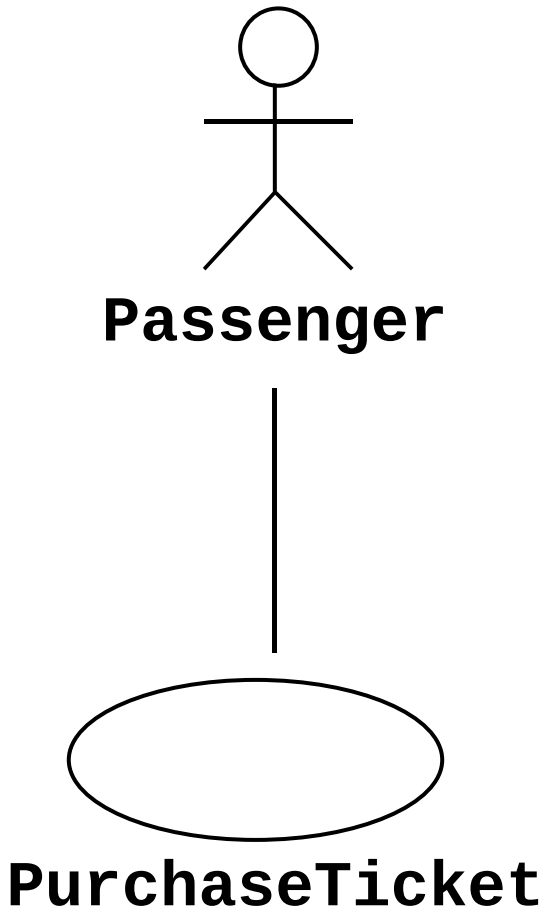
UML Core Conventions

- Rectangles are classes or instances
- Ovals are functions or use cases
- Instances are denoted with an underlined names
  - **`myWatch:SimpleWatch`**
  - **`joe:Firefighter`**
- Types are denoted with nonunderlined names
  - **`SimpleWatch`**
  - **`Firefighter`**
- Diagrams are graphs
  - **Nodes are entities**
  - **Arcs are relationships between entities**

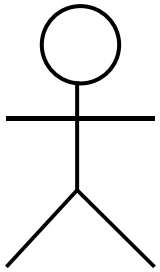
UML Second Pass: Use Case Diagrams



Used during requirements elicitation to represent external behavior

- ***Actors*** represent roles, that is, a type of user of the system
- ***Use cases*** represent a sequence of interaction for a type of functionality
- The use case model is the set of all use cases. It is a complete description of the functionality of the system and its environment

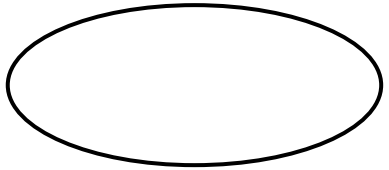
Actors



Passenger

- An actor models an external entity which communicates with the system:
  - **User**
  - **External system**
  - **Physical environment**
- An actor has a unique name and an optional description.
- Examples:
  - **Passenger: A person in the train**
  - **GPS satellite: Provides the system with GPS coordinates**

Use Case



`PurchaseTicket`

A use case represents a class of functionality provided by the system as an event flow.

A use case consists of:

- Unique name
- Participating actors
- Entry conditions
- Flow of events
- Exit conditions
- Special requirements

Use Case Example

Name: `Purchase ticket`

Participating actor: `Passenger`

Entry condition:

- `Passenger` standing in front of ticket distributor.
- `Passenger` has sufficient money to purchase ticket.

Exit condition:

- `Passenger` has ticket.

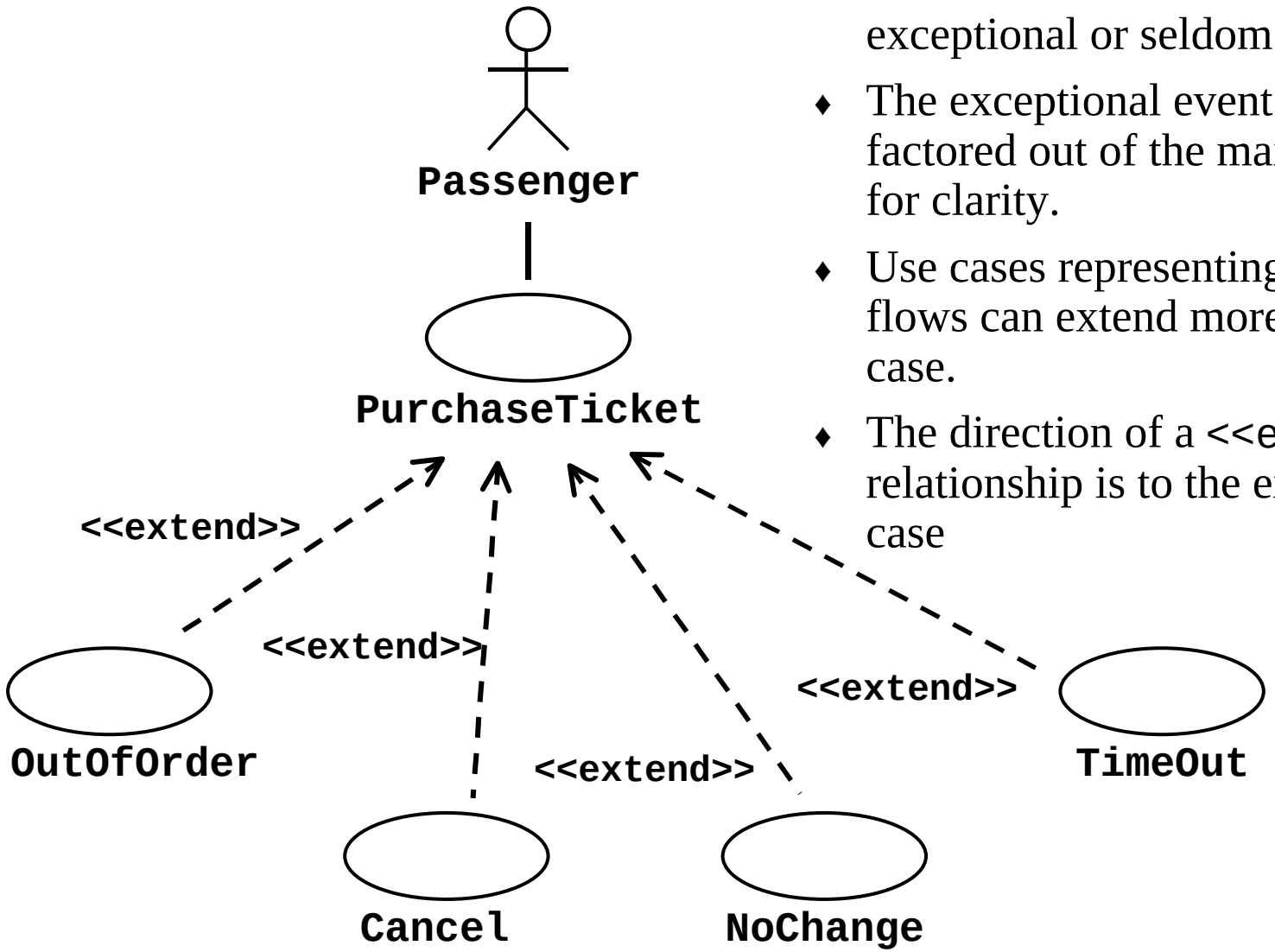
Event flow:

1. `Passenger` selects the number of zones to be traveled.
2. Distributor displays the amount due.
3. `Passenger` inserts money, of at least the amount due.
4. Distributor returns change.
5. Distributor issues ticket.

Anything missing?

