SysML Behavioural Diagrams

Sequence Diagrams

Introduction to Systems Engineering
I2ISE

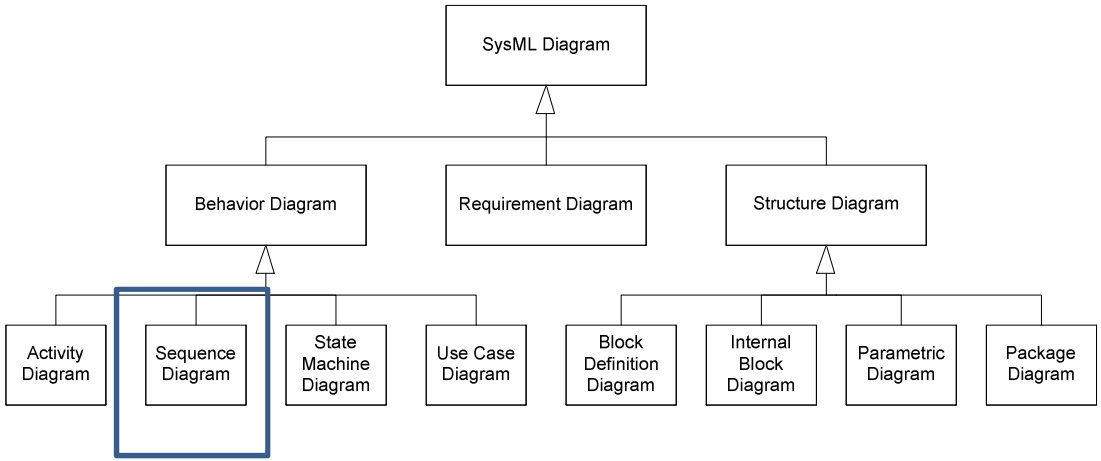
SysML Stucture vs. behaviour

- We have learned a lot about how to model *structure* in SysML
  - Block Definition Diagrams
  - Internal Block Diagrams

- Now, we will look at how we can model *behaviour* in SysML
  - Sequence diagrams
  - State Machines

Sequence diagrams

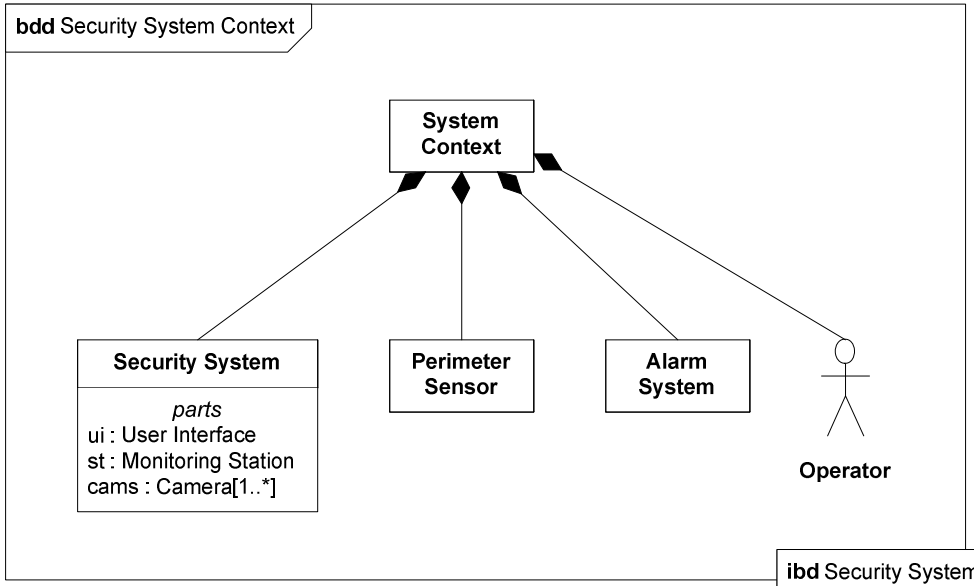
- Sequence diagrams (diagram type *sd*) model interactions between *parts* of a block

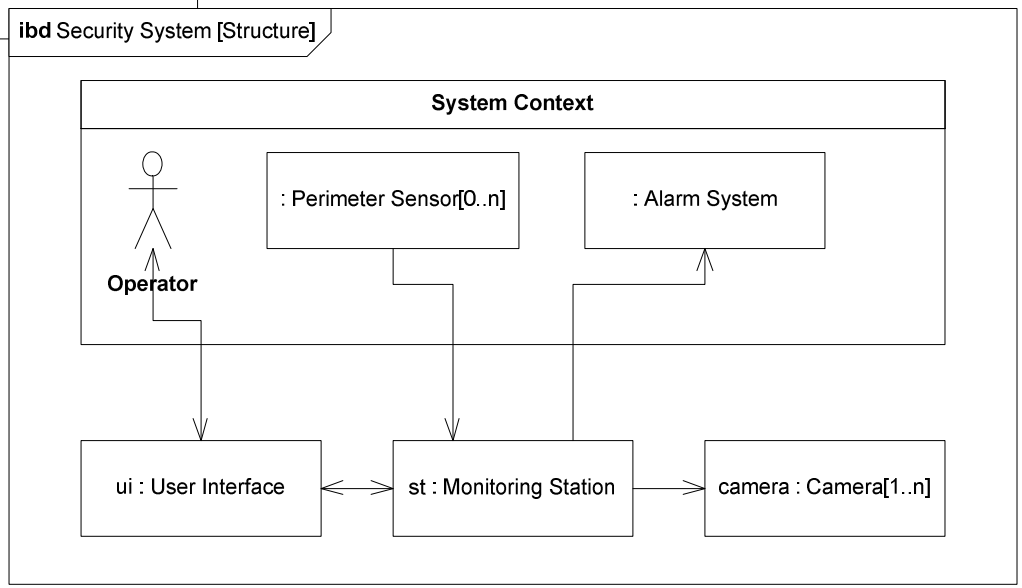


Sequence diagrams

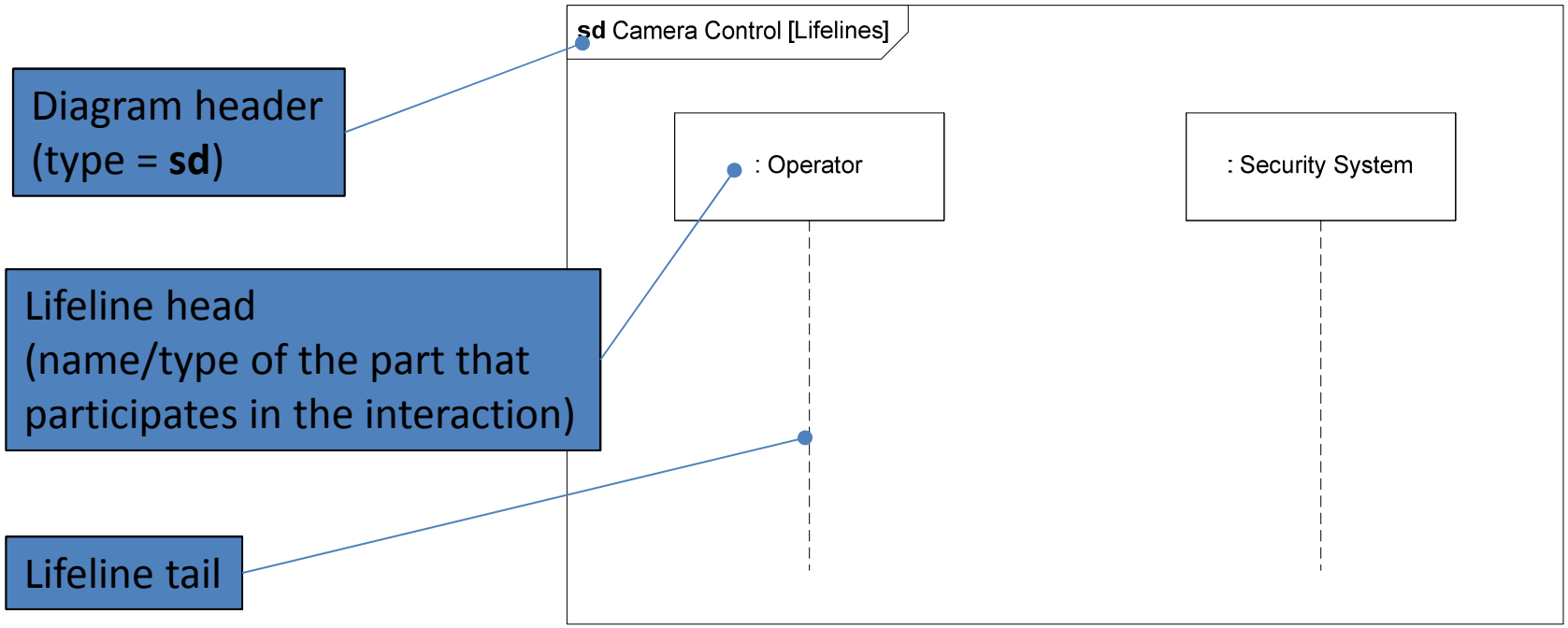
- Sequence diagrams are used to model *message*-based behaviour
- The interactions take place within a block between its elements of internal structure (parts)
- The basic diagram consists of *lifelines* with *messages* between them.

SD's – example system (structure)

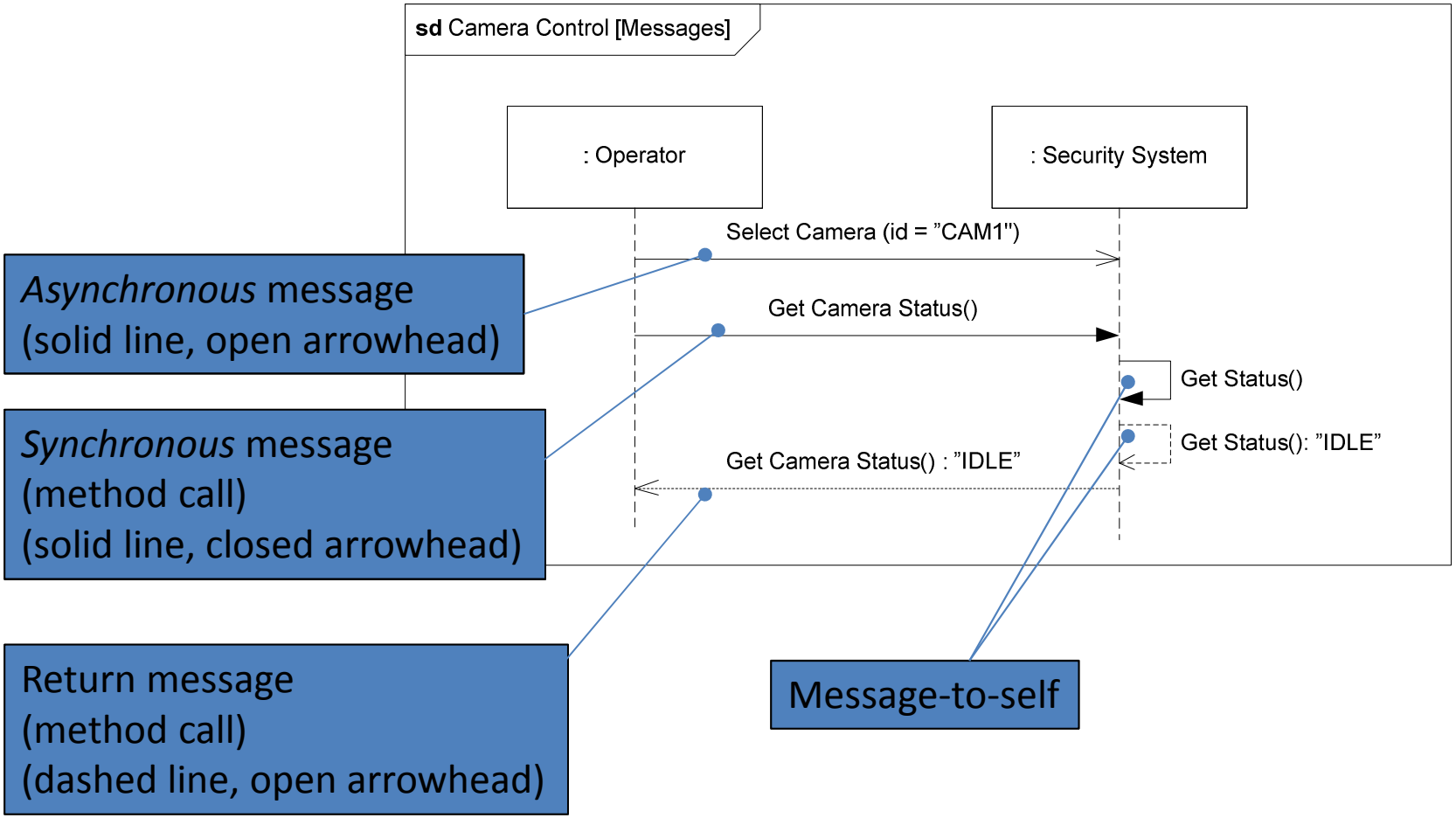




SD’s – lifelines



SD's – messages



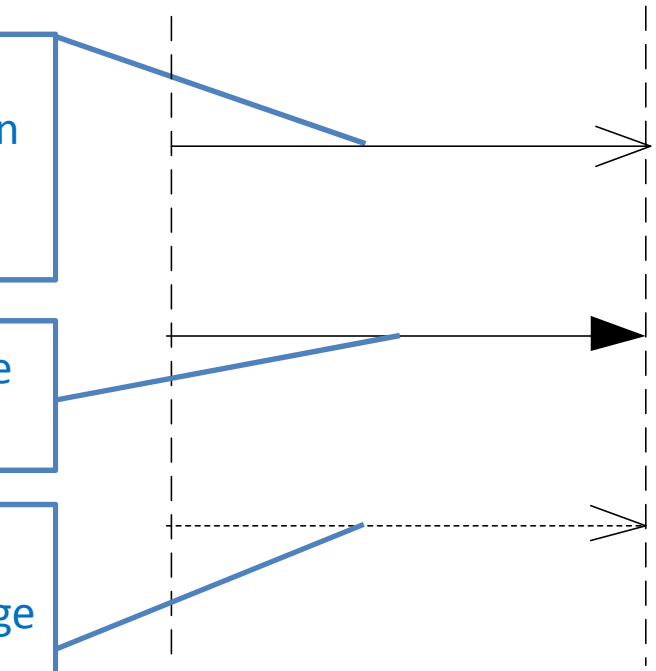
SD's – async/sync/reply

There are two basic types of messages: **asynchronous** and **synchronous**. A sender of an asynchronous message continues to execute immediately after sending the message, whereas a sender of a synchronous message waits until it receives a reply from the receiver.

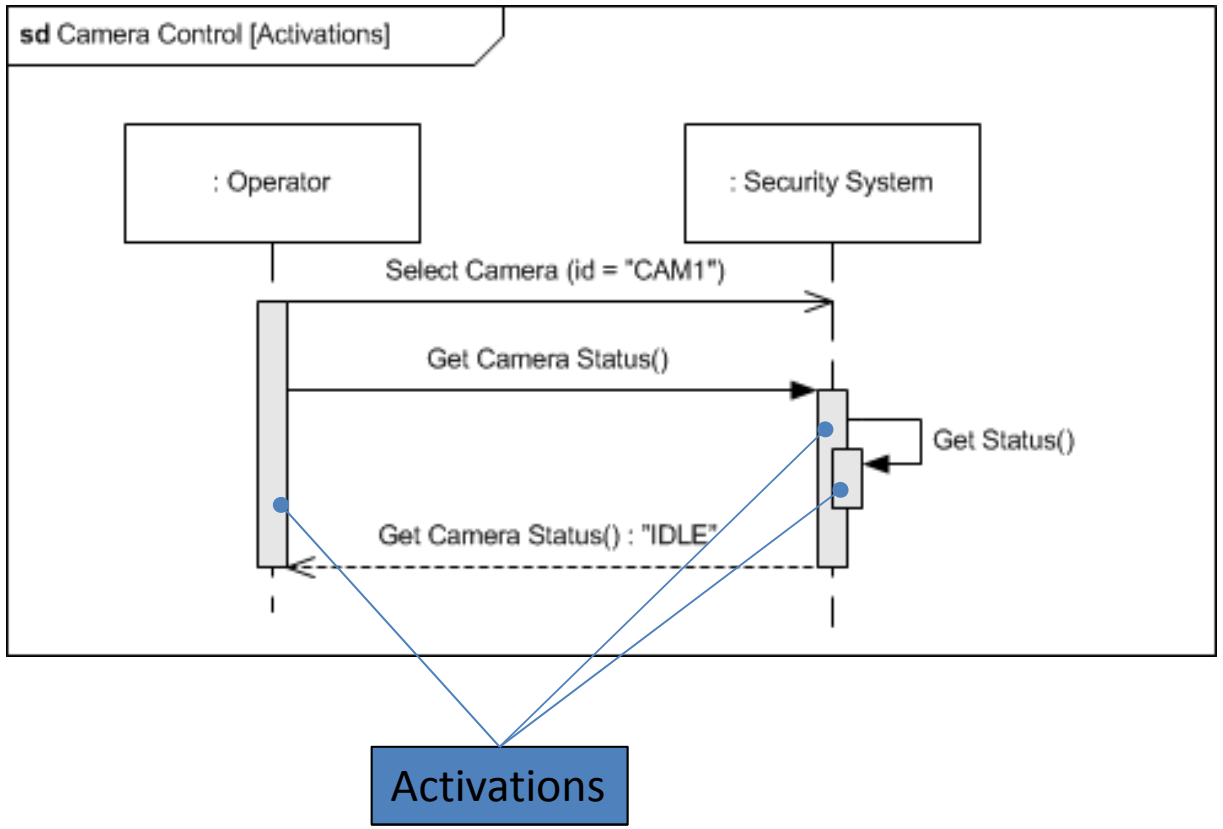
An open arrowhead means an **asynchronous message**. Input arguments associated with the message are shown in parentheses as a comma-separated list after the message name.

