IDEA Progetto
Design Thinking ; Requisiti ; Kanban ; Pitch

D1

Implementazione

Git / GitHub
User Stories
RESTful API
OpenAPI
Web 2.0 JavaScript
WebAPI Node.js
MongoDB
Authentication JWT + GoogleAuth
Frontend
Deplyment & CI-CD
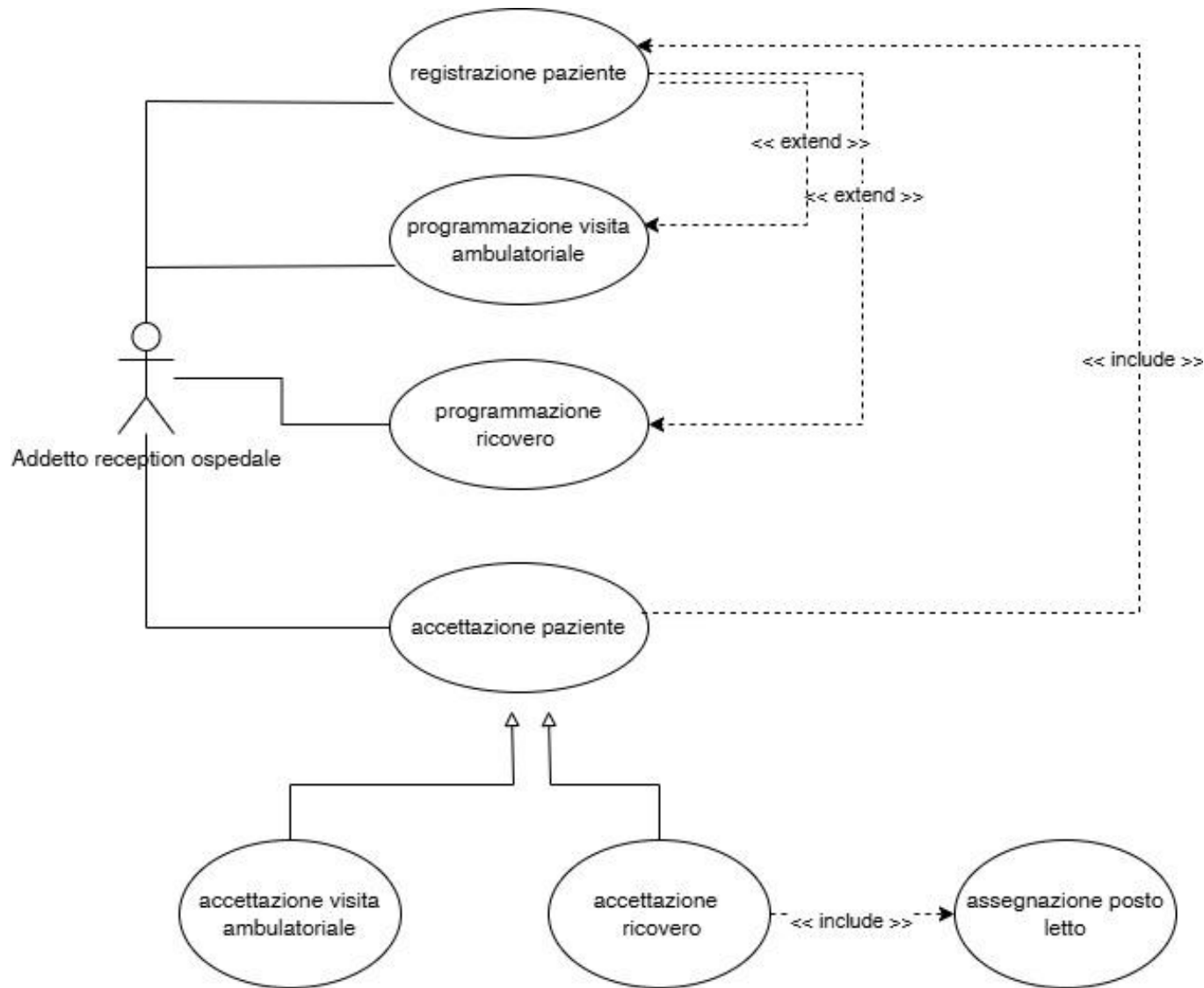Testing Jest

D3

Analisi e Progettazione

Processi di sviluppo
Agile
Linguaggi di modellazione
Use Case Diagram
Sequence + Activity Diagram
Architetture
Component Diagram
Class Diagram
Class Diagram -> API
Testing

D2

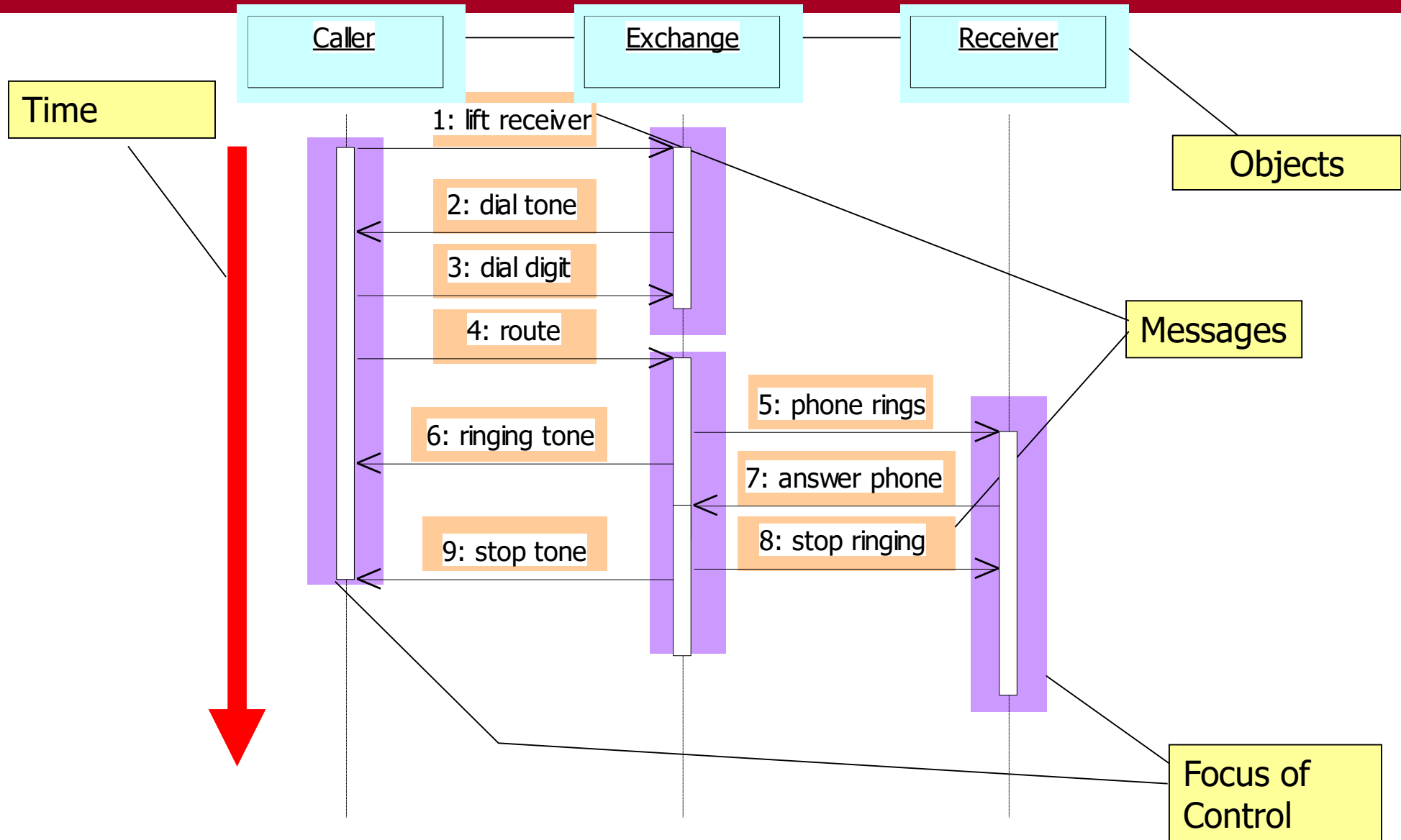**Report finale**

D4

Use Case Generalization: example

Software Engineering

Sequence Diagrams

Sequence Diagrams: motivations

- Use Case Diagrams present an outside view of the system

- Sequence Diagrams are good at representing how objects collaborate.

Sequence Diagram: example

Sequence Diagrams: ingredients

- **Time:** either unspecified or in fixed units

- **Messages:** a communication between two objects that conveys information with expectation that action will ensue

- **Focus of Control:** period during which an object is performing an action either directly or through a subordinate procedure.

Sequence Diagrams vs. Use Case Diag.

Objects	Actors
Messages	Use Cases

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
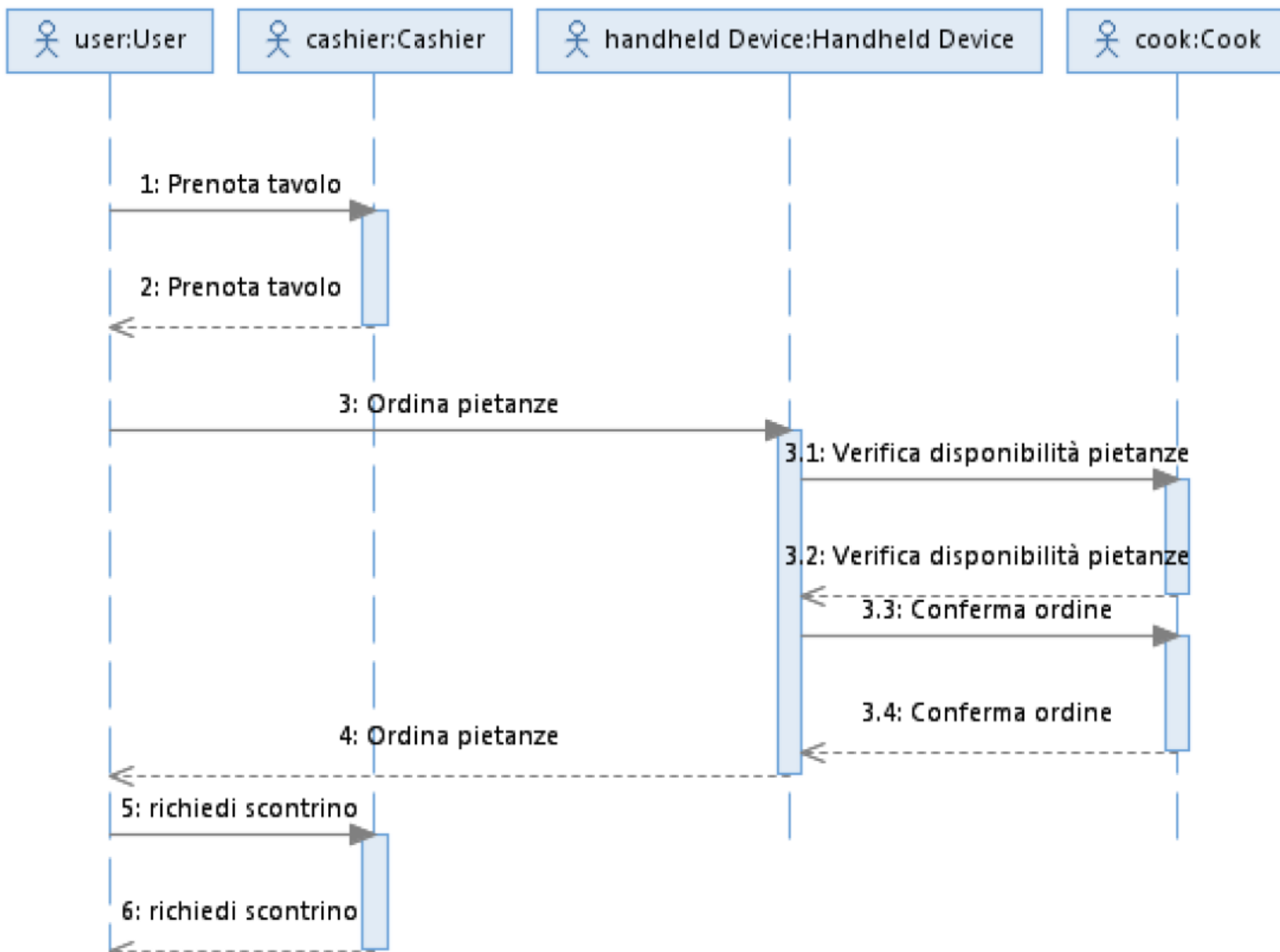
Software Engineering

