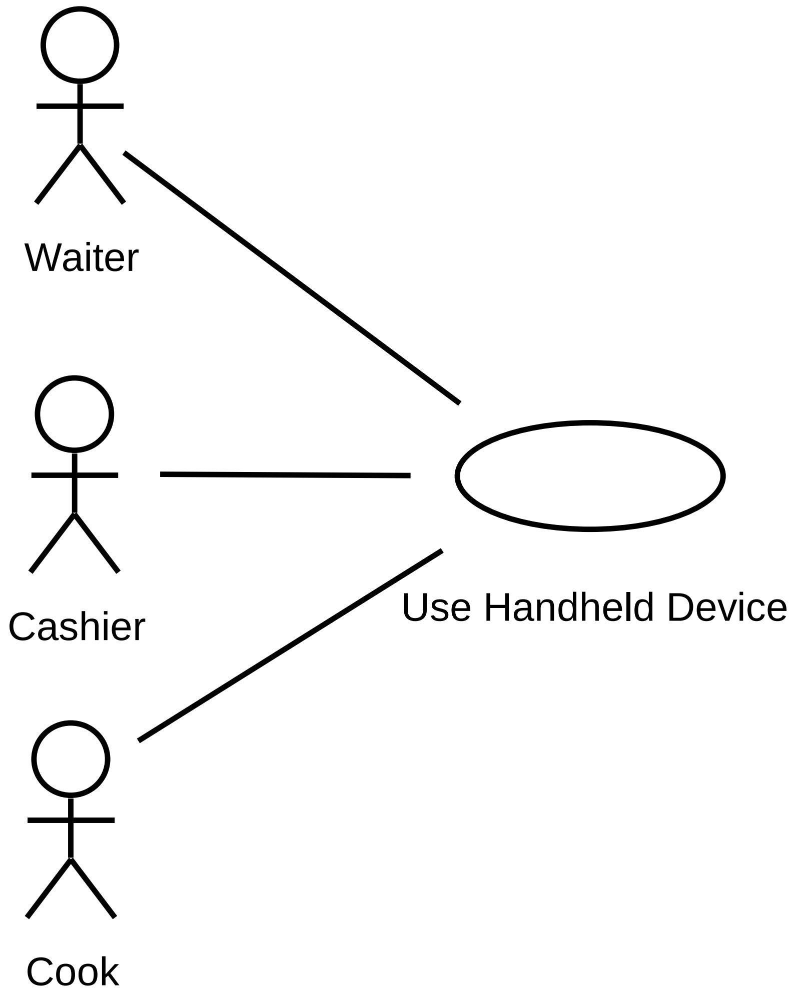
Restaurant Management System

Restaurant Management System

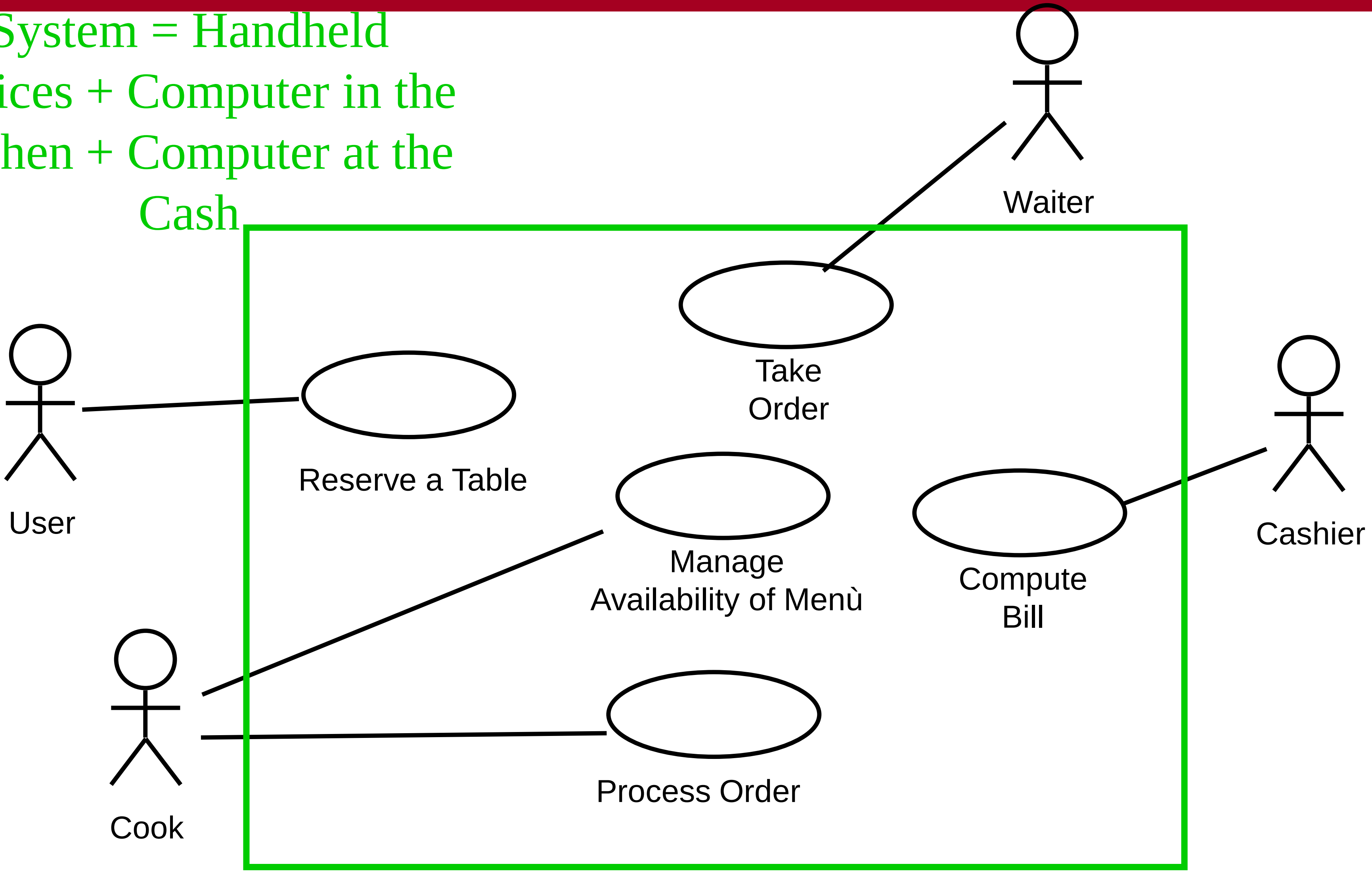
We have been asked to build a system to automate the ordering and billing activities of a restaurant. The system is distributed: waiters and waitresses are provided with handheld devices to take orders. The handheld devices communicate orders to the kitchen and to the cashier. The handheld devices receive real-time information about availability of the different items in the menu. Once placed, orders can be changed by the customers, within a time frame from the order (5 minutes) or after the time-out, if the corresponding order has not yet been processed by the Cook. The system computes bills and is also used to manage reservations of tables. Reservations can either happen by phone or via the internet.

Restaurant Management System



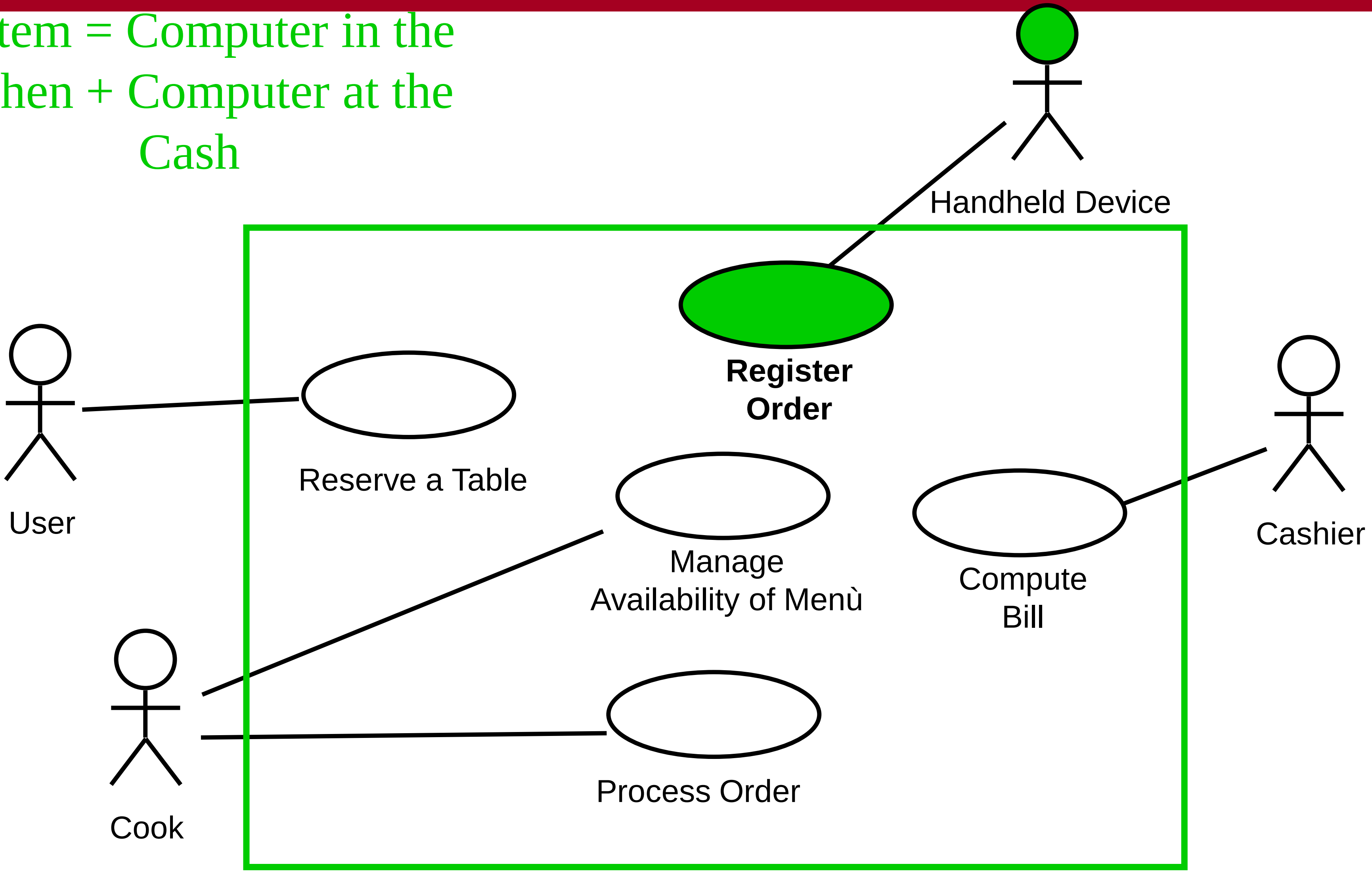
Resturant Management System

System = Handheld devices + Computer in the kitchen + Computer at the Cash



Resturant Management System

System = Computer in the kitchen + Computer at the Cash

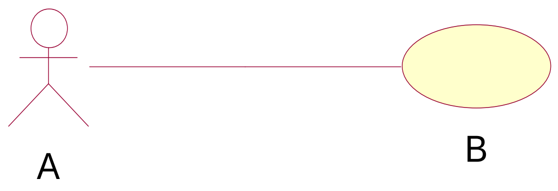


Software Engineering

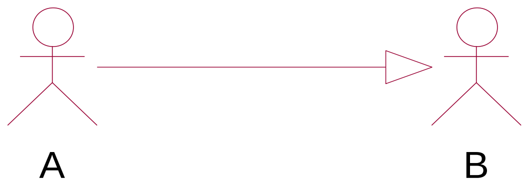
Use Case Diagram: Relationships

Use Case Diagram: Relationships

- **Actors Relationships**

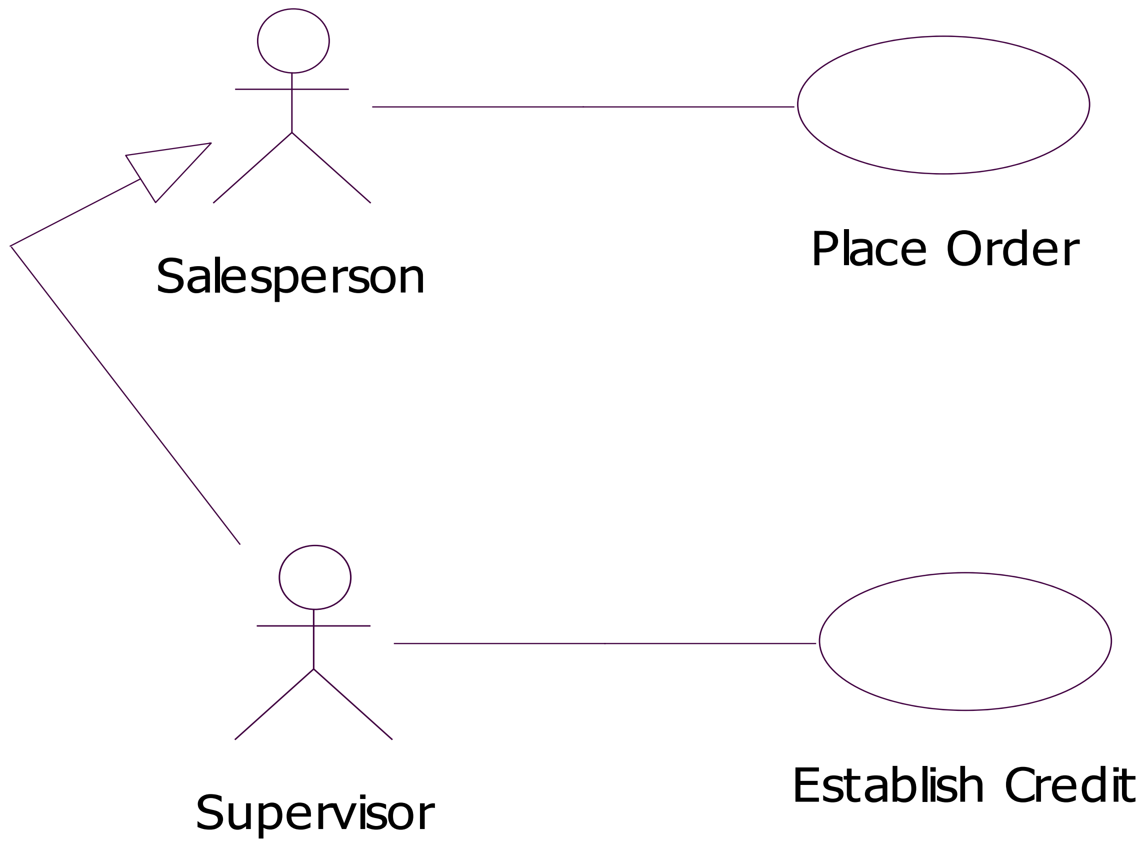


Association:
The participation of an actor in a use case. **This is the only relationship between Actors and Use Cases**



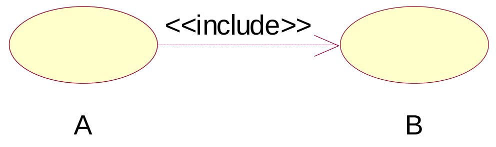
Generalization:
A generalizes B implies that A can perform the use cases of B.

Actors Relationships: Example



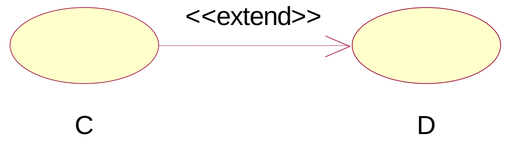
Use Case Diagrams: Relationships

- **Use Case Relationships**



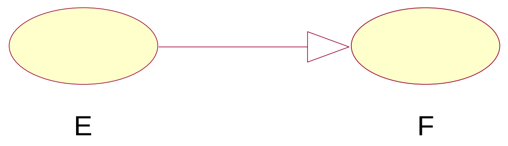
Include:

a step of A is the execution of B



Extend:

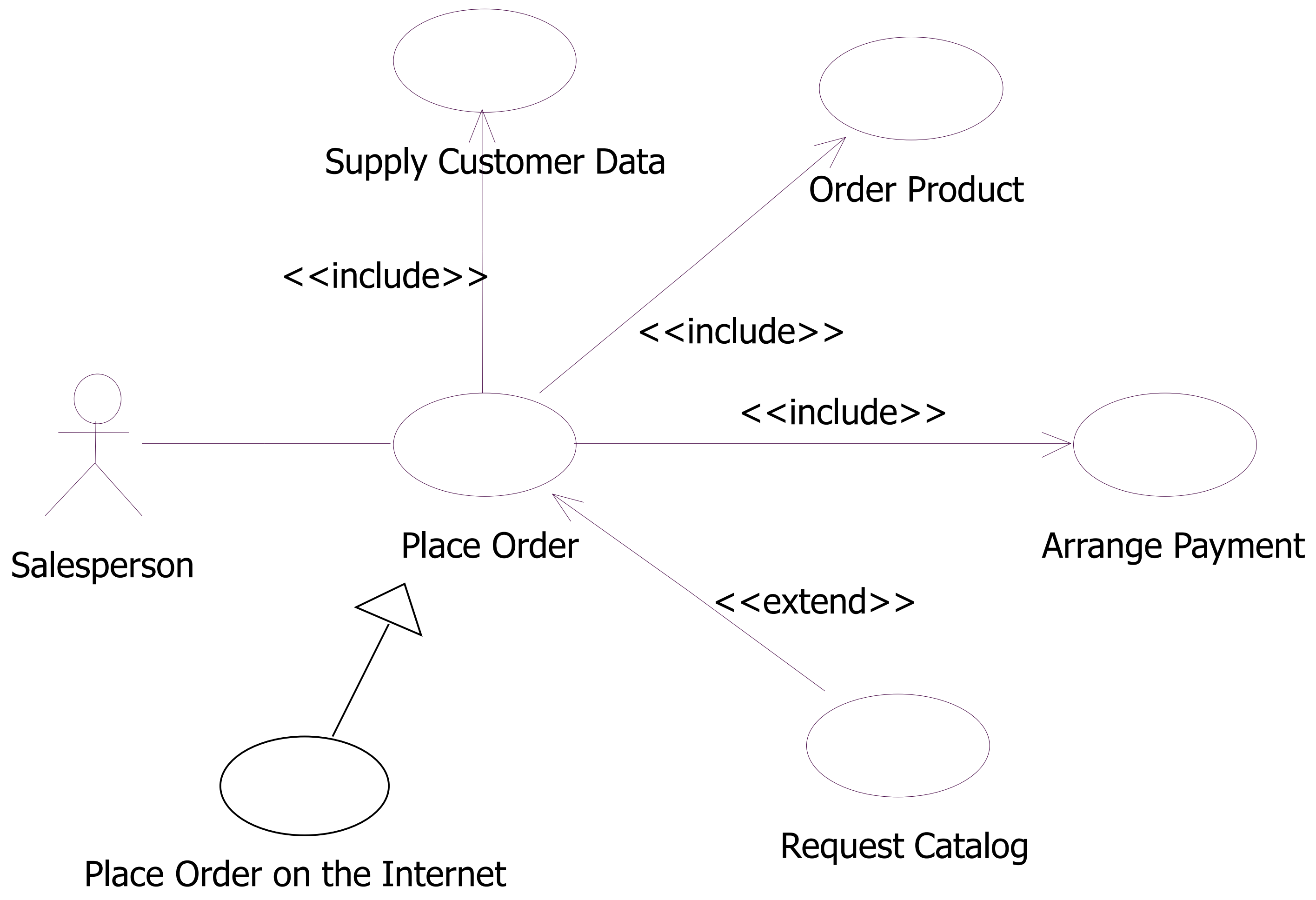
C does a little more than D does



Generalize:

E does something more specific than F

Use Case Relationships: Example



Hint 1: Actors

- Actors are external to the system:
  - They help setting the boundaries of the system…
  - … as a consequence: the “system” itself can’t be actor!
- Actors need not be human:
  - Authentication System, Billing System, …

Hint 2: Actors

Sometimes you need to represent functionality that take place on a regular basis (e.g. backup)

Who is the actor for such a use case?

- **Solution 1.** Who benefits from the execution of the regular action (System Administrator)
- **Solution 2.** What triggers the action, namely: “Time” or “Clock”.

… both 1 and 2 are ok!

Software Engineering

Organizing your Requirements

Organizing Your Requirements

To limit the number of Use Cases

- Make it an Iterative Process

Factorize Use Cases and Actors using relationships

- Organize and Classify your Use Cases

Package Diagram or Subsystem

“Strategic” and “System” Use Cases

Revise and Factorize UC/UCD

- **In the Use Case Diagrams**
  - Include
  - Extend

- **In the Use Case**
  - Description
  - Extension
  - Exceptions

1. The user executes any of the use cases “Save”, “Load”, …

[extension 1] if The User chooses “Save As”, then he can specify a name for the file.

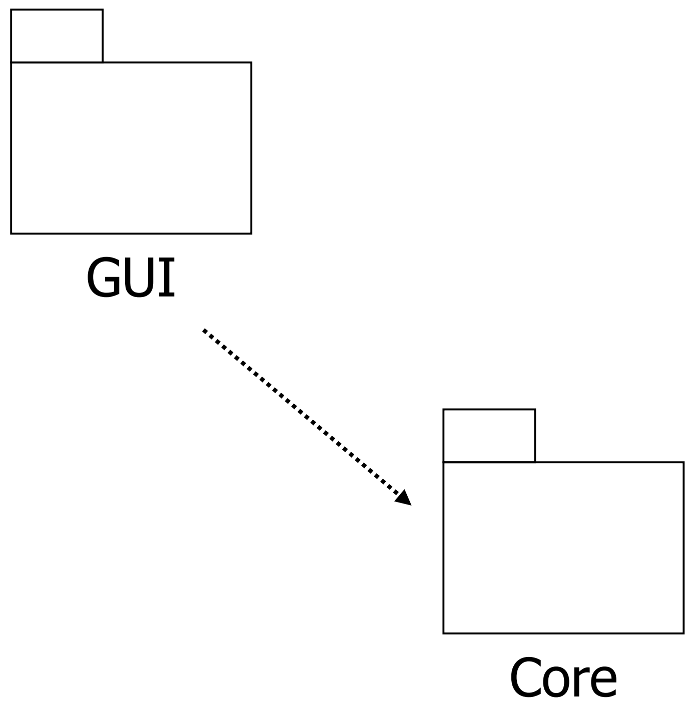
1. The user selects a name for the file **[exception 1]**

[exception 1] If the file name contains “/” an error is reported.

Organize and Classify UC/UCD

Package Diagrams or Subsystem

Packages allows to split diagrams in smaller parts.



Key Ingredients:

- Package or Subsystem
- Dependency

Organize and Classify UC/UCD

- Strategic Use Cases

High-level use cases. They provide an overview of the system.

Useful to communicate with the stakeholder.

- System Use Cases

Lower-level use cases. They refine the strategic use cases and provide a more fine grained view of the system.

Use them to decide how to allocate the work.

Organize and Classify UC/UCD

… caveat:

“Strategic” and “System” are just one of the ways (possibly the simplest) to organize your requirements.

Software Engineering

Organizing your Requirements: example

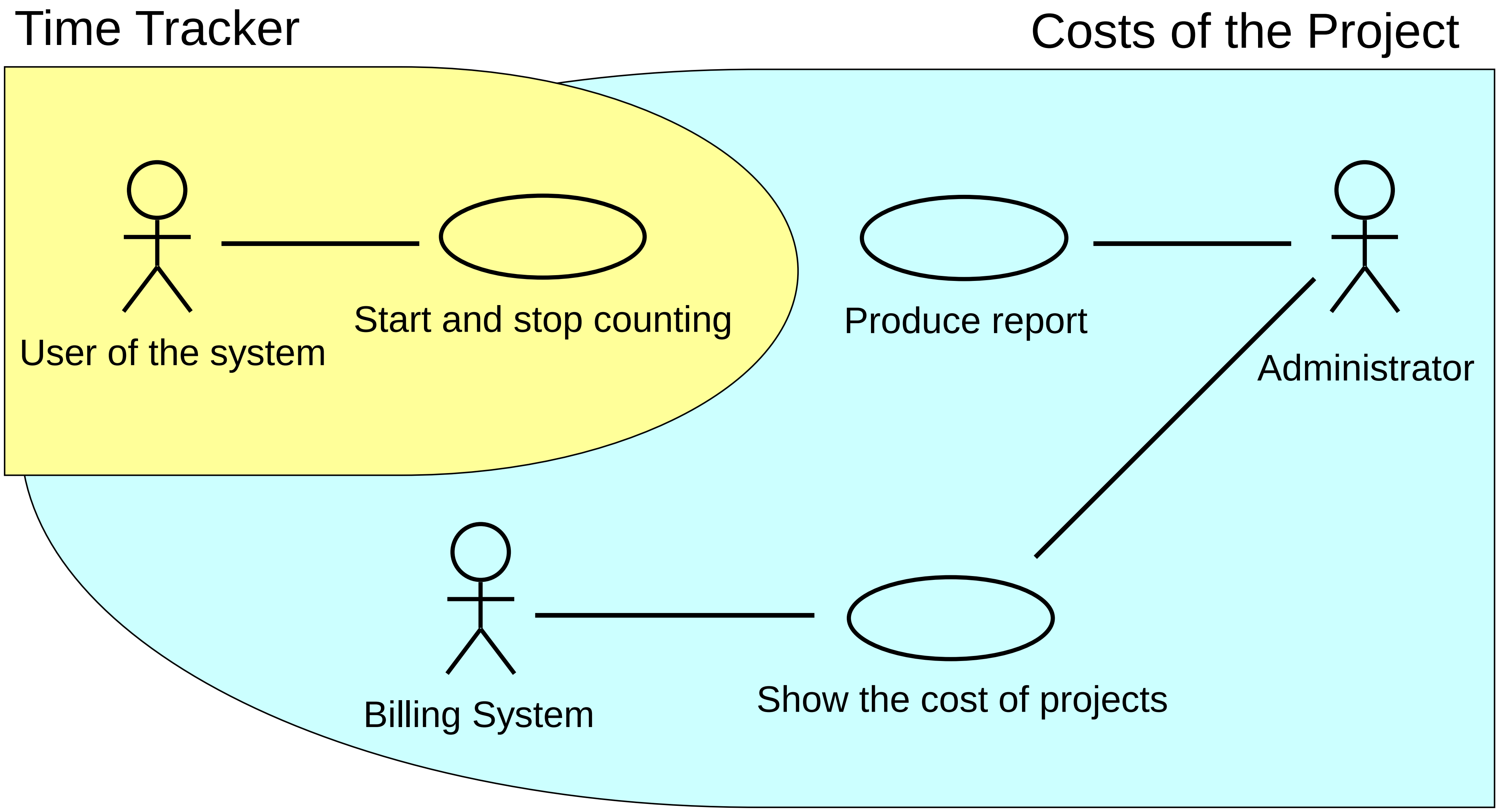
Example: Time Tracker

We have been contacted by a small software firm.

They want us to build a system for letting employees track how they spend their time when working on a computer. The idea is that of a stop-watch: the users of the system can start and stop counting the time spent on different activities; the system logs such activities and can be used to produce reports.

The system can also be integrated with a billing system. The billing system receives all the information about the time spent by programmers on the different projects and computes the cost of projects. This information is then used to charge clients.

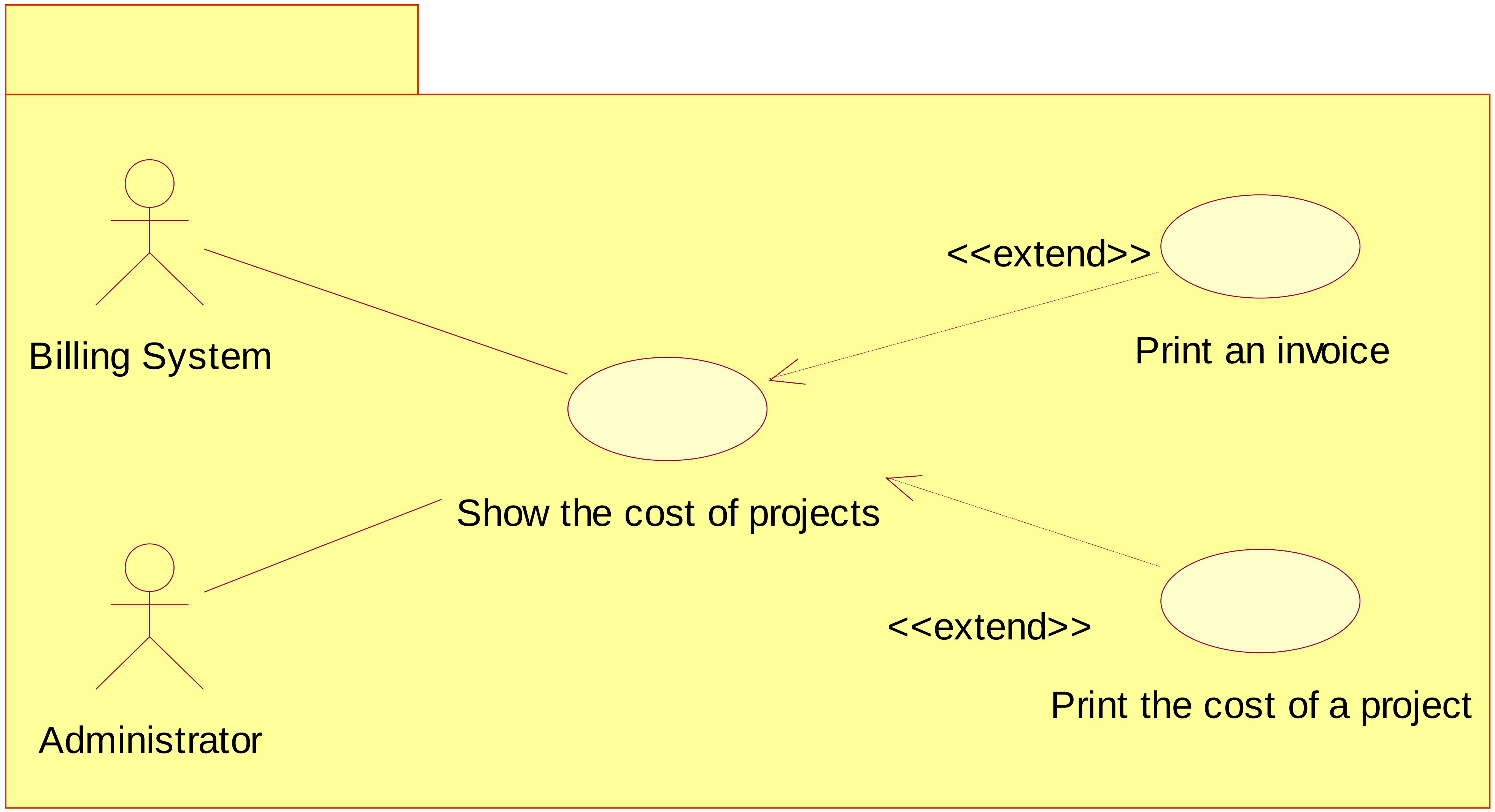
Use Case Diagram



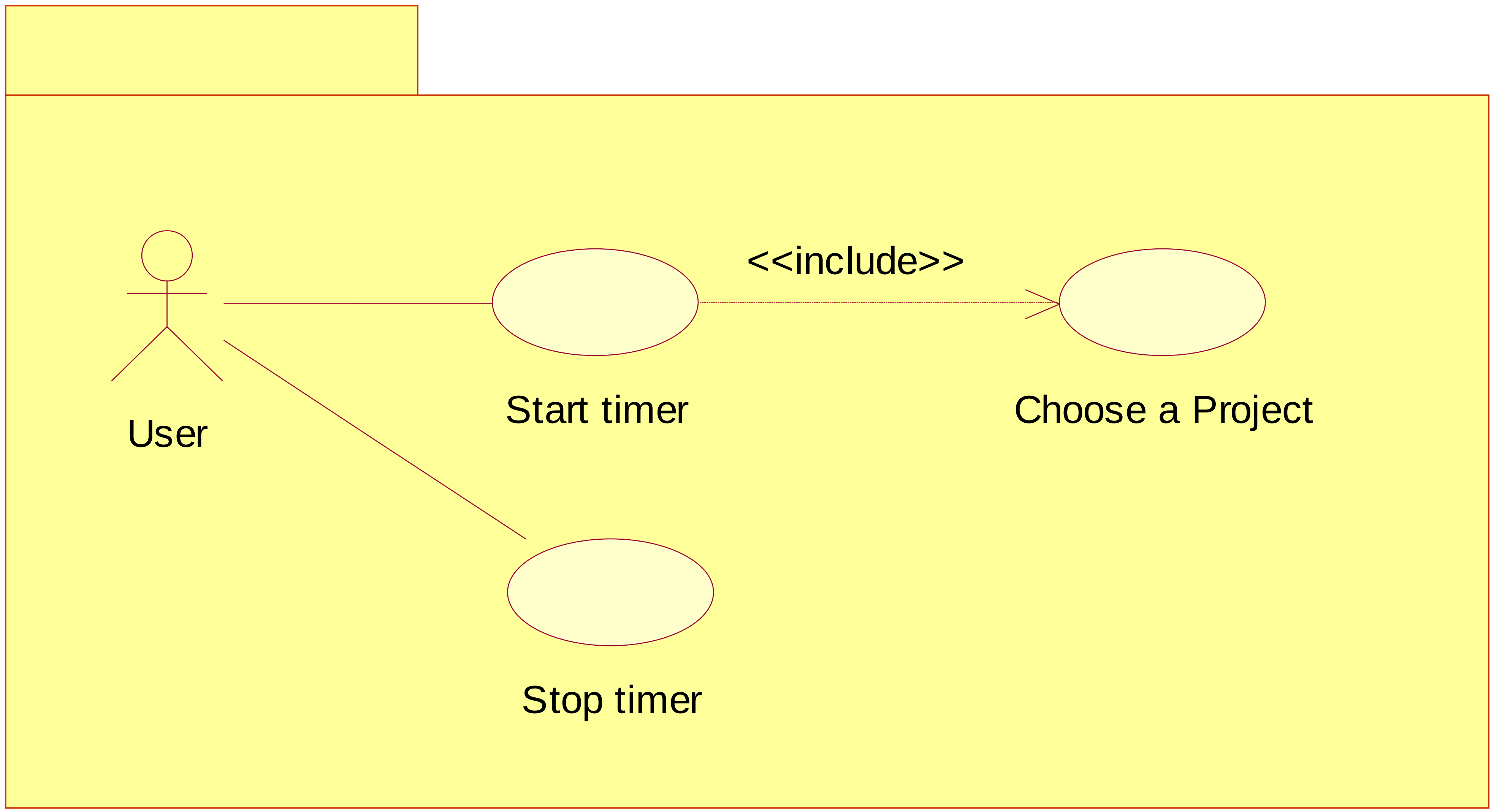
Use Case Diagram with Subsystems

Costs of the projects
Time Tracker

Costs of the Projects



Time Tracker



Exercise:

Reservation System

We want to build a system for the electronic reservation of seats of a group of movie theatres. The users can either buy tickets for a particular show or buy subscriptions, also by phone. Payments can be performed in cash or by credit card. A supervisor can print the list of seats available for a particular show and see the status of sales performed so far.

A secretary updates the list of shows (by adding/deleting/...)