System sequence diagram

What is a SSD

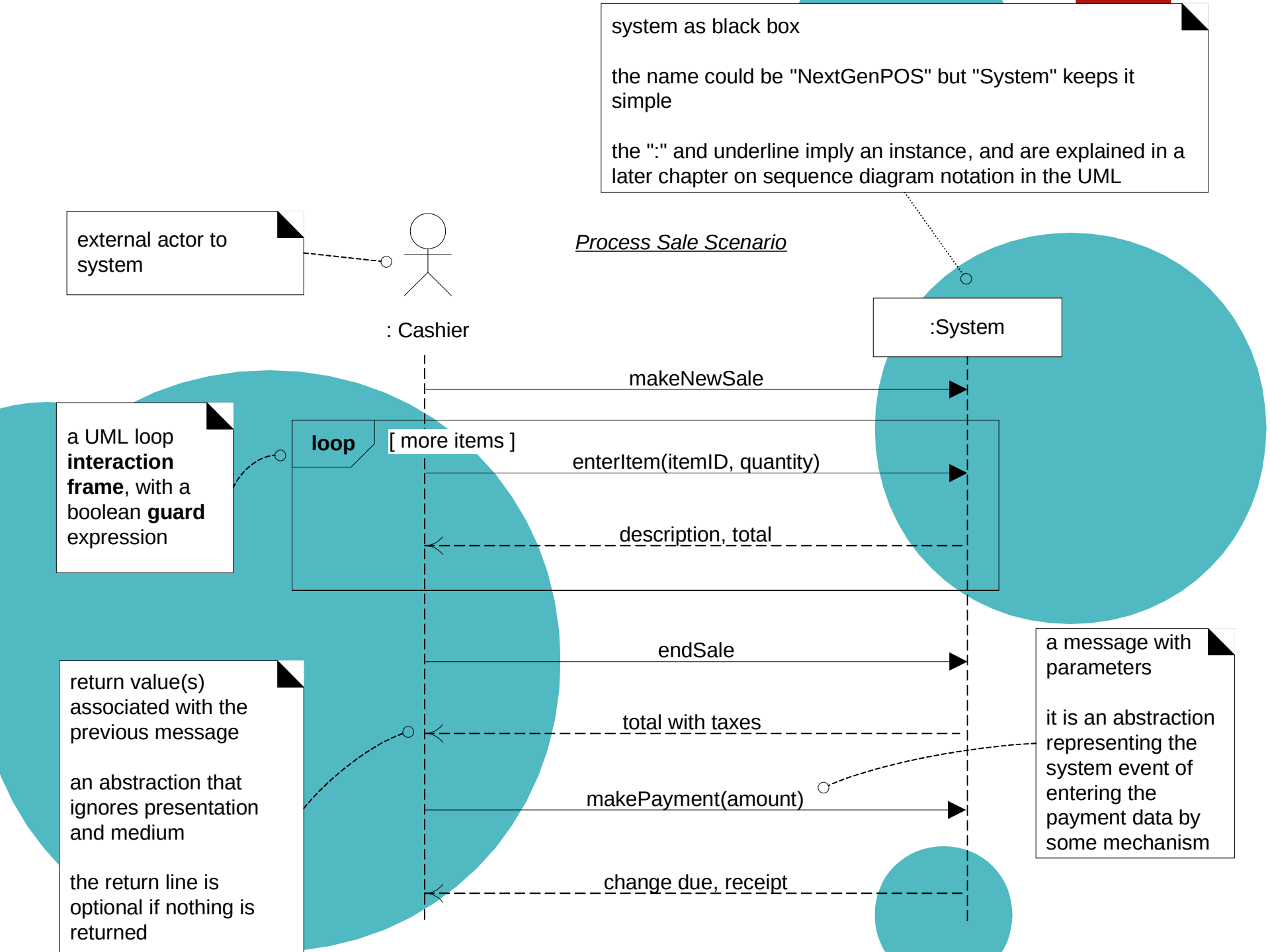
- A way of modeling input and output events related to systems
- It is a picture that shows, for one particular use case scenario, the events an external actor generates and in what order
- Draw SSD for the main success scenario and complex alternatives
- They show the system as a black box
- There should be one for the main success scenario of a use case
- Like Domain Models – very high level with a lower level counterpart

Process Sale Use Case

Simple cash-only Process Sale scenario:

1. Customer arrives at a POS checkout with goods and/or services to purchase.
2. Cashier starts a new sale.
3. Cashier enters item identifier.
4. 4. System records sale line item and presents item description, price, and running total.
   Cashier repeats steps 3-4 until indicates done.
5. System presents total with taxes calculated.
6. Cashier tells Customer the total, and asks for payment.
7. Customer pays and System handles payment.

