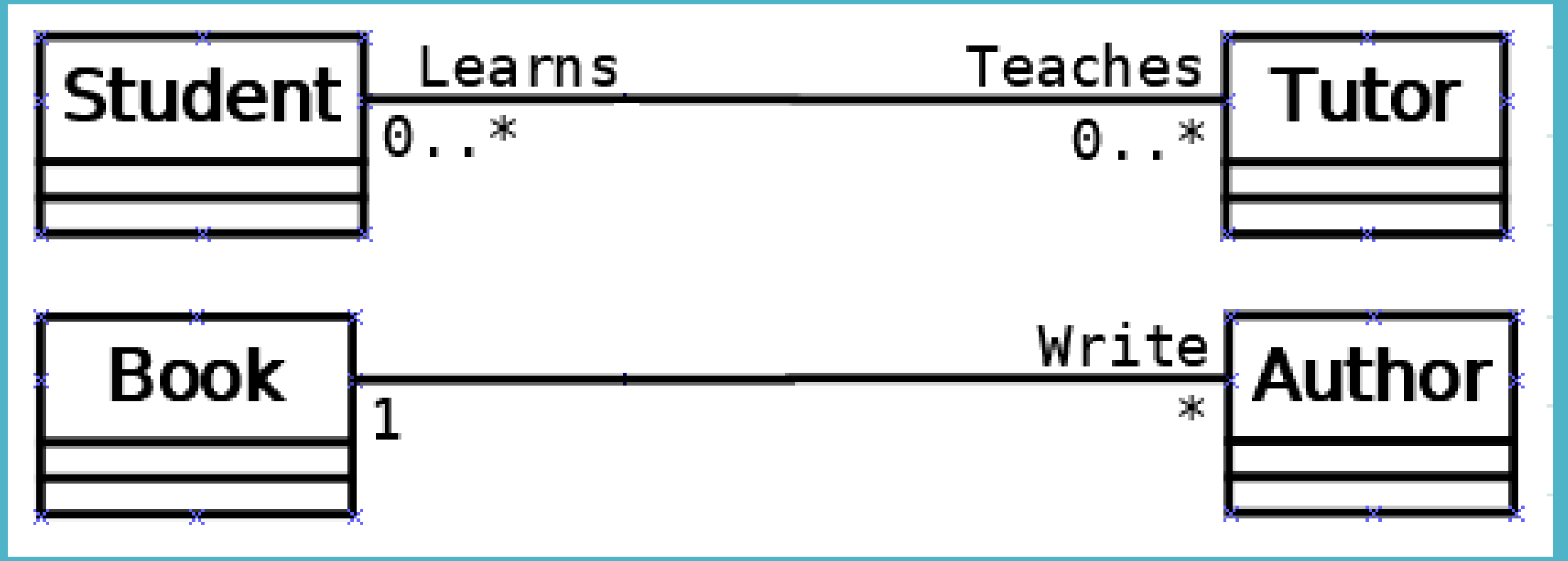
•Ассоциация (Association): Линия, соединяющая два класса, показывает отношение между ними. Например, связь между "Книгой" и "Автором".

•Наследование (Inheritance): Стрелка с пустым треугольником указывает на отношение наследования между суперклассом и подклассом.

•Агрегация (Aggregation): Линия со стрелкой, указывающей на "целое", показывает агрегацию, когда один класс является частью другого.

