# Lab 4: Process/Data Consistency Checking

- Why checking consistency? To verify that the process considers all data dependencies (deadlock avoidance)
- How to check consistency? By comparing the processes and related state transition Diagrams

- **Object Lifecycle Conformance (OLC):** A process fits to a data object, if the process requires only those state changes of the object that are defined in the life cycle of the object.
  *"Check the state transition diagram with the object life cycle"*
- **Object Lifecycle Coverage (OLCC):** A process covers the behavior of a data object, if it allows all possible state transitions of the object.
  *"Check the object life cycle with the state transition diagram"*

**Example 1:** Derive data object life cycle from the following process model

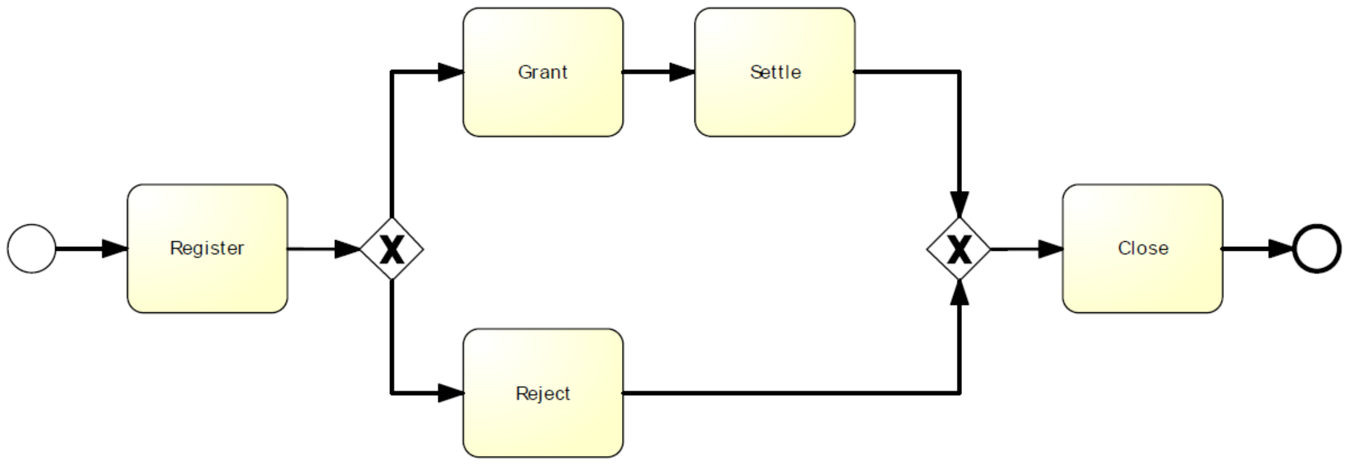


Figure 1: Example 1

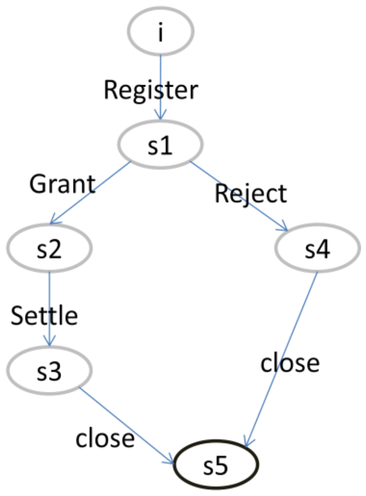


Figure 2: State transition diagram for Example 1

**Example 2: Book Lending Process**

Check the following process model and the state diagrams and determine on each if it is OLC or OLCC fulfilled

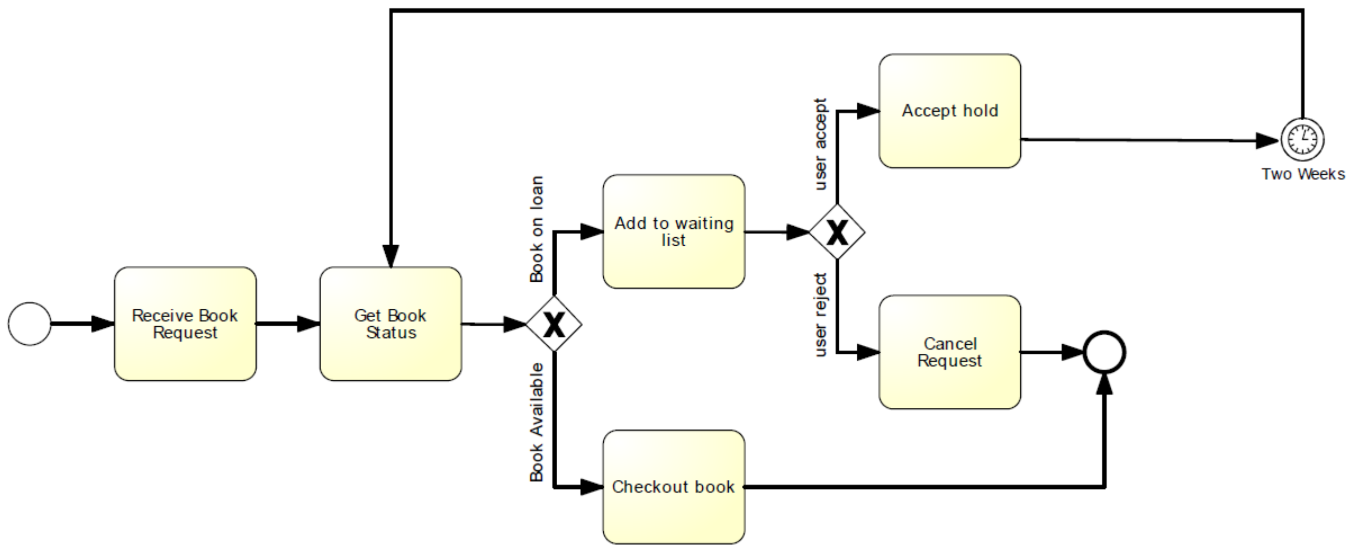


Figure 3: Example 2 BPMN

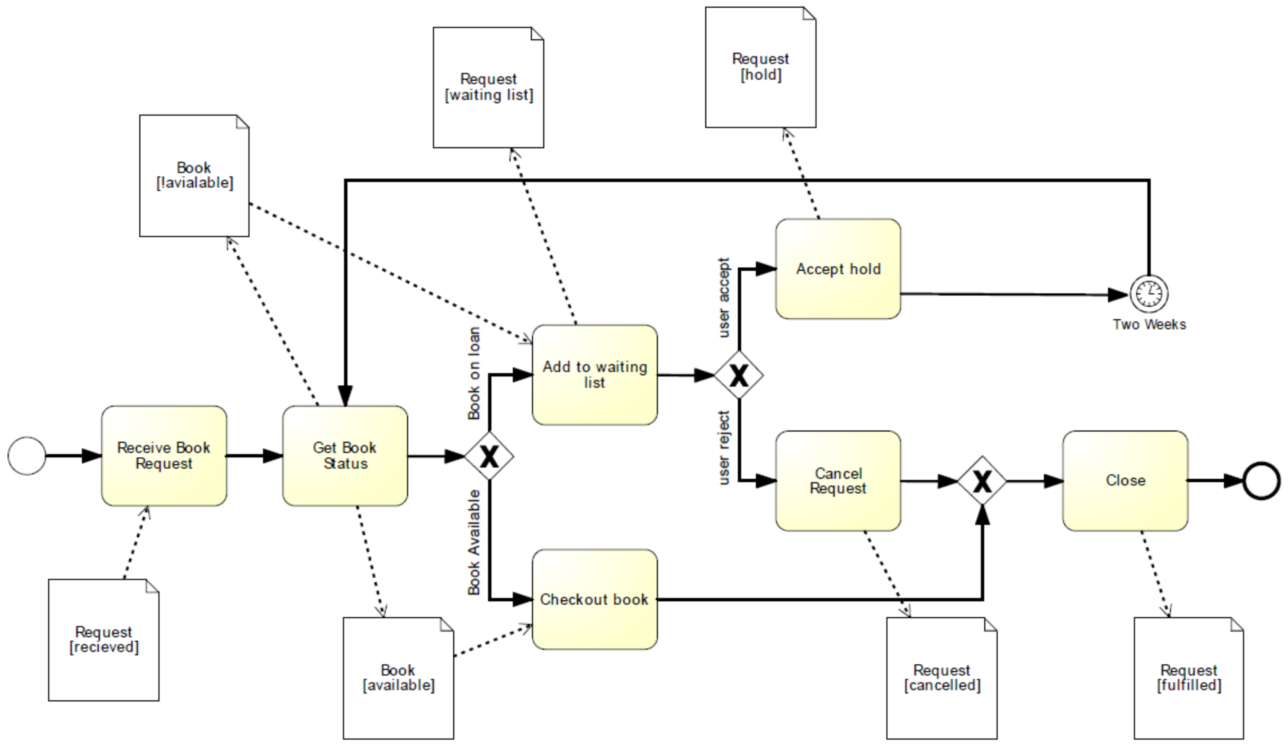


Figure 4: Example 2 BPMN with data objects

## Request object life cycle:

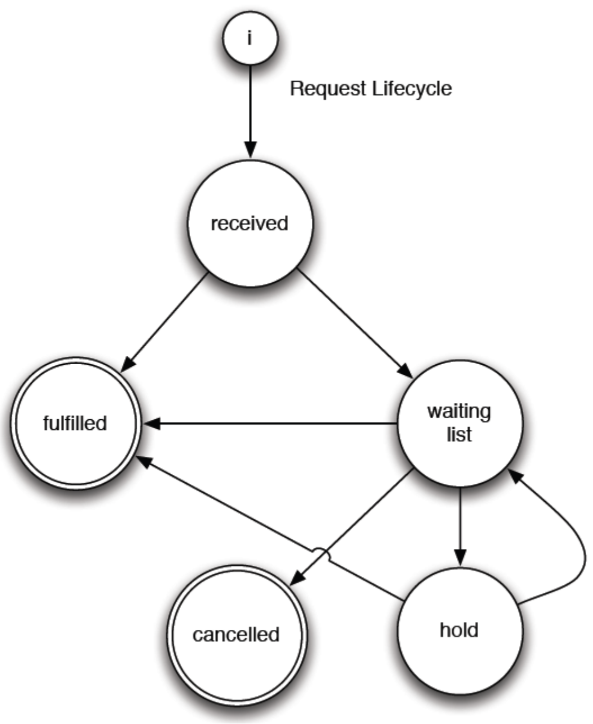


Figure 5: Request object -> OLC not fulfilled and OLCC not fulfilled

## Book object life cycle:

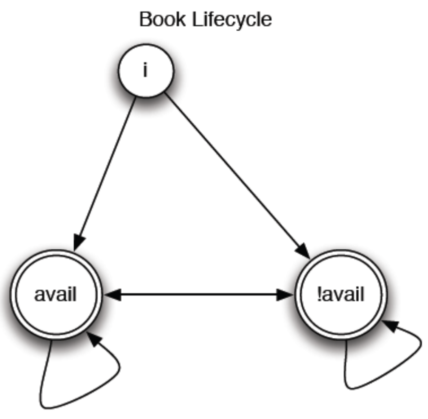


Figure 6: Book object -> OLC fulfilled but OLCC not fulfilled

Figure 6 is not OLCC as our BPMN doesn't cover the state transition from available to !available nor the cycle of available.

***Can we make any changes to make it OLCC fulfilled?***

The following image presents the state transition from available to !available (below Red circled)
This change means our end state will always be the "!available" but we still can't have the "available" loop

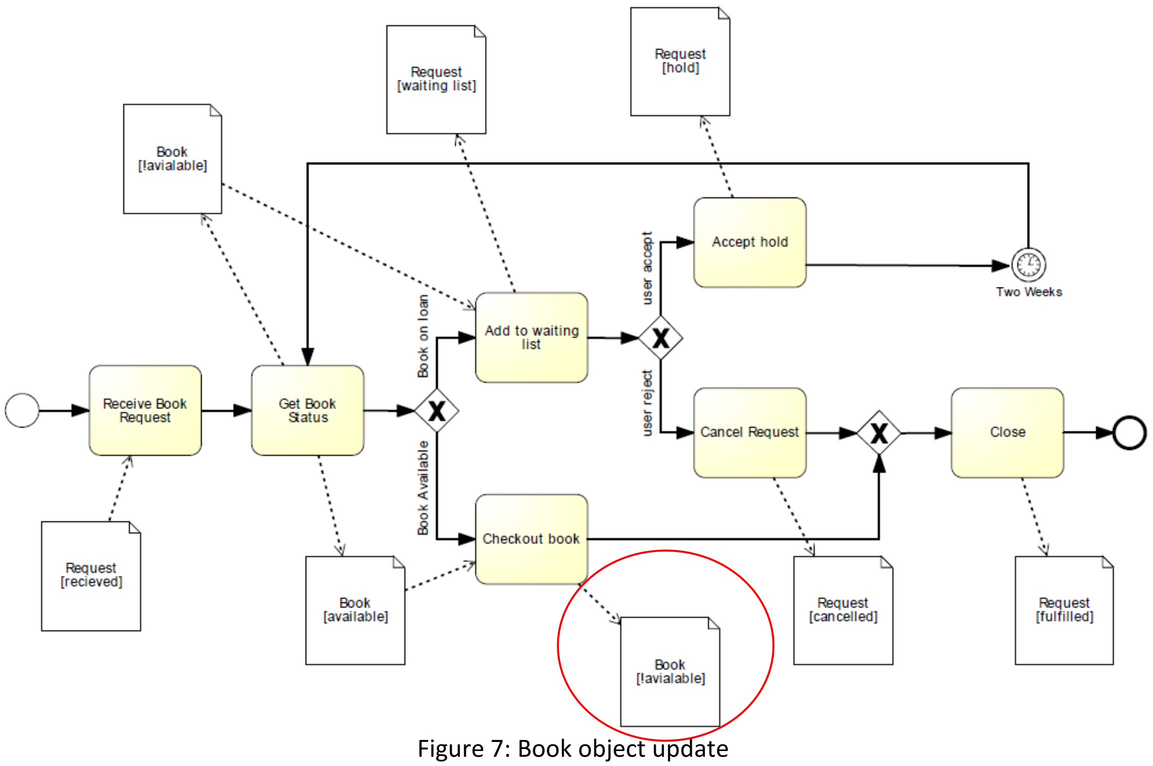


Figure 7: Book object update

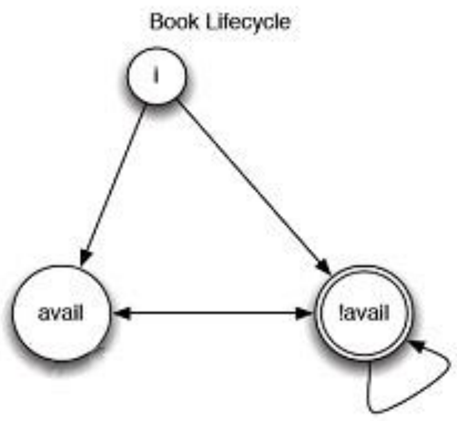
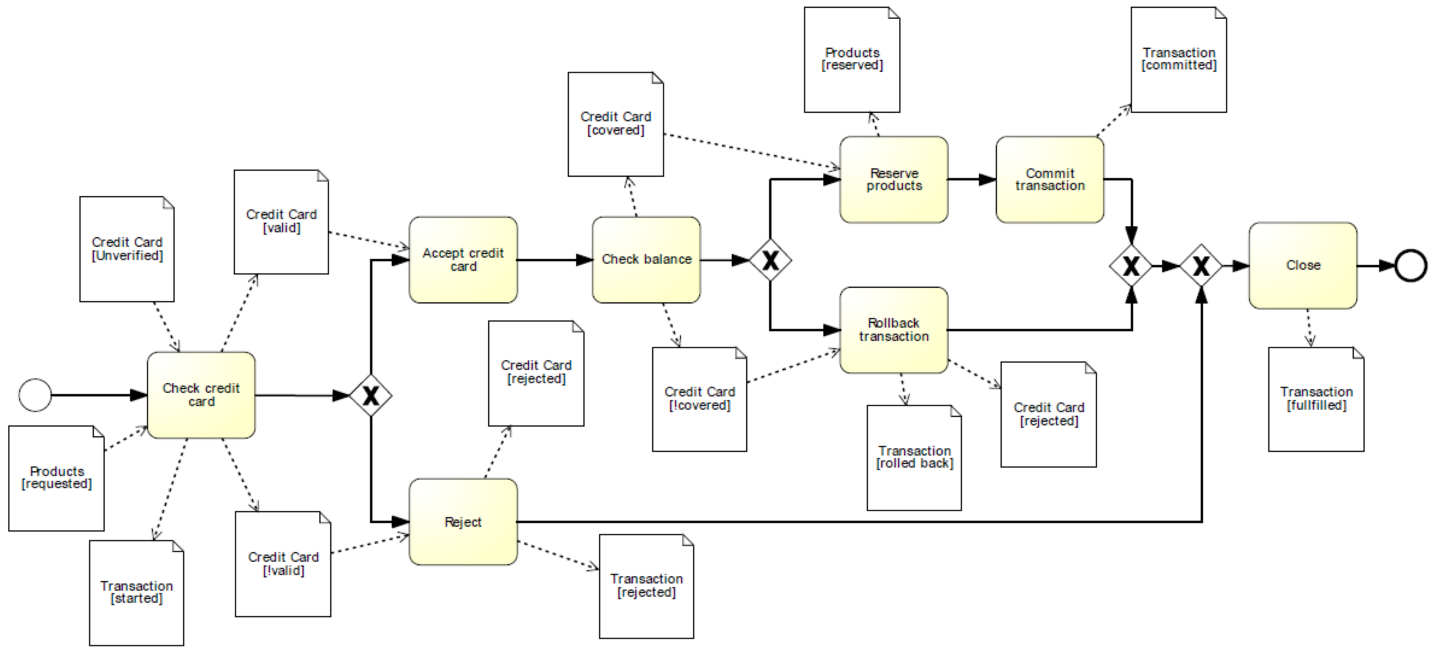


