6SENG006W Concurrent Programming

FSP Process Composition Analysis & Design Form

Name	Fathima Rifa Rameez
Student ID	20200856/ W1836099
Date	11/01/2024

1. FSP Composition Process Attributes

Attribute	Value
Name	TICKET_SYSTEM
Description	
Sub-processes (List them.)	Alphabet: { a.acquireMachine, a.acquireRefillPaper, a.acquireRefillToner, a.print, a.refillPaper, a.refillToner, a.releaseMachine, a.releaseRefillPaper, a.releaseRefