# Weekly Homework 5

Yevgeniy Terentyev, Miriam Gaetano
Softwaretechnik

June 13, 2025

## 5-1

### a) What is a use case in the context of software development (independent of UML), and what is it for?

A use case is a textual description of sequences of actions (scenarios) that the system performs in other words, the expected **behavior** of a system in response to an interaction with a user or another external actor. We care about *what* the system does, not *how* it does it.

Use cases are used to describe and verify the understanding of user requirements during early analysis. For this reason, they must be simple and understandable. They are later used to validate the architecture and verify the system behavior.

### b) Research principles for writing good use cases

**1. Product Perspective:** What makes a use case excellent *when completed*? Here are three specific quality criteria (from the perspective of the reader/evaluator):

1. The actor's goals must be clear, and internal system operations should not be mentioned (e.g., "Make a payment" instead of "Update transaction database").
2. The scenario flow must be correct and only describe observable interactions between system and actor, without technical implementation details.
3. The flow must be coherent and alternative flows should be clearly handled.

**2. Process Perspective:** How do you write a good use case? Here are at least three key writing guidelines (from the writer's point of view):

1. Start with identifying the main actor and their goal, using simple and non-technical language. Then proceed to the main scenario, and finally the extensions.
2. Avoid technical structures or pseudo-code (e.g., IF-THEN, REPEAT), as the description must be understandable by non-technical stakeholders.

3. Take into account variants and error conditions.

4. Structure the use case from the user's point of view, focusing on how they interact with the system to ensure clarity and relevance.

### c) What is the difference between a scenario and a use case?

A **scenario** is a sequence of actions that represents a *single concrete interaction* between one or more actors and the system. It is a *linear and specific* example of behavior, with no alternatives or error handling. It focuses on what happens *externally*, without describing internal logic.
A **use case**, by contrast, is a more *abstract and complete* description of how an actor achieves a goal through the system. It includes the actors, preconditions, goals, and *multiple scenarios*, including alternatives and exceptions — all under the same use case.

**Example Scenario: Online Order**

1. The customer requests the product list

2. The system displays the available products

3. The customer selects the desired products

4. The system shows the total cost

5. The customer confirms the order

6. The system sends an order confirmation

## 5-2

### Learning Objective

To recognize when and for what purposes UML use case diagrams are appropriate.

### a) What is a use case diagram compared to a use case? What are the advantages and disadvantages of UML diagrams?

A use case diagram is a graphical representation of the various relationships between **actors and use cases**, without detailing the interaction. In contrast, a use case is a textual description that explains in detail the interaction between the **actor and the system**.

**Advantages:**
Being graphical, it is more intuitive to understand the various functionalities and the involved actors.

**Disadvantages:**
It only shows actors, use cases, and their relationships, but does not describe the system's behavior. Therefore, if not accompanied by textual use cases, it is insufficient for requirement analysis.

## b) What does the actor model element represent in a use case diagram?

An actor is an external entity that interacts with the system to achieve a goal. It is not necessarily a person, but rather a **role that someone or something plays in relation to the system**. In diagrams, actors are represented using a stick figure.

An actor must have an active role and a goal-oriented interest.

*Examples:*

- **Use case:** "Purchase product"
  → **Primary actor:** Buyer (goal: to make a purchase)
- **Use case:** "Record data in a management system"
  → **Actor:** Employee

In these cases, the actor name identifies a functional role, not a specific individual. For example, if a customer only receives an automatic email and does not interact with the system, they are **not considered an actor**.

## c) What other model elements does UML provide for use case diagrams, besides the actor and the use case?
## What types of relationships can exist between two use cases in a UML diagram?

1. **System Boundary:** A rectangle that encloses all the use cases, helping to distinguish what is internal to the system from external elements such as actors and external systems.
2. **<<include>>** relationship: This indicates that a use case **always includes** another use case as part of its main flow. It is similar to a function call in programming. *Example:* "Place Order" <<include>> "Select Payment Method"
3. **<<extend>>** relationship: This represents alternative or optional behaviors that are executed **only under certain conditions**. *Example:* "View Order History" <<extend>> "View Order Details"
   In this case, "View Order Details" is extended only if the actor chooses to see the order history.
4. **Generalization:** It represents inheritance between actors or between use cases, and is used to specialize roles or functionalities.

# 5-3

## a) Use Case – Start a Pomodoro Session

**Learning Objective:** Formulate use cases in a structured way and relate them in a UML.

| | |
|---|---|
| **Use Case** | Start a Pomodoro Session<br>History: Created 15/05/2025 – Author: [hv] |
| **Description** | The user starts a timed Pomodoro session to perform focused work. A session usually lasts 25 minutes and may be linked to a task. The timer runs until completion and alerts the user when the session ends.<br>Sources: *Pomodoro Technique – Cirillo 2006* |
| **Actors** | User (primary)<br>Application Timer Module<br>Notification System |
| **Preconditions** | - The system is already running.<br>- The timer is not already active.<br>- Session timing and notification mechanisms are reliable. |
| **Main Scenario (Steps)** | 1. User opens the Pomodoro app.<br>2. The system displays the timer interface and the list of available tasks.<br>3. User optionally selects a task.<br>4. User presses the "Start" button.<br>5. Timer countdown begins and is displayed on the screen, using the defined duration (25 min).<br>6. When the timer reaches zero, the system notifies the user with sound and/or popup. |
| **Variations (Extensions)** | #3: No task selected → The session starts without a linked task.<br>#4: Timer fails to start due to internal error → System shows an error and allows retry.<br>#5: Notifications are muted → Only a visual notification is shown. |
| **Issues** | What happens if the user misses the notification at the end of the Pomodoro session? Should the timer automatically start the next phase or wait for explicit user confirmation? |

## b) Use Case – UML relationships

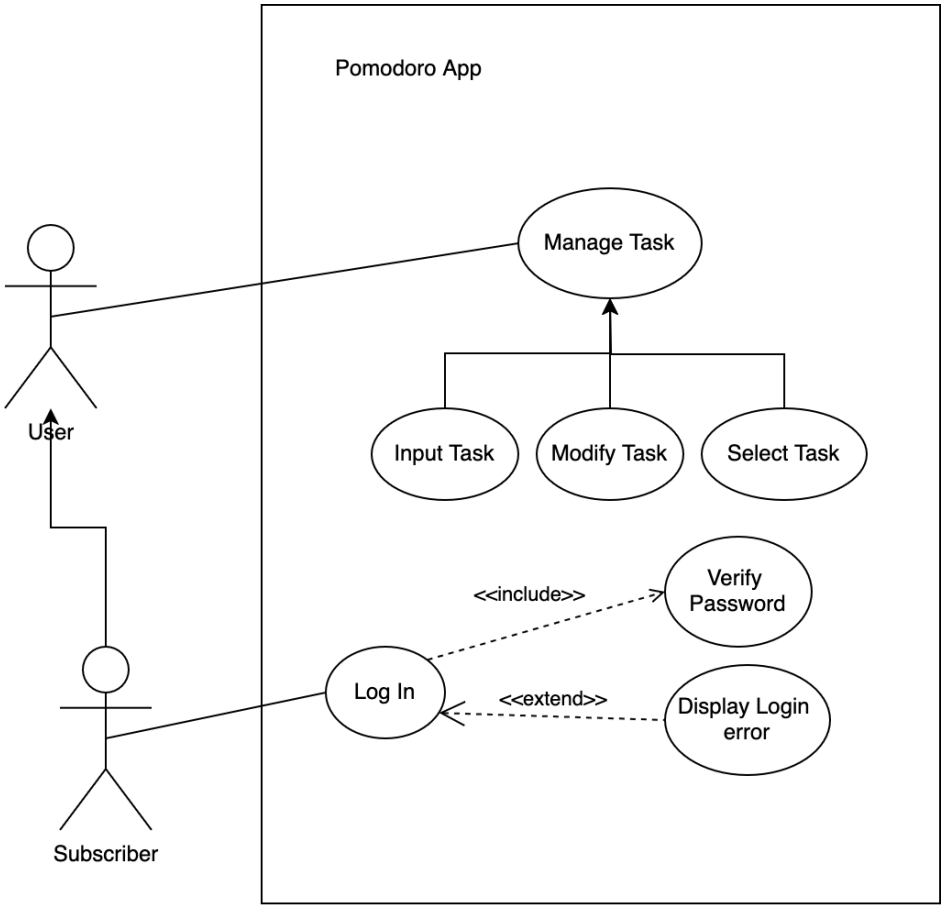


Figure 1: UML relationships between use cases