A closed arrowhead means a **synchronous message**. The notation is the same as for asynchronous messages.

An open arrowhead on a dashed line shows a **reply message**. Output arguments associated with the message are shown in parentheses after the message name, and the return value, if any, is shown after the argument list.

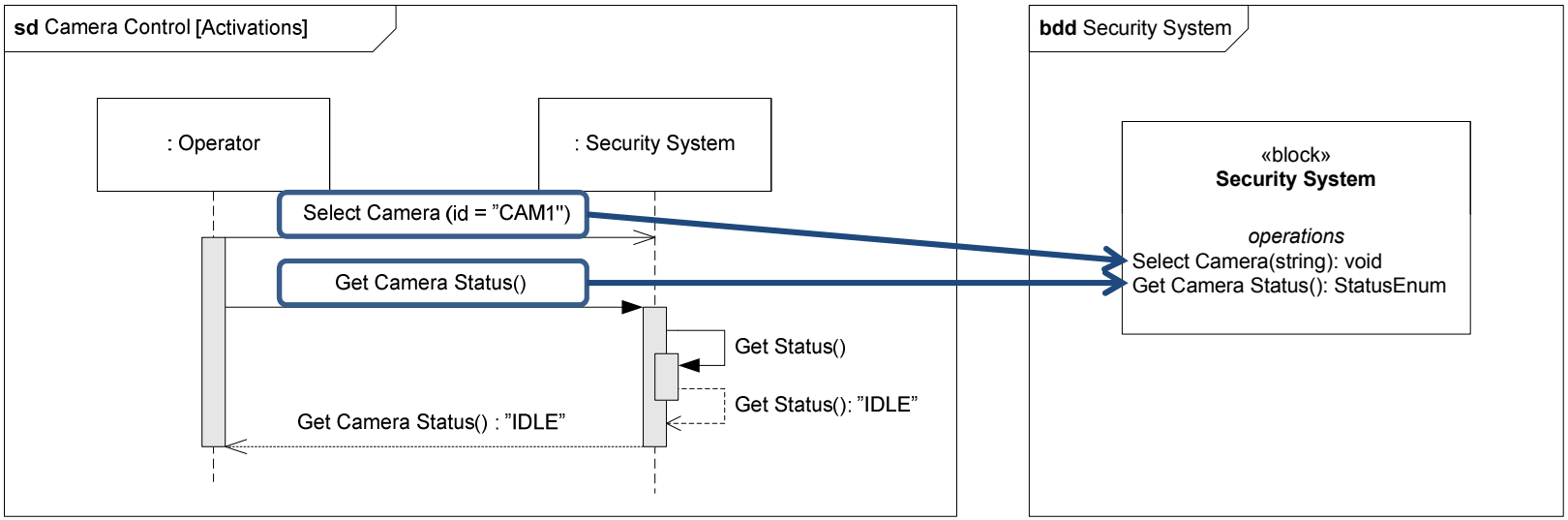


SD’s – activations



Structure and behaviour – a link

- The behaviour (operations) used in an SD are found in the operations compartment of the enclosing block