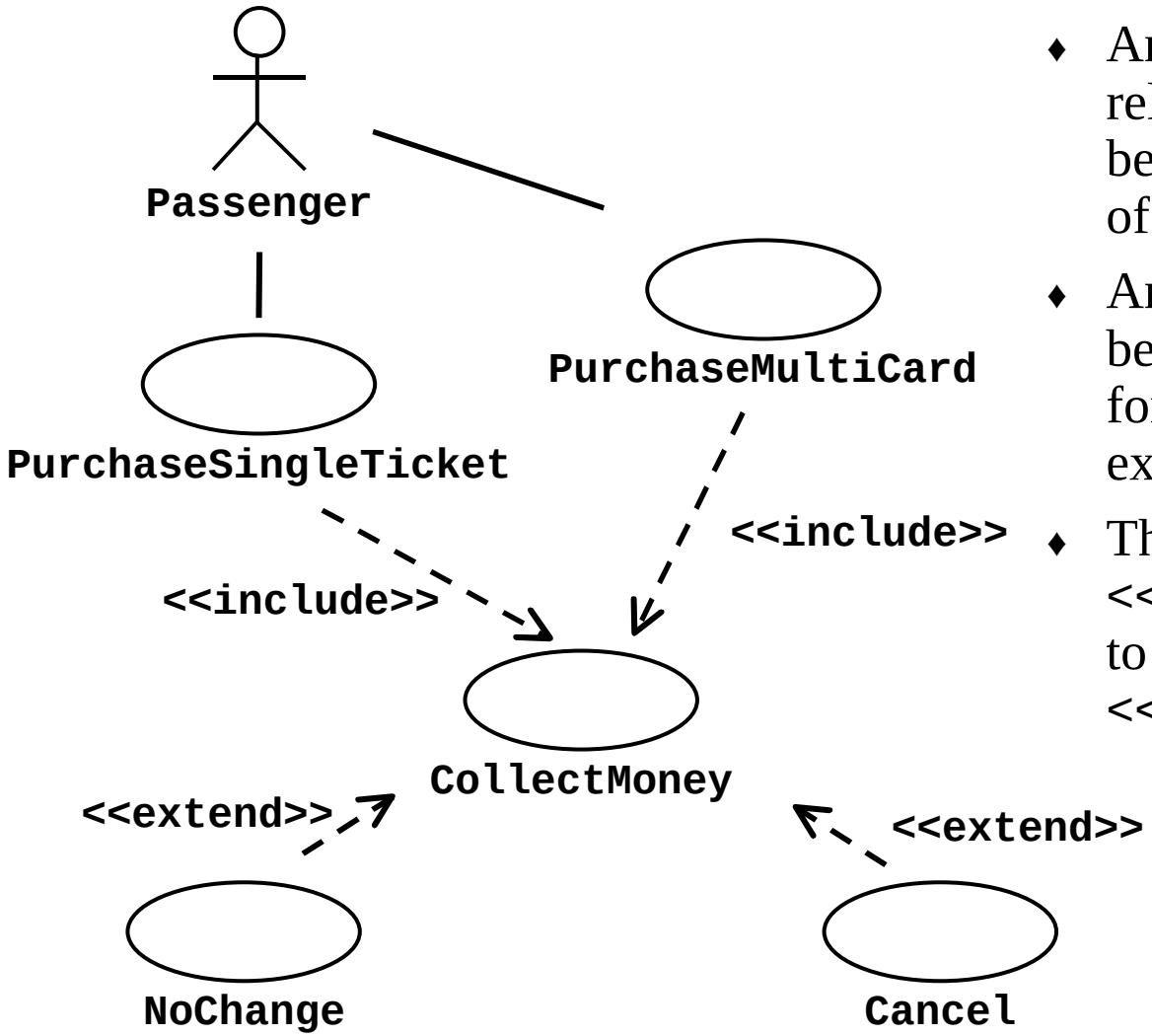
Exceptional cases!

The <<extend>> *Relationship*



- <<extend>> relationships represent exceptional or seldom invoked cases.
- The exceptional event flows are factored out of the main event flow for clarity.
- Use cases representing exceptional flows can extend more than one use case.
- The direction of a <<extend>> relationship is to the extended use case

The <<include>> Relationship



- An <<include>> relationship represents behavior that is factored out of the use case.
- An <<include>> represents behavior that is factored out for reuse, not because it is an exception.
- The direction of a <<include>> relationship is to the using use case (unlike <<extend>> relationships).