Project Report Part-2:

Project Title: Restaurant Management System

Course Title: Software Quality Assurance
Course Code: CSE435
Semester: SUMMER'21

SUBMITTED TO,

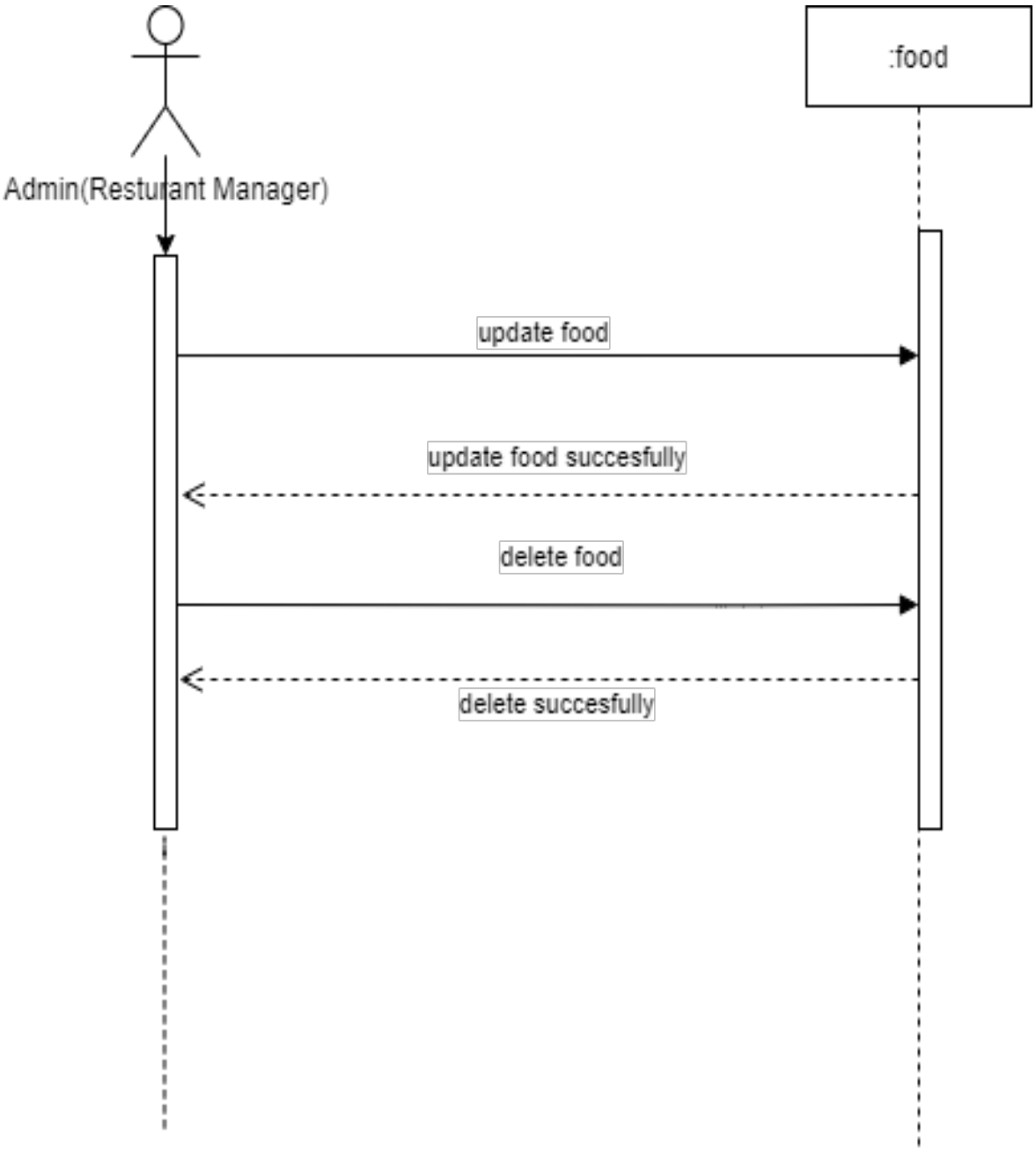
Dr. Shamim H Ripon,
Professor, Department of Computer Science & Engineering,
East West University, Dhaka, Bangladesh.

SUBMITTED BY:

- **Maria Mehjabin Shenjuti (2018-1-60-244)-Sec02**
- **Rafina Afreen (2018-1-60-119)-Sec03**

For Update & Delete:

Sequence Diagram:



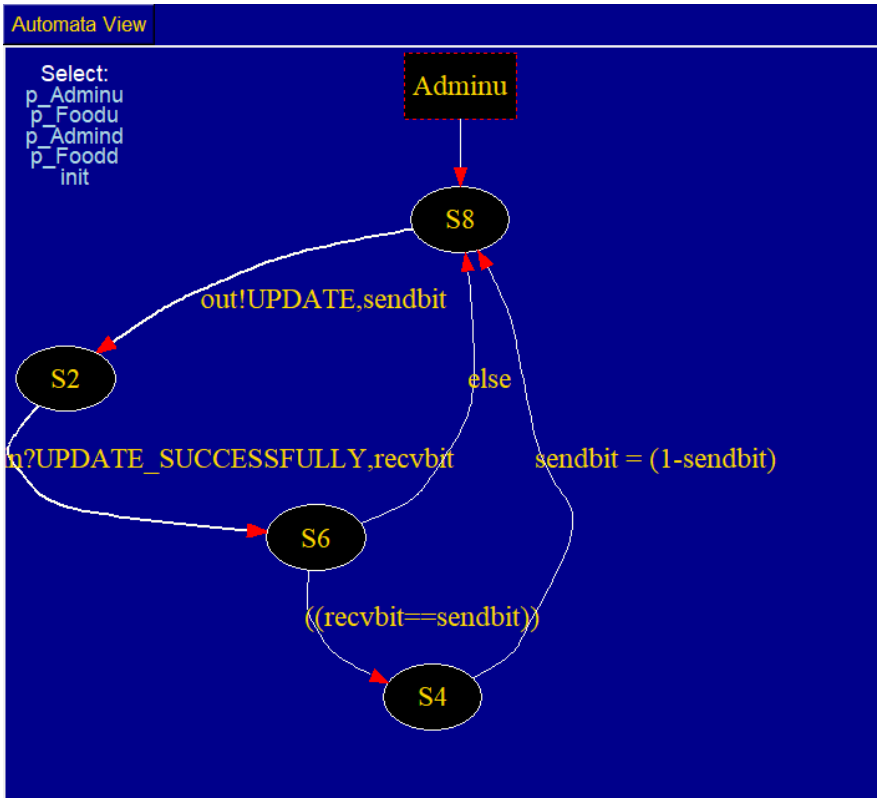
Promela Code:

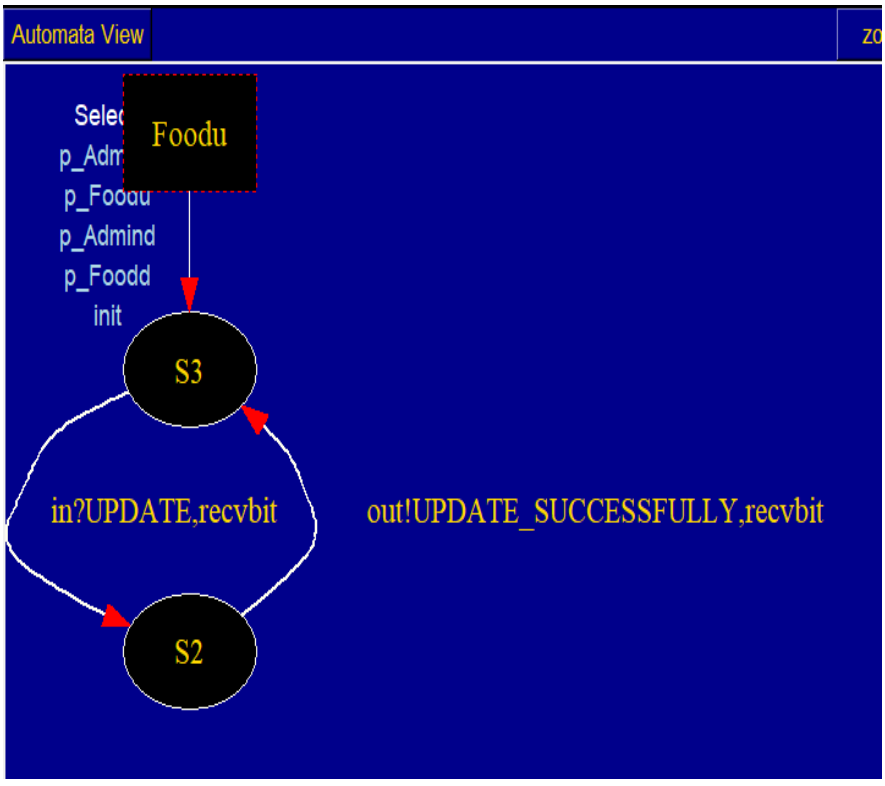
Edit/View | Simulate / Replay | Verification | Swarm Run | <Help> | Save Session | Restore Session | <Quit>

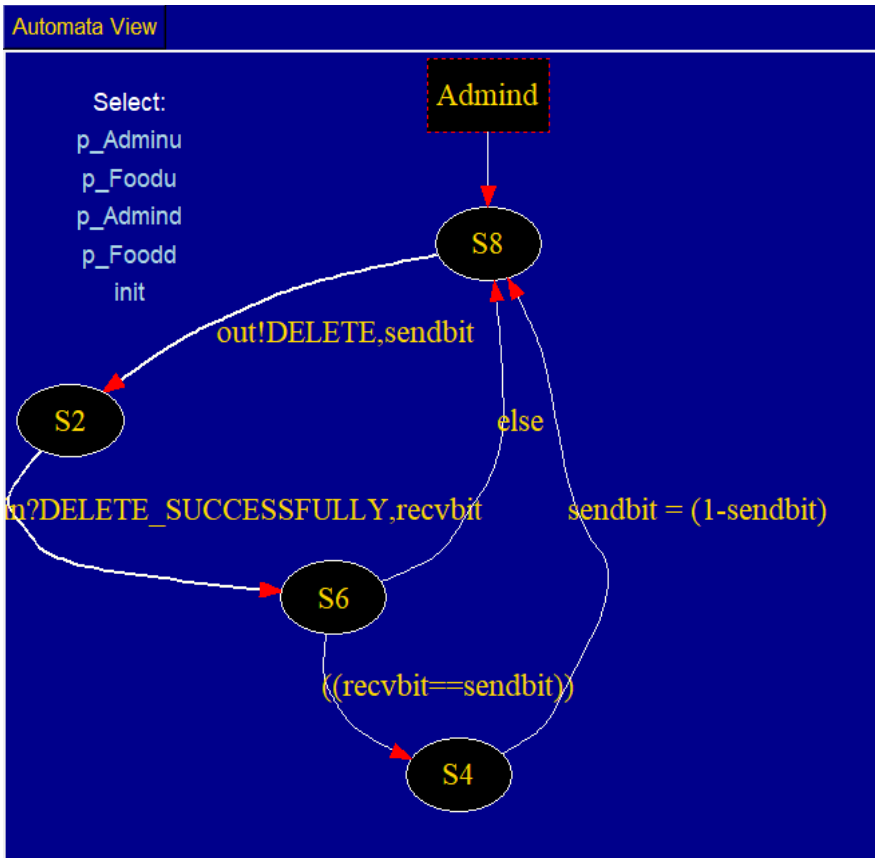
Open... | ReOpen | Save | Save As... | Syntax Check | Redundancy Check | Symbol Table | Find:

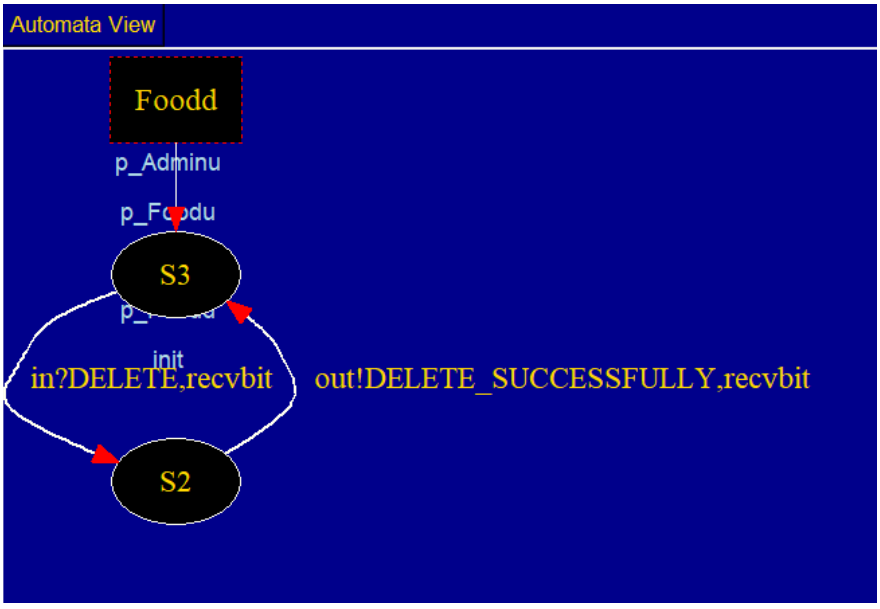
```
1   mtype {UPDATE, UPDATE_SUCCESSFULLY}
2   mtype {DELETE, DELETE_SUCCESSFULLY}
3   chan toAu = [2] of {mtype,bit};
4   chan toFu = [2] of {mtype,bit};
5   chan toAd = [2] of {mtype,bit};
6   chan toFd = [2] of {mtype,bit};
7   proctype Adminu(chan in, out)
8   {
9           bit sendbit,recvbit;
10          do
11          :: out !UPDATE, sendbit ->
12                  in ? UPDATE_SUCCESSFULLY,recvbit;
13                  if
14                  :: recvbit == sendbit ->
15                          sendbit = 1-sendbit
16                  :: else
17                  fi
18          od
19  }
20
21  proctype Foodu(chan in, out)
22  {
23          bit recvbit
24          do
25          :: in ? UPDATE(recvbit) ->
26                  out ! UPDATE_SUCCESSFULLY(recvbit);
27          od
28  }
29  proctype Admind(chan in, out)
30  {
31          bit sendbit,recvbit;
32          do
33          :: out !DELETE, sendbit ->
34                  in ? DELETE_SUCCESSFULLY,recvbit;
35                  if
36                  :: recvbit == sendbit ->
37                          sendbit = 1-sendbit
38                  :: else
39                  fi
40          od
41  }
42
43  proctype Foodd(chan in, out)
44  {
45          bit recvbit
46          do
47          :: in ? DELETE(recvbit) ->
48                  out ! DELETE_SUCCESSFULLY(recvbit);
49          od
50  }
51  init
52  {
53          run Adminu(toAu, toFu);
54          run Foodu(toFu, toAu);
55          run Admind(toAd, toFd);
56          run Foodd(toFd, toAd);
57  }
```

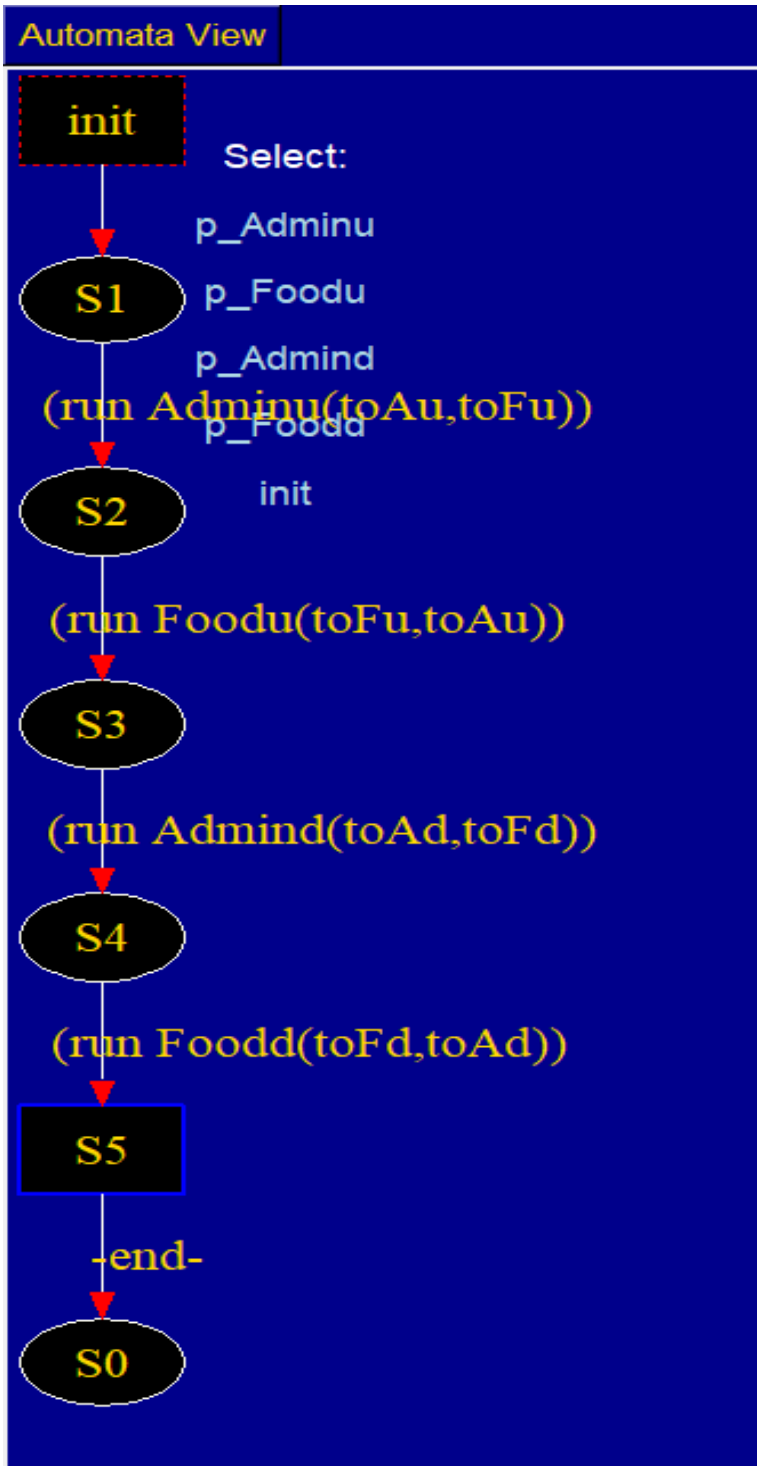
Automata of each process:



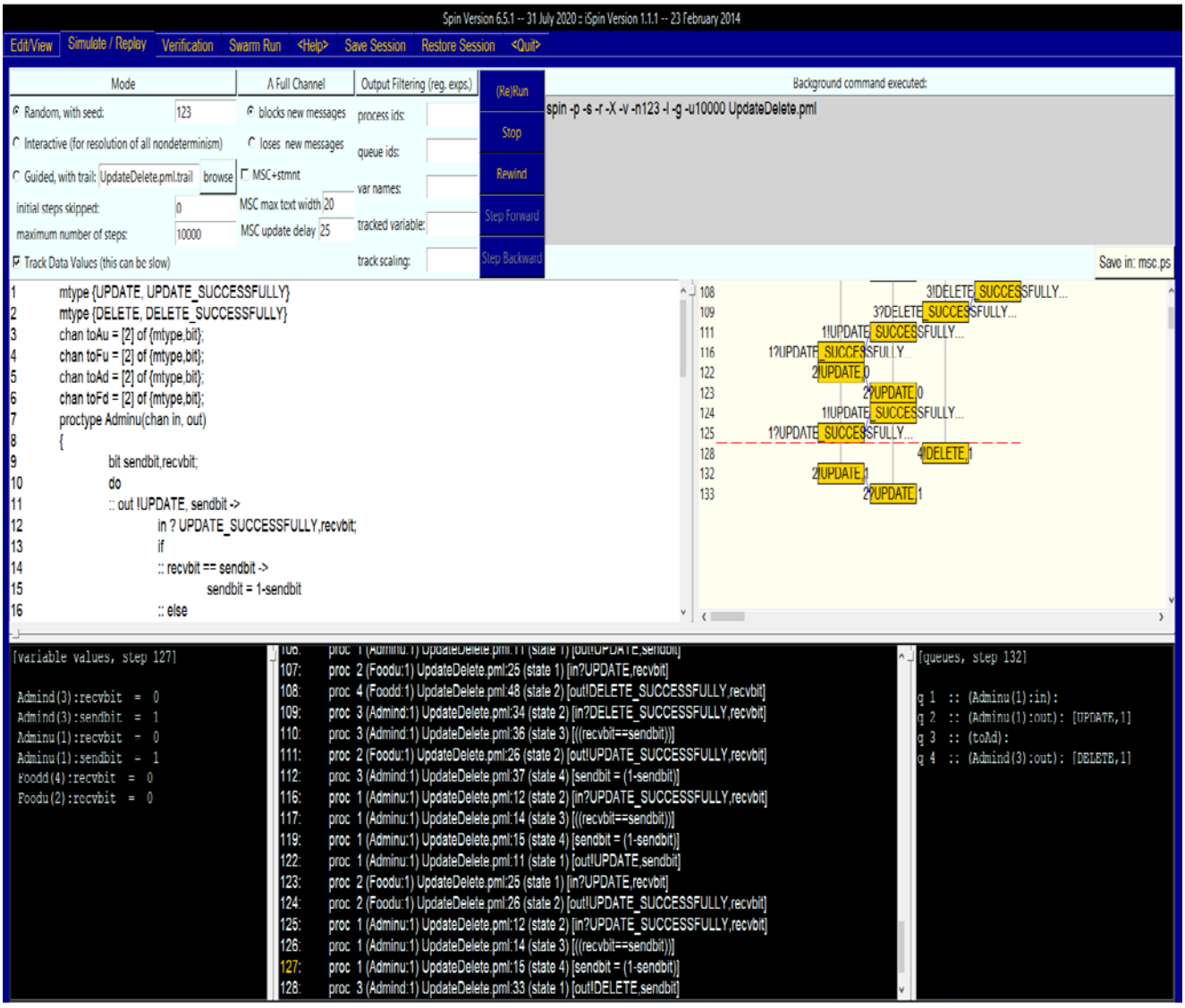



Automata View
Admind
Select:
p_Adminu
p_Foodu
p_Admind
p_Foodd
init
S8
out!DELETE,sendbit
S2
else
in?DELETE_SUCCESSFULLY,recvbit
sendbit = (1-sendbit)
S6
((recvbit==sendbit))
S4


Automata View
Foodd
p_Adminu
S3
init
in?DELETE,recvbit
out!DELETE_SUCCESSFULLY,recvbit
S2

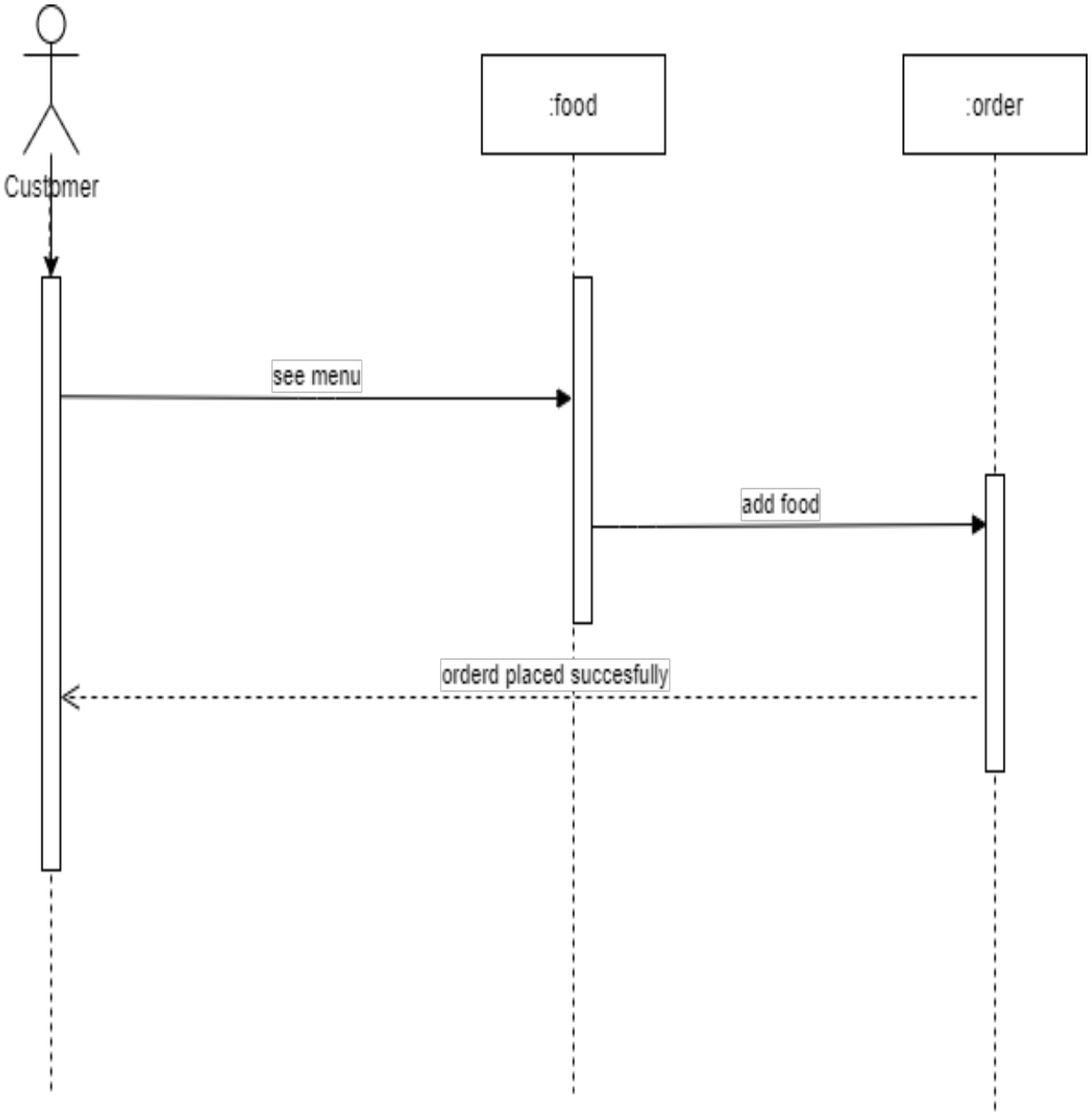

Automata View
init
Select:
p_Adminu
S1
p_Foodu
p_Admind
(run Adminu(toAu,toFu))
p_Foodd
S2
init
(run Foodu(toFu,toAu))
S3
(run Admind(toAd,toFd))
S4
(run Foodd(toFd,toAd))
S5
-end-
S0

Simulation run:



For Order Place:

Sequence Diagram:



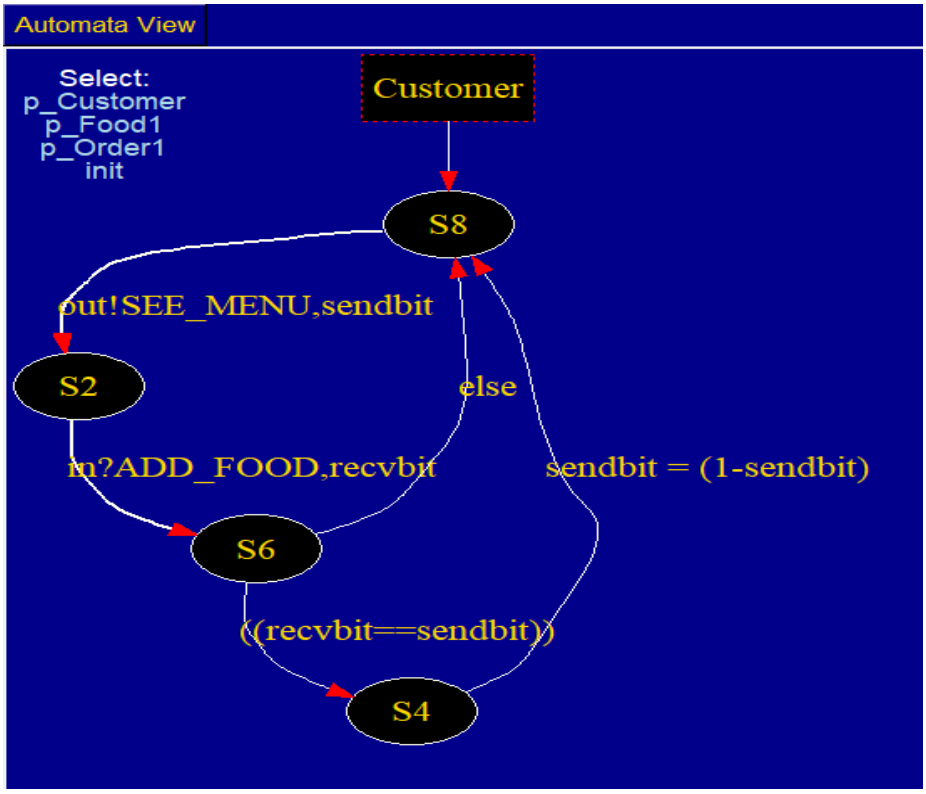
Promela Code:

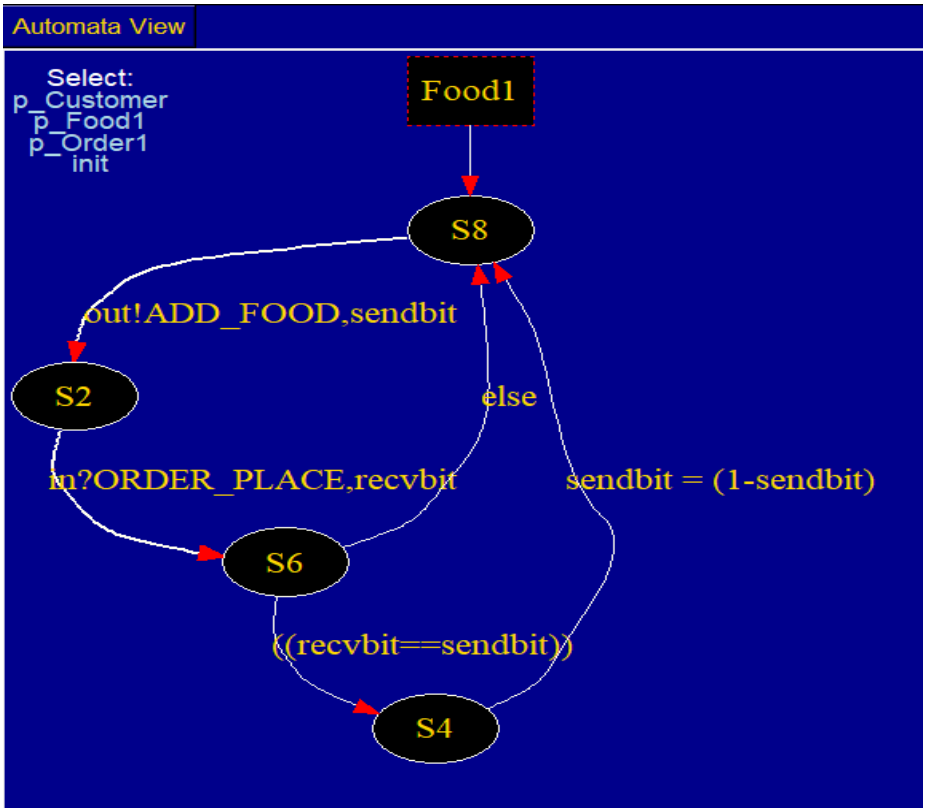
Edit/View | Simulate / Replay | Verification | Swarm Run | <Help> | Save Session | Restore Session | <Quit>

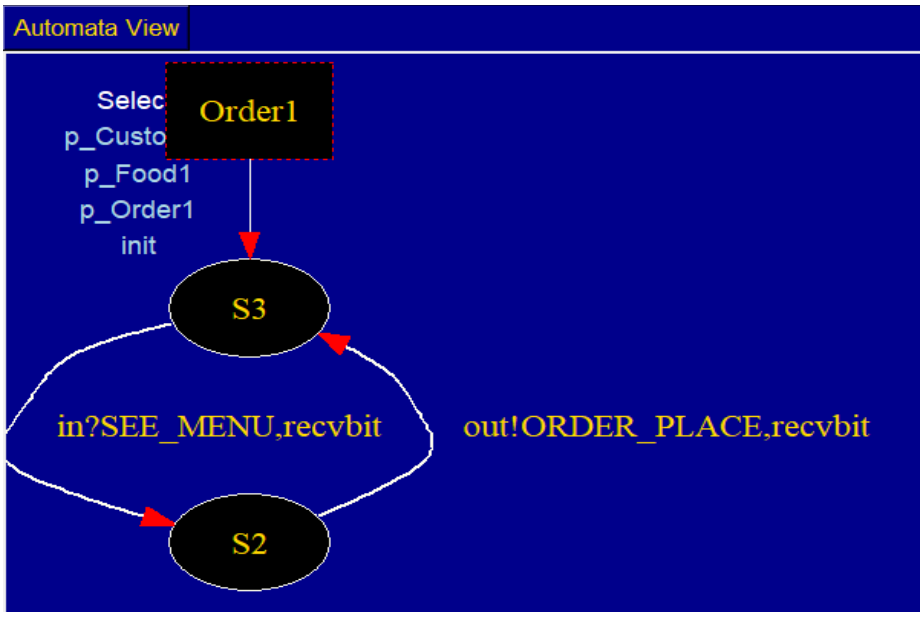
Open... | ReOpen | Save | Save As... | Syntax Check | Redundancy Check | Symbol Table | Find:

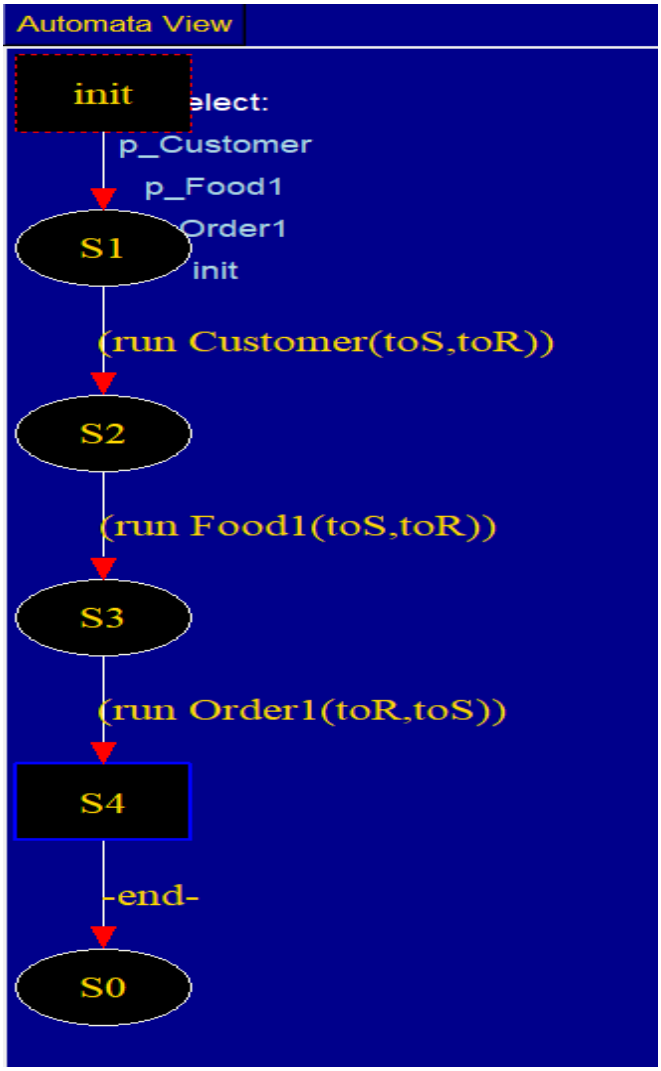
```
1   mtype {SEE_MENU,ADD_FOOD,ORDER_PLACE}
2
3
4   chan toS = [3] of {mtype,bit,bit};
5   chan toR = [3] of {mtype,bit,bit};
6
7   proctype Customer(chan in, out)
8   {
9           bit sendbit,recvbit;
10          do
11          :: out !SEE_MENU, sendbit ->
12                  in ? ADD_FOOD,recvbit;
13                  if
14                  :: recvbit == sendbit ->
15                          sendbit = 1-sendbit
16                  :: else
17                  fi
18          od
19  }
20  proctype Food1(chan in, out)
21  {
22          bit sendbit,recvbit;
23          do
24          :: out !ADD_FOOD, sendbit ->
25                  in ? ORDER_PLACE,recvbit;
26                  if
27                  :: recvbit == sendbit ->
28                          sendbit = 1-sendbit
29                  :: else
30                  fi
31          od
32  }
33
34  proctype Order1(chan in, out)
35  {
36          bit recvbit
37          do
38          :: in ? SEE_MENU(recvbit) ->
39                  out ! ORDER_PLACE(recvbit);
40          od
41  }
42  init
43  {
44          run Customer(toS, toR);
45          run Food1(toS, toR);
46          run Order1(toR, toS);
47  }
```

Automata of each process:

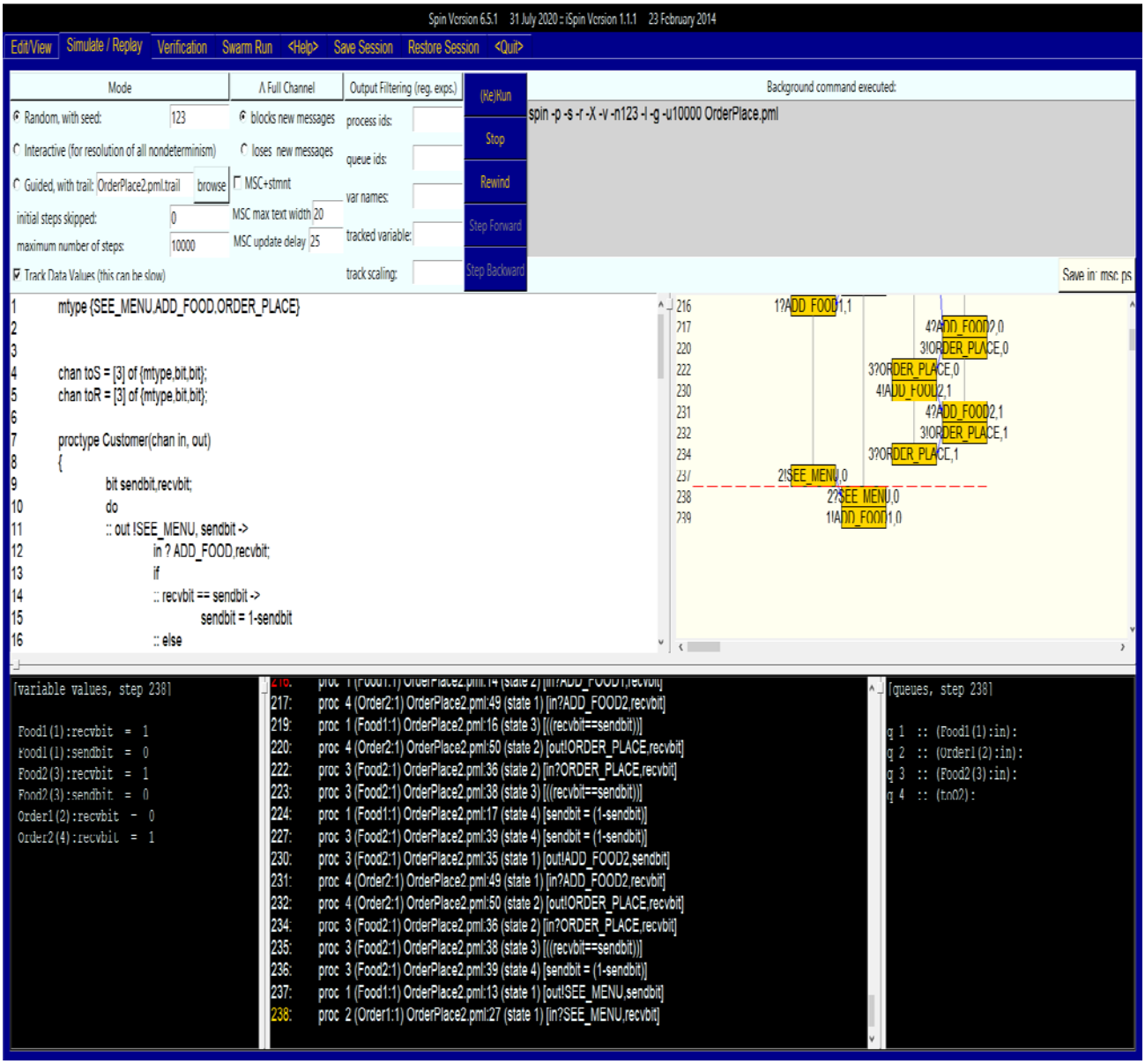





Automata View
Selec
p_Custo
p_Food1
p_Order1
init
Order1
S3
in?SEE_MENU,recvbit
out!ORDER_PLACE,recvbit
S2

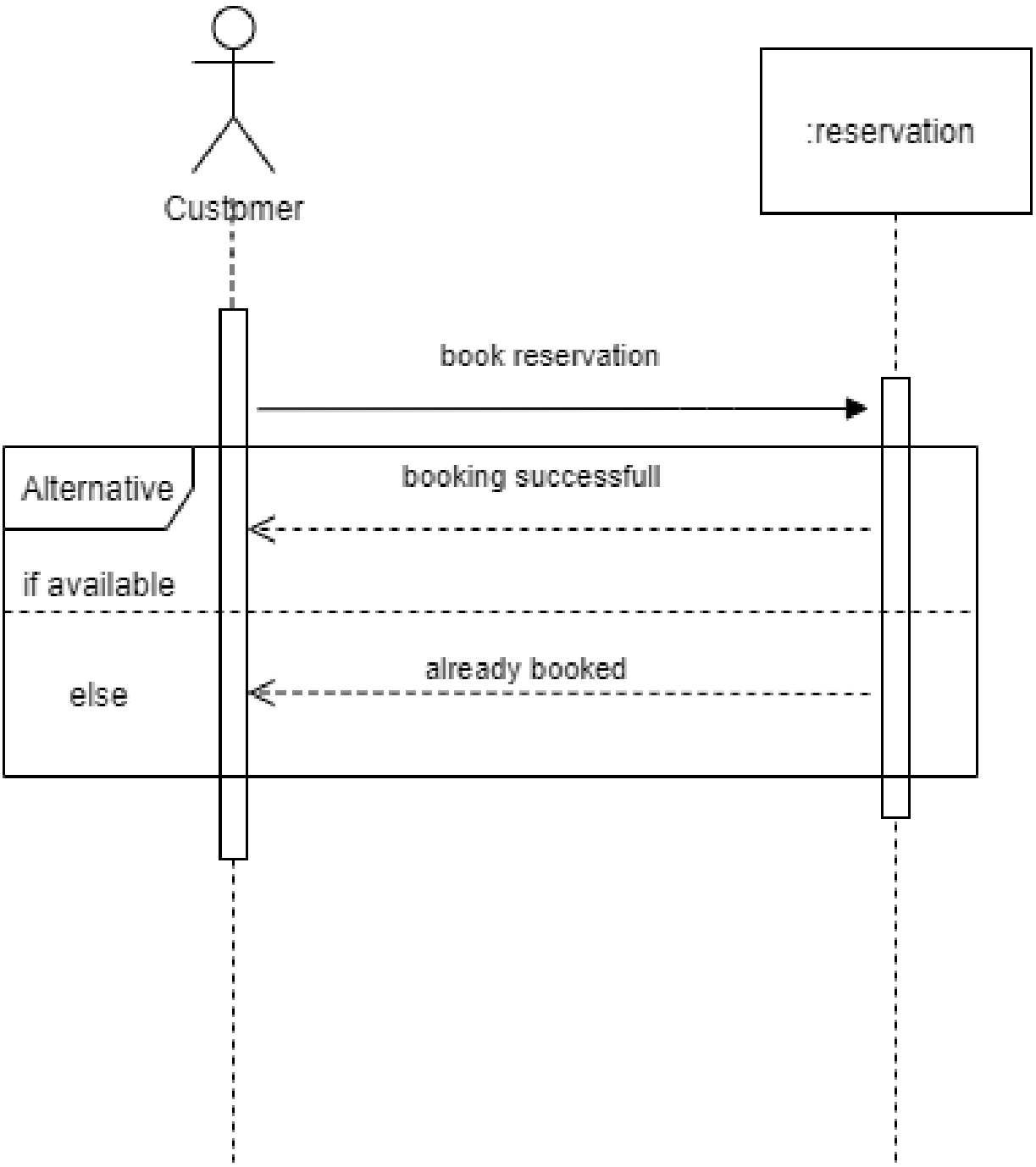

Automata View
init
elect:
p_Customer
p_Food1
Order1
init
S1
(run Customer(toS,toR))
S2
(run Food1(toS,toR))
S3
(run Order1(toR,toS))
S4
-end-
S0

Simulation run:



For Reservation:

Sequence Diagram:



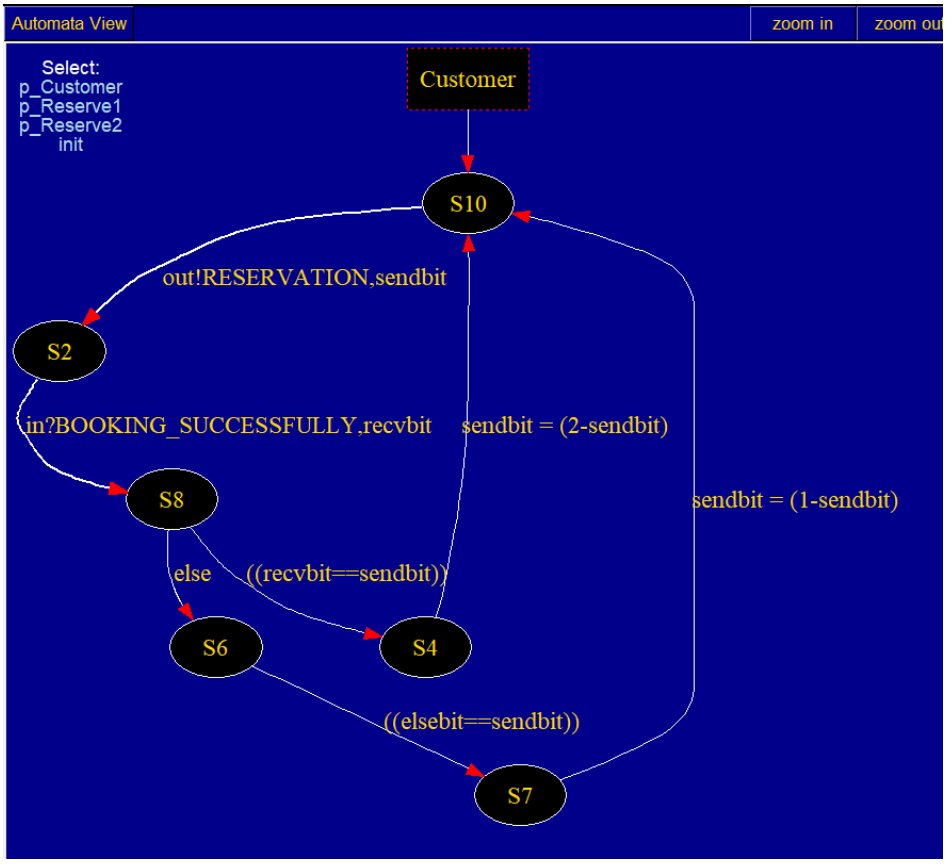
Promela Code:

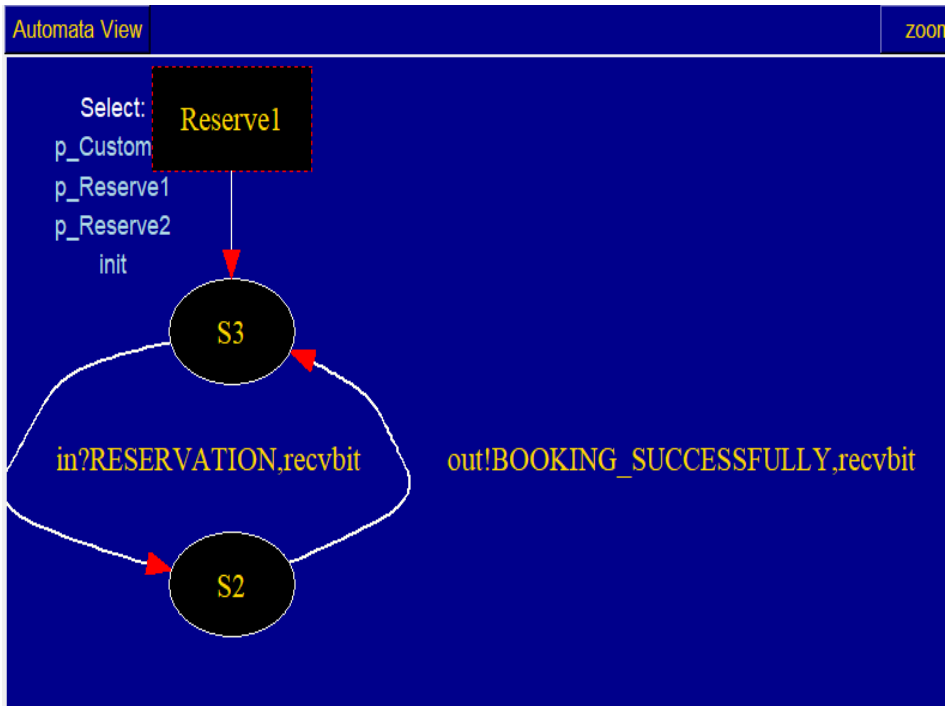
Edit/View | Simulate / Replay | Verification | Swarm Run | <Help> | Save Session | Restore Session | <Quit>