Figure 8: Actual state transition diagram of Example 2

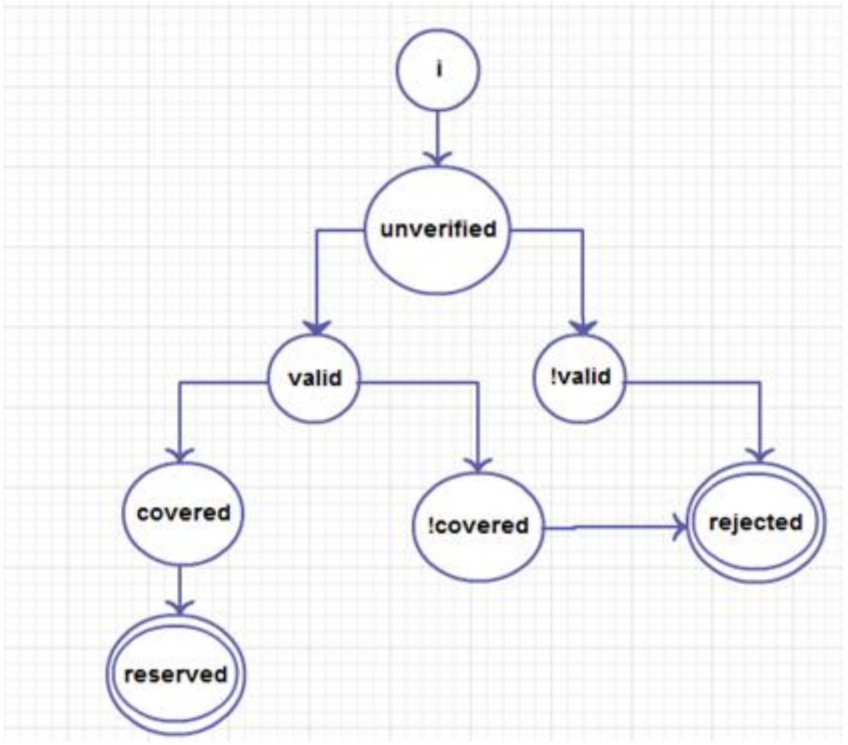
Figure 8 is both not OLC and not OLCC, as our BPMN doesn't cover the state transition from available to !available nor the cycle of available, besides available is not termination state anymore.

*Example (3):*



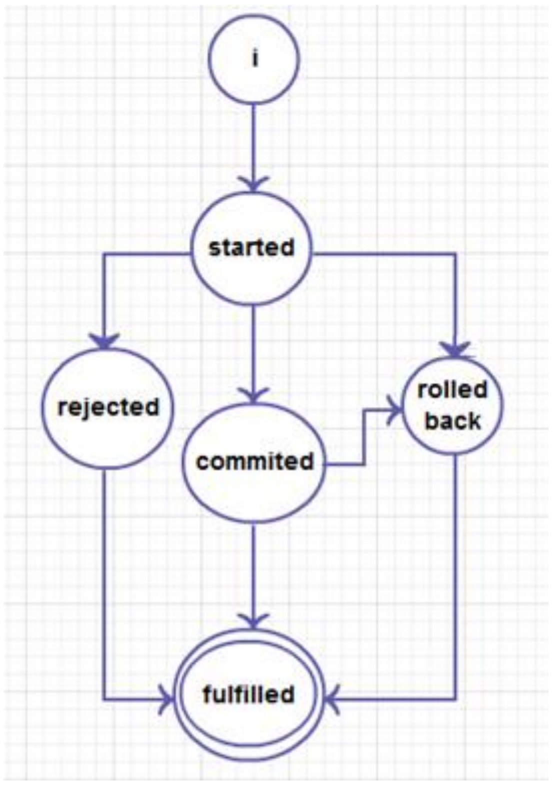
**Credit Card object life cycle:**

It’s not OLC and not OLCC (reserved is an extra state; covered should be the end state)



**Transaction object life cycle:**

Its' OLC but not OLCC (link between commit and rolled back not supported)


i
started
rejected
commited
rolled back
fulfilled