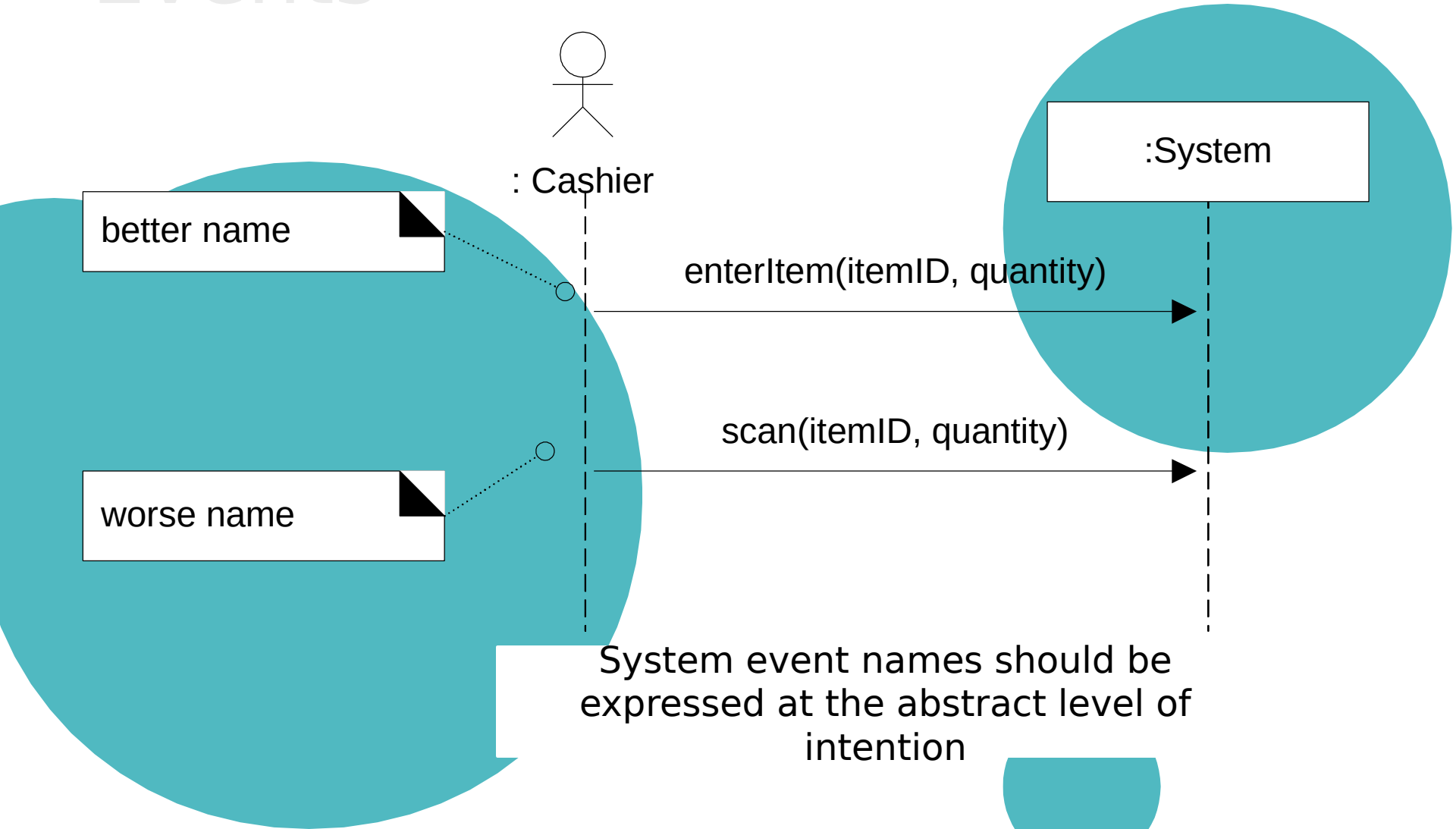
...


