Use Cases

Use case analysis

- Integral part of RE in user-centered analysis approaches
- Uses cases
  - ways in which the system is being used
  - useful means of expressing requirements on the functionality of the system
  - written using simple prose that is understandable by a wide range of stakeholders

Use cases (examples of definitions)

- A use case describes the possible sequence of interactions among the system and one or more actors in response to some initial stimulus by one of the actors [Rumbaugh]
- A use case is a sequence of actions a system performs that yields an observable result of valuable to a particular actor
    - An actor is someone or something outside the system that interacts with the system [Kruchten: RUP]

Use cases, scenarios, use case models

- The *use case* describes a flow of events, the sequence of actions between the actor and the system, e.g. using an ATM includes depositing, withdrawing, etc.
- A *use case class* is a grouping of all possible alternate flows in a use case;
- A *scenario* is an instance of the use case class is one specific flow of events, e.g. depositing funds
- The *use case model* describes all use cases for the system (or a portion of the system) together with the set of all actors that interact with these use cases; it describes the entire system functionality

Elements of a use case

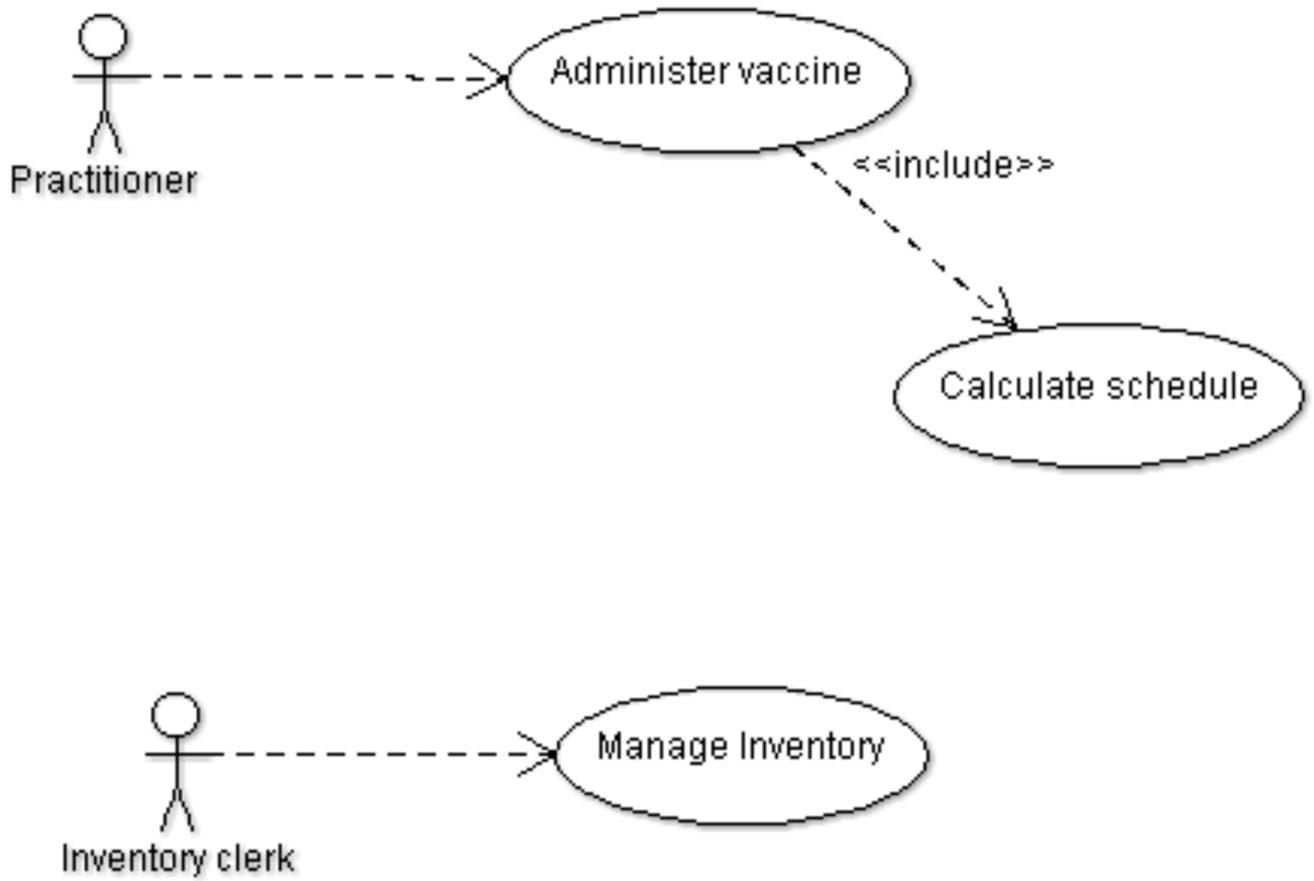
- Use case
  - title
  - summary
  - actors
  - preconditions
  - description
  - exceptions
  - postconditions

Writing use cases

- For each actor, think of fundamentally different ways in which the actor uses the system
- Make up specific scenarios for each use case
- Determine the interaction sequences
- Write a prose description of the use case
- Consider all the exceptions that can occur in handling a transaction.

