Drawing System Sequence Diagrams

Chapter 10

Applying UML and Patterns

-Craig Larman

Objectives

- Identify System Events.
- Create System Sequence diagrams for the events.

Iteration 1

- First real development iteration.
- The requirement work done during inception phase was to decide if the project was worth more serious investigation.
- Before starting iteration 1 design work, further investigation of the problem domain is useful such as clarification of the input and output system events, related to the system.

SSD versus Sequence Diagram

- A System Sequence Diagram is an artifact that illustrates input and output events related to the system under discussion.
- System Sequence Diagrams are typically associated with use-case realization in the logical view of system development.
- Sequence Diagrams (Not *System* Sequence Diagrams) display object interactions arranged in time sequence.

Sequence Diagrams

- Sequence Diagrams depict the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the system.
- Sequence diagrams can be used to drive out testable user interface requirements.

SSD—System Behavior

- System behaves as “Black Box”.
- Interior objects are not shown, as they would be on a Sequence Diagram.

System Sequence Diagrams

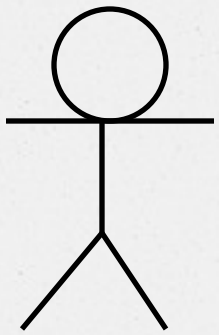
- Use cases describe-
  - How actors interact with system.
  - Typical course of events that external actors generate and
  - The order of the events.
- System Sequence Diagrams should be done for the main success scenario of the use-case, and frequent and alternative scenarios.

Notation

- **Object:** Objects are instances of classes. Object is represented as a rectangle which contains the name of the object underlined.
- Because the system is instantiated, it is shown as an object.

Notation (2)

- **Actor:** An Actor is modeled using the ubiquitous symbol, the stick figure.



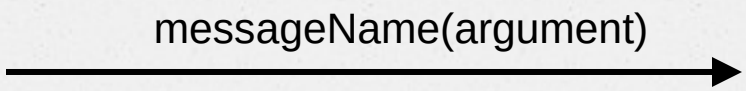
actor1

Notation (3)

- **Lifeline:** The Lifeline identifies the existence of the object over time. The notation for a Lifeline is a vertical dotted line extending from an object.

Notation (4)

- **Message:** Messages, modeled as horizontal arrows between Activations, indicate the communications between objects.



Example of an SSD

- Following example shows the success scenario of the Process Sale use case.
- Events generated by cashier (actor)-

  makeNewSale()

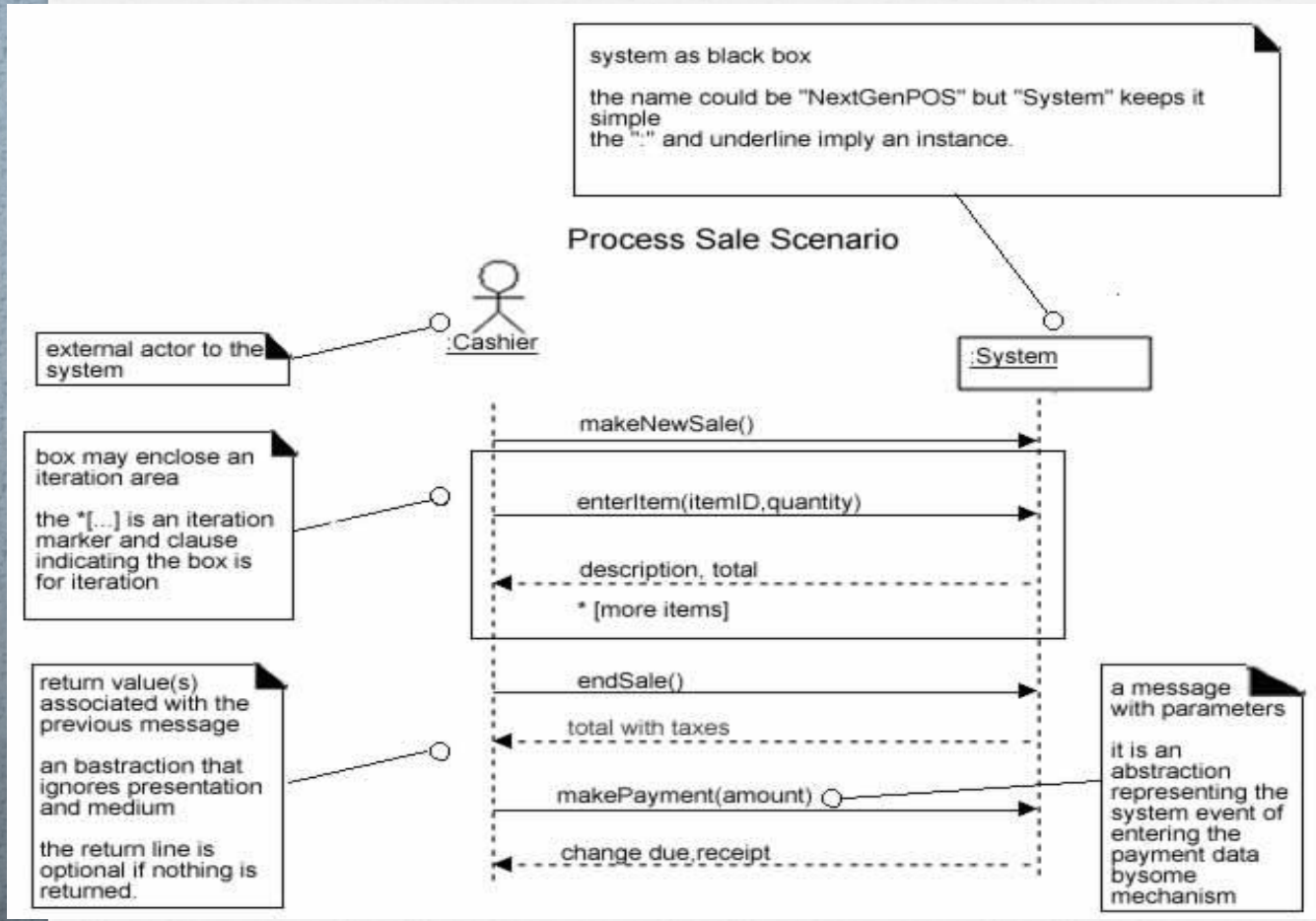
  enterItem(itemID, quantity)

  endSale()

  and

  makePayment(amount).

SSD for Process Sale scenario



System Sequence Diagrams and Use Cases

- System Sequence Diagram is generated from inspection of a use case.
- Constructing a systems sequence diagram from a use case-

1. Draw a line representing the system as a black box.
2. Identify each actor that directly operates on the system. Draw a line for each such actor.

System Sequence Diagrams and Use Cases

3. From the use case, typical course of events text, identify the system (external) events that each actor generates. They will correspond to an entry in the right hand side of the typical use case. Illustrate them on the diagram.
4. Optionally, include the use case text to the left of the diagram.

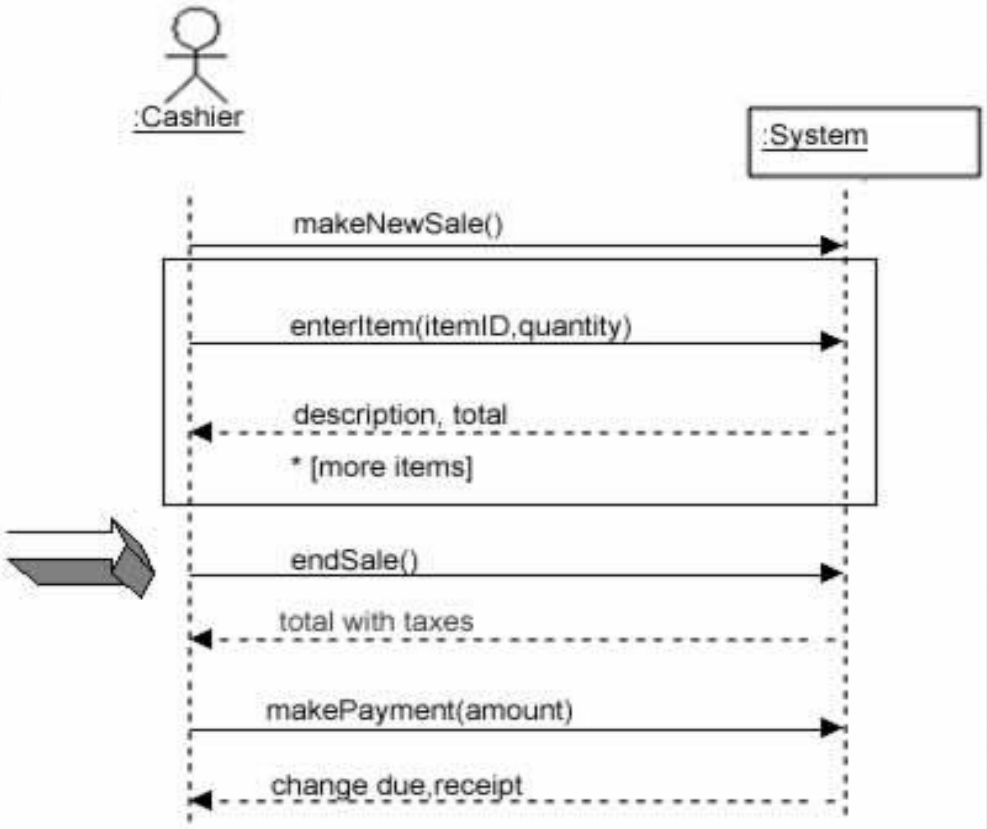
SSDs are derived from use cases.

Simple Cash-only Process Sale scenario:

1.customer arrives at aPOS check out with goods and/or services to purchase.
2.Cashier starts a new sale.
3.Cashier enters a new item identifier.
4.System records new sale line item and presents item description,price and running total.

Cashier repeats steps 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.
....

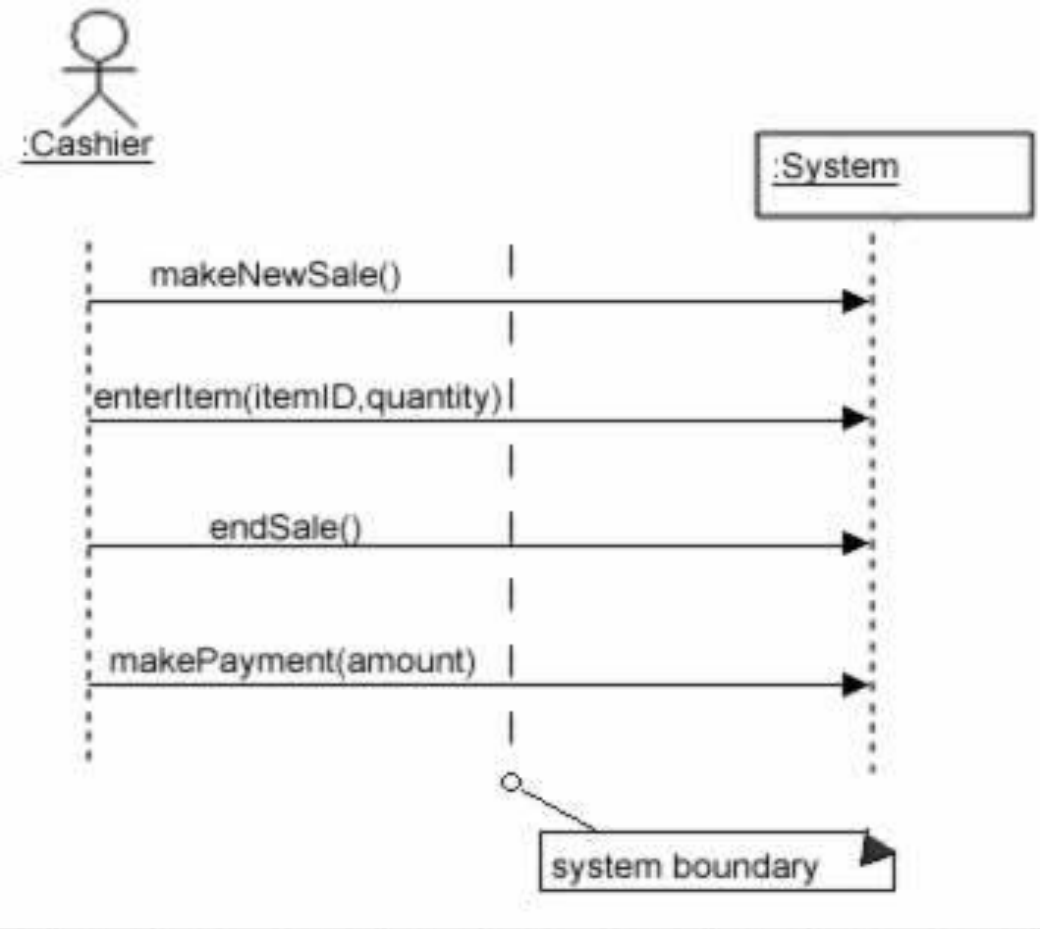


System Events and System Boundary

- Identifying the System events-

1. Determine the actors that directly interact with the system.
2. In the process Sale example, the customer does not directly interact with the POS system. Cashier interacts with the system directly. Therefore cashier is the generator of the system events.

Defining system boundary (system itself)



Naming System Events and Operations

System event

- External input event generated by an actor.
- Initiates a responding operation by system.

System operation

- Operation invoked in response to system event.

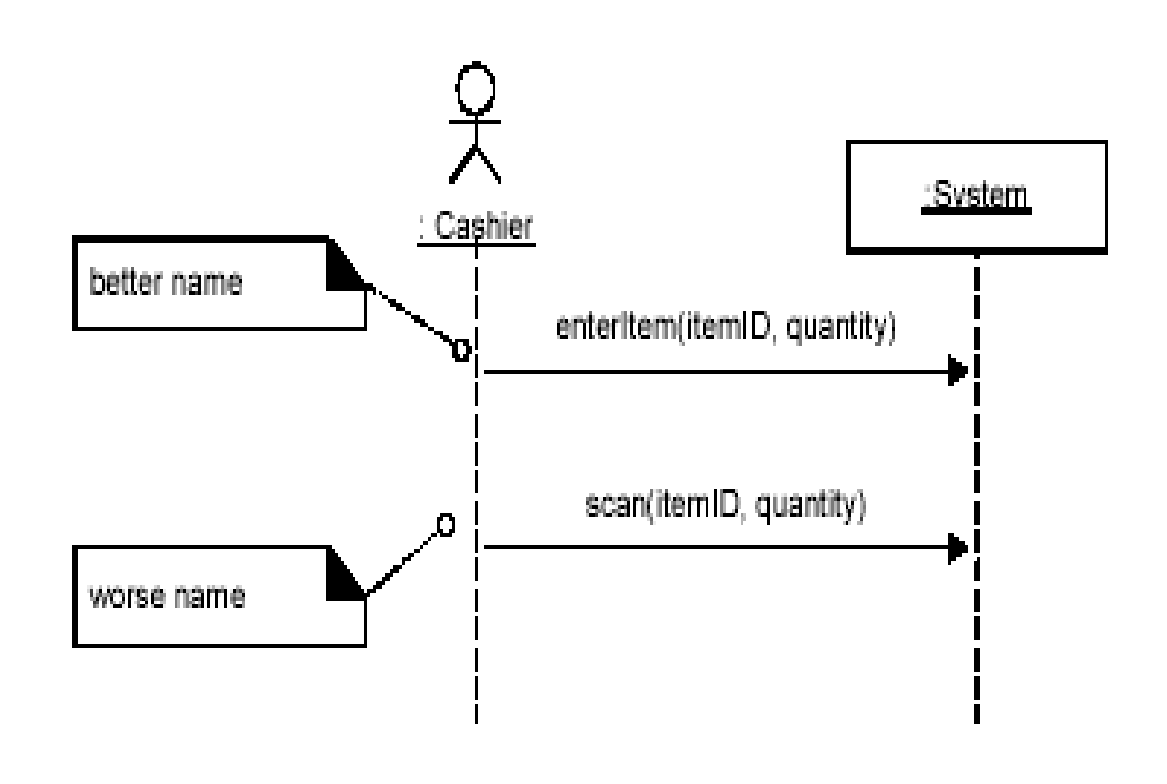
Naming System Events and Operations(2)

- System events and their associated system operations should be expressed at the level of intent rather than in terms of the physical input medium or widget.
- In order to improve the clarity, it is appropriate to start the name of the system event with a verb (for example- add....,enter....,end....,make.... etc.,). It also emphasizes the command origination of these events.

Naming System Events and Operations(3)

- For example “enterItem” is better than “scan” as it captures the intent of operation rather than what interface is used to capture the system event (design choice).

Choose event and operation names at an abstract level



Showing Use Case Text

- It is desirable to show at least fragments of use case text for the scenario.
- The text provides the details and context, while the diagram visually summarizes the interaction.

SSD with use case text

Simple Cash-only Process Sale scenario:

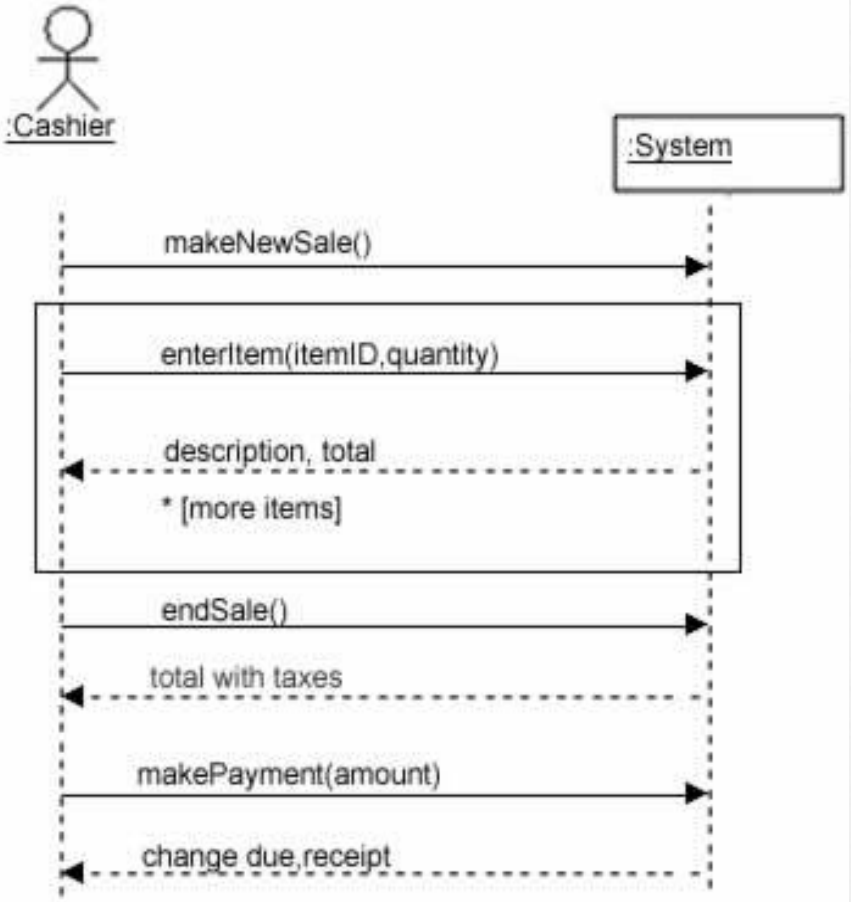
1.customer arrives at aPOS check out with goods and/or services to purchase.
2.Cashier starts a new sale.

3.Cashier enters a new item identifier.
4.System records new sale line item and presents item description,price and running total.

Cashier repeats steps 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.
....



SSD and the Glossary

- Since the terms used in SSDs are terse, they may need proper explanation. If these terms are not explained in use cases, glossary could be used.
- A glossary is less formal, it is easier to maintain and more intuitive to discuss with external parties such as users and customers.
- However, glossary data should be meaningful, otherwise it will be an unnecessary work.

System Sequence Diagrams Within the Unified Process

- System Sequence Diagrams are a visualization of the interactions implied in the use cases.
- System Sequence Diagrams were not explicitly mentioned in the original UP description.
- **In UP Phases**

  1.Inception: System Sequence Diagrams are not usually motivated in inception.

System Sequence Diagrams Within the Unified Process

Elaboration: It is useful to create System Sequence Diagrams during elaboration in order to -

- Identify the system events and major operations.
- To write system operation contracts (Contracts describe detailed system behavior) and
- To support estimation.