State Machine Diagram
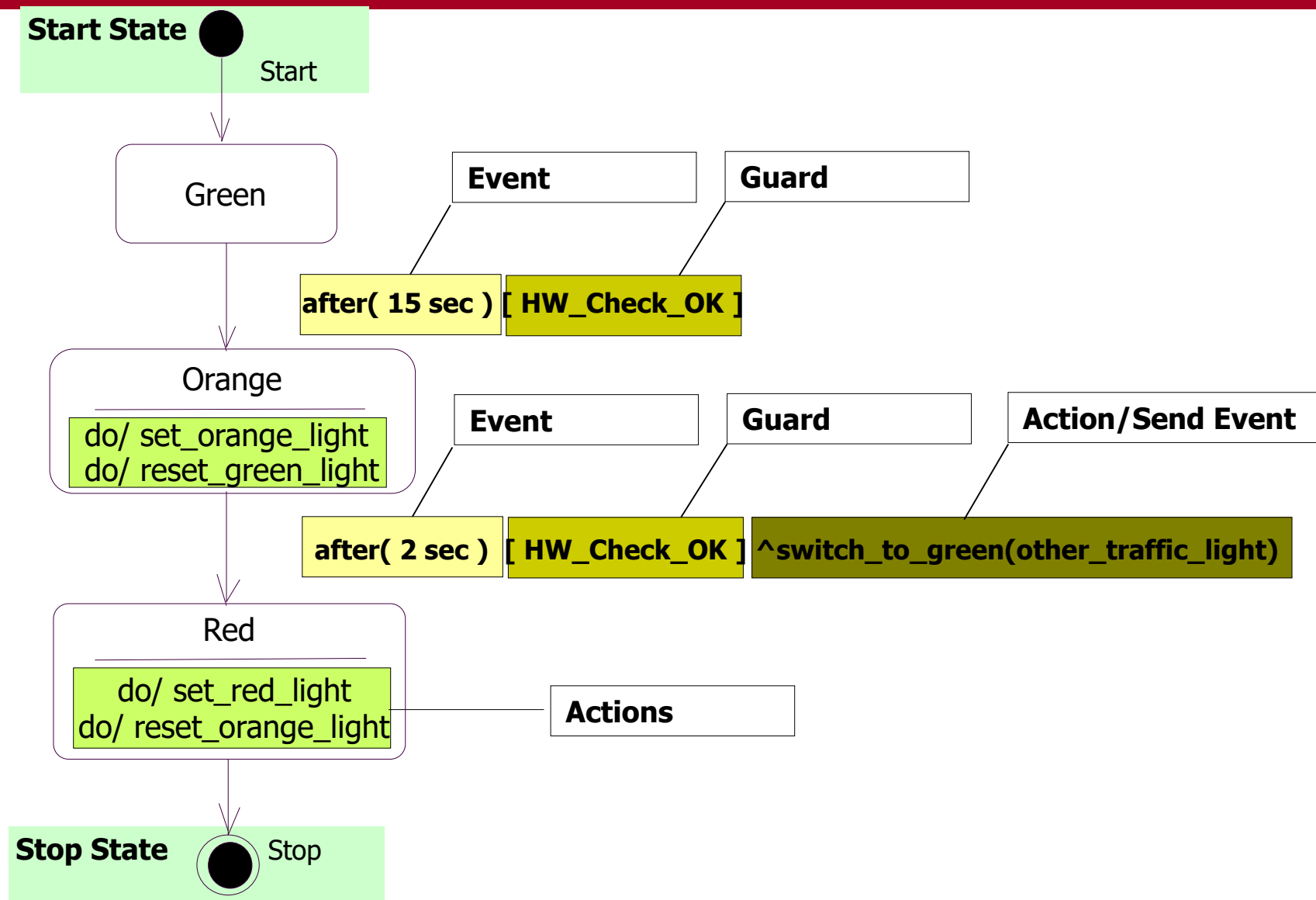
State Machine Diagram

Definition:

A State Machine Diagram describes the sequences of states and actions through which an element (e.g. of the design) can proceed during its lifetime as a result to reacting to dicrete events.

Key ingredients:

➢ States (simple and composite)

➢ Transitions (event, guard, actions)

State Machine Diagram: example

Start State ● Start

Green

Event	Guard

after(15 sec) **[HW_Check_OK]**

Orange
do/ set_orange_light
do/ reset_green_light

Event	Guard	Action/Send Event

after(2 sec) **[HW_Check_OK]** **^switch_to_green(other_traffic_light)**

Red
do/ set_red_light
do/ reset_orange_light

Actions

**Stop State** ● Stop

State Machine Diagram: ingredients

- ➢ **Start State and Stop State:** obvious

- ➢ Transition
  - ➢ **Event:** what may cause the transition
  - ➢ **Guard:** upon "event", if true allows transition, if false prevents transition
  - ➢ **Action/Send Event:** if transition, then execute action/send event

- ➢ Actions within a state
  - ➢ **entry/** do action when entering the state
  - ➢ **do/** keep doing action while in state
  - ➢ **exit/** do action on leaving state

State Machine Diagram: loop
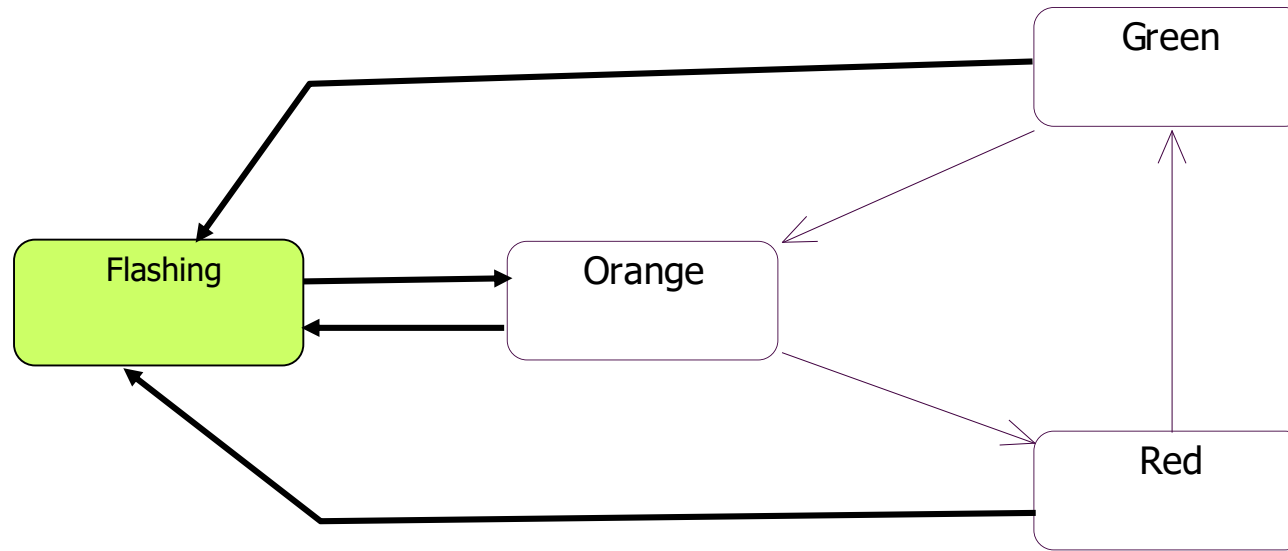
Issues:

- Cluttering of the diagram

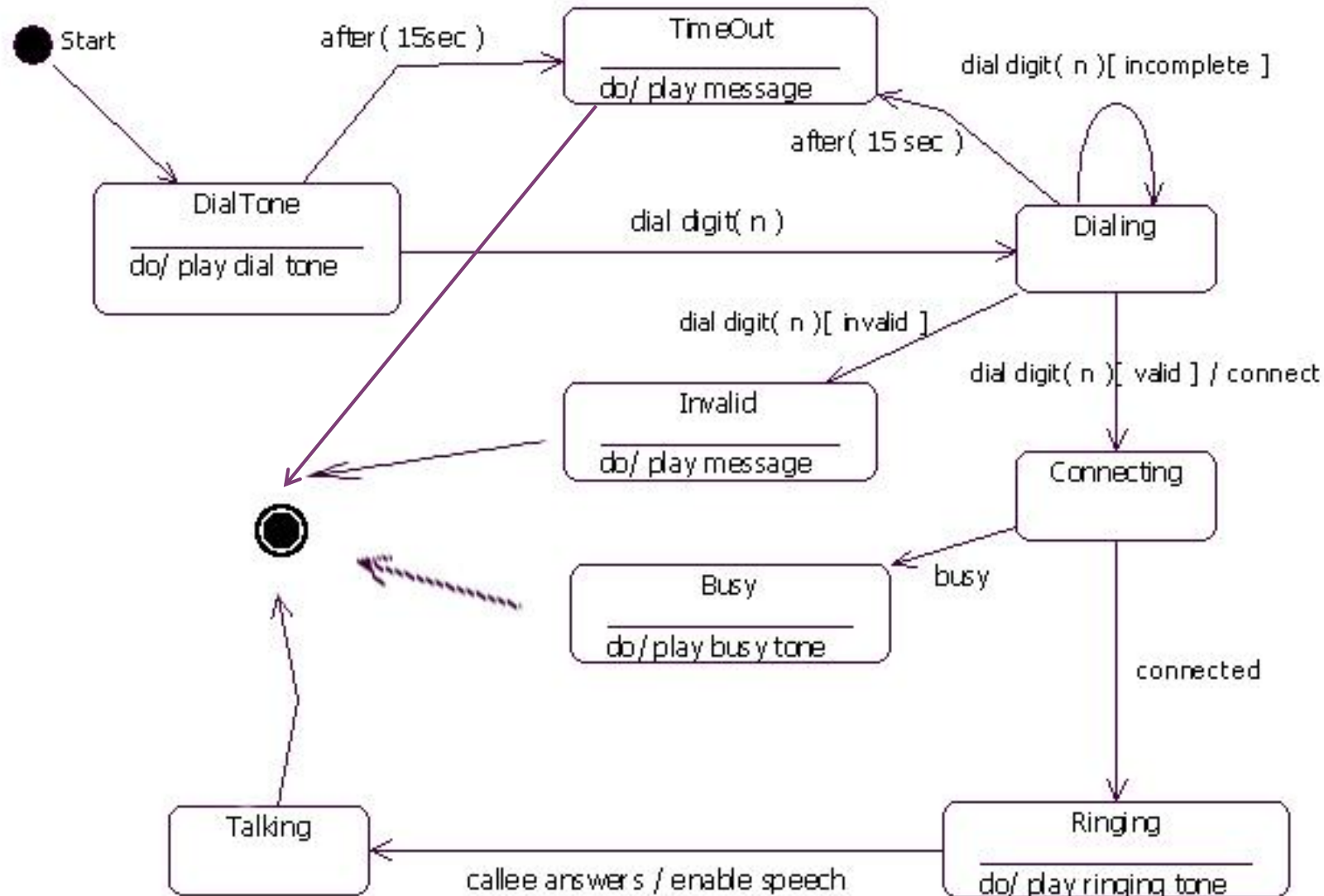- Structural information is lost

Software Engineering

State Machine Diagram: example

State Machine Diagram: example

Phone Dialing

Specify, using State Machine Diagram, the process of placing a call with a telephone (e.g. when you lift the receiver, a dial tone is activated; then you can either …).

State Machine Diagram: example

Software Engineering

Activity Diagram

Activity Diagram

Definition

An activity diagram is a variation of a State Machine Diagram in which the states represent the performance of actions and the transitions are triggered by the completion of the actions.

Key Ingredients

➢ Activities (States)

➢ Transitions (event, guard, actions)

➢ Merge/Fork/Join

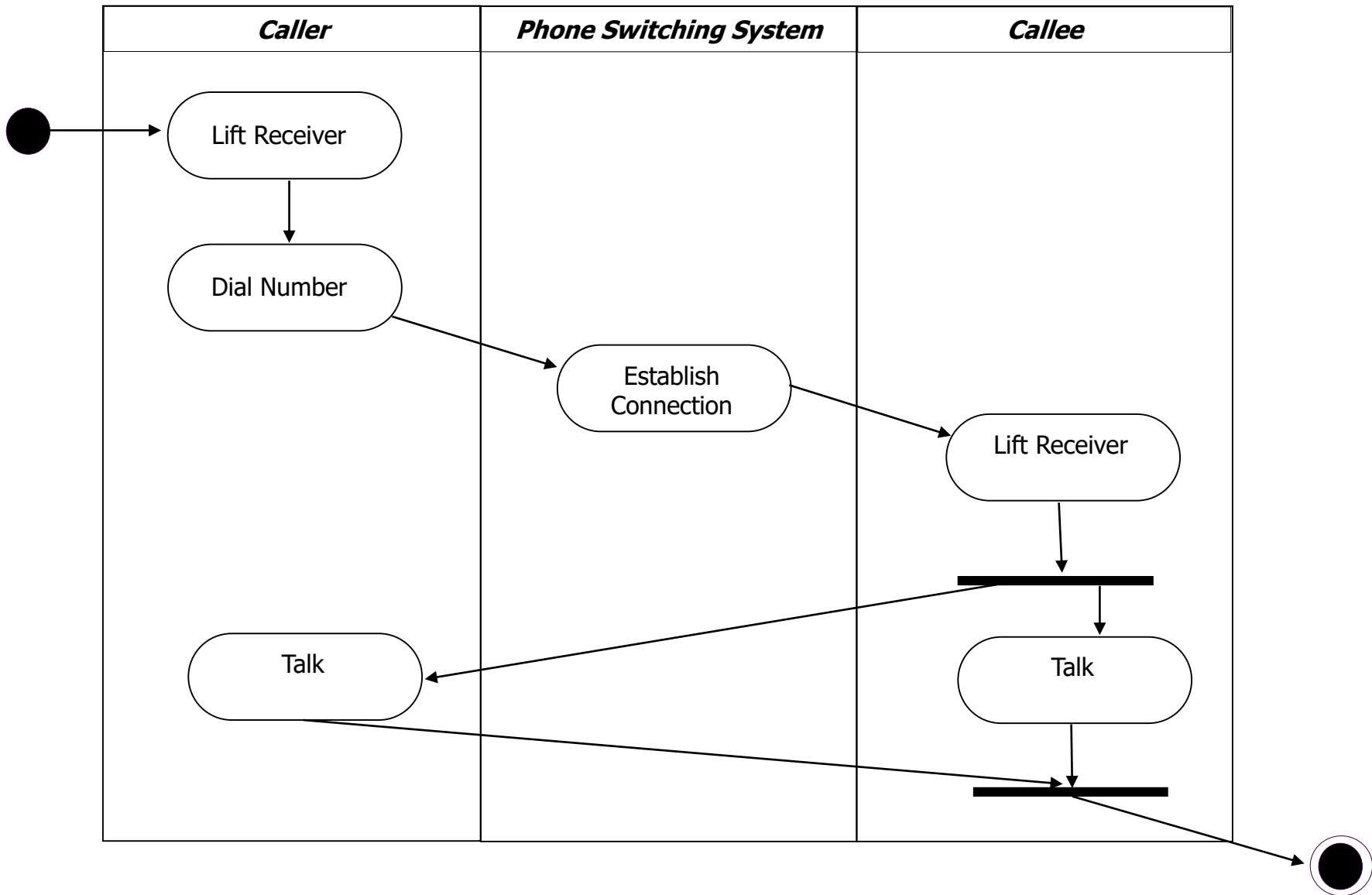# Activity Diagram: explanation

➢ **Start/Stop/Activity/Guard:** obvious

➢ **Fork:** allows for the parallel decomposition of activities.

➢ **Join:** synchronises parallel activities.

➢ **Decision:** conditional behavior.

Swimlanes

Sometime it is useful to highlight responsibilities in activity diagrams

Activities diagrams allow to do that with "swimlanes"

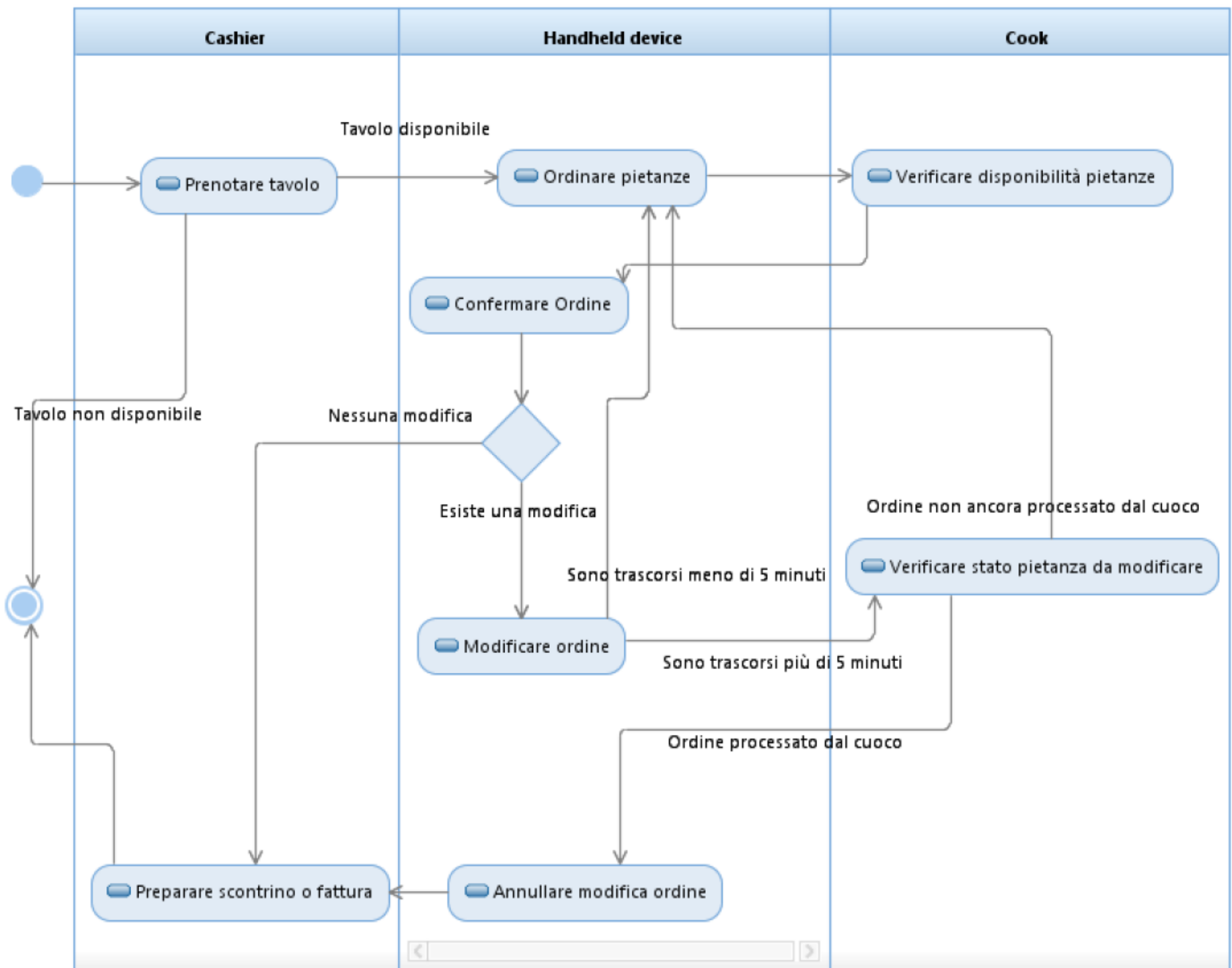# Swimlanes: example

Caller	Phone Switching System	Callee

- Lift Receiver
- Dial Number
- Establish Connection
- Lift Receiver
- Talk
- Talk

Software Engineering

Activity Diagram: example

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.

Cashier	Handheld device	Cook

Tavolo disponibile

Prenotare tavolo → Ordinare pietanze → Verificare disponibilità pietanze

Confermare Ordine

Tavolo non disponibile

Nessuna modifica

Esiste una modifica

Ordine non ancora processato dal cuoco

Sono trascorsi meno di 5 minuti

Verificare stato pietanza da modificare

Modificare ordine

Sono trascorsi più di 5 minuti

Ordine processato dal cuoco

Preparare scontrino o fattura ← Annullare modifica ordine

Sequence Diagrams vs Activity Diag.

- Sequence Diagram: description of **the whole system**

- Activity Diagram: description of **a part of the system** (usually an Use Case)

Sequence and Activity Diagrams

Use them when really necessary!!!

(e.g. complex interactions, (in)formal verification, business process definition and enhancement, …)