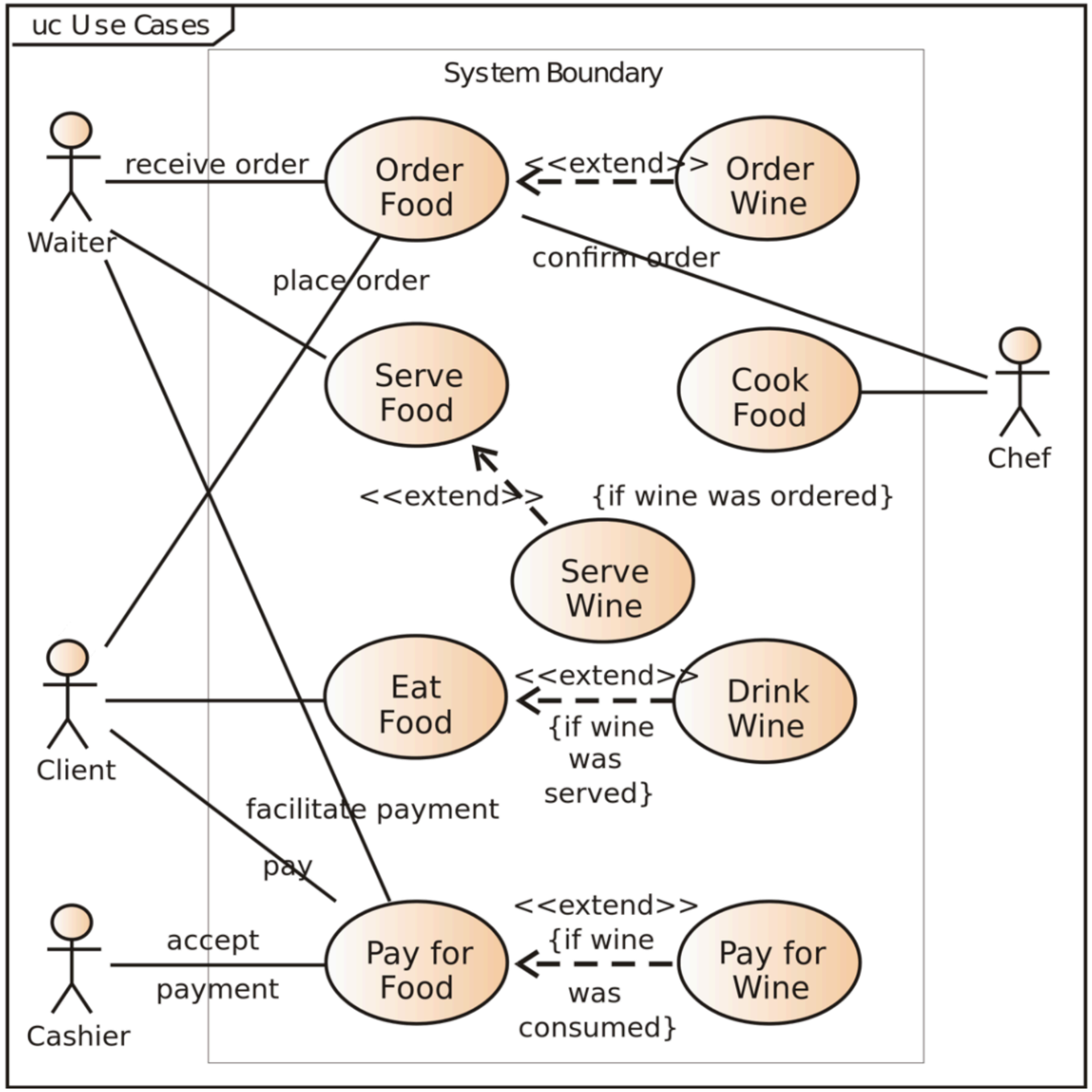
Use cases — how do they look like?

Use case diagrams



Use Case *Model*

- UML 2.0 use case diagram



Creating the use case model from Requirements

- Find actors (roles)
- Identify possible actions
- Narrow actions to specific, unique, & significant (to a given actor)
- Draw use case model with actors, use cases, and relationships
- Write each use case specification

The use case specification should include :

- Actors – primary, possibly secondary
- Preconditions – What has to have happened first
- Steps (numbered and may have sub-steps) – List the actions of the actor
- Success condition – What has to result for this to be successful
- Alternate paths (if any)
  – Branch off at the appropriate numbered step

Write a use case: more specifics

- Pick one of the actors' potential goals (as in a functional requirement)
- Write down the pre-conditions for that goal (if applicable)
- Write the main flow for the interaction
- Identify the possible failures at each step – think about the alternative scenarios they can lead to
- For the main failure scenarios create alternative flows

Actors

- There is not necessarily a 1-to-1 relationship between users and actors. (All actors are users but not all users are actors)
- Actors are not necessarily human users, they can be systems.
- Actors are defined by the way the system recognizes them, through the use cases they participate in.
- Tip: Think about the end-users you have defined and whether they will use exactly the same use cases in the same way. If yes, they are the same actor.

Use case Main flow

- It’s a story of an actor using the system to accomplish a goal. It is an outline of an expected sequence of events.
- It always starts with an actor’s action. It consists of numbered steps.
- Represents what the sequence will be if the user and the system behave in the way that the system was designed for.
- Tip: Pick a purpose for which an actor might use your system and think of all the necessary steps they will have to go through, and how the system will respond to each of their actions.

Alternate flows

- Alternative flows correspond to “if” statements. They cover conditions under which either the actor or the system will behave differently than expected, and that will change the sequence of events in their interaction.
- Alternative flows are not standalone, they exist only as exceptions of the main flow.
- They are branches off the main flow, they consist of numbered steps and may or may not return to it.

The “Include” relationship

- Functionality (sequence of steps) that is repeated in more than 2 use cases, can be factored out into a separate use case. That will make it easier to reuse.
- The included use case is a standalone use case, complete with its own alternative flows, if applicable. The original use case cannot function without the included one, it will be missing steps.
- Syntax:
  “Include **<use case name>**”.