FSP Process Composition Analysis & Design Form

1. FSP Composition Process Attributes

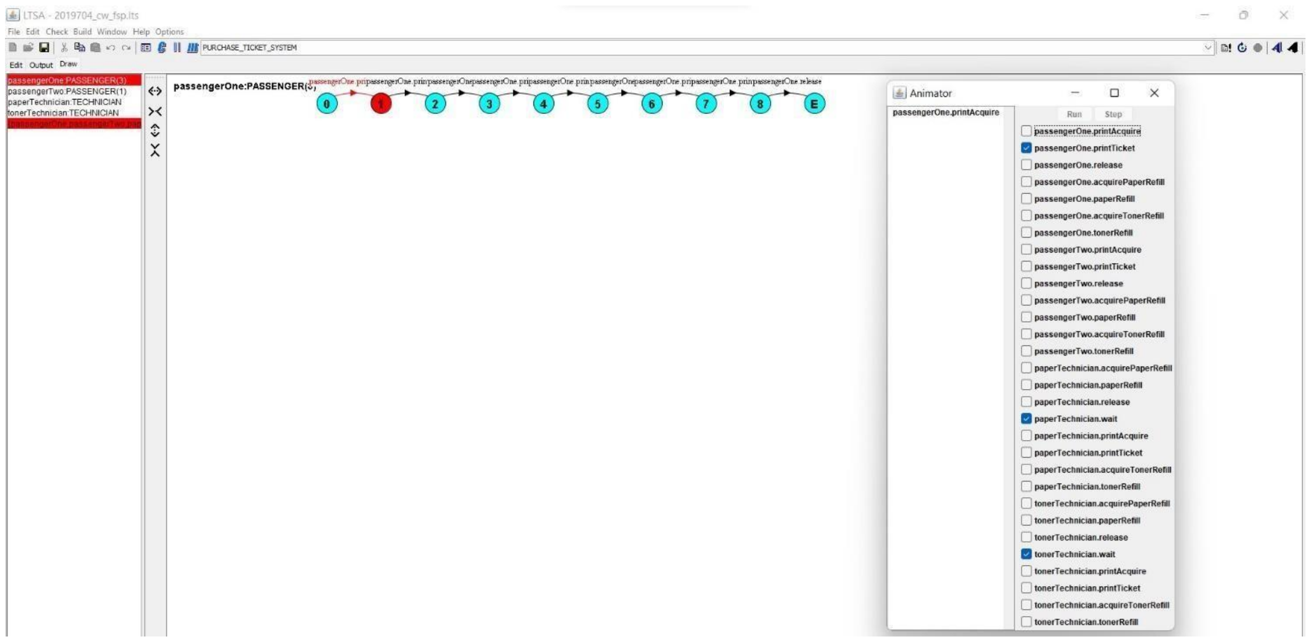
Attribute	Value
Name	PURCHASE_TICKET_SYSTEM
Description	Represents a ticket purchase system with passengers, ticket machines, and technicians.
Alphabet (Use LTSA's compressed notation if alphabet is large.)	{printAcquire, printTicket, acquirePaperRefill, paperRefill, acquireTonerRefill, tonerRefill, release, wait, printTicket[1..3]}
Sub-processes (List them.)	PASSENGER, PASSENGER, TICKETMACHINE
Deadlocks (yes/no)	No Deadlocks /Errors (Screenshots provided)
Deadlock Trace(s) **(If applicable)**	N/A

2. FSP "main" Program Code

FSP Program:

```
const PAPERS = 3     // Define a constant representing the initial number of papers.
const TONERS = 3     // Define a constant representing the initial number of toners.

// Define a set of actions or events that can be performed (operations).
set THINGS = {printAcquire, printTicket, acquirePaperRefill, paperRefill, acquireTonerRefill, tonerRefill,
release}

// Define a process for a ticket machine with a specified number of papers.
TICKETMACHINE (PAPER = PAPERS) = TICKETMACHINE[PAPER],
TICKETMACHINE[p : 0..PAPER] =
    if (p > 0)
    then (printAcquire -> printTicket -> release -> TICKETMACHINE[p-1])
    else (acquirePaperRefill -> paperRefill -> release -> TICKETMACHINE[PAPERS]).

// Define a process for a passenger with a specified number of tickets.
PASSENGER(TICKET_COUNT = 1) = PASSENGER[TICKET_COUNT],
PASSENGER[t : 1..TICKET_COUNT] = (
    printAcquire -> printTicket[t] ->
    if (t > 1)
    then (release -> PASSENGER[t-1])
    else (release -> END)
) + THINGS / {printTicket/printTicket[t:1..TICKET_COUNT]}.

// Define a process for a technician that involves acquiring, refilling, and releasing resources.
TECHNICIAN = (acquirePaperRefill -> paperRefill -> release -> TECHNICIAN | wait -> TECHNICIAN) +
THINGS.

// Define a composed system representing the entire purchase ticket system.
||PURCHASE_TICKET_SYSTEM =
    (passengerOne: PASSENGER(3)
    || passengerTwo: PASSENGER(1)
    || paperTechnician: TECHNICIAN
    || tonerTechnician: TECHNICIAN
    || {passengerOne, passengerTwo, paperTechnician, tonerTechnician} :: TICKETMACHINE).
```

3. Output Results


LTSA - 2019704_cw_fsp.lts

PURCHASE_TICKET_SYSTEM

```
passengerOne:PASSENGER(3)
passengerTwo:PASSENGER(1)
paperTechnician:TECHNICIAN
tonerTechnician:TECHNICIAN
{passengerOne,passengerTwo,pap
```

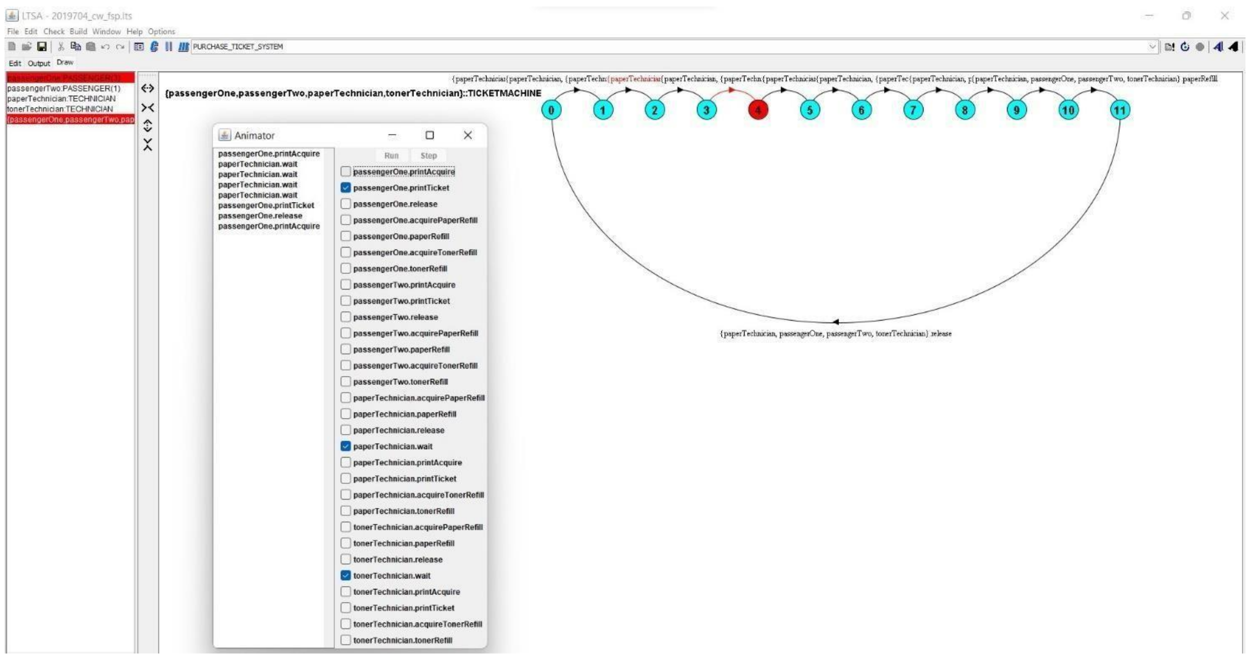
passengerOne:PASSENGER(3)

passengerOne.pri passengerOne.prin passengerOne passengerOne.pri passengerOne.prin passengerOne passengerOne.pri passengerOne.prin passengerOne.release
0
1
2
3
4
5
6
7
8
E

Animator

```
passengerOne.printAcquire
```

```
[ ] passengerOne.printAcquire
[✓] passengerOne.printTicket
[ ] passengerOne.release
[ ] passengerOne.acquirePaperRefill
[ ] passengerOne.paperRefill
[ ] passengerOne.acquireTonerRefill
[ ] passengerOne.tonerRefill
[ ] passengerTwo.printAcquire
[ ] passengerTwo.printTicket
[ ] passengerTwo.release
[ ] passengerTwo.acquirePaperRefill
[ ] passengerTwo.paperRefill
[ ] passengerTwo.acquireTonerRefill
[ ] passengerTwo.tonerRefill
[ ] paperTechnician.acquirePaperRefill
[ ] paperTechnician.paperRefill
[ ] paperTechnician.release
[✓] paperTechnician.wait
[ ] paperTechnician.printAcquire
[ ] paperTechnician.printTicket
[ ] paperTechnician.acquireTonerRefill
[ ] paperTechnician.tonerRefill
[ ] tonerTechnician.acquirePaperRefill
[ ] tonerTechnician.paperRefill
[ ] tonerTechnician.release
[✓] tonerTechnician.wait
[ ] tonerTechnician.printAcquire
[ ] tonerTechnician.printTicket
[ ] tonerTechnician.acquireTonerRefill
[ ] tonerTechnician.tonerRefill
```


LTSA - 2019704_cw_fsp.lts

PURCHASE_TICKET_SYSTEM

```
passengerOne:PASSENGER(3)
passengerTwo:PASSENGER(1)
paperTechnician:TECHNICIAN
tonerTechnician:TECHNICIAN
{passengerOne,passengerTwo,pap
```

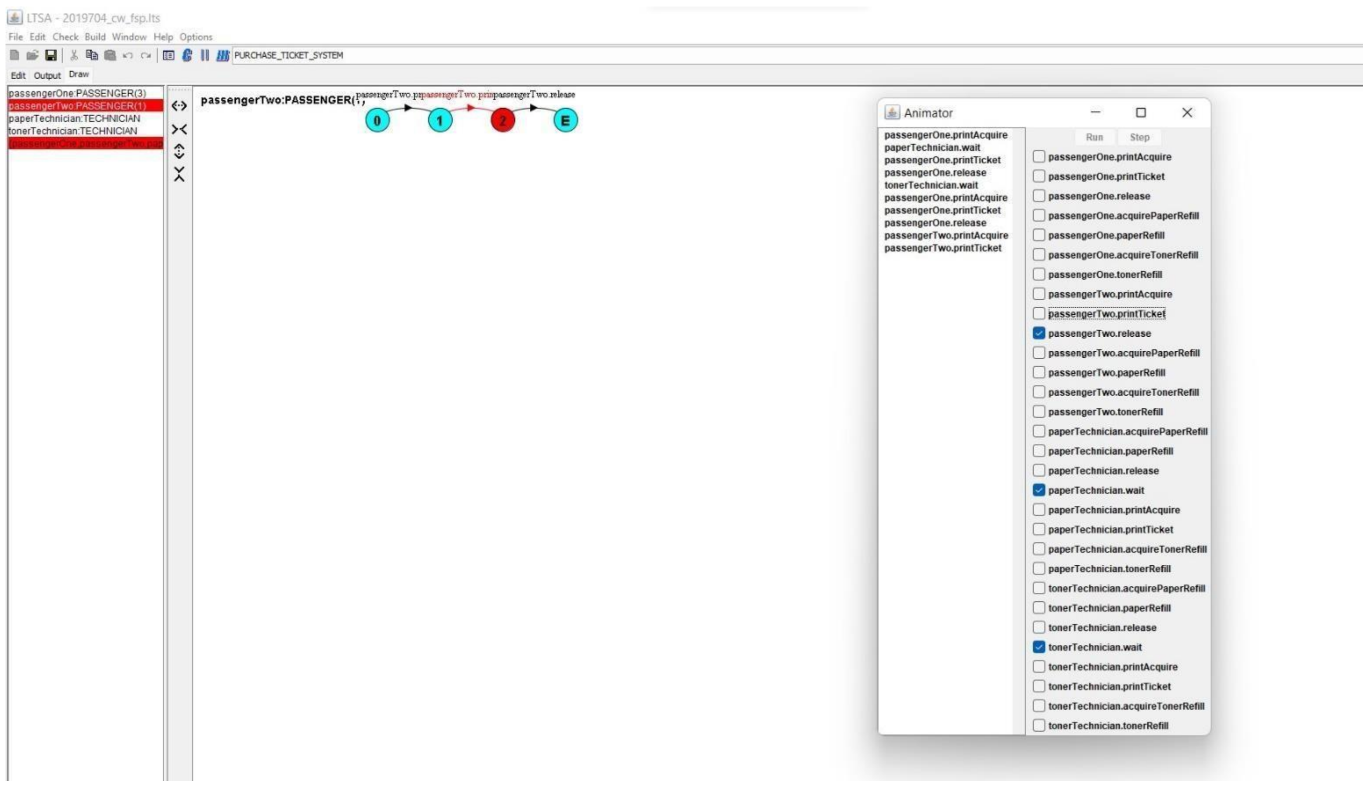
{passengerOne,passengerTwo,paperTechnician,tonerTechnician}::TICKETMACHINE

{paperTechnicia{paperTechnician, {paperTechn{paperTechnicia{paperTechnician, {paperTechn{paperTechnicia{paperTechnician, {paperTec{paperTechnician, p{paperTechnician, passengerOne, passengerTwo, tonerTechnician}.paperRefill
0
1
2
3
4
5
6
7
8
9
10
11
{paperTechnician, passengerOne, passengerTwo, tonerTechnician}.release

Animator

```
passengerOne.printAcquire
paperTechnician.wait
paperTechnician.wait
paperTechnician.wait
paperTechnician.wait
passengerOne.printTicket
passengerOne.release
passengerOne.printAcquire
```

```
[ ] passengerOne.printAcquire
[✓] passengerOne.printTicket
[ ] passengerOne.release
[ ] passengerOne.acquirePaperRefill
[ ] passengerOne.paperRefill
[ ] passengerOne.acquireTonerRefill
[ ] passengerOne.tonerRefill
[ ] passengerTwo.printAcquire
[ ] passengerTwo.printTicket
[ ] passengerTwo.release
[ ] passengerTwo.acquirePaperRefill
[ ] passengerTwo.paperRefill
[ ] passengerTwo.acquireTonerRefill
[ ] passengerTwo.tonerRefill
[ ] paperTechnician.acquirePaperRefill
[ ] paperTechnician.paperRefill
[ ] paperTechnician.release
[✓] paperTechnician.wait
[ ] paperTechnician.printAcquire
[ ] paperTechnician.printTicket
[ ] paperTechnician.acquireTonerRefill
[ ] paperTechnician.tonerRefill
[ ] tonerTechnician.acquirePaperRefill
[ ] tonerTechnician.paperRefill
[ ] tonerTechnician.release
[✓] tonerTechnician.wait
[ ] tonerTechnician.printAcquire
[ ] tonerTechnician.printTicket
[ ] tonerTechnician.acquireTonerRefill
[ ] tonerTechnician.tonerRefill
```


