Object-Oriented Software Analysis and Design

School of Computer Science
University of Windsor

Sample UP artifact influence

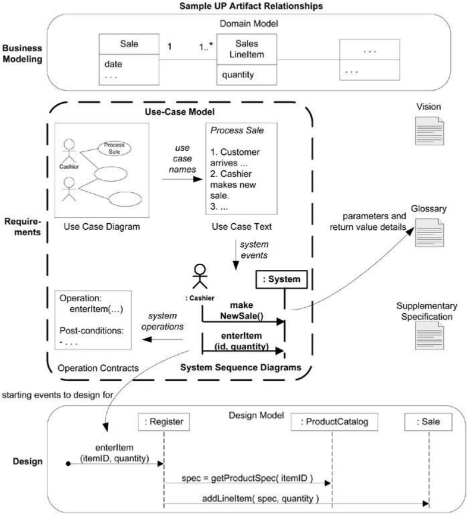


Figure: *Sample UP artifact influence.*

System Sequence Diagrams (SSD)

- A **system sequence diagram (SSD)** is a fast and easily created artifact that illustrates input and output events related to the systems under discussion.
- They are **input** to operation contracts and most importantly—object design.
- The use case text and its implied system events are **input** to SSD creation.
- The SSD operations (such as `enterItem`) can in turn be analyzed in the operation contracts, detailed in the Glossary, and **most important serve as the starting point for designing collaborating objects**.

Example: NextGen SSD

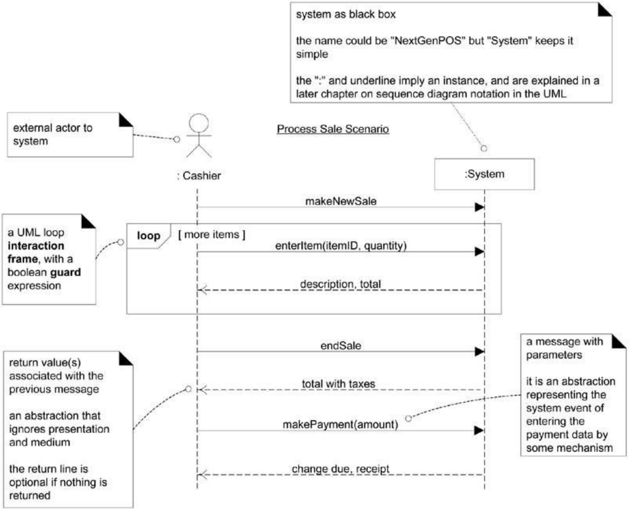

system as black box
the name could be "NextGenPOS" but "System" keeps it simple
the ":" and underline imply an instance, and are explained in a later chapter on sequence diagram notation in the UML
external actor to system
Process Sale Scenario
: Cashier
:System
makeNewSale
a UML loop **interaction frame**, 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

Figure: *SSD for a Process Sale scenario.*

What are System Sequence Diagrams?

- Use cases describe how external actors interact with the software system we are interested in creating.
- During this interaction an actor generates **system events** to a system, usually requesting some **system operation** to handle the event.

  For example,

  when a cashier enters an item's ID, the cashier is requesting the POS system to record that item's sale (the `enterItem` event). That event initiates an operation upon the system.

What are System Sequence Diagrams? (contd.)

- The UML includes **sequence diagrams** as a notation that can illustrate actor interactions and the operations initiated by them.
- A **system sequence diagram** is a picture that shows, for one particular scenario of a use case, the events that external actors generate, their order, and inter-system events.
- All systems are treated as a black box; the emphasis of the diagram is events that cross the system boundary from actors to systems.
- Guideline: Draw an SSD for a main success scenario of each use case, and frequent or complex alternative scenarios.

Why Draw an SSD?

- What events are coming in to our system? Why?

- Because we have to design the software to handle these events (from the mouse, keyboard, another system, ...) and execute a response.

- Basically, a software system reacts to three things:
  1. external events from actors (humans or computers),
  2. timer events, and
  3. faults or exceptions (which are often from external sources).