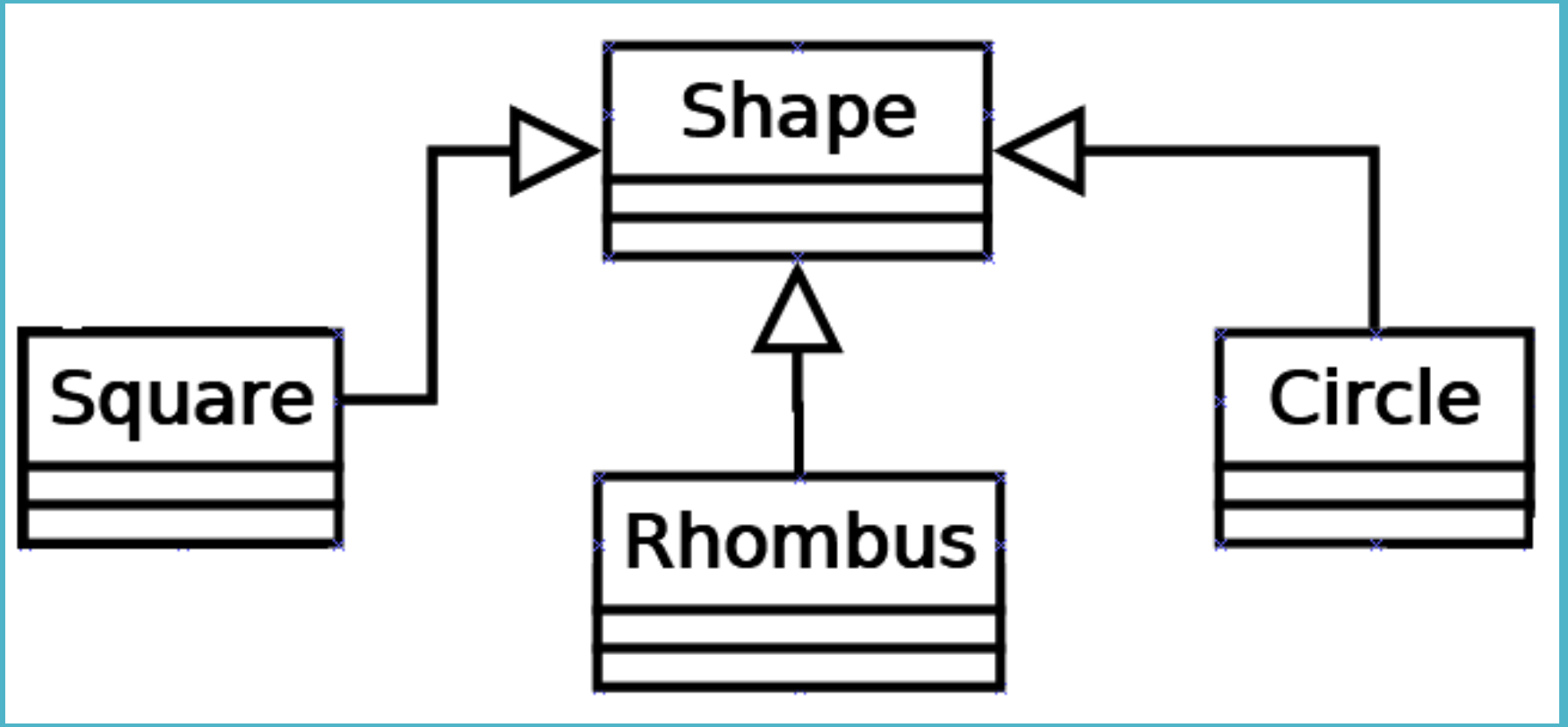
•Композиция (Composition): Линия со стрелкой и зачеркнутым ромбом, указывающим на "часть", показывает композицию, когда объекты одного класса являются частями другого объекта.

Ассоциация (Association):