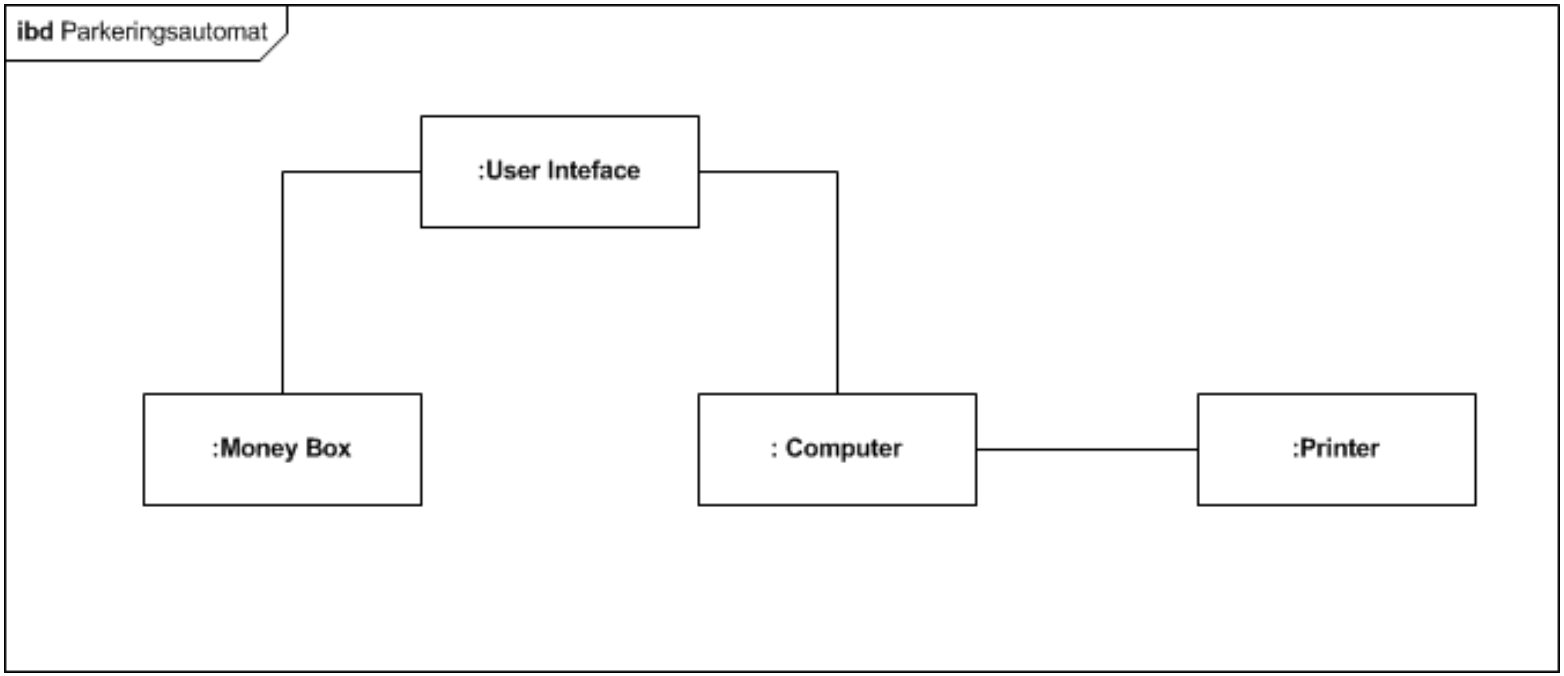
SD's – your turn – Parking Machine

- Draw a sequence diagram for buy ticket
- Use actor and blocks:
  User Interface, Computer, Money Box and Printer

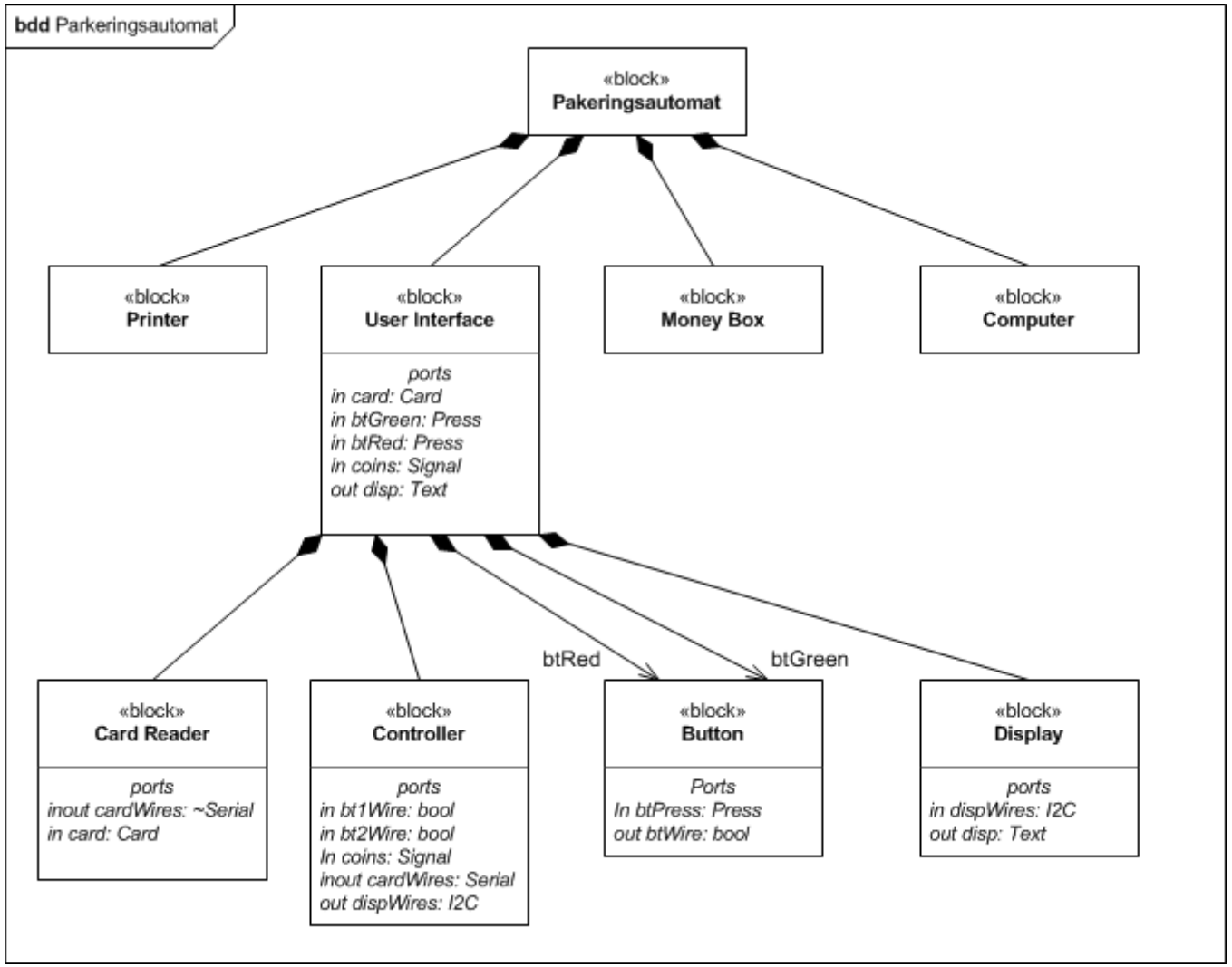
UC: Buy Ticket - Main scenario

1. User inserts coins in the Parking Machine
2. Parking Machine displays total amount and time
3. User press the pay button (green)
4. Parking Machine prints a ticket