- It is useful to know what, precisely, are the external (input) events—the system events. They are an important part of analyzing system behavior.

Why Draw an SSD? (contd.)

- **System behavior** is a description of what a system does, without explaining how it does it.
- One part of that description is a system sequence diagram.
- Other parts include the use cases and system operation contracts.

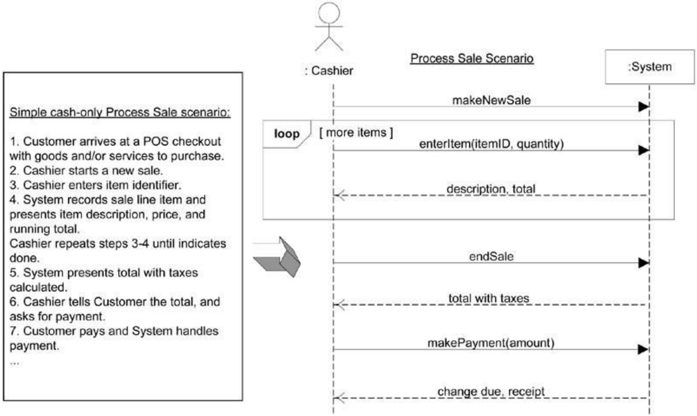
Applying UML: Sequence Diagrams

- The UML does not define something called a “system” sequence diagram but simply a “sequence diagram.”

- The qualification is used to emphasize its application to systems as black boxes.

- *Later, sequence diagrams will be used in another context to illustrate the design of interacting software objects to fulfill work.*

What is the Relationship Between SSDs and Use Cases?



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. 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.

...

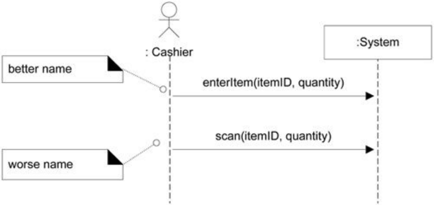
Figure: *SSDs are derived from use cases; they show one scenario.*

Applying UML: Should We Show Use Case Text in the SSD?

- Not usually. If you name the SSD appropriately, you can indicate the use case; for example, “Process Sale Scenario”.

How to Name System Events and Operations?

- Which is better, `scan(itemID)` or `enterItem(itemID)`?



- System events should be expressed at the abstract level of intention rather than in terms of the physical input device.

- It also improves clarity to start the name of a system event with a **verb** (add. . . , enter. . . , end. . . , make. . .), since it emphasizes these are commands or requests.

How to Model SSDs Involving Other External Systems?

- SSDs can also be used to illustrate collaborations between systems, such as between the NextGen POS and the external credit payment authorizer.

What SSD Information to Place in the Glossary?

- The elements shown in SSDs (**operation name**, **parameters**, **return data**) are terse.

- These may need proper explanation so that during design it is clear what is coming in and going out. The **Glossary** is a great place for these details.

  For example,

  in SSD, there is a return line containing the description "change due, receipt."

- That's a vague description about the receipt—a complex report. So, the UP Glossary can have a receipt entry, that shows sample receipts (perhaps a digital picture), and detailed contents and layout.

- Guideline: In general for many artifacts, show details in the Glossary.

Iterative and Evolutionary SSDs

- **Don't create SSDs for all scenarios.** Rather, draw them only for the scenarios chosen for the next iteration.

- And, they shouldn't take long to sketch - perhaps a few minutes or a half hour.

- SSDs are also very useful when you want to understand the interface and collaborations of existing systems, or to document the architecture.

SSDs Within the UP

- SSDs are part of the **Use-Case Model**—a visualization of the interactions implied in the scenarios of use cases.

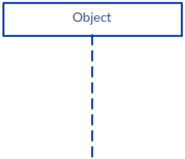
- SSDs are an example of the many possible skillful and widely used analysis and design artifacts or activities that the UP documents do not mention.

- But the UP, being very flexible, encourages the inclusion of any and all artifacts and practices that add value.