system as black box
the name could be "NextGenPOS" but "System" keeps it simple
the ":" and underline imply an instance, and are explained in a later chapter on sequence diagram notation in the UML
external actor to system
Process Sale Scenario
: Cashier
:System
makeNewSale
a UML loop interaction frame, with a boolean guard expression
loop
[more items]
enterItem(itemID, quantity)
description, total
endSale
a message with parameters
it is an abstraction representing the system event of entering the payment data by some mechanism
return value(s) associated with the previous message
an abstraction that ignores presentation and medium
the return line is optional if nothing is returned
total with taxes
makePayment(amount)
change due, receipt

Why Draw SSDs

- Easy way to capture external events like “customer arrives at CheckOut”
- A description of “what” the system does but with some time aspects

How to Name System Events



System event names should be expressed at the abstract level of intention

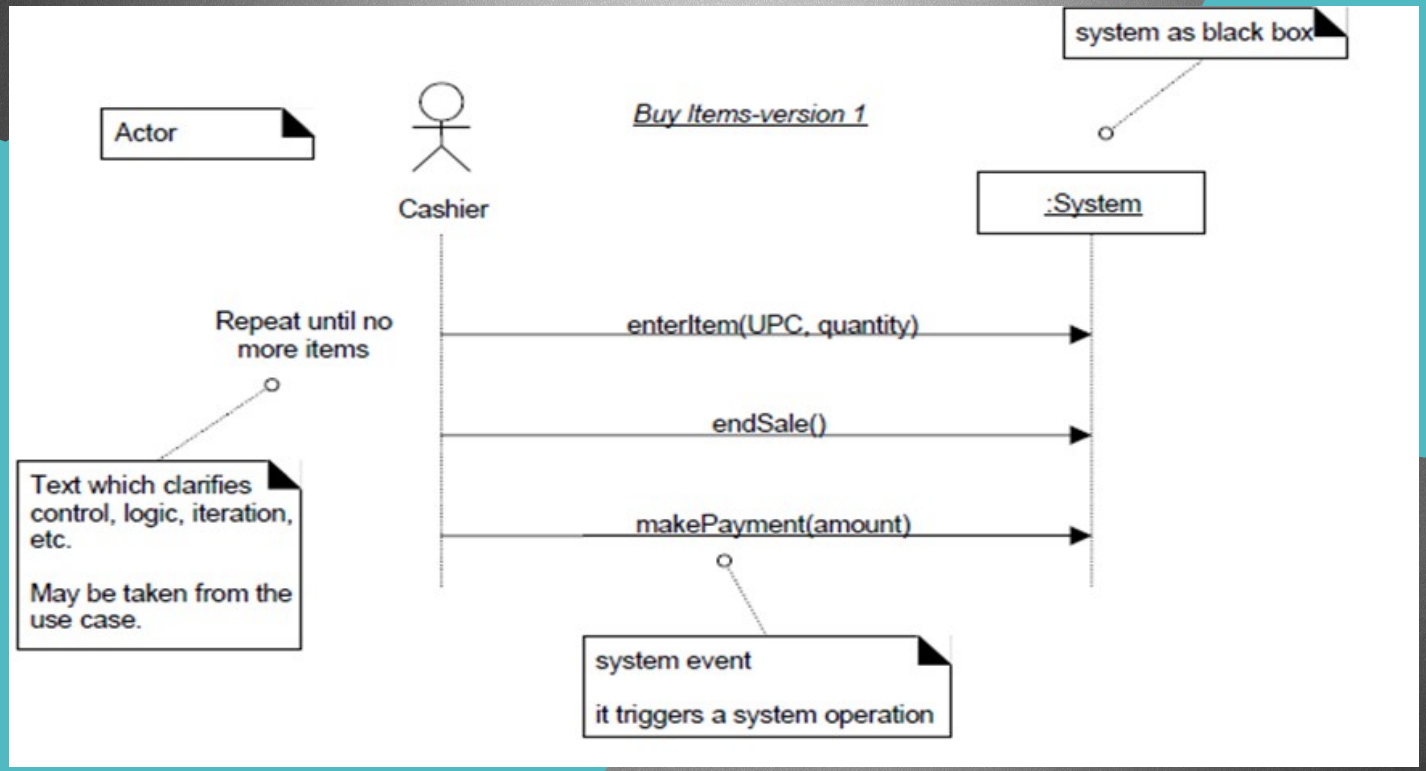
System sequence diagram

- SSD is a picture that shows for one particular scenario of a use case, the events that external actors generate which trigger some inter-system events
- This diagram treats the system as black box and only emphasize on events that cross the system boundary from actor to system
- It illustrates inputs and outputs to the system

Motivation behind SSD

- The motivation behind SSD is to design a pattern for handling external system requests and producing proper response
- *Events may be external (triggered by human or computer), time events or fault / exception events*
- It is useful to investigate and define system's behavior as "black box" before proceeding with the detailed design
- It shows events from one scenario of a use-case diagram and operations performed by system in response

System sequence diagram

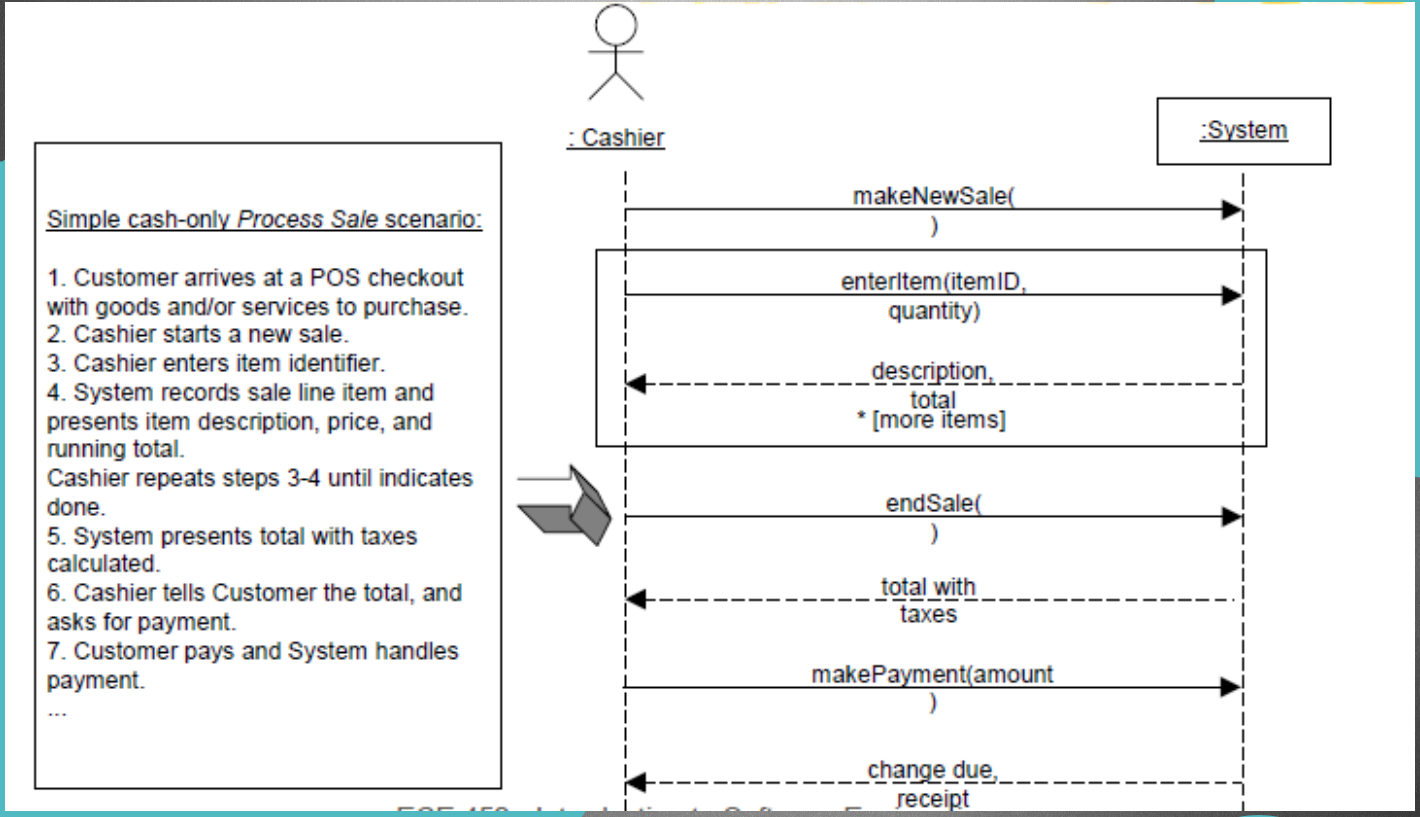


System events

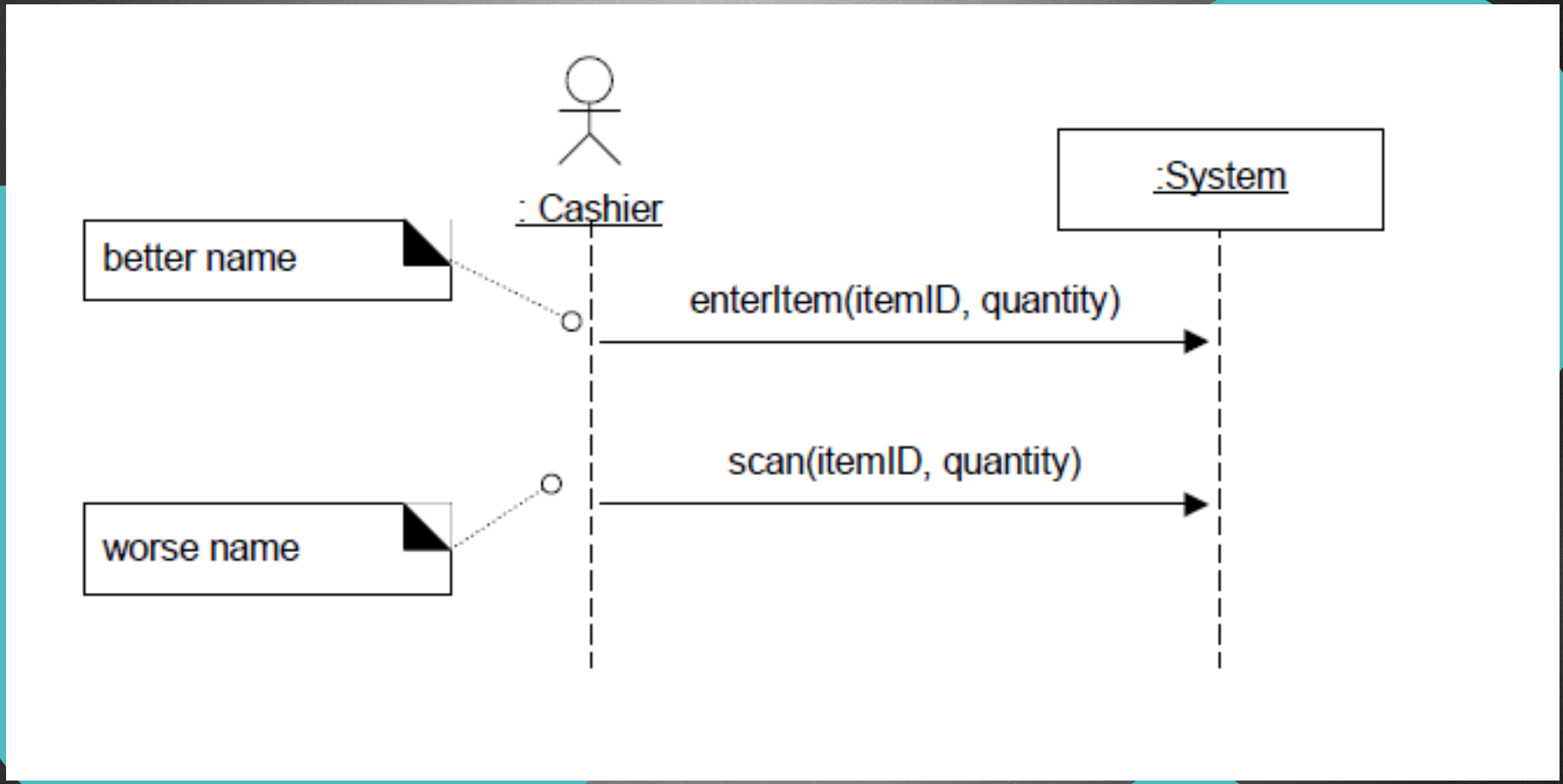
- **System events:** External input to system generated by actor
- **System operation:** Methods invoked in response to system events

- System events may have arguments
  - enterItem(UPC, quantity)
  - raise(money)

System sequence diagram



Naming system events and operations



Difference with sequence diagram

- Sequence diagram shows the internal operations of the system triggered by events.
- It ignores the external events source
- Operations are drawn between different objects of the system
- The flow of messaging is maintained

A Sequence Diagram