# 3.2 Activity Diagrams

Subject/Topic/Focus:

- Events and workflow behavior

Summary:

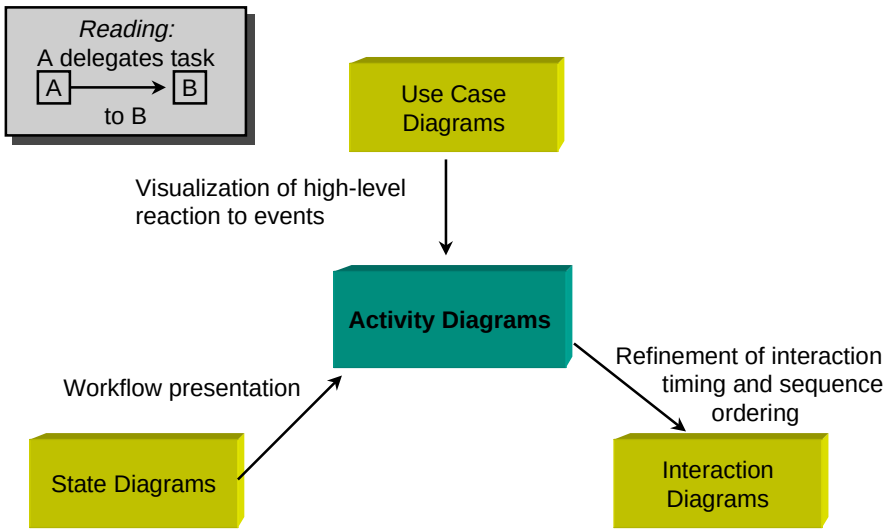
- Activities, Decisions
- Concurrency & Synchronization
- Swimlanes

Literature:

- Fowler



# Role of Activity Diagrams in UML



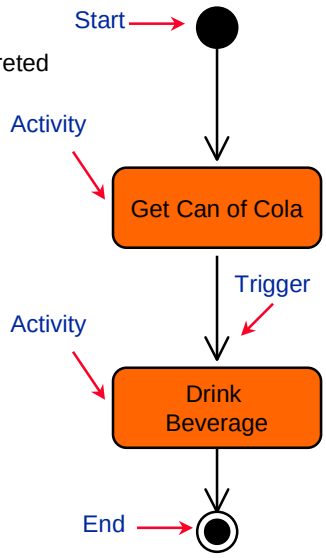
# Activity Diagrams

The core symbol is the **activity** box.

Depending on the perspective, the activity is interpreted differently.

- **Conceptual** perspective:
  - An activity is some **task** that needs to be done whether by a human or a computer.
- **Specification** perspective:
  - An activity is a **method** on a class.

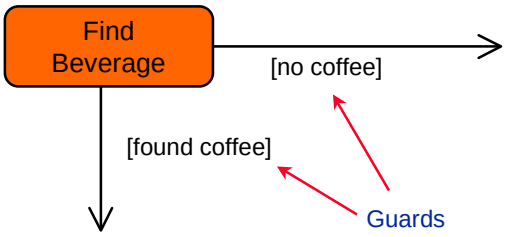
The links between the activities are the **triggers**.






# Guards

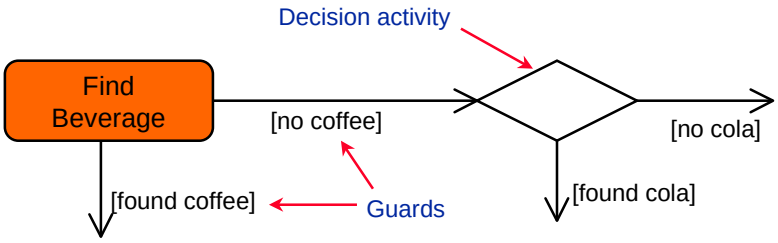
Just like in state diagrams, guards determine which trigger is used.




At a single point in time, only one trigger can occur.
Therefore **guards** essentially need to be **exclusive** !

# Decision Activity

To describe nested decisions, UML activity diagrams offer the decision-diamond activity symbol.




Coding:

```
if (found_coffee()) { ... }
else
  if (found_cola()) {  ... }
  else { ... }
```



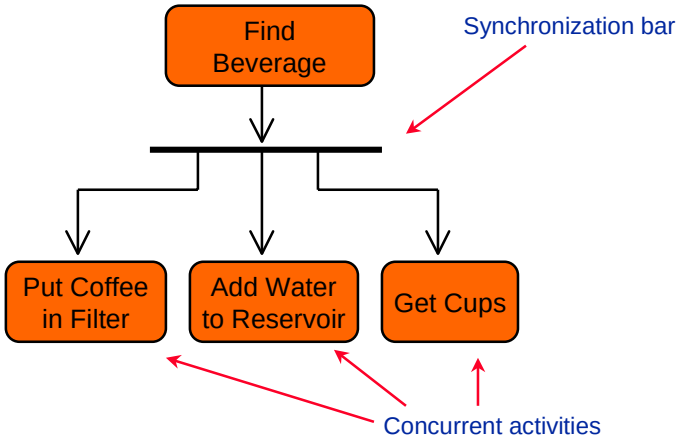
# Concurrent Activities

The difference between flowcharts and activity diagrams is that in activity diagrams parallel behavior can be expressed.

- This is important for **business modeling**, where unnecessary sequential processes can be designed for parallel execution.
- This improves the **efficiency** and **responsiveness** of business processes.
- Activity diagrams are also useful for concurrent programs, since you can graphically lay out what **threads** you have and when they need to **synchronize**.

# Synchronization Bars (1)

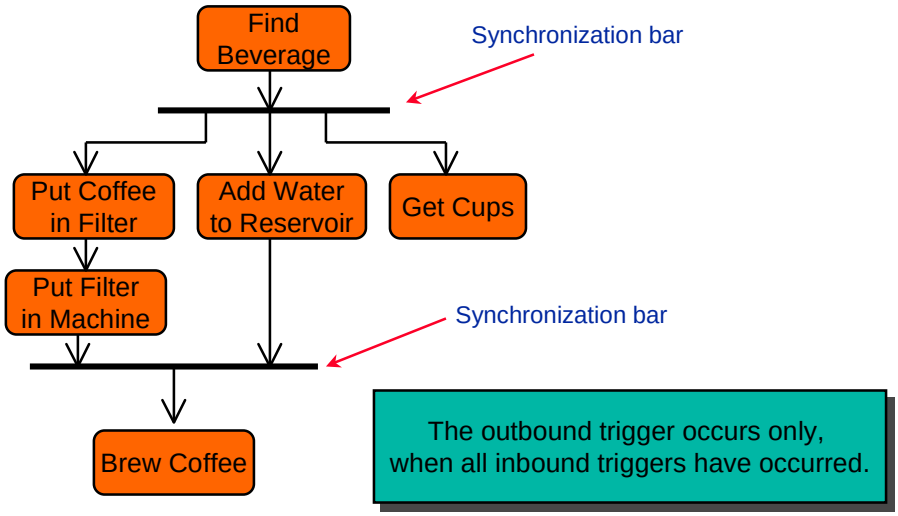
Synchronization bars initiate concurrent sections in an activity diagram.
In these concurrent sections, triggers can occur in parallel and no sequential order is established.





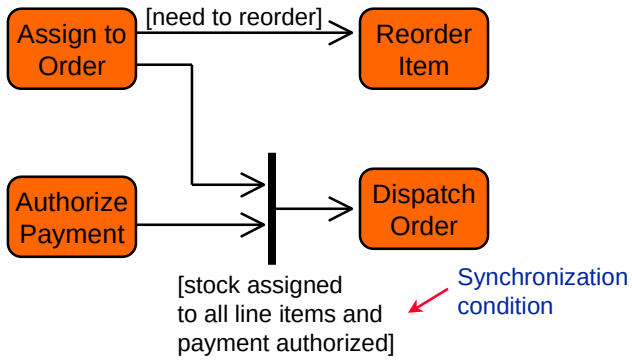

# Synchronization Bars (2)

Synchronization bars synchronize concurrent activities.



# Synchronization Conditions

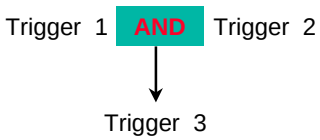
- The default behavior of synchronization bars is that the outbound trigger occurs as soon as all inbound triggers have occurred.
- In addition to this condition, you can specify an **extra synchronization condition** which is checked every time an inbound trigger occurs.

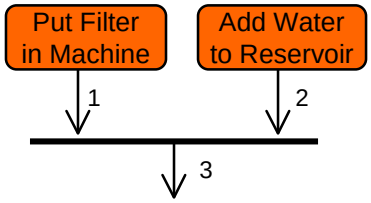




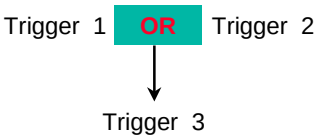

# Multiple Incoming Triggers

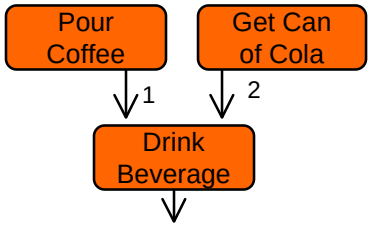
For synchronization bars, the outbound trigger occurs only if all inbound triggers have occurred.




For activities the trigger semantics is different.
Activities occur as soon as one
(of many possible) inbound trigger occurs.



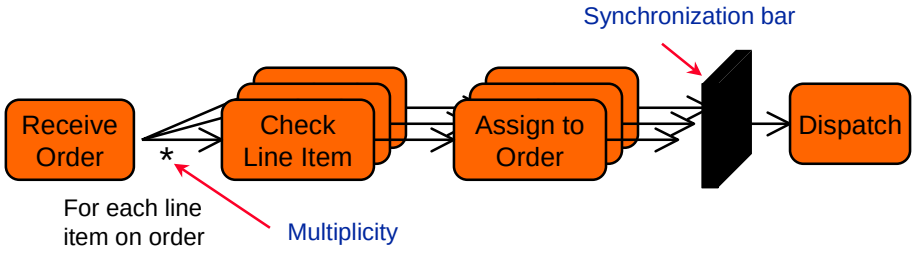



# Multiple Triggers

The second source of **parallelism** in activity diagrams are **multiple triggers**.

The multiple trigger with the multiplicity marker (*) is used just like it is done in class diagrams.

- Although it is not part of the UML, you should state what the basis of the multiple trigger is, to improve readability and understandability.
- To synchronize multiple threads initiated by a multiplicity marker again a synchronization bar is used.






# Swimlanes

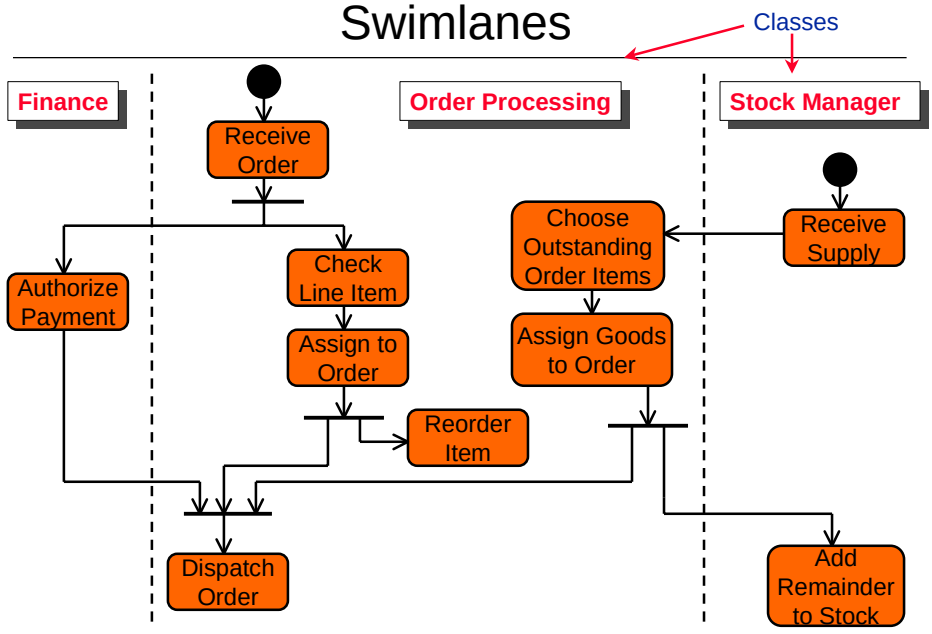
Activity diagrams tell you what happens, but they do not tell you **who** does what.

- From the **implementation** perspective, this means, that the diagram does not convey which **class** is **responsible** for which activity.
- From the **domain** view point, this means that the diagram does not show which **people** or departments are responsible for which activity.

**Swimlanes** are a way around this.

- Swimlanes are indicated by vertical dashed lines which separate the diagram into **zones**.
- Each zone represents a particular class, person or department, etc.

# Swimlanes






# When to Use Activity Diagrams

- Activity diagrams show **behavior** that **spans** over **multiple use cases** to describe the **workflow** of the overall process.
- For multiple **objects** and their high-level **interaction**, activity diagrams are particularly helpful for representing an overview of **concurrent processes**.

- Do **not** use activity diagrams to see how objects **collaborate**. An interaction diagram is simpler and gives you a clearer picture of collaborations.
- Activity diagrams are **not** accurate for describing how an **object behaves** over its lifetime. Use a state diagram instead.