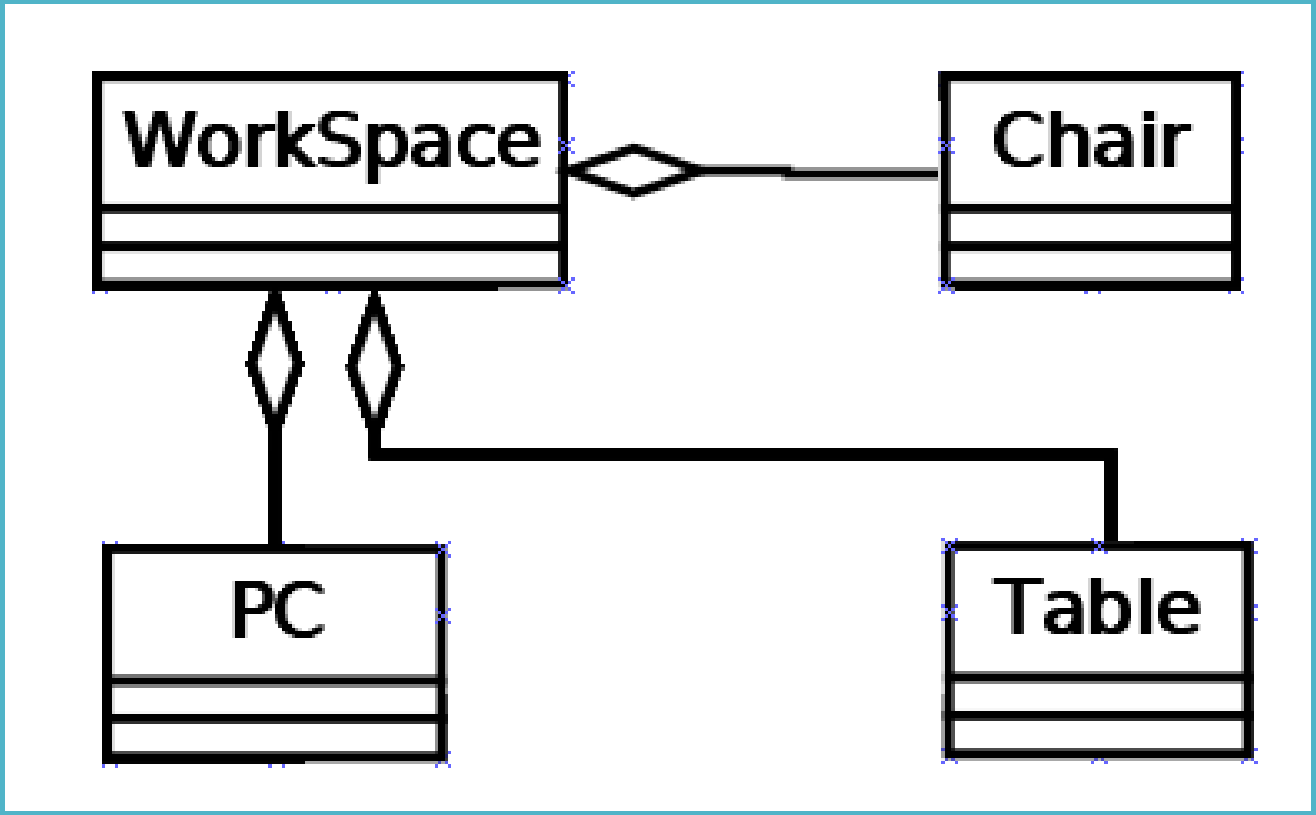
Это широкий термин, который охватывает практически любую логическую связь между классами.