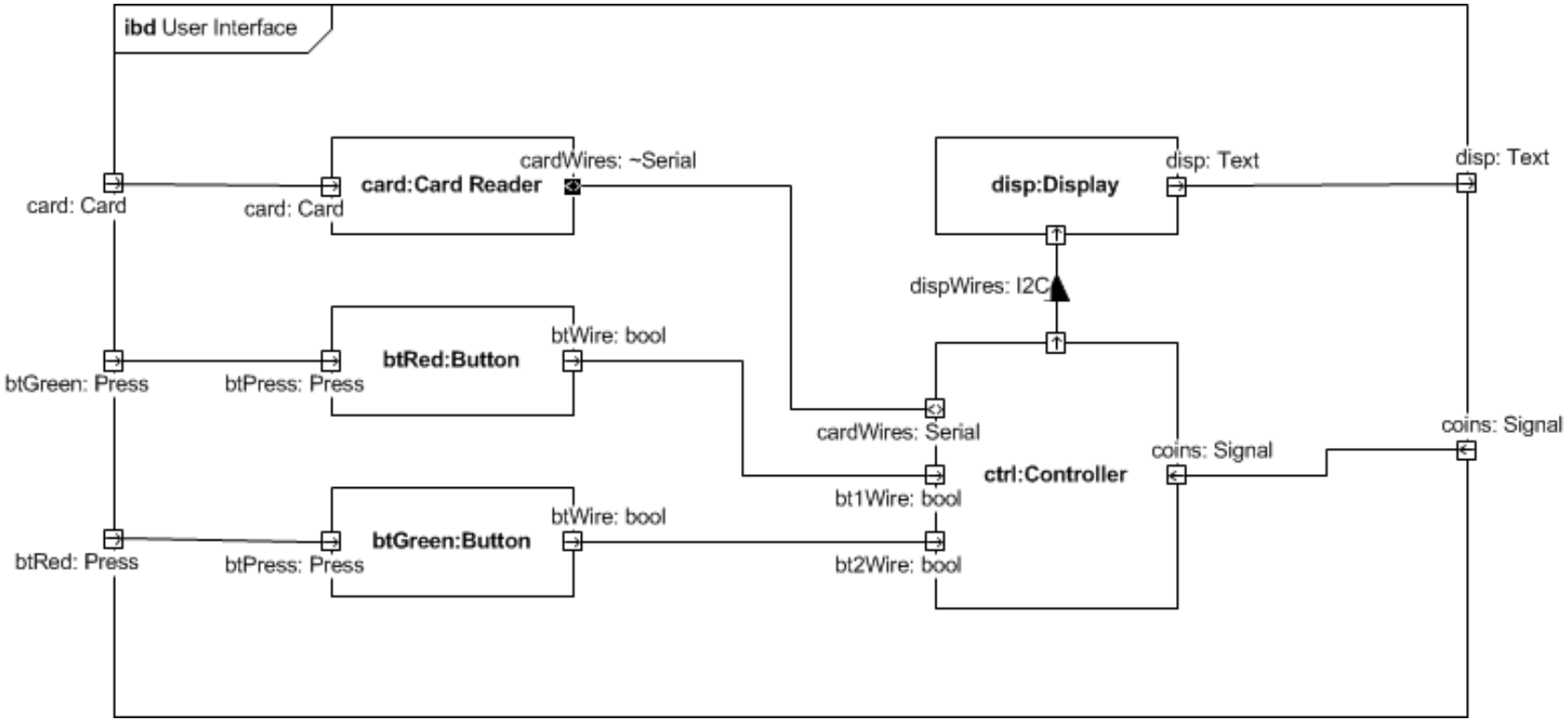
ibd Parkeringsautomat



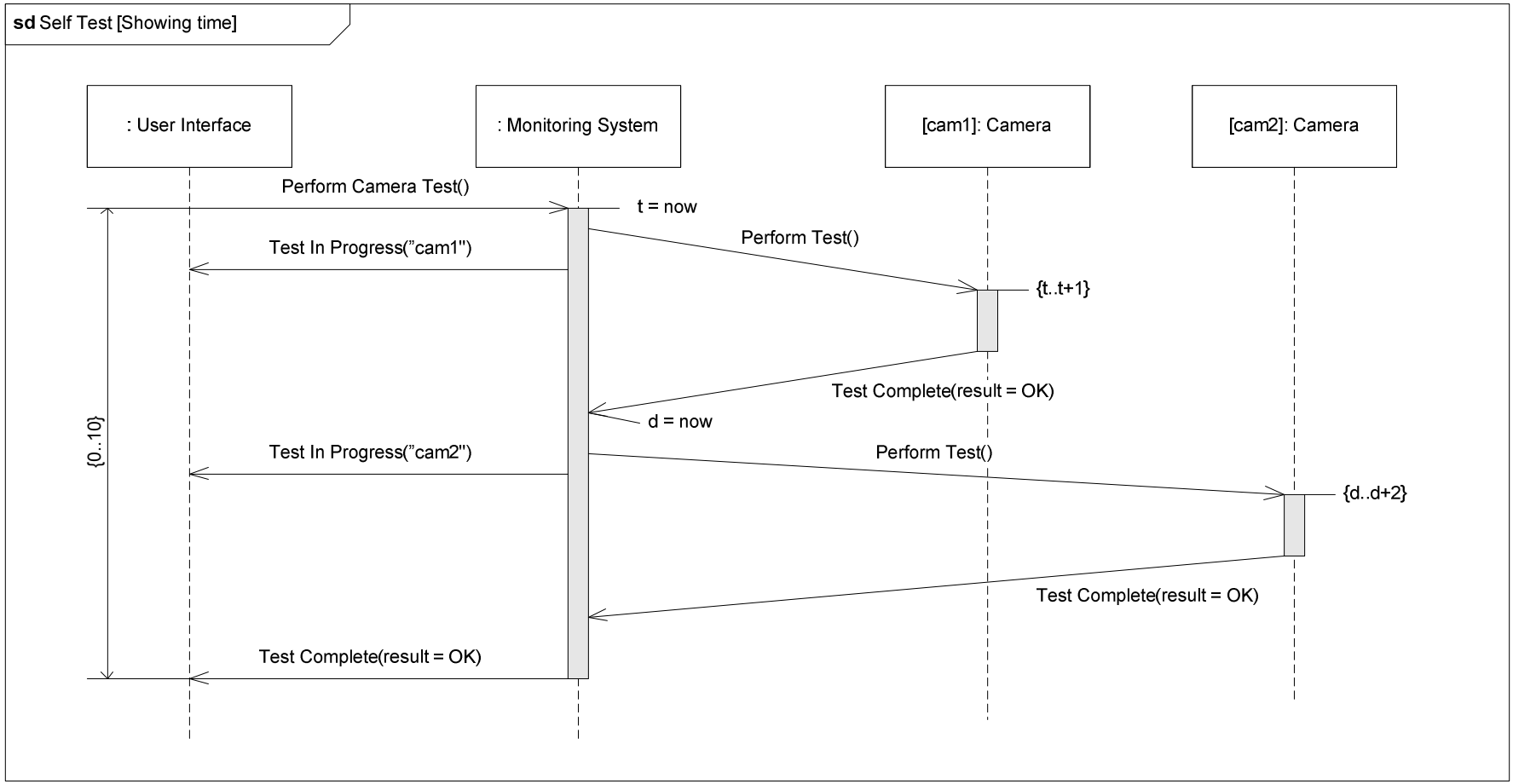
bdd Parkeringsautomat



ibd User Interface



SD's – representing time

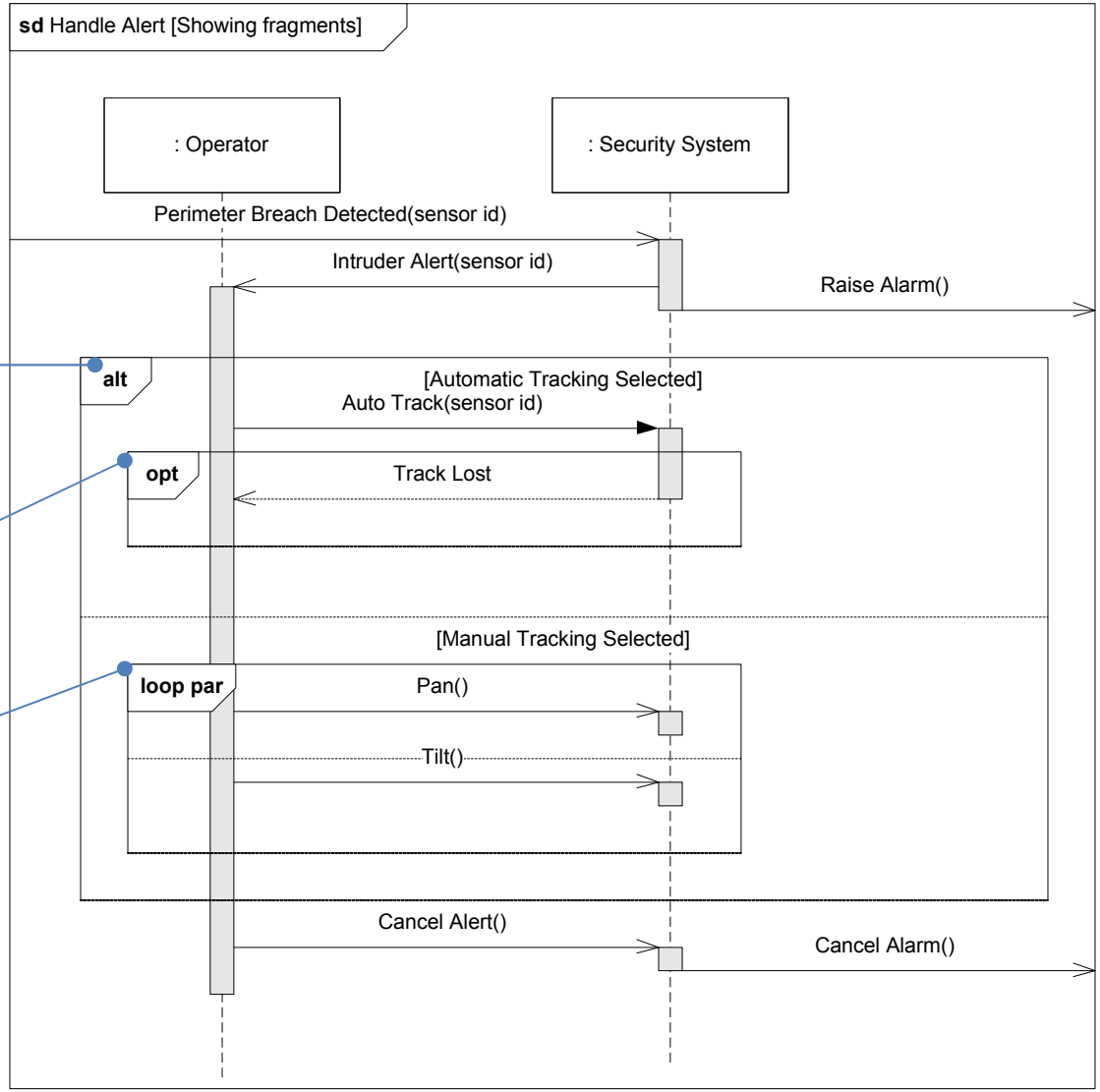


SD's – fragments

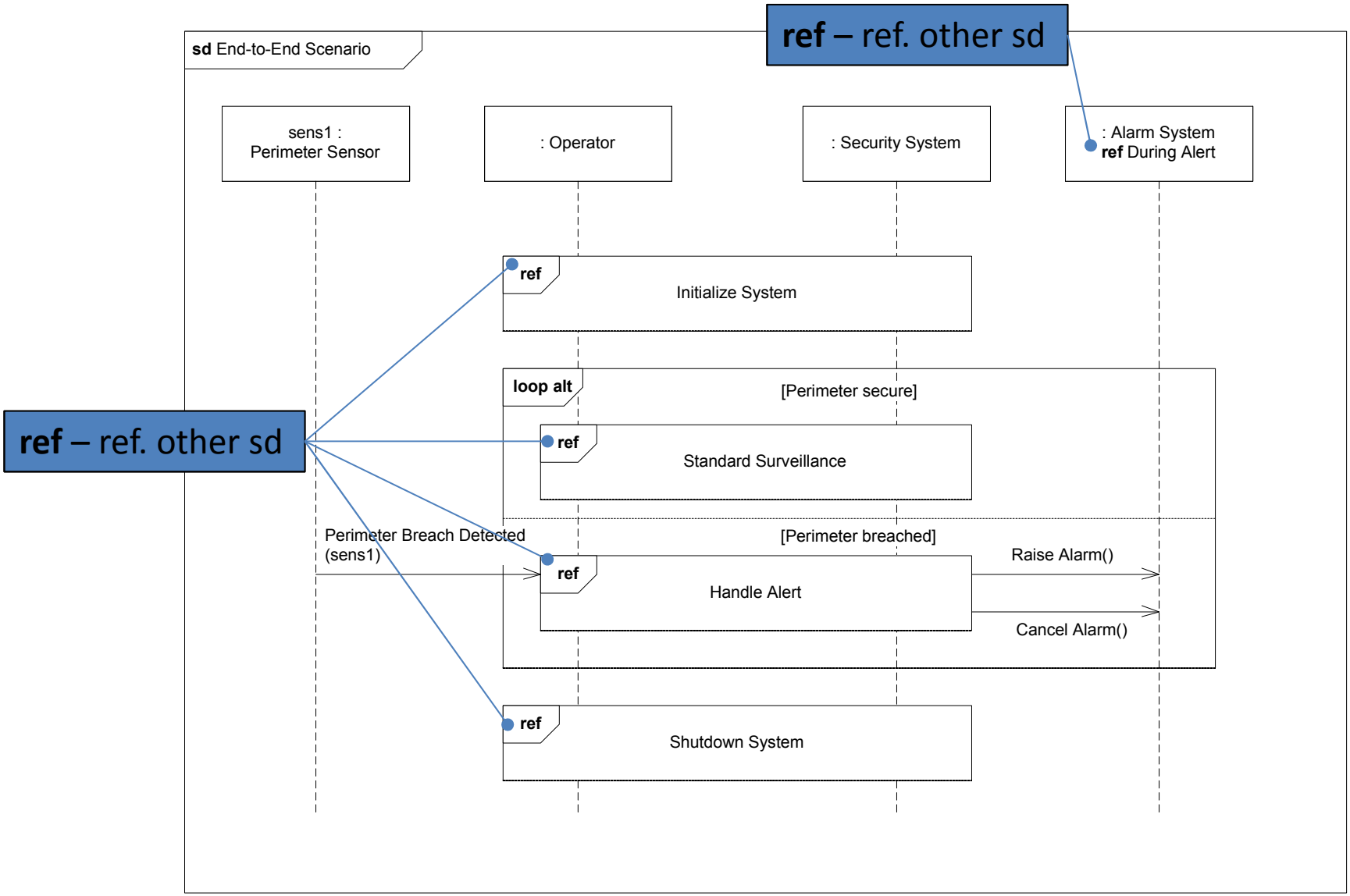
alt - Alternative activities

opt - Optional activity

loop – loop activity
par – parallel activities



SD's – reference blocks



SD's – your turn!

- Create a sequence diagram for the RVM scenario *Recycle Containers* below
  - Participants: *User* and *RVM*
- Add operations to the RVM on a BDD

Main Scenario for Use Case *Recycle containers*

1. User arrives at RVM and is informed to insert containers.
2. User places container in the in-feed.
3. RVM scans container and *either*
   a) accepts the container, collects the container from the in-feed, adds the return deposit to the collected amount, and displays the type and value of the accepted container and the total collected amount; *or*
   b) does not accept the container, rejects the container to User, and displays that the container is not accepted and the total collected amount.

Step 2 through 3 is repeated until User is done feeding containers.

1. User request the return deposit receipt.
2. RVM prints out the return deposit receipt, and resets the collected amount.