Library USE model with State Machines

We will now consider state machines or state transition diagrams for individual objects or classes in USE. A sequence diagram shows the interactions between object, but not the internal changes of state within an object. A state machine diagram is required for this.

Consider the following USE model, where only one library class is considered so as to reduce the complexity. An enumerated type is also used here to show you how it's done.

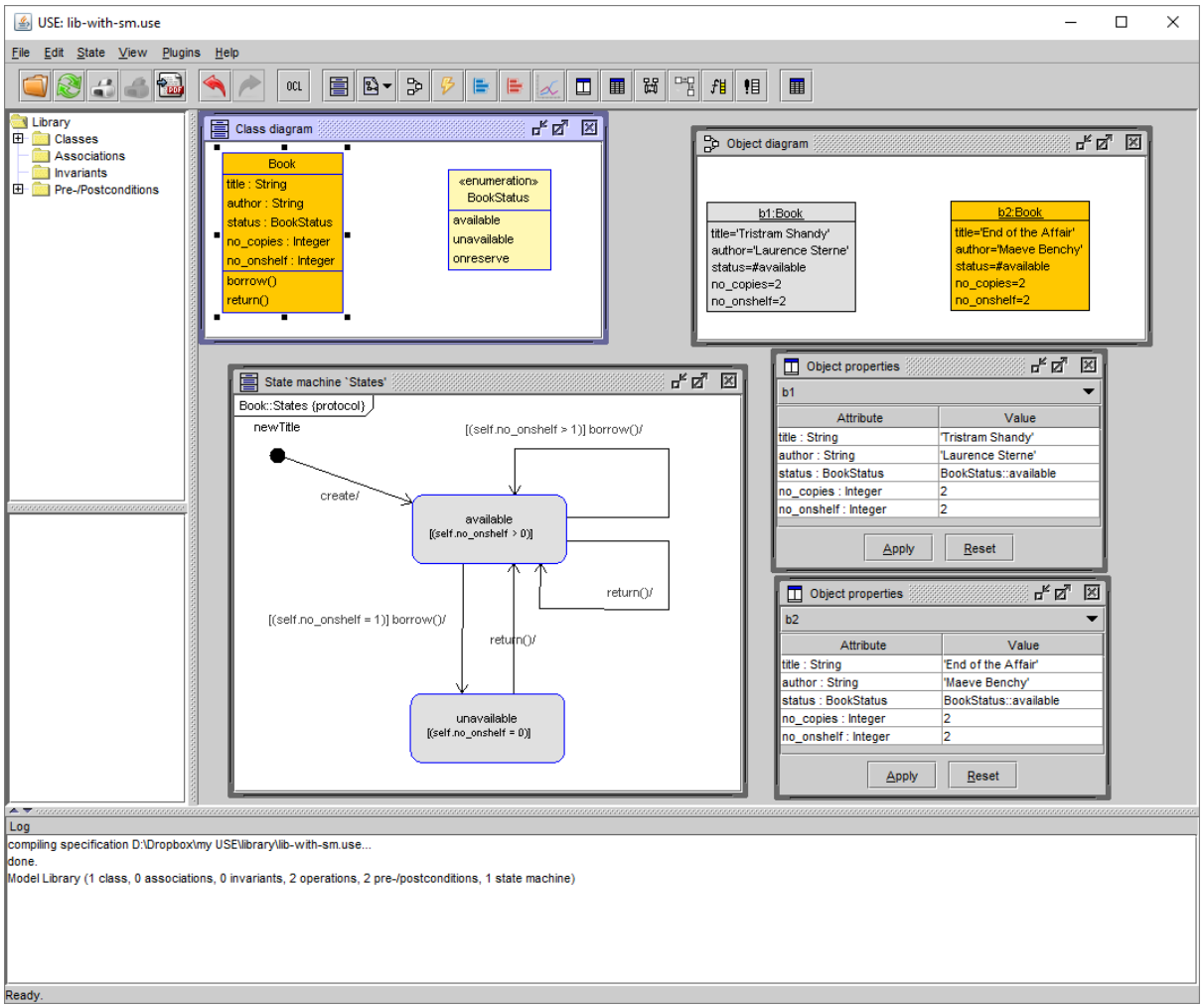
```
model Library
-- reduced version showing how to use state machines for Book class

enum BookStatus { available, unavailable, onreserve}

class Book
    attributes
        title : String
        author : String
        status : BookStatus init = #available
        no_copies : Integer init = 2
        no_onshelf : Integer init = 2

    operations
        borrow()
        begin
            self.no_onshelf := self.no_onshelf - 1;
            if (self.no_onshelf = 0) then
                self.status := #unavailable
            end
        end

        return() begin
            self.no_onshelf := self.no_onshelf + 1;
            self.status := #available
        end
        post: no_onshelf = no_onshelf@pre + 1

    statemachines
        psm States
        states
            newTitle : initial
            available       [no_onshelf > 0]
            unavailable     [no_onshelf = 0]
        transitions
            newTitle ->  available  { create }
            available -> unavailable { [no_onshelf = 1] borrow() }
            available -> available { [no_onshelf > 1] borrow() }
            available -> available { return() }
            unavailable -> available { return() }
        end
end
```

Notice that there are no preconditions for the borrow() method, the state machine obviates their necessity. The operation return() does have a postcondition which could be moved to the state machine.

Execute the following SOIL command to animate the model.

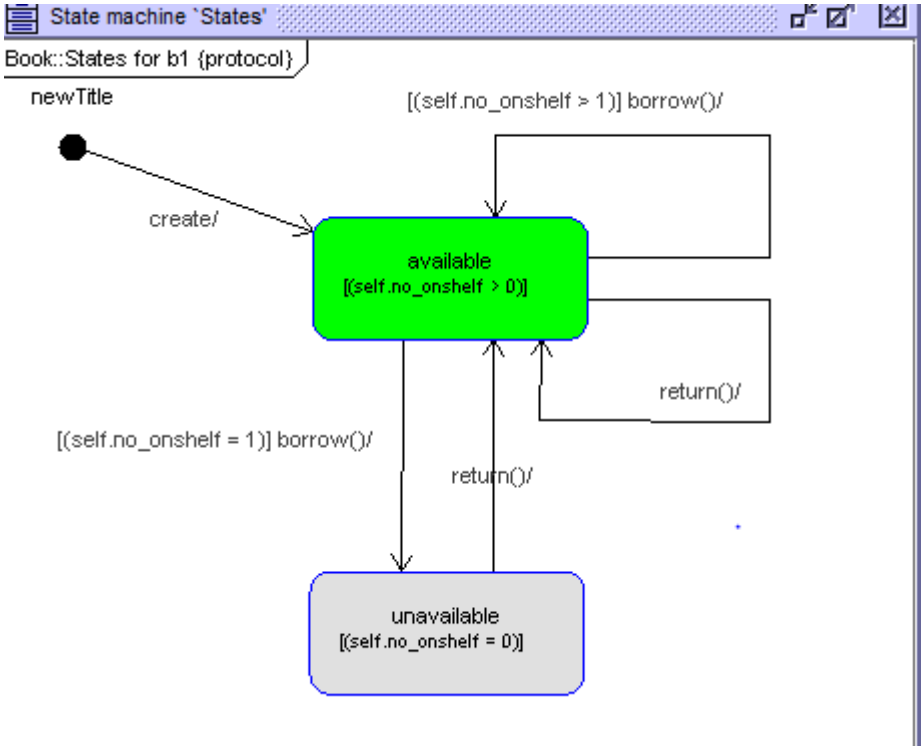
```
!new Book('b1')
!b1.title := 'Tristram Shandy'
!b1.author := 'Laurence Sterne'

!new Book('b2')
!b2.title := 'End of the Affair'
!b2.author := 'Maeve Benchy'
```

When the model is opened in USE and the objects created, select and right-click on the Book class in the class diagram window, then select **Show protocol state machine / States**. Also open the properties windows for both object by simply double-clicking on them. Layouts can then be rearranged to get something like:



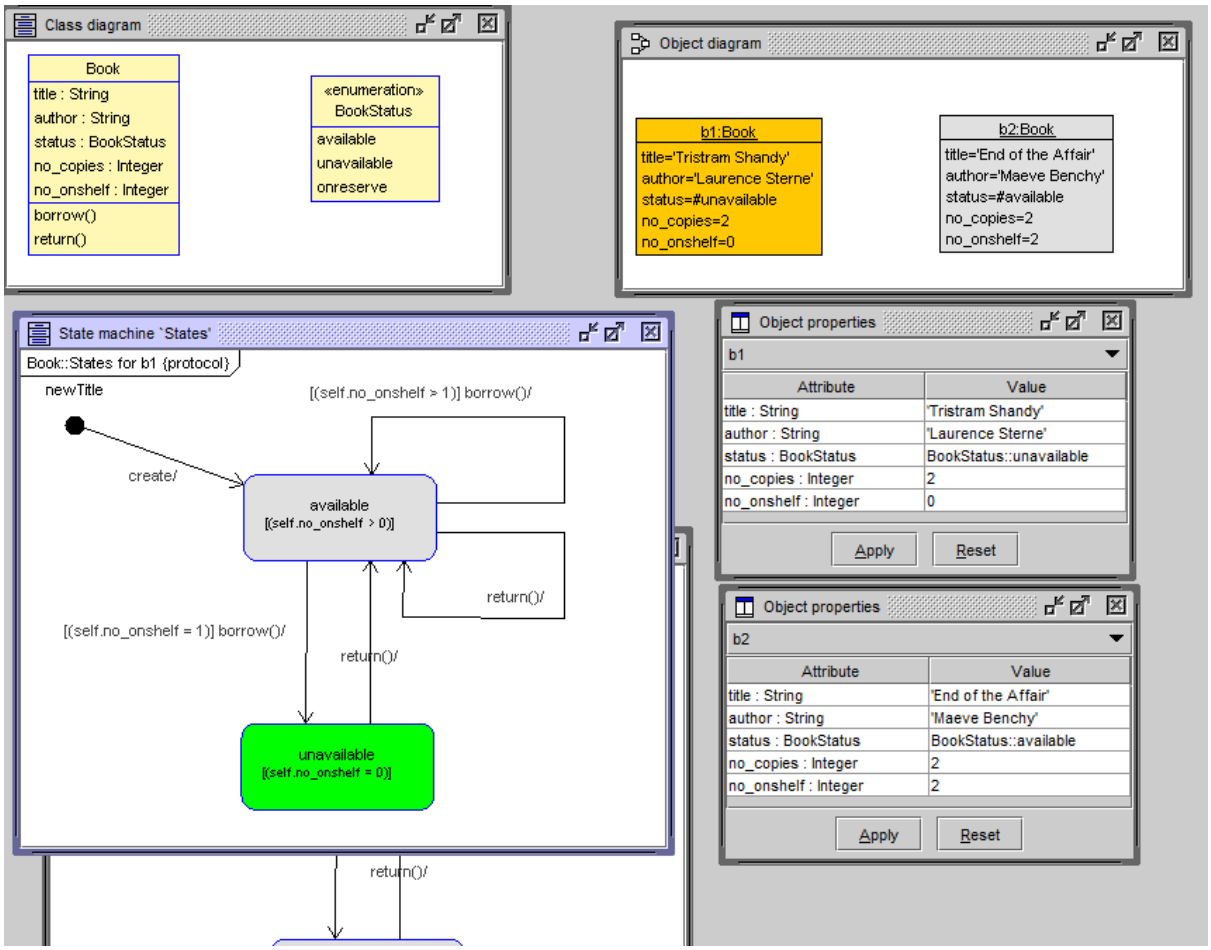
Note that each book is listed by default to have 2 copies on the shelf when created.

Next select and right-click on the **b1** object, and open a state machine for it.



Then using the command prompt, enter the SOIL command: `!b1.borrow()`

The state machine for b1 should be unchanged. Try `!b1.borrow()` again. Things should then look like:



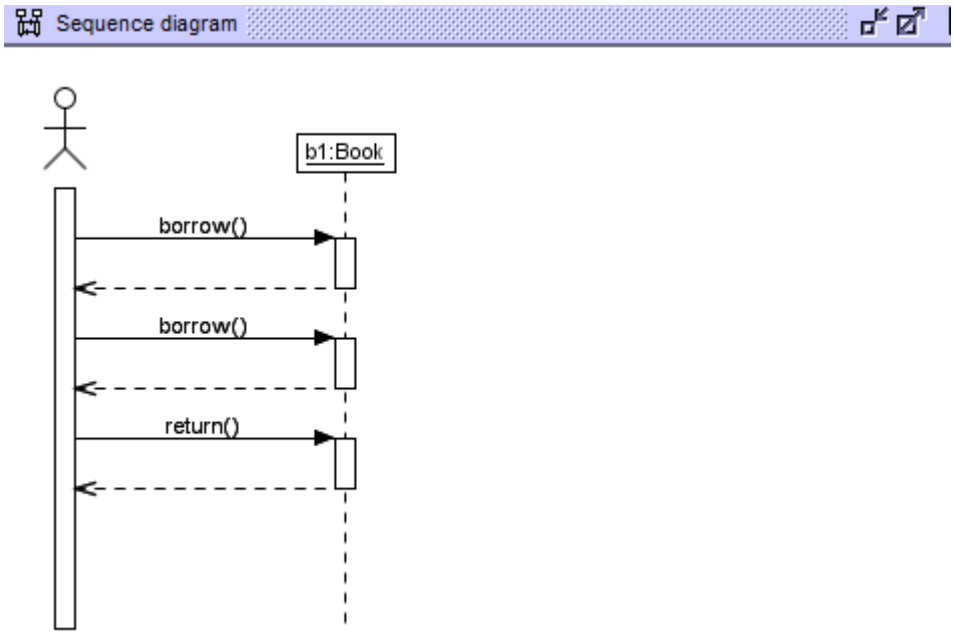
If you try to borrow a third time, you will get an error message like:

```
use> !b1.borrow()
use> !b1.borrow()
use> !b1.borrow()
Error: No valid transition available in protocol state machine `Book::States [se
lf: b1, current state: unavailable]' for operation call Book::borrow(self:b1).
use>
```

So the state machine won't allow this operation as there is no corresponding state transition. Normally an approriate precondition would pick this up.

Next try `!b1.return()` and see what happens to the state machine.

Then open a sequence diagram view.



Exercise

Return to your original Library model that includes the Copy class, and construct a state machine for Copy. Then test it.