Наследование (Inheritance)



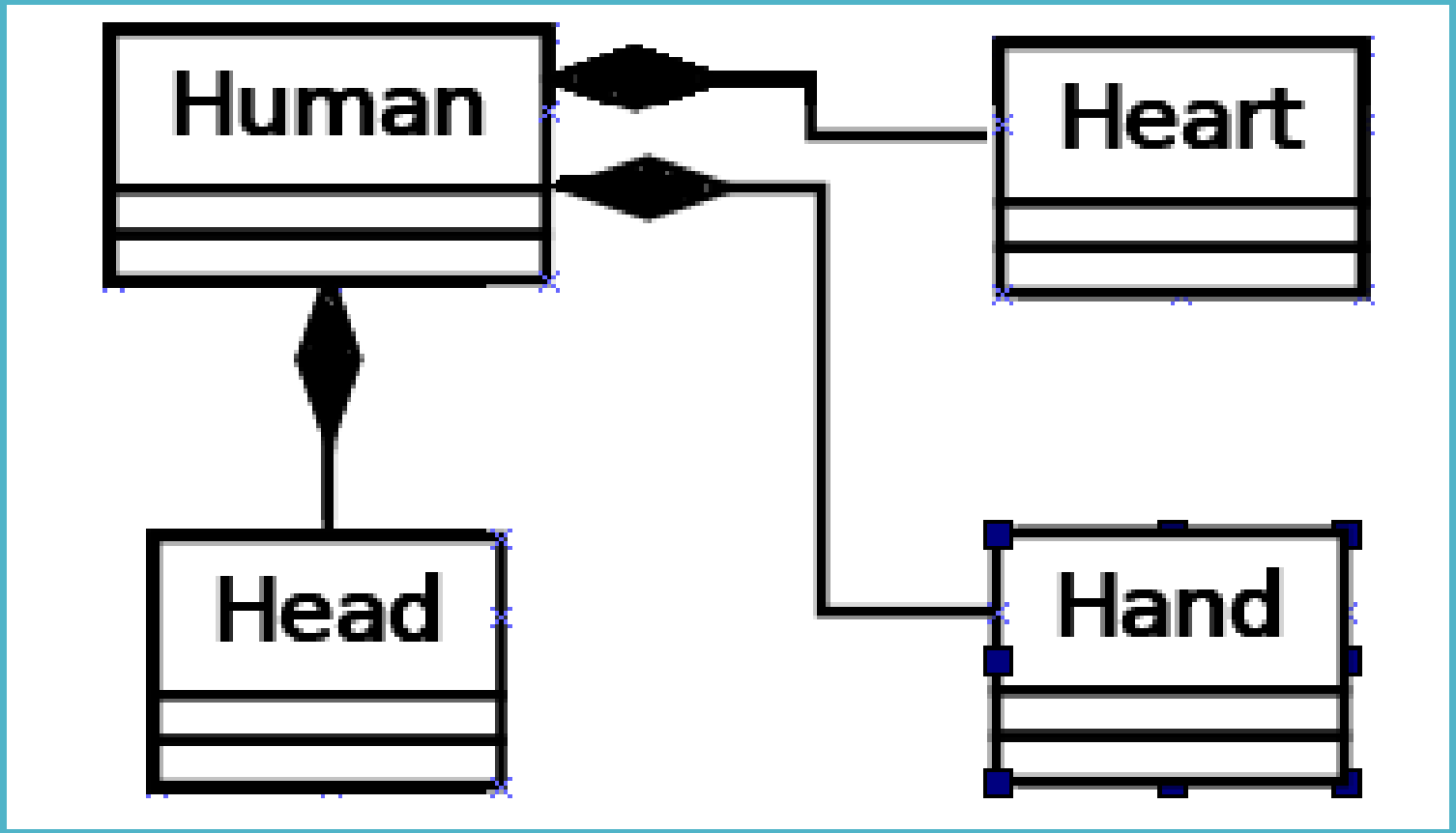
Тип отношений, при которых один связанный класс является дочерним по отношению к другому.

Агрегация (Aggregation)

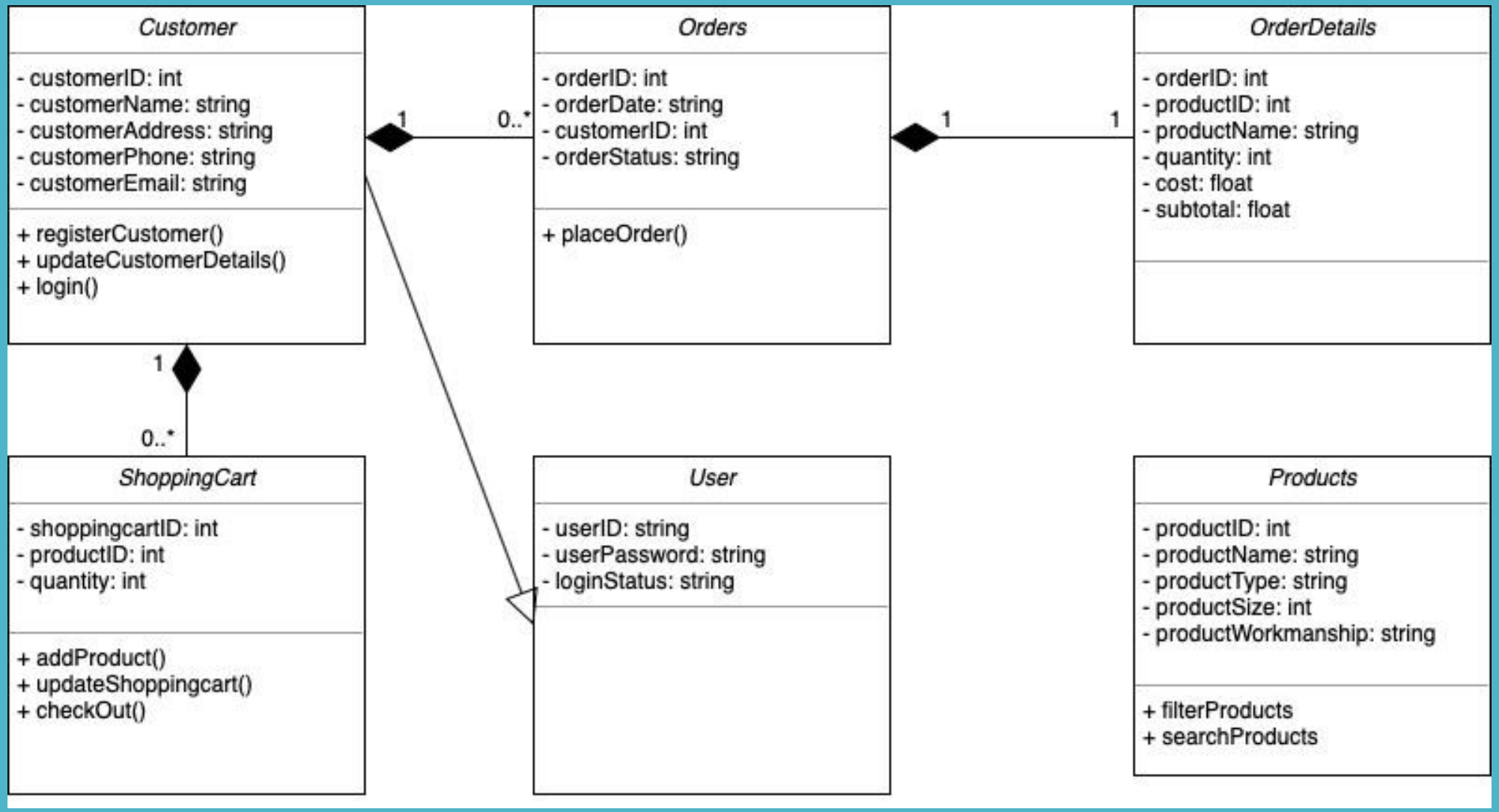


Формирование определенного класса в результате агрегирования одного класса или создания коллекции. При удалении класса WorkSpace, классы PC, Table, Chair будут существовать.

Композиция (Composition)



Похоже на отношение агрегирования, но при уничтожении класса-агрегатора (Human) входящие в него классы тоже уничтожаются.



Пример диаграммы классов.

3. Разработка технического задания.

Техническое задание (ТЗ) на разработку - это документ, который содержит детальное описание требований и спецификаций для проекта разработки программного обеспечения, веб-приложения, мобильного приложения или другого технического продукта. Он служит основой для команды разработчиков, чтобы понять, что должно быть реализовано, какие функциональности должны быть включены и как проект должен быть реализован.

Техническое задание обычно разрабатывается заказчиком или бизнес-аналитиками и направляется разработчикам или команде разработки для выполнения задачи. Оно должно быть четким, конкретным и содержать достаточно деталей, чтобы разработчики поняли, какие результаты должны быть достигнуты и каким образом.

Основные компоненты технического задания:

1.Введение: Общее описание проекта, его целей и контекста.

2.Описание требований: Подробное описание функциональных и нефункциональных требований к проекту.

3.Архитектура и дизайн: Описание архитектуры системы, включая компоненты, модули, базы данных, интерфейсы и другие технические аспекты проекта.

4.Функциональные требования: Описание основных функций и возможностей, которые должны быть реализованы в проекте.

Основные компоненты технического задания:

5.Нефункциональные требования: Требования к производительности, безопасности, масштабируемости, доступности, совместимости и другим аспектам проекта, не связанным с его функциональностью.

6.Интерфейсы: Требования к пользовательскому интерфейсу, включая внешний вид, навигацию, элементы управления и другие аспекты, которые важны для удобства использования проекта.

7.Тестирование: Описание требований к тестированию, включая тестовые сценарии и критерии приемки.

Основные компоненты технического задания:

8.Ресурсы: Перечень требуемых ресурсов, таких как аппаратное и программное обеспечение, базы данных, сторонние библиотеки и другие компоненты, необходимые для разработки и запуска проекта.

9.Графики и диаграммы: Визуальные представления, такие как диаграммы классов, диаграммы последовательности или прототипы пользовательского интерфейса, которые помогают визуализировать проект.

10.График разработки: План разработки, включая этапы, сроки и задачи, необходимые для завершения проекта.

Пример технического задания:

1.Введение:
Цель проекта: Разработка полнофункционального интернет-магазина для продажи товаров народного потребления.

Контекст проекта: Интернет-магазин будет интегрироваться с существующей системой управления складом и системой обработки платежей.

Заказчиком является компания “Roga & Kopyta".

Описание требований:

2.1 Функциональные требования:

Регистрация и аутентификация пользователей: Пользователи должны иметь возможность создать учетную запись, войти в систему и управлять своим профилем.

Каталог товаров: Показывать список товаров с описанием, изображениями, ценами и доступностью на складе. Пользователи могут осуществлять поиск и фильтрацию товаров.

Корзина покупок: Пользователи могут добавлять товары в корзину, управлять количеством товаров и оформлять заказы.

Описание требований:

Оформление заказа: Пользователи могут выбрать способ доставки и оплаты, ввести данные для доставки и подтвердить заказ.

Интеграция с платежной системой: При оформлении заказа пользователи должны иметь возможность выбрать способ оплаты и осуществить платеж.

Управление заказами: Администраторы могут просматривать, обрабатывать и отслеживать статусы заказов.

Управление контентом: Администраторы могут добавлять, редактировать и удалять товары, категории, акции и другой контент.

2.2 Нефункциональные требования:

- Интернет-магазин должен обеспечивать быструю загрузку страниц, обработку запросов и отображение товаров. Отклик на действия пользователя не должен превышать 1 секунду.
- Защита данных пользователей, использование шифрования для передачи конфиденциальной информации, защита от взлома и злоумышленников.
- Платформа должна поддерживать минимум 1000 пользователей одновременно.
- Система должна быть способна масштабироваться для обработки растущего числа пользователей и товаров.
- Интерфейс должен быть доступен и понятен для пользователей с ограниченными возможностями (например, поддержка скринридеров).

3.Сценарии использования:

3.1 Регистрация и аутентификация пользователя:

•Пользователь открывает страницу регистрации.

•Пользователь вводит свои данные, включая имя, адрес электронной почты и пароль.

•Система проверяет введенные данные и создает учетную запись пользователя.

•Пользователь получает подтверждение регистрации по электронной почте.

3.2 Покупка товара:

•Пользователь переходит на страницу товара.

•Пользователь выбирает количество товара и добавляет его в корзину.

•Пользователь переходит в корзину и просматривает выбранные товары.

•Пользователь нажимает кнопку "Оформить заказ" и выбирает способ доставки и оплаты.

•Пользователь вводит данные для доставки и подтверждает заказ.

•Система обрабатывает заказ, резервирует товары и перенаправляет пользователя на страницу оплаты.

•Пользователь осуществляет платеж через выбранную платежную систему(продолжение)

•Система подтверждает успешный платеж и отправляет пользователю подтверждение заказа по электронной почте.

4.Архитектура и дизайн:

- Интернет-магазин будет разработан на основе клиент-серверной архитектуры.
- Фронтенд будет разработан с использованием HTML, CSS, JavaScript и фреймворка React.
- Бэкенд будет разработан с использованием языка программирования Python и фреймворка Django.
- Для хранения данных будет использоваться реляционная база данных PostgreSQL.
- Интерфейс будет разработан с учетом принципов юзабилити и современного дизайна.

5.Интерфейсы:

- Пользовательский интерфейс должен быть интуитивно понятным и удобным для пользователей.
- Административный интерфейс должен обеспечивать управление товарами, заказами и другим контентом.

6.Тестирование:

- Разработчики должны разработать план тестирования, включающий функциональное тестирование, интеграционное тестирование и нагрузочное тестирование.
- Тестирование должно быть проведено для проверки работоспособности всех функций и соответствия требованиям.

7.Ресурсы:

•Для разработки интернет-магазина требуются следующие ресурсы: серверы для размещения фронтенда и бэкенда, база данных, доменное имя, SSL-сертификат для обеспечения безопасного соединения.

8.Ограничения и допущения:

- Бюджет проекта ограничен [сумма].
- Разработка должна быть завершена в течение [сроки].
- Допущение: Пользователи имеют базовые знания компьютерной грамотности.

9.Графики и диаграммы :

Визуальные представления, такие как диаграммы классов, диаграммы последовательности или прототипы пользовательского интерфейса, которые помогают визуализировать проект.

10.График разработки:

- Разработка интернет-магазина будет выполнена в три фазы: анализ и проектирование, разработка и тестирование, развертывание и запуск.
- Каждая фаза будет иметь определенные этапы и сроки выполнения.

Домашнее задание:

1.Создайте диаграмму последовательности для простой системы, такой как автомат для продажи кофе. Определите объекты и их взаимодействия в системе.

2.Создайте диаграмму классов для колледжа. Опишите основные классы, их атрибуты (поля) и методы (операции). Укажите основные связи между классами, такие как наследование, ассоциация, композиция и агрегация.

Рекомендуемая литература:

1.Программная инженерия. Визуальное моделирование программных систем. Е.А. Черткова. Учебник для СПО. 2019г.

2.Введение в UML. Курс лекций. НГТУ им. Н.И. Лобачевского