LTSA - 2019704_cw_fsp.lts
File Edit Check Build Window Help Options
PURCHASE_TICKET_SYSTEM
Edit Output Draw
Animator
Run Step
0
1
2
E

passengerOne:PASSENGER(3)
passengerTwo:PASSENGER(1)
paperTechnician:TECHNICIAN
tonerTechnician:TECHNICIAN
(passengerOne,passengerTwo,pap

passengerTwo:PASSENGER(1)
passengerTwo.pr passengerTwo.prin passengerTwo.release

```
passengerOne.printAcquire
paperTechnician.wait
passengerOne.printTicket
passengerOne.release
tonerTechnician.wait
passengerOne.printAcquire
passengerOne.printTicket
passengerOne.release
passengerTwo.printAcquire
passengerTwo.printTicket
```

- [ ] passengerOne.printAcquire
- [ ] passengerOne.printTicket
- [ ] passengerOne.release
- [ ] passengerOne.acquirePaperRefill
- [ ] passengerOne.paperRefill
- [ ] passengerOne.acquireTonerRefill
- [ ] passengerOne.tonerRefill
- [ ] passengerTwo.printAcquire
- [ ] passengerTwo.printTicket
- [✓] passengerTwo.release
- [ ] passengerTwo.acquirePaperRefill
- [ ] passengerTwo.paperRefill
- [ ] passengerTwo.acquireTonerRefill
- [ ] passengerTwo.tonerRefill
- [ ] paperTechnician.acquirePaperRefill
- [ ] paperTechnician.paperRefill
- [ ] paperTechnician.release
- [✓] paperTechnician.wait
- [ ] paperTechnician.printAcquire
- [ ] paperTechnician.printTicket
- [ ] paperTechnician.acquireTonerRefill
- [ ] paperTechnician.tonerRefill
- [ ] tonerTechnician.acquirePaperRefill
- [ ] tonerTechnician.paperRefill
- [ ] tonerTechnician.release
- [✓] tonerTechnician.wait
- [ ] tonerTechnician.printAcquire
- [ ] tonerTechnician.printTicket
- [ ] tonerTechnician.acquireTonerRefill
- [ ] tonerTechnician.tonerRefill

4. Analysis of Combined Process Actions

• **Synchronous** actions are performed by at least two sub-processes in combination, represented by the TICKETMACHINE process, which includes actions like printAcquire, printTicket, acquirePaperRefill, paperRefill, acquireTonerRefill, tonerRefill, and release.

• **Blocked Synchronous** actions cannot be performed, as at least one of the sub-processes cannot perform them due to constraints, such as attempting to print a ticket when there is no paper available ($p > 0$ condition in TICKETMACHINE).

• **Asynchronous** actions are performed independently by a single sub-process.
For example, the TECHNICIAN process involves acquiring, refilling, and releasing resources asynchronously.

Grouped actions include ticket printing for multiple passengers, e.g., printTicket[t:1..TICKET_COUNT], representing the indexed actions for printing tickets for different passengers.

Constants PAPERS and TONERS are defined to represent the initial number of papers and toners, respectively.

The entire ticket purchase system is composed of various processes, including multiple passengers (passengerOne and passengerTwo), paper technician (paperTechnician), and toner technician (tonerTechnician), all interacting with the TICKETMACHINE process to simulate the purchase ticket system.

Synchronous Actions	Synchronised by Sub-Processes (List)
printAcquire	printAcquire in PASSENGER and TECHNICIAN
printTicket	printTicket in PASSENGER
acquirePaperefill	acquirePaperRefill in TICKETMACHINE and TECHNICIAN
paperRefill	paperRefill in TICKETMACHINE and TECHNICIAN
release	release in PASSENGER, TICKETMACHINE, and TECHNICIAN