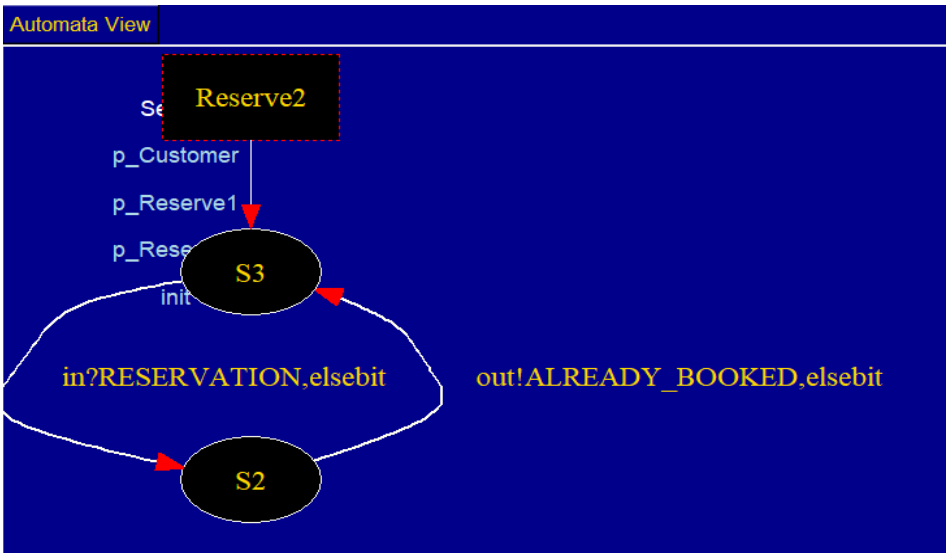
Open... | ReOpen | Save | Save As... | Syntax Check | Redundancy Check | Symbol Table | Find:

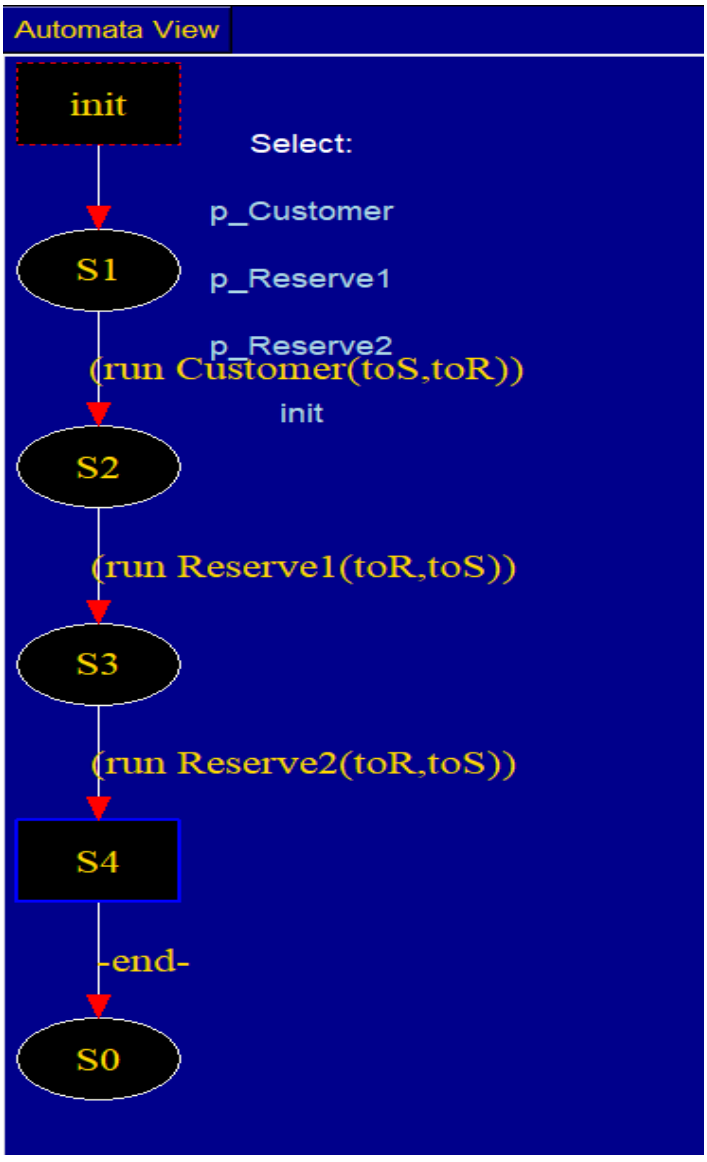
```
1   mtype {RESERVATION, BOOKING_SUCCESSFULLY, ALREADY_BOOKED}
2   chan toS = [3] of {mtype,bit,bit};
3   chan toR = [3] of {mtype,bit,bit};
4
5   proctype Customer(chan in, out)
6   {
7           bit sendbit,recvbit,elsebit;
8           do
9           :: out !RESERVATION, sendbit ->
10                  in ? BOOKING_SUCCESSFULLY,recvbit;
11                  if
12                  :: recvbit == sendbit ->
13                          sendbit = 2-sendbit
14                  :: else -> elsebit == sendbit ->
15                          sendbit = 1-sendbit
16                  fi
17          od
18  }
19  proctype Reserve1(chan in, out)
20  {
21          bit recvbit
22          do
23          :: in ? RESERVATION(recvbit) ->
24                  out ! BOOKING_SUCCESSFULLY(recvbit);
25          od
26  }
27  proctype Reserve2(chan in, out)
28  {
29          bit elsebit
30          do
31          :: in ? RESERVATION(elsebit) ->
32                  out ! ALREADY_BOOKED(elsebit);
33          od
34  }
35  init
36  {
37          run Customer(toS, toR);
38          run Reserve1(toR, toS);
39          run Reserve2(toR, toS);
40  }
```

Automata of each process:






Automata View
Reserve2
p_Customer
p_Reserve1
S3
init
in?RESERVATION,elsebit
out!ALREADY_BOOKED,elsebit
S2


Automata View
init
Select:
p_Customer
S1
p_Reserve1
p_Reserve2
(run Customer(toS,toR))
init
S2
(run Reserve1(toR,toS))
S3
(run Reserve2(toR,toS))
S4
-end-
S0

Simulation run: