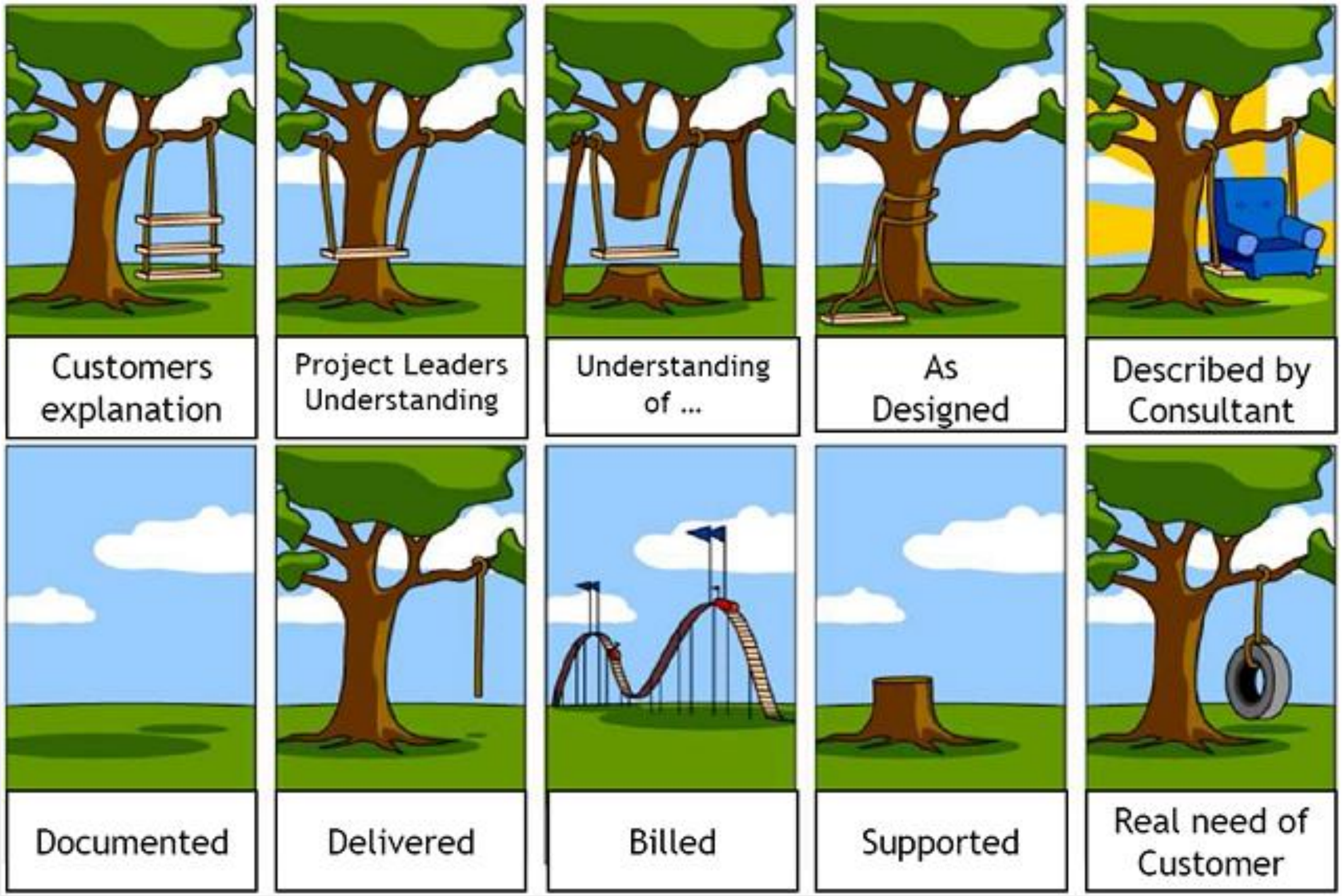
Glachs D., Lettner J., Kiefel A.

Formatvorlage des Untertitelmasters durch Klicken bearbeiten

Software Design
Session 1 –
UML Use Case and Activity Diagrams

ITS Software Design – Why?

Fachhochschule Salzburg University of Applied Sciences

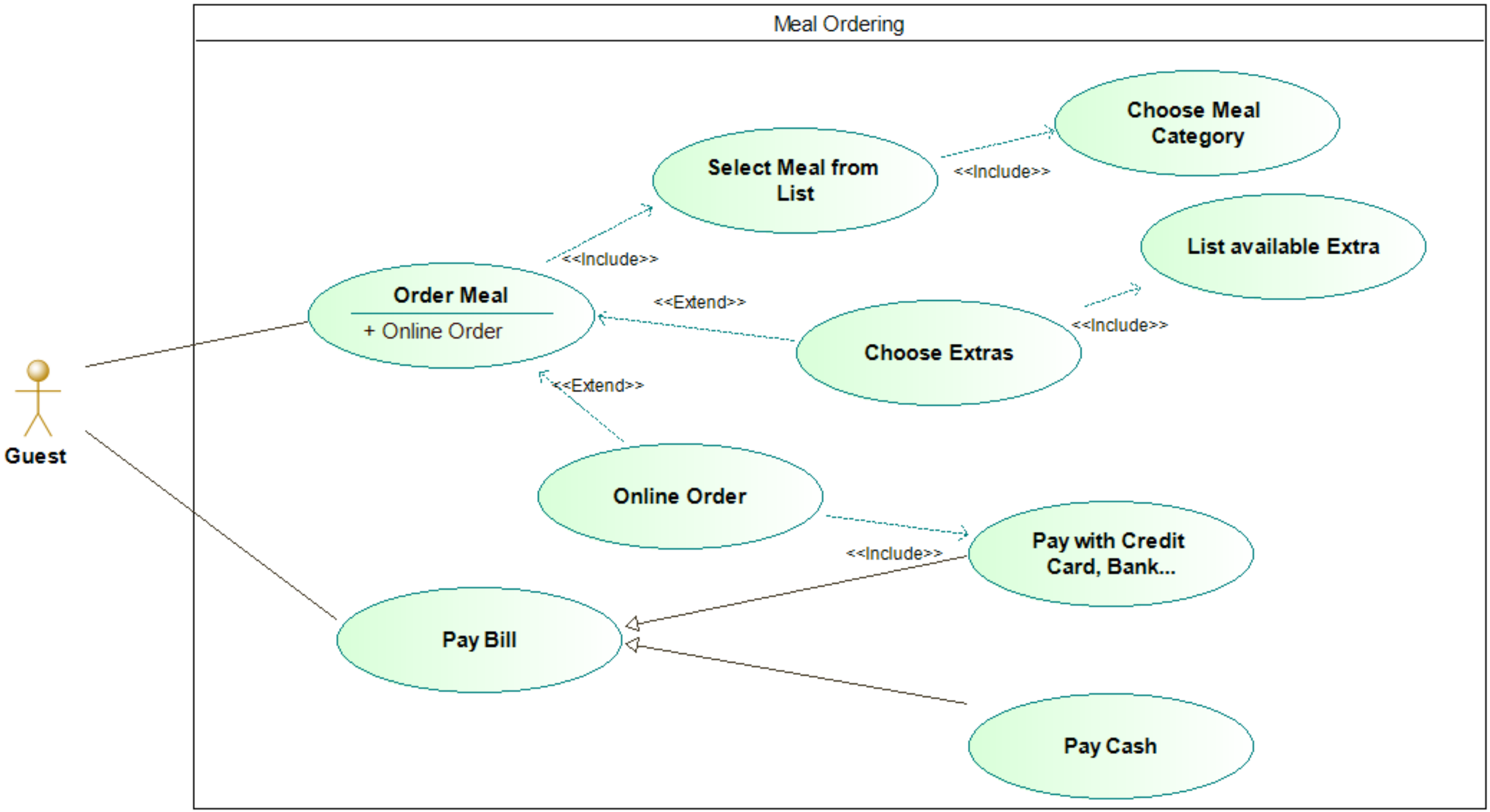


USE CASE DIAGRAMS

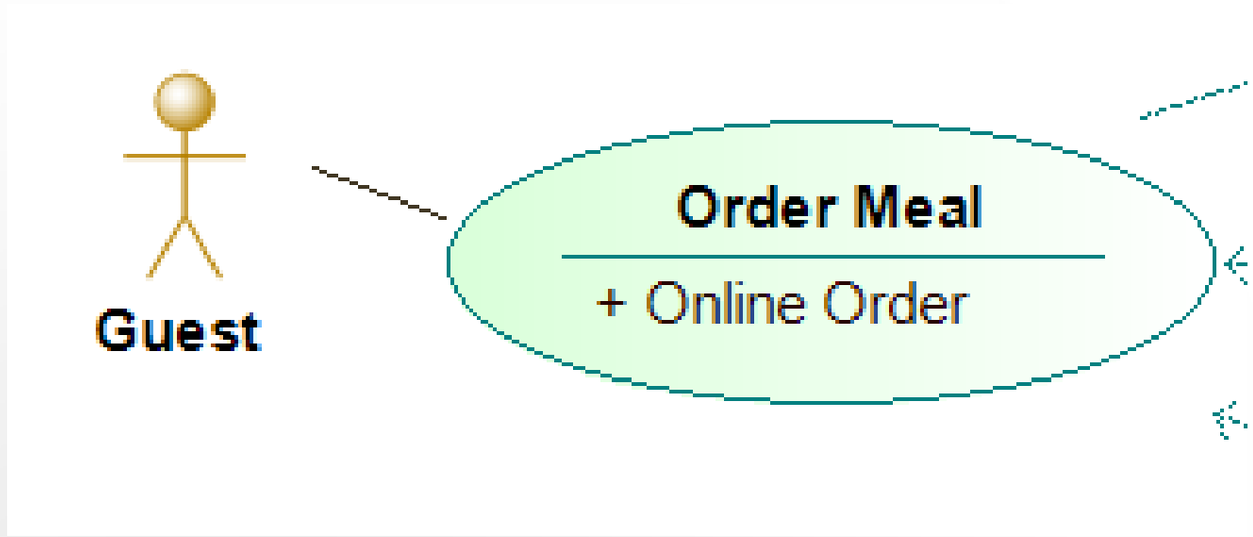
ITS Software Design
Use Case Diagrams

Fachhochschule Salzburg University of Applied Sciences

ITS Software Design
Use Case Diagrams

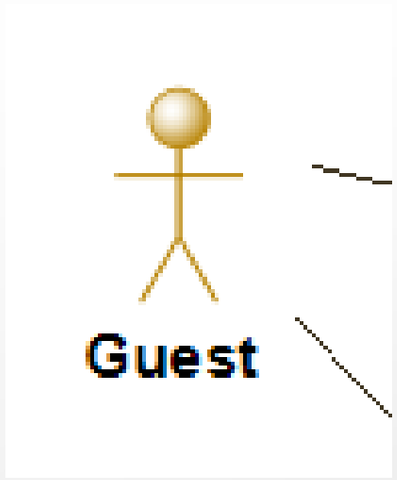


- What is a Use Case?
  - A use case is a single **unit of meaningful work**
  - High-level view of behaviour **observable from outside** of the system
  - **Identify users, user groups and systems** interacting with the system
  - **Define the expected functionality of the system** based on system requirements defined by stakeholders
  - **Actors are stakeholders!**

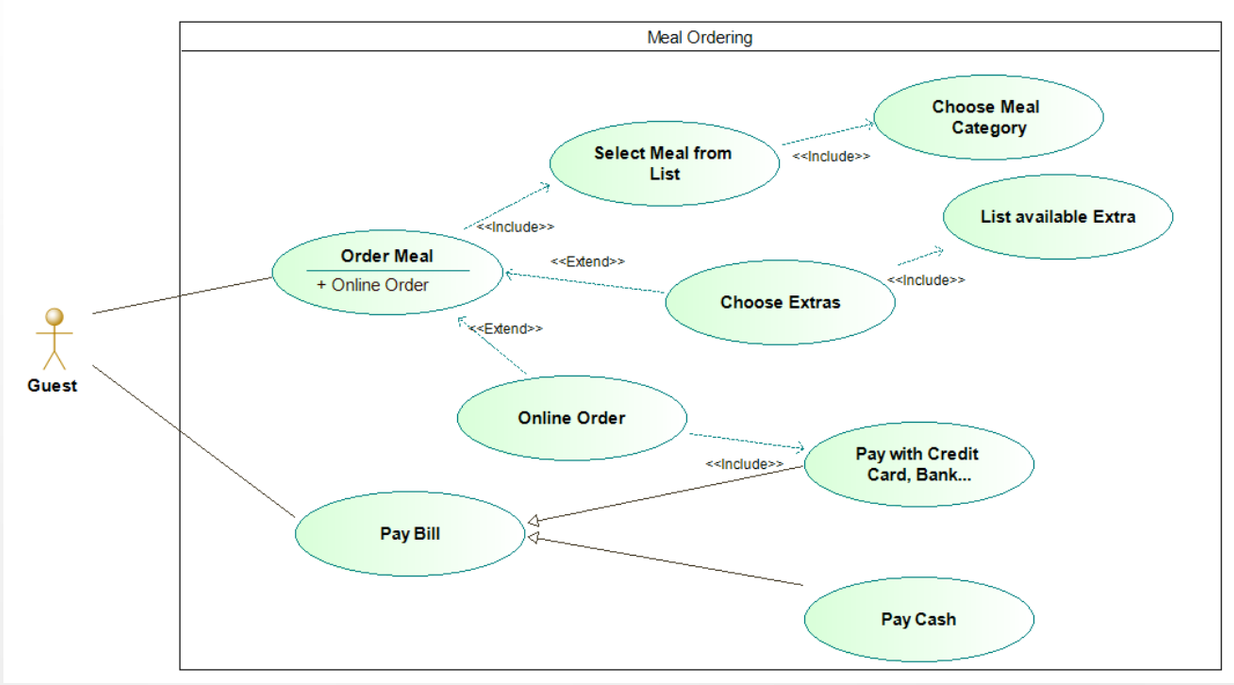


- What is an Actor?

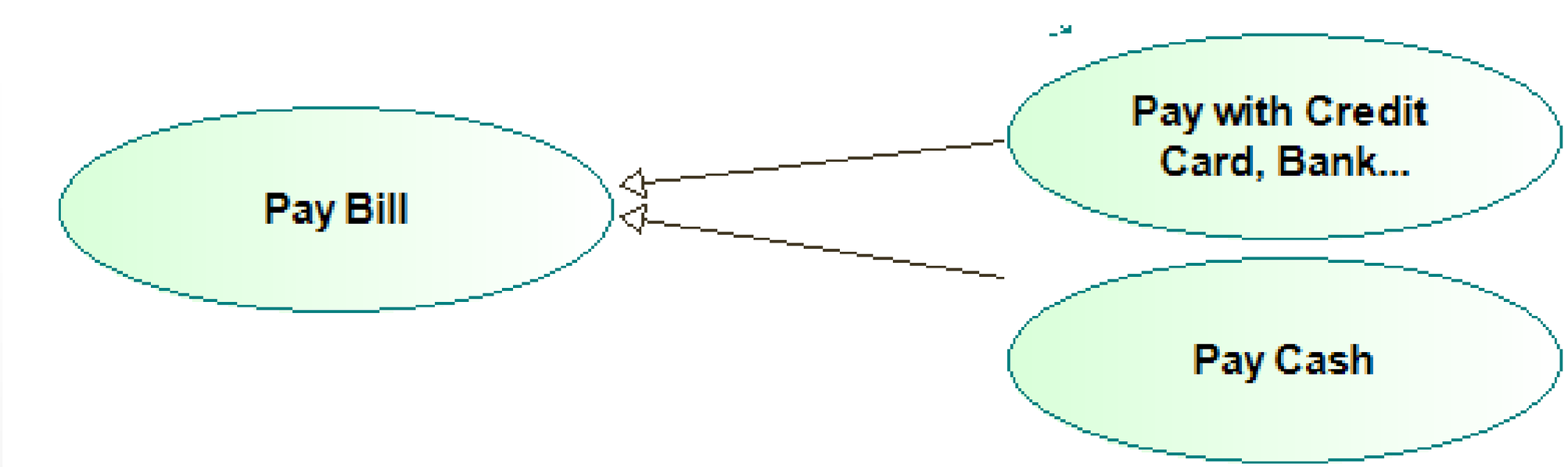
  - An **actor is human or system entities** that interact with the system in context of a use case.
  - The set of **use cases an actor is connected to define their overall role in the system** and the scope of their action



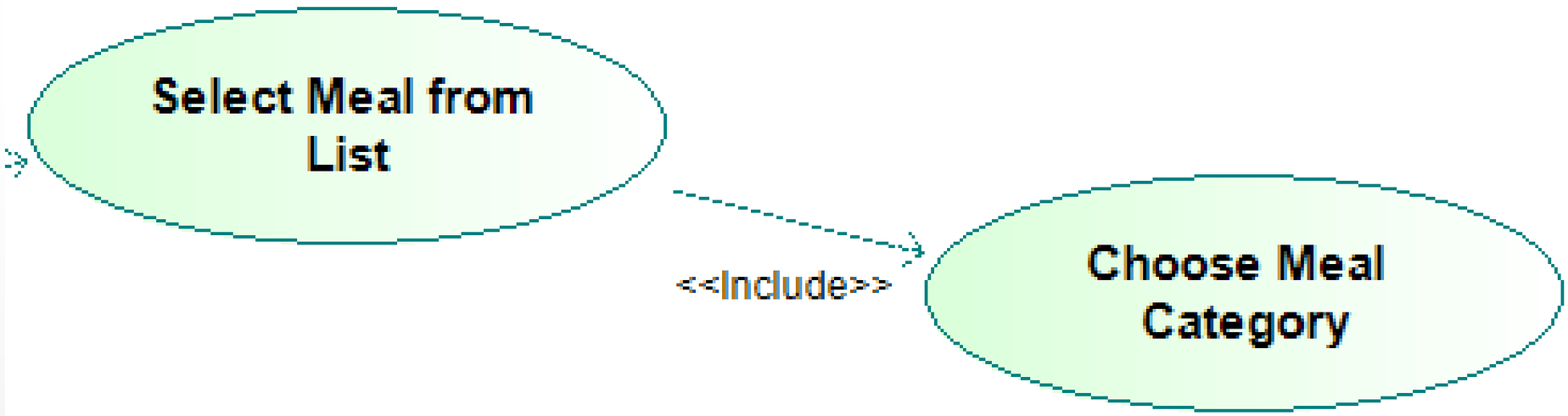
- What is a System Boundary?
  - The system boundary limits the scope of a system
  - **Use cases** are displayed within the system boundaries to show that this **functionality is part of the system**
  - **Actors (human or an other system)** are displayed outside the system boundries as they are **not part of the system** but interacting with it → Interfaces needed to make interaction possible: Graphical UI, Webservices, ...



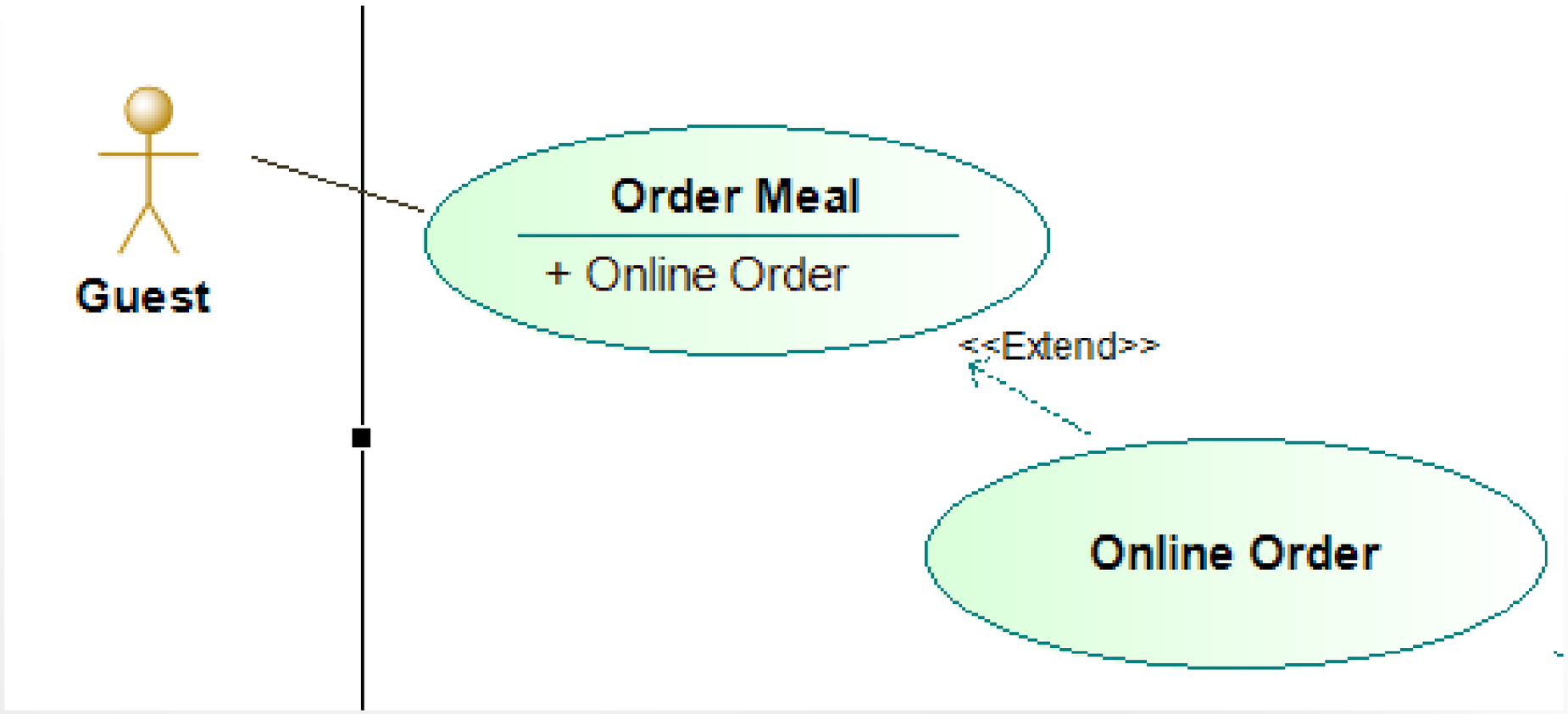
- Inheritence relationship
  - General behaviour can be grouped in global use cases
  - Use cases with detailed behaviour are related to the general use cases via inheritence.



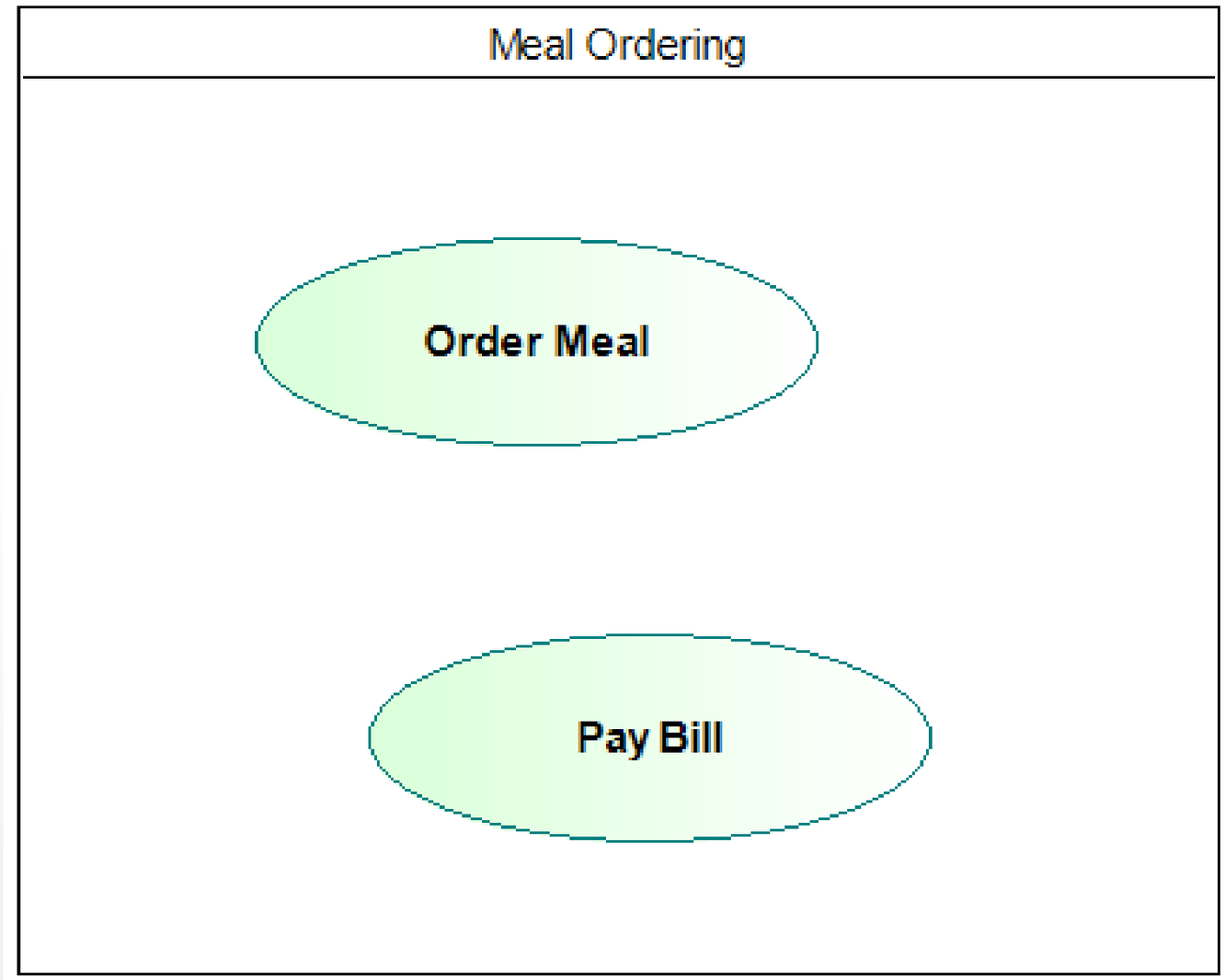
- Include relationship
  - Some use cases **include one or more use cases**.
  - The included use cases describe one action in the use case more detailed.



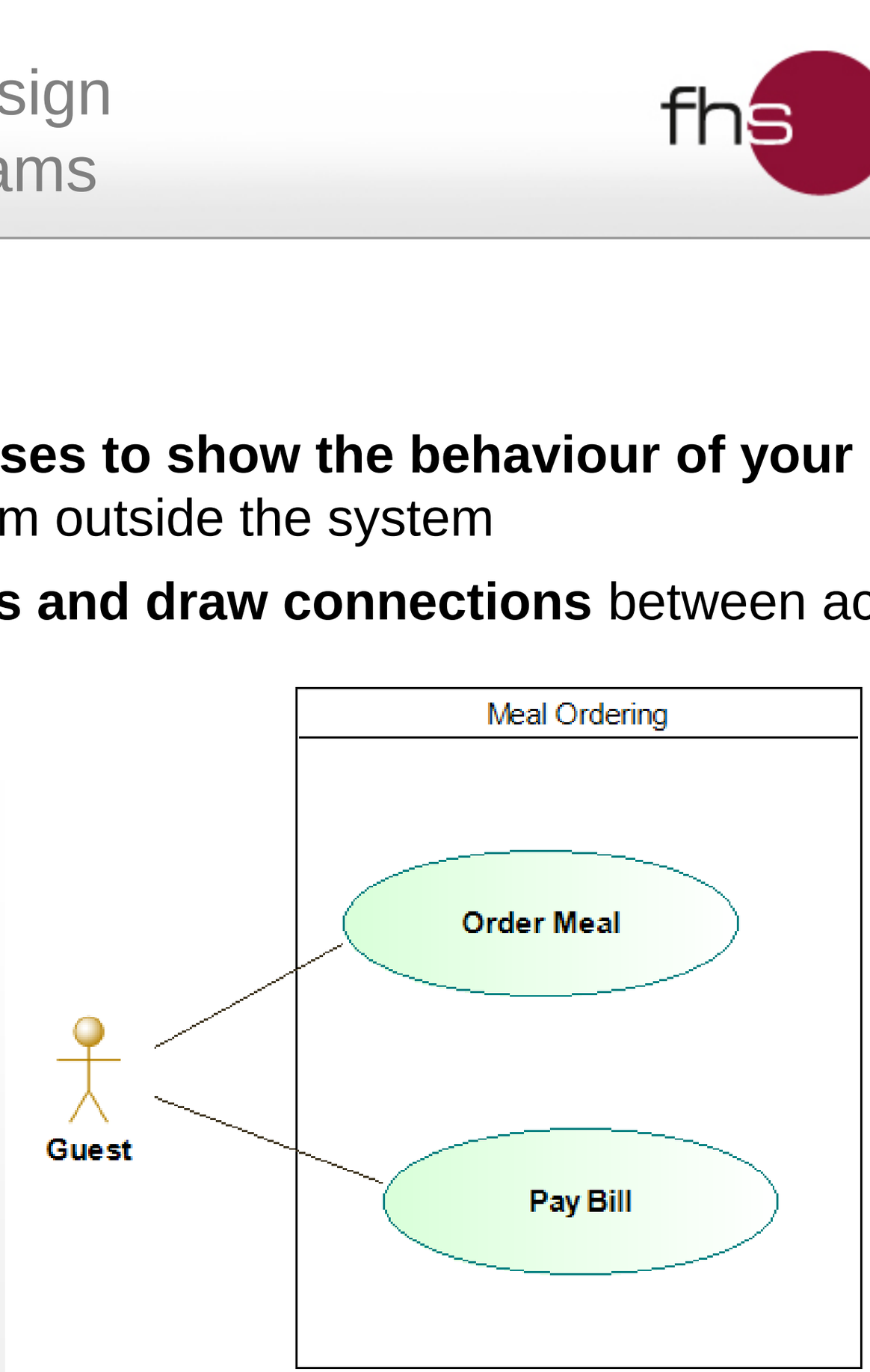
- Extends relationship
  - The extends relationship shows that a use case can be extended by the functionality of an other use case.
  - The extension of a use case only happens under certain conditions described in the extension point.



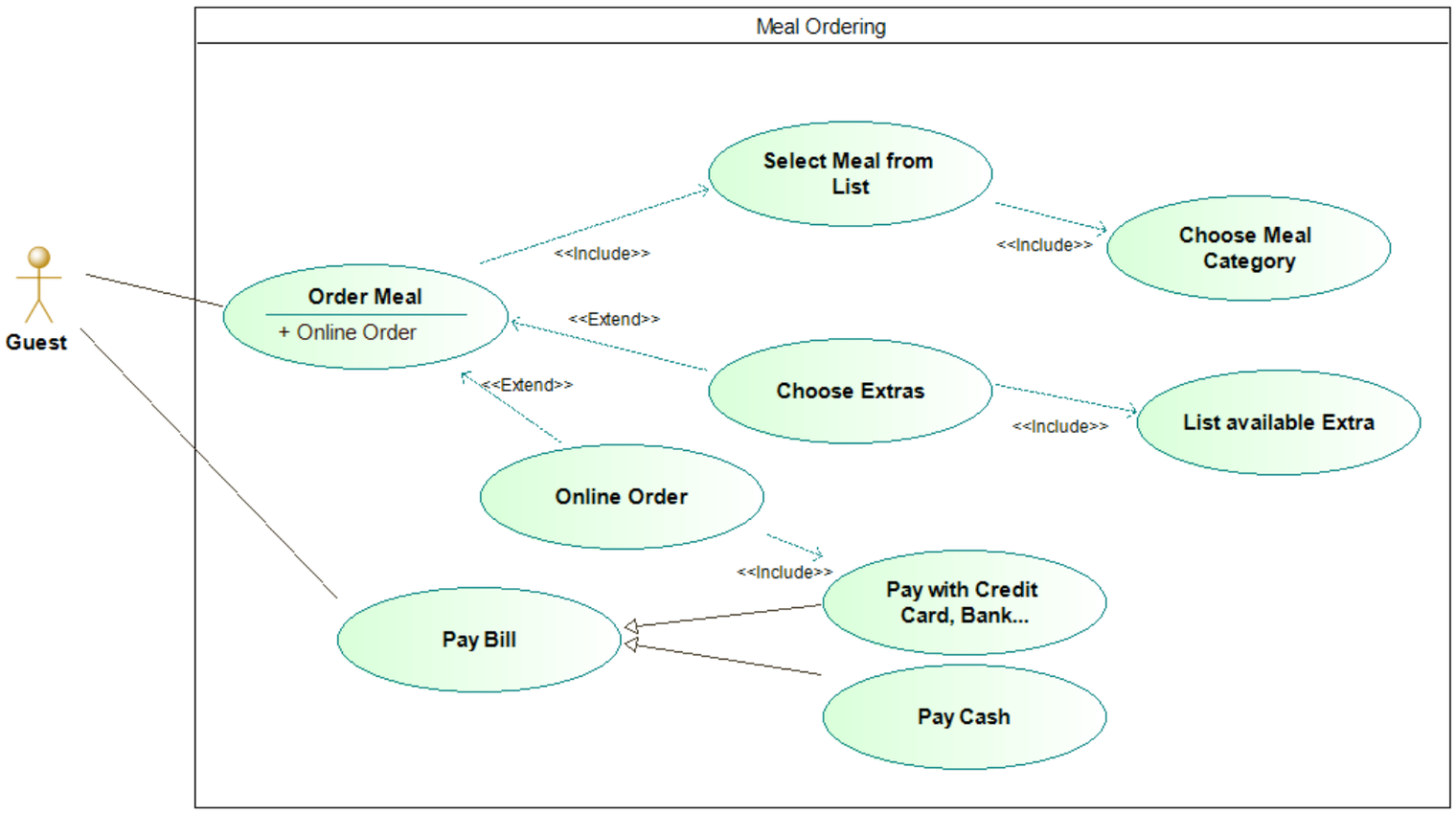
- Design steps
  - Create **use cases to show the behaviour of your system** observable from outside the system



- Design steps
  - Create **use cases to show the behaviour of your system** observable from outside the system
  - **Identify actors and draw connections** between actors and use cases



- Design steps
  - Create **use cases to show the behaviour of your system** observable from outside the system
  - **Identify actors and draw connections** between actors and use cases,
  - **Group common functions to common use cases**, define



- Design steps
  - **Create use cases to show the behaviour of your system** observable from outside the system
  - **Identify actors and draw connections** between actors and use cases,
  - **Group common functions** to common use cases
  - **Describe the use case in detail** – Create use case descriptions

Use Case ID:			
Use Case Name:			
Created By:		Last Updated By:	
Date Created:		Date Last Updated:	

Actors:	
Description:	

Trigger:	
Pre-Conditions:	1.
Post-Conditions:	1.
Normal Flow:	1.
Alternative Flows:	
Exceptions:	
Includes:	
Priority:	
Frequency of Use:	
Business Rules:	
Special Requirements:	
Assumptions:	
Notes and Issues:	

Fachhochschule Salzburg University of Applied Sciences

- # School Administration System
  - Create a use case diagram outlining the CDG's administration system by taking the following constraints into account:
    - The administration staff can manage students, teachers, classrooms, schedules etc. working groups etc. in a dedicated administration application

A student, teacher may ask for her/his personal schedule via a public web page – Feature: using natural language

(Hint: https://api.ai/)

- Create Use Case Diagram
  - Login with user and password
    - Student
    - Teacher
  - Show schedule preview for next 2 weeks
  - Security:
    - Students see only their own schedule
    - Teachers can see all schedules from students and teachers

Fachhochschule Salzburg University of Applied Sciences

ACTIVITY DIAGRAMS

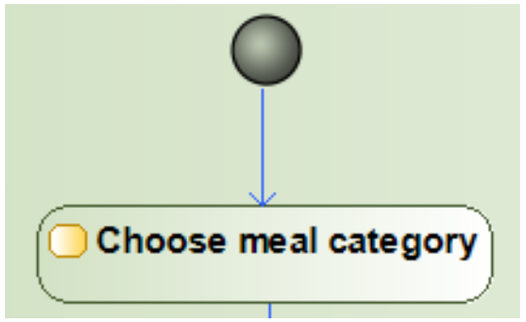
- What is an Activity Diagram good for?
  - Make the interaction process visible
  - Process start, process steps and process end
  - What is the expected result at the end of the process?
  - Who is triggering the process?
  - How is the process flow – sequence of actions?
  - When and how does the process end?

- Define the expected behaviour of the system more detailed

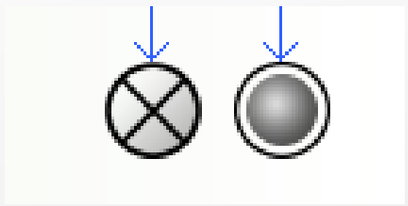
Hint: Switch from a bird‘s eye view – outside the system – to a inside view of the process

- Activity and Action
  - An **activity describes a bigger task** – on top level it describes a use case
  - An action is a smaller task in the activity flow.
  - **A task** is defined by
    - **Which object** is handled?
    - **What** has to be done?
    - **By whom**?
- Each action needs to receive **information at the beginning**
- **Information** can also be a **result of an action**

- Inital Node, Flow, Final Node
  - The **initial node shows the starting point** of the process
  - The **flow indicates the direction** of the process flow, it also shows the flow of information and, in case of an object flow, the transfer of objects like documents or goods

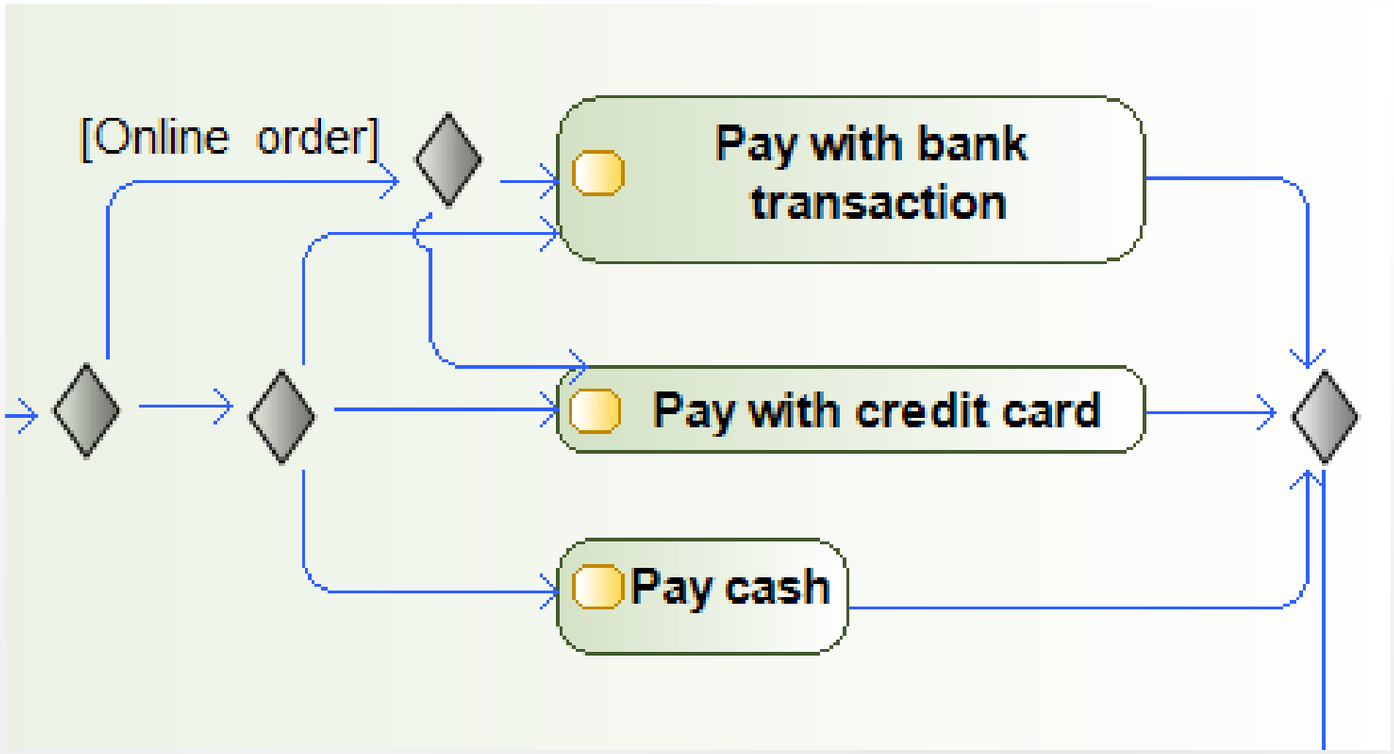


  - The **final node indicates the end of the process**. At the end of the main path of the process, also called „Happy Path“ or „Normal Flow“ the expected output of the process is reached.

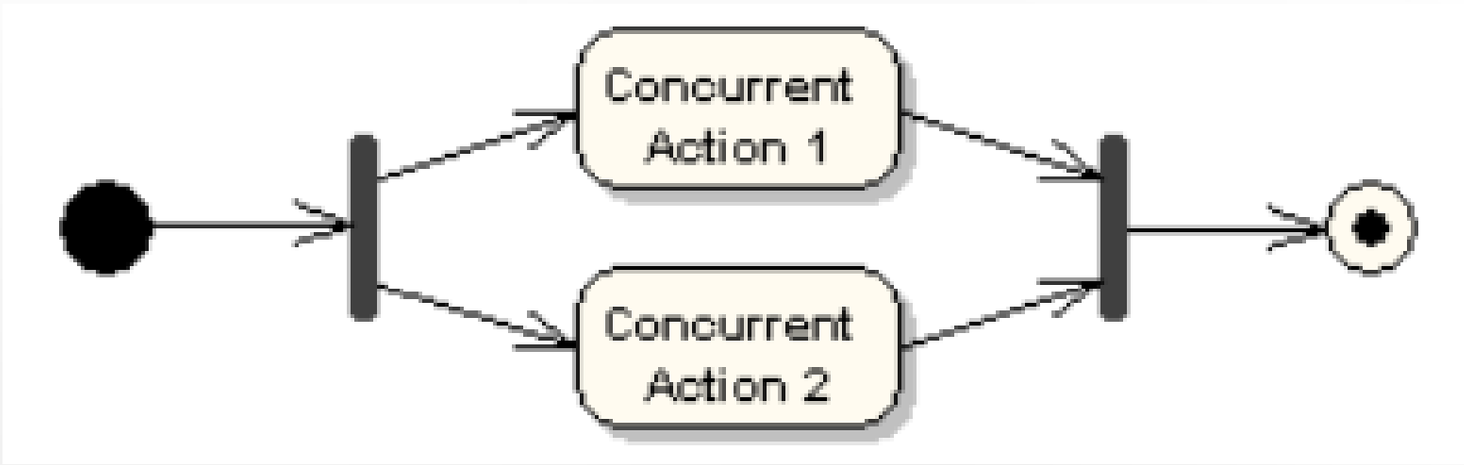


  - **A top level process flow starts and ends at the user**

- Decision and Merge Node
  - **Logical OR Relationship** – next to each other
  - **One** of two or more possible actions **has to be executed**
  - **Decision node splits the flow** into several flows
  - **Merge node ingerates the flows** back to one flow
- Shaped as diamond

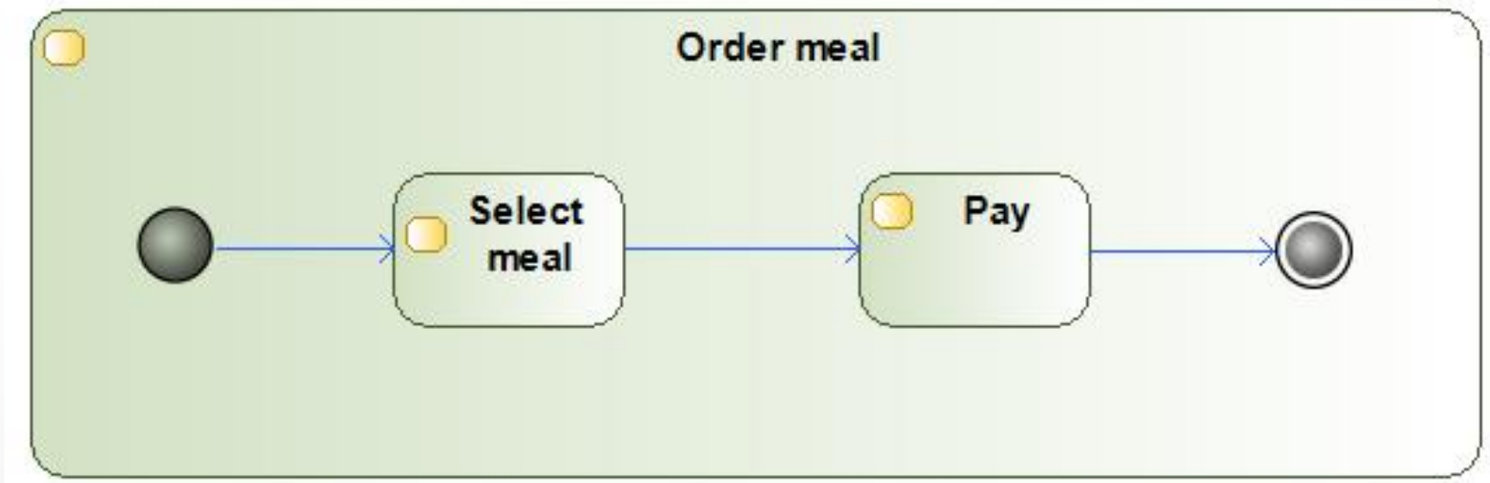


- Fork and Join Node
  - **Logical AND** relationship next to each other
  - **Fulfill** two or more actions/tasks **in parallel**
  - All actions **start together**
  - **All actions end together**
    - **Next task is not stared before all concurrent actions are done**
- Shaped as horizontal or vertical bar

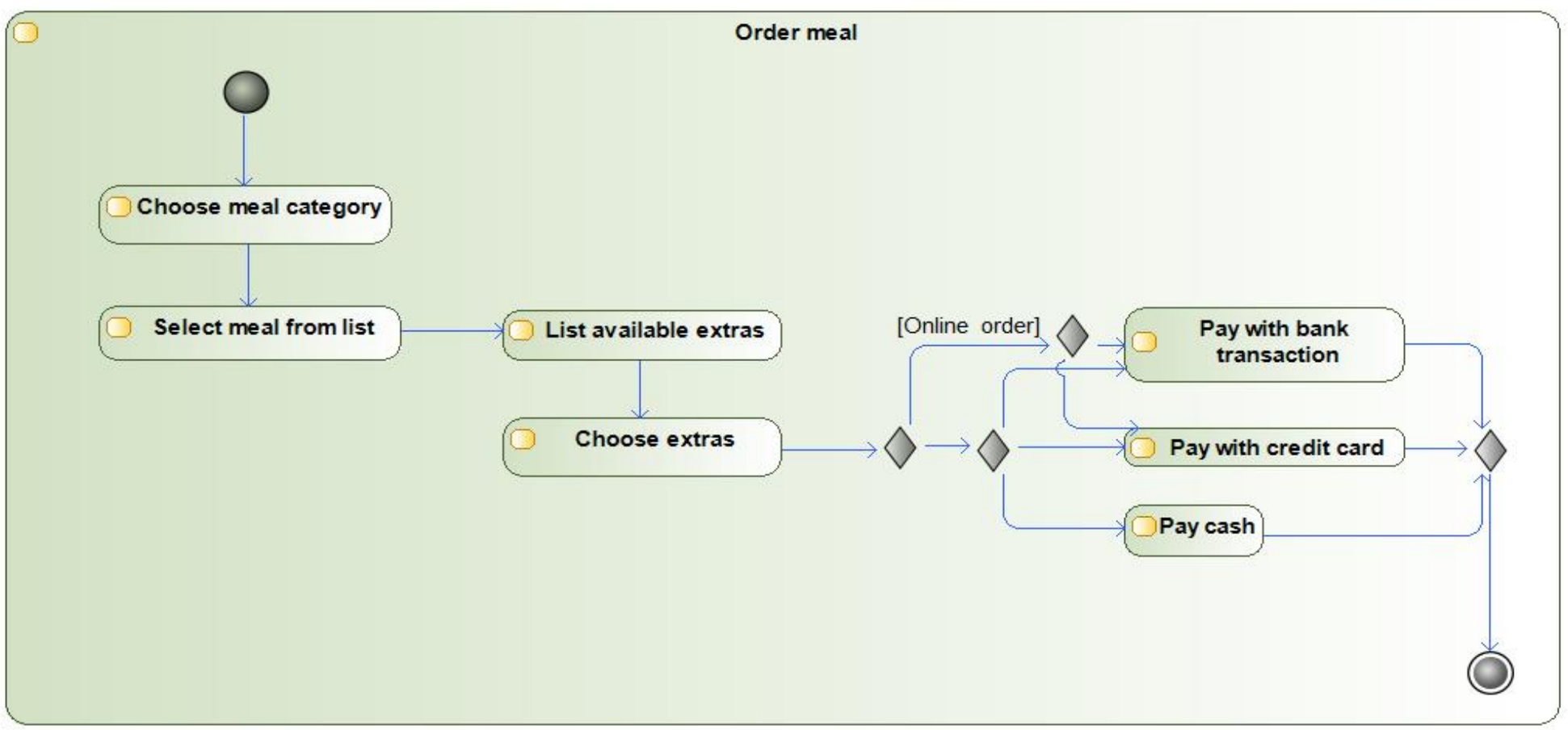


- Design steps
  - **Identify the expected output** of the activity
    - Meal ordered
    - Meal paid

- Design steps
  - **Identify the expected output** of the activity
  - **Define a logical main flow (Happy path)** of actions to reach this output



- Design steps
  - **Identify the expected output** of the activity
  - **Define a logical main flow** of actions to reach this output
  - **Define alternative flows of actions** to avoid exceptions or dead ends



- Design steps
  - **Identify the expected output** of the activity
  - **Define a logical main flow** of actions to reach this output
  - **Define alternative flows** of actions to avoid exceptions or dead ends
  - **Verify your process using paper prototypes**
    - https://balsamiq.com/
    - https://marvelapp.com/pop/

- # School Administration System
  - Create an activity diagram outlining the CDG's administration system by taking the following constraints into account:
    - The administration staff can manage students, teachers, classrooms, schedules etc. working groups etc. in a dedicated administration application
    - Teachers can accept or decline proposed schedules

Create an Activity Diagram

Create a new plan for next years lectures of teacher Mr. Heistracher

- 2 Lectures (OOP 2 lessons per week, SWD 3 lessons per week)
- Assign Rooms (check availability)
- Only tuesdays and wednesdays
- Block holidays (personal preferences, vacation,)
- Teacher, administration office and principal of the school have to confirm the time schedule in parallel
- Assign students