4.3 A central computer connected to remote terminals via communication links is used to automate seat reservations for a concert hall. A booking clerk can display the current state of reservations on the terminal screen. To book a seat, a client chooses a free seat and the clerk enters the number of the chosen seat at the terminal and issues a ticket. A system is required which avoids double-booking of the same seat whilst allowing clients free choice of the available seats. Construct a model of the system and demonstrate that your model does not permit double-bookings. (Hint: It is only necessary to model a few terminals and a few seats. Remember, a seat can appear to be free although it is booked or being booked by another clerk).

```
/*
* ADAPTED FROM:
* Concurrency: State Models & Java Programs
*  Jeff Magee & Jeff Kramer
*  Solutions to Exercises
*
*  Exercise 4.3
*/

const False = 0
const True  = 1
range Bool =  False..True

set Terminals = {a,b}
const SeatNumber = 2

LOCK = (acquire -> release -> LOCK).

SEAT = SEAT[False],
SEAT[reserved:Bool]
     = ( when (!reserved) reserve -> SEAT[True]
       | query[reserved] -> SEAT[reserved]
       | when (reserved) reserve -> ERROR
       ).

range Seats = 0..SeatNumber-1
||SEATS = (seat[Seats]:SEAT).

TERMINAL = (choose[s:Seats] -> acquire -> BOOKING[s]),
BOOKING[s:Seats] = (
                seat[s].query[False] -> seat[s].reserve -> release -> ok -> TERMINAL
                |
                seat[s].query[True] -> release -> ko -> TERMINAL
                ).

||CONCERT = (Terminals:TERMINAL || Terminals::SEATS || Terminals::LOCK).
```