Biểu đồ trường hợp sử dụng - Ngôn ngữ mô hình hóa thống nhất (UML)

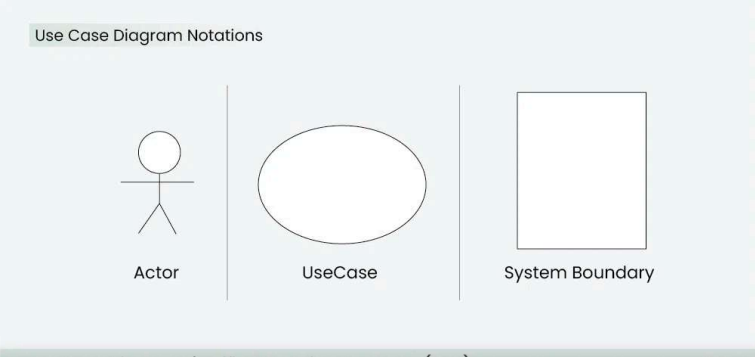
Cập nhật lần cuối: 23 tháng 7 năm 2025

Sơ đồ Trường hợp Sử dụng trong Ngôn ngữ Mô hình Thống nhất (UML) là một biểu diễn trực quan minh họa tương tác giữa người dùng (tác nhân) và hệ thống. Sơ đồ này nắm bắt các yêu cầu chức năng của hệ thống, cho thấy cách người dùng khác nhau tương tác với các trường hợp sử dụng khác nhau, hoặc các chức năng cụ thể, trong hệ thống. Sơ đồ trường hợp sử dụng cung cấp cái nhìn tổng quan về hành vi của hệ thống, giúp các bên liên quan, nhà phát triển và nhà phân tích hiểu cách hệ thống được thiết kế để vận hành theo góc nhìn của người dùng và cách các quy trình khác nhau liên quan đến nhau. Chúng rất quan trọng trong việc xác định phạm vi và yêu cầu của hệ thống.

Mục lục

Biểu đồ trường hợp sử dụng trong UML là gì?

Biểu đồ Trường hợp Sử dụng (Use Case Diagram) là một loại biểu đồ Ngôn ngữ Mô hình Thống nhất (UML) thể hiện sự tương tác giữa các tác nhân (người dùng hoặc hệ thống bên ngoài) và hệ thống đang được xem xét để đạt được các mục tiêu cụ thể. Biểu đồ này cung cấp góc nhìn tổng quan về chức năng của hệ thống bằng cách minh họa các cách khác nhau mà người dùng có thể tương tác với hệ thống.



Sơ đồ trường hợp sử dụng hữu ích trong một số trường hợp. Dưới đây là những trường hợp bạn nên cân nhắc sử dụng chúng:

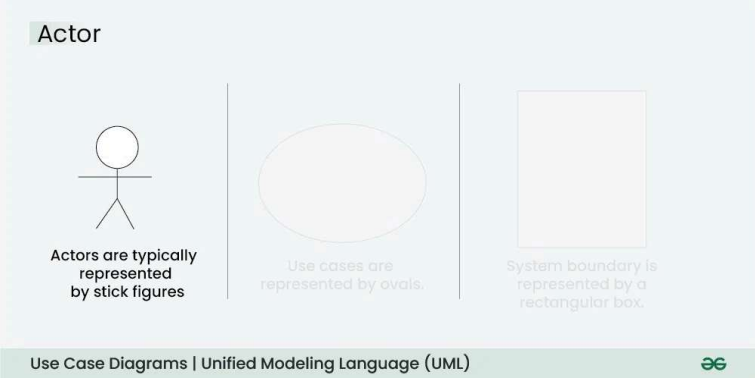
- Khi bạn cần thu thập và làm rõ các yêu cầu của người dùng, sơ đồ trường hợp sử dụng sẽ giúp hình dung cách những người dùng khác nhau tương tác với hệ thống.
- Nếu bạn đang làm việc với nhiều nhóm khác nhau, bao gồm cả những bên liên quan không chuyên về kỹ thuật, các sơ đồ này sẽ cung cấp cách rõ ràng và đơn giản để truyền đạt chức năng của hệ thống.
- Trong giai đoạn thiết kế hệ thống, sơ đồ trường hợp sử dụng giúp phác thảo tương tác của người dùng và lập kế hoạch các tính năng, đảm bảo thiết kế phù hợp với nhu cầu của người dùng.
- Khi xác định những gì được bao gồm trong hệ thống so với những gì bên ngoài, sơ đồ trường hợp sử dụng giúp làm rõ các ranh giới này.

Ký hiệu sơ đồ trường hợp sử dụng

Ký hiệu UML cung cấp ngôn ngữ trực quan cho phép các nhà phát triển phần mềm, nhà thiết kế và các bên liên quan khác giao tiếp và ghi lại thiết kế, kiến trúc và hành vi của hệ thống theo cách nhất quán và dễ hiểu.

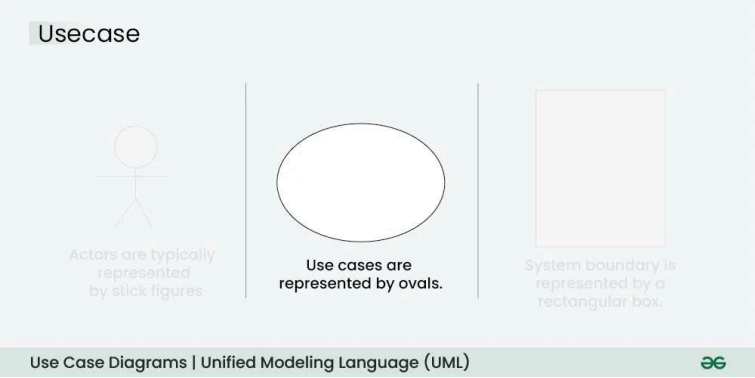
1. Diễn viên

Actors are external entities that interact with the system. These can include users, other systems, or hardware devices. In the context of a Use Case Diagram, actors initiate use cases and receive the outcomes. Proper identification and understanding of actors are crucial for accurately modeling system behavior.



2. Use Cases

Use cases are like scenes in the play. They represent specific things your system can do. In the online shopping system, examples of use cases could be "Place Order," "Track Delivery," or "Update Product Information". Use cases are represented by ovals.

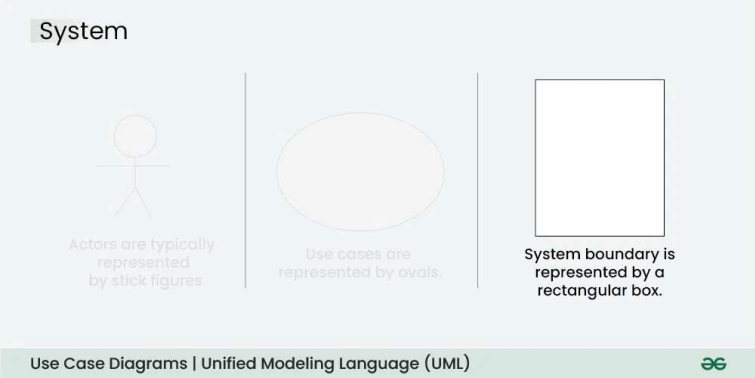


3. System Boundary

The system boundary is a visual representation of the scope or limits of the system you are modeling. It defines what

rectangular box that surrounds all the use cases of the system.

> *The purpose of system boundary is to clearly outlines the boundaries of the system, indicating which components are internal to the system and which are external actors or entities interacting with the system.*



Use Case Diagram Relationships

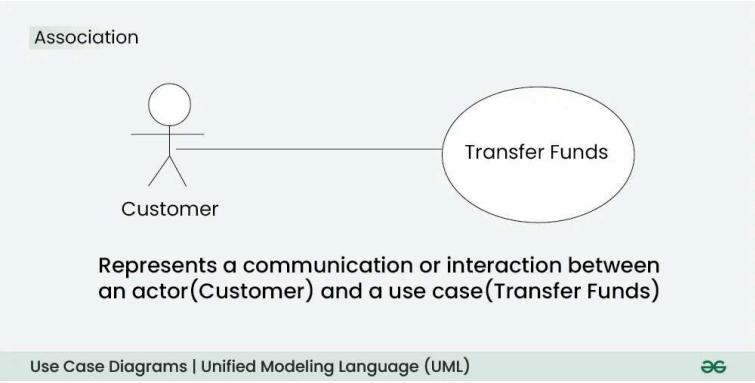
In a Use Case Diagram, relationships play a crucial role in depicting the interactions between actors and use cases. These relationships provide a comprehensive view of the system's functionality and its various scenarios. Let's delve into the key types of relationships and explore examples to illustrate their usage.

1. Association Relationship

The Association Relationship represents a communication or interaction between an actor and a use case. It is depicted by a line connecting the actor to the use case. This relationship signifies that the actor is involved in the functionality described by the use case.

> ***Example: Online Banking System***

- **Actor:** Customer
- **Use Case:** Transfer Funds
- **Association:** A line connecting the "Customer" actor to the "Transfer Funds" use case, indicating the customer's involvement in the funds transfer process.

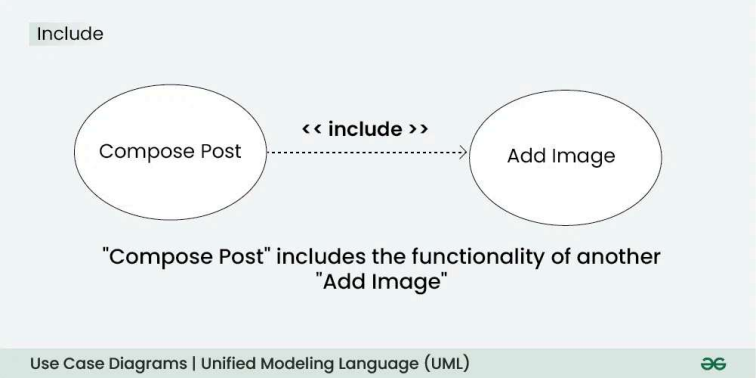


2. Include Relationship

The Include Relationship indicates that a use case includes the functionality of another use case. It is denoted by a dashed arrow pointing from the including use case to the included use case. This relationship promotes modular and reusable design.

> ***Example: Social Media Posting***

- **Use Cases:** Compose Post, Add Image

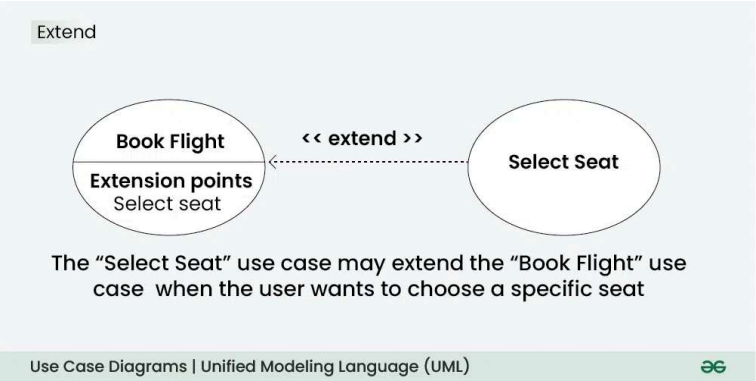


3. Extend Relationship

The Extend Relationship illustrates that a use case can be extended by another use case under specific conditions. It is represented by a dashed arrow with the keyword "extend." This relationship is useful for handling optional or exceptional behavior.

Example: Flight Booking System

- **Use Cases:** Book Flight, Select Seat
- **Extend Relationship:** The "Select Seat" use case may extend the "Book Flight" use case when the user wants to choose a specific seat, but it is an optional step.

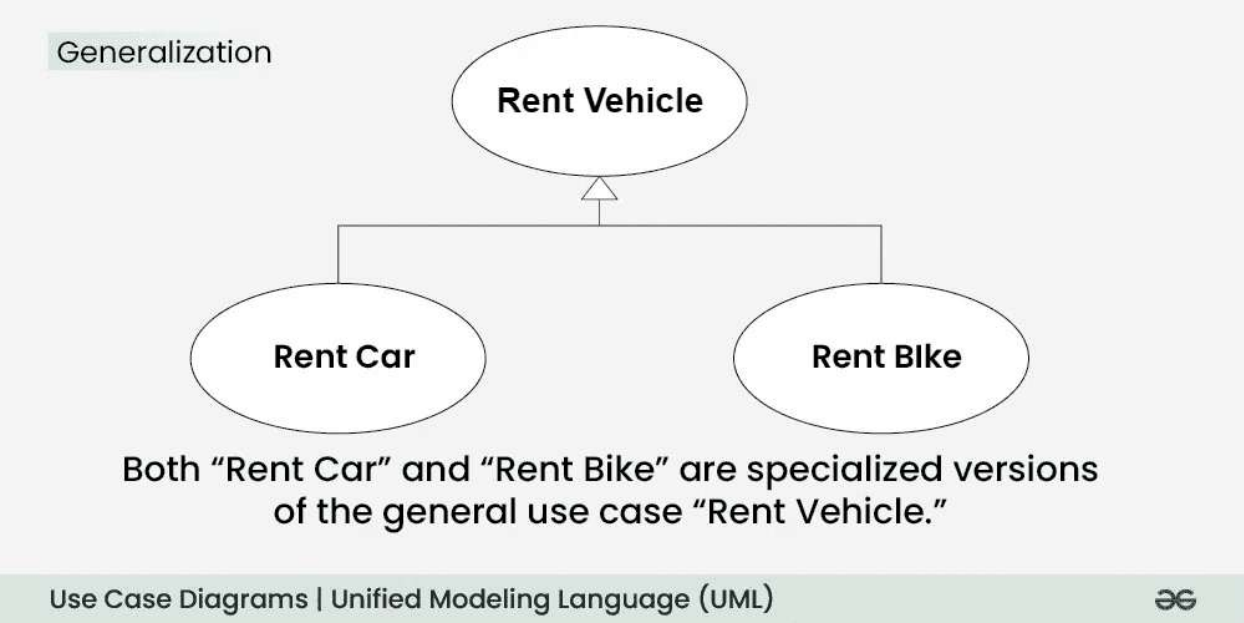


4. Generalization Relationship

The Generalization Relationship establishes an "is-a" connection between two use cases, indicating that one use case is a specialized version of another. It is represented by an arrow pointing from the specialized use case to the general use case.

Example: Vehicle Rental System

- **Use Cases:** Rent Car, Rent Bike
- **Generalization Relationship:** Both "Rent Car" and "Rent Bike" are specialized versions of the general use case "Rent Vehicle."



Generalization Relationship

How to draw a Use Case diagram in UML?

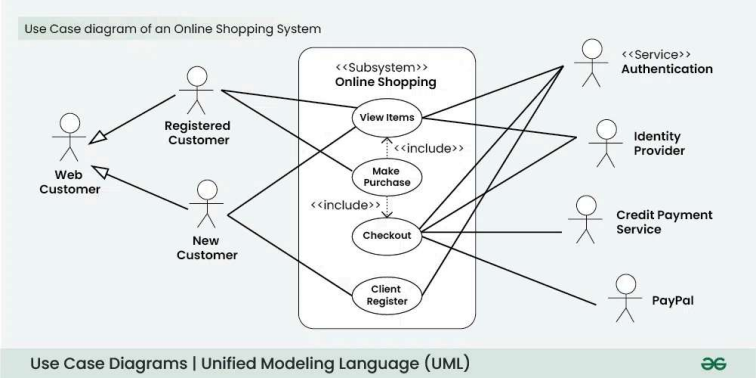
Below are the main steps to draw use case diagram in UML:

- **Step 1: Identify Actors**: Determine who or what interacts with the system. These are your actors. They can be users, other systems, or external entities.
- **Step 2: Identify Use Cases**: Identify the main functionalities or actions the system must perform. These are your use cases. Each use case should represent a specific piece of functionality.
- **Step 3: Connect Actors and Use Cases**: Draw lines (associations) between actors and the use cases they are involved in. This represents the interactions between actors and the system.
- **Step 4: Add System Boundary**: Draw a box around the actors and use cases to represent the system boundary. This defines the scope of your system.
- **Step 5: Define Relationships**: If certain use cases are related or if one use case is an extension of another, you can indicate these relationships with appropriate notations.
- **Step 6: Review and Refine**: Step back and review your diagram. Ensure that it accurately represents the interactions and relationships in your system. Refine as needed.
- **Step 7: Validate**: Share your use case diagram with stakeholders and gather feedback. Ensure that it aligns with their understanding of the system's functionality.

Use Case Diagram example(Online Shopping System)

Let's understand how to draw a Use Case diagram with the help of an Online Shopping System:

- **Actors:**
  - Customer
  - Admin
- **Use Cases:**
  - Browse Products
  - Add to Cart
  - Checkout
  - Manage Inventory (Admin)
- **Relations:**
  - The Customer can browse products, add to the cart, and complete the checkout.
  - The Admin can manage the inventory.



What are common Use Case Diagram Tools and Platforms?

Several tools and platforms are available to create and design Use Case Diagrams. These tools offer features that simplify the diagram creation process, facilitate collaboration among team members, and enhance overall efficiency. Here are some popular Use Case Diagram tools and platforms:

- **Lucidchart:**
    - Cloud-based collaborative platform.
-
    - Real-time collaboration and commenting.
    - Templates for various diagram types.
- **draw.io:**
    - Free, open-source diagramming tool.
    - Works offline and can be integrated with Google Drive, Dropbox, and others.
    - Offers a wide range of diagram types, including Use Case Diagrams.
- **Microsoft Visio:**
    - Part of the Microsoft Office suite.
    - Supports various diagram types, including Use Case Diagrams.
-
    - Extensive shape libraries and templates.
- **SmartDraw:**
    - User-friendly diagramming tool.
    - Templates for different types of diagrams, including Use Case Diagrams.
    - Integration with Microsoft Office and Google Workspace.
- **PlantUML:**
    - Open-source tool for creating UML diagrams.
    - Text-based syntax for diagram specification.
    - Supports collaborative work using version control systems.

What are Common Mistakes while making Use Case Diagram?

Avoiding common mistakes ensures the accuracy and effectiveness of the Use Case Diagram. Here are key points for each mistake:

- Adding too much detail can confuse people.
- Unclear connections lead to misunderstandings about system interactions.
- Different names for the same elements create confusion.
- Incorrectly using generalization can misrepresent relationships.
- Failing to define the system's limits makes its scope unclear.
- Treating the diagram as static can make it outdated and inaccurate.

Best Practices for Use Case Diagram

- Use Case Diagram focus on capturing the core functions of the system, avoiding extraneous details.
- They uses a uniform naming scheme for use cases and actors throughout the diagram to enhance clarity and prevent misunderstandings.
- They ensure uniformity in the appearance of elements such as ovals (for use cases), stick figures (for actors), and connecting lines to create a polished presentation.
- They help in organizing use cases into coherent groups that represent distinct modules or subsystems within the overall system.
- Use Case Diagrams adopt an iterative method, updating the diagram as the system changes or as new information emerges.

What is the Purpose and Benefits of Use Case Diagrams?

The Use Case Diagram offers numerous benefits throughout the system development process. Here are some key advantages of using Use Case Diagrams:

- Use Case Diagrams offer a clear visual representation of a system's functions and its interactions with external users. This representation helps stakeholders, including those without technical expertise, in grasping the system's overall behavior.
- They establish a shared language for articulating system requirements, ensuring that all team members have a common understanding.
- Use Case Diagram illustrate the different ways users engage with the system, contributing to a thorough comprehension of its functionalities.
- In the design phase, Use Case Diagrams help outline how users (actors) will interact with the system. They support the planning of user interfaces and aid in structuring system functionalities.

Conclusion

In conclusion, a Use Case Diagram in UML serves as a powerful tool for capturing and visualizing the functional requirements and interactions within a system. By representing actors, use cases, and their relationships in a clear and concise manner, this diagram provides a high-level overview of the system's behavior.

Next Article

UML Full Form

Similar Reads

What are UML Diagrams

Unified Modeling Language (UML) Diagrams

Unified Modeling Language (UML) is a general-purpose modeling language. The main aim of UML is to define a standard way to visualize the way a system has been designed. It is quite similar to blueprints used in other fields of engineering. UML is not a programming language, it is rather a visual lan

14 min read

UML Full Form

The full form of UML is "Unified Modeling Language". It is a general-purpose modeling language. The main aim of UML is to define a standard way to visualize how a system has been designed. It is quite similar to blueprints used in other fields of engineering. UML is not a programming language, it is

3 min read

Structural Diagrams

Class Diagram | Unified Modeling Language (UML)

A UML class diagram is a visual tool that represents the structure of a system by showing its classes, attributes, methods, and the relationships between them. It helps everyone involved in a projectâ€"like developers and designersâ€"understand how the system is organized and how its components interact

12 min read

8 min read

Deployment Diagram in Unified Modeling Language(UML)

A Deployment Diagram is a type of Structural UML Diagram that shows the physical deployment of software components on hardware nodes. It illustrates the mapping of software components onto the physical resources of a system, such as servers, processors, storage devices, and network infrastructure.Ta

8 min read

Package Diagram â€“ Unified Modeling Language (UML)

Biểu đồ gói là một loại biểu đồ cấu trúc trong UML (Ngôn ngữ mô hình hóa thống nhất) dùng để tổ chức và nhóm các lớp và thành phần liên quan thành các gói. Nó thể hiện trực quan các mối quan hệ phụ thuộc và mối quan hệ giữa các gói này, giúp minh họa cách các phần khác nhau của hệ thống tương tác với nhau.

7 phút đọc

Biểu đồ hành vi

Biểu đồ hành vi | Ngôn ngữ mô hình thống nhất (UML)

Các ứng dụng phức tạp cần sự hợp tác và lập kế hoạch từ nhiều nhóm, do đó cần một phương thức giao tiếp rõ ràng và súc tích giữa các nhóm. Vì vậy, UML trở nên thiết yếu để giao tiếp với những người không phải lập trình viên về các yêu cầu, chức năng và quy trình thiết yếu của hệ thống. UML được liên kết với

7 phút đọc

Địa chỉ Công ty & Truyền thông:
A-143, Tầng 7, Tòa nhà Sovereign Corporate, Khu 136, Noida, Uttar Pradesh (201305)

Địa chỉ đăng ký:
K 061, Tháp K, Căn hộ Gulshan Vivante, Khu 137, Noida, Gautam Buddh Nagar, Uttar Pradesh, 201305

NextJS
Khởi động
Thiết kế web

Khoa học máy tính

Hệ điều hành
Mạng máy tính
Hệ thống quản lý cơ sở dữ liệu
Kỹ thuật phần mềm
Thiết kế logic số
Toán Kỹ thuật
Phát triển phần mềm
Kiểm thử phần mềm

Thiết kế hệ thống

Thiết kế cấp cao
Thiết kế cấp thấp
Biểu đồ UML
Hướng dẫn phỏng vấn
Mẫu thiết kế
OOAD
Trại huấn luyện thiết kế hệ thống
Câu hỏi phỏng vấn

Môn học ở trường

Toán học
Vật lý
Hoá học
Sinh vật học
Khoa học xã hội
Ngữ pháp tiếng Anh
Thương mại
GK thế giới

Câu hỏi phỏng vấn Python
Django

DevOps

Git
Linux
AWS
Docker
Kubernetes
Xanh lam
GCP
Lộ trình DevOps

Chuẩn bị phỏng vấn

Lập trình cạnh tranh
DS hoặc Algo hàng đầu cho CP
Quy trình tuyển dụng theo công ty
Chuẩn bị theo từng công ty
Chuẩn bị năng khiếu
Câu đố

Video GeeksforGeeks

DSA
Trăn
Java
C++
Phát triển web
Khoa học dữ liệu
Môn Khoa học máy tính