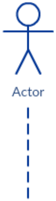
UP Phases

- ***Inception***
  - SSDs are not usually motivated in inception, unless you are doing rough estimating (don't expect inception estimating to be reliable) involving a technique that is based on identifying system operations.
- ***Elaboration***
  - Most SSDs are created during elaboration, when it is useful to identify the details of the system events to clarify what major operations the system must be designed to handle, write system operation contracts, and possibly to support estimation.

Applying UML: Sequence Diagrams

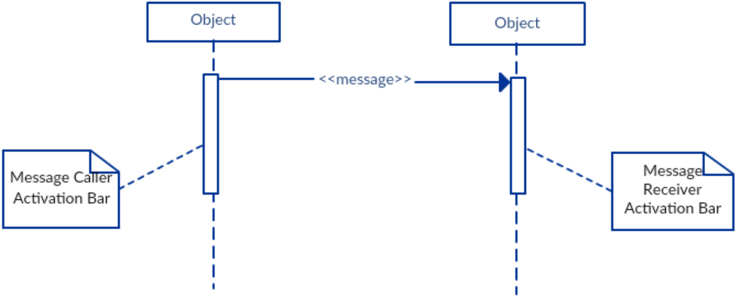


(a)

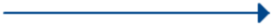


(b)

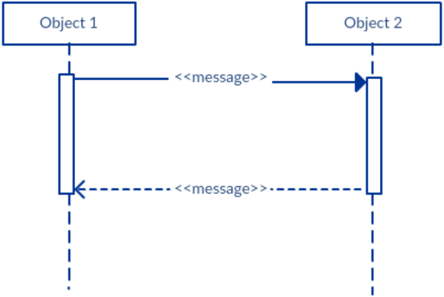
Sequence Diagram: Activation Bars



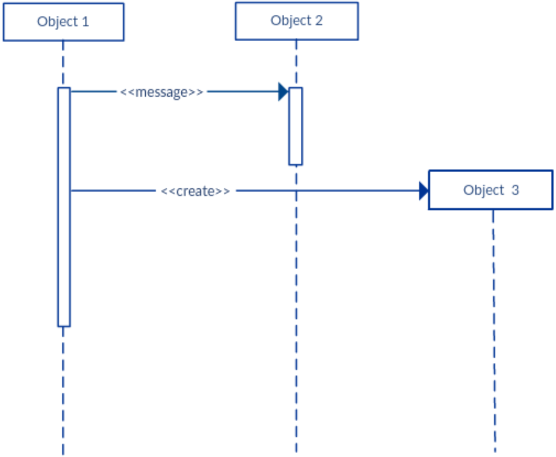
Sequence Diagram: A Message



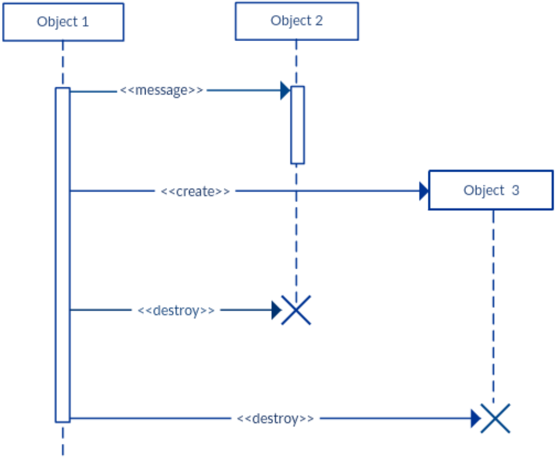
Sequence Diagram: Return Message



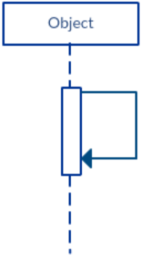
Sequence Diagram: Participant Creation Message



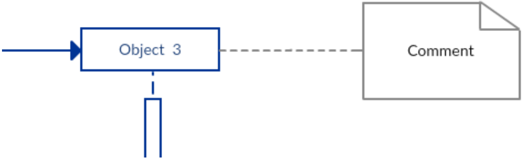
Sequence Diagram: Participant Destruction Message



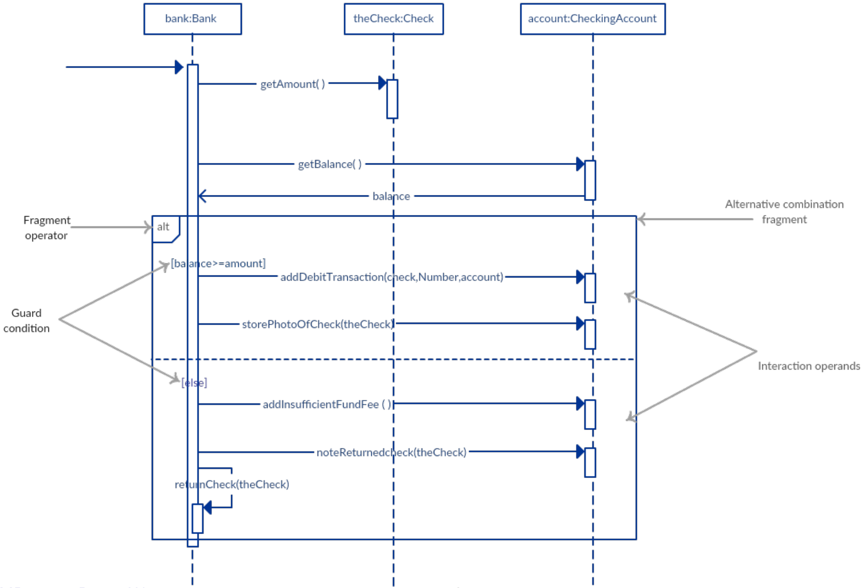
Sequence Diagram: Reflexive Message



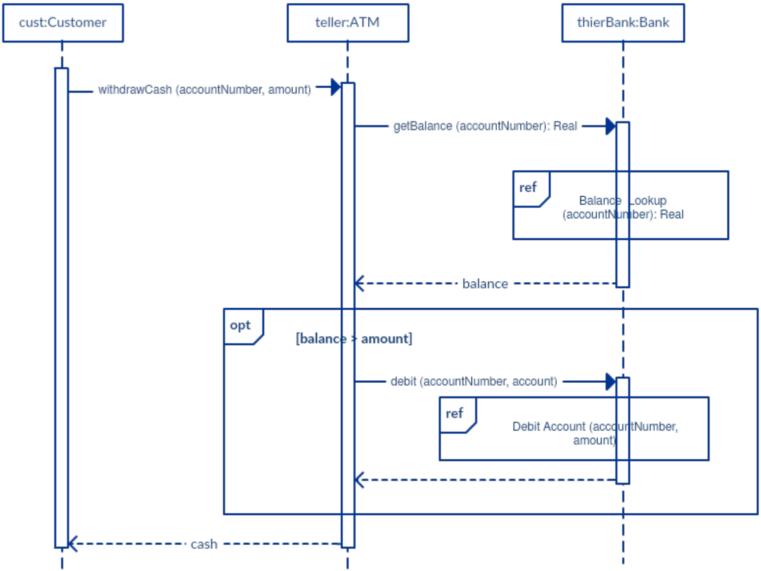
Sequence Diagram: Comment



Sequence Diagram: Alternatives



Sequence Diagram: Options


cust:Customer
teller:ATM
thierBank:Bank
withdrawCash (accountNumber, amount)
getBalance (accountNumber): Real
ref
Balance Lookup
(accountNumber): Real
balance
opt
[balance > amount]
debit (accountNumber, account)
ref
Debit Account (accountNumber,
amount)
cash

It's Quiz Time

1. Draw an SSD for a main success scenario of each use case, and frequent or complex alternative scenarios. (True or False)

2. SSDs are not part of the Use-Case Model. (True or False)

3. Use cases describe how external actors interact with the software system we are interested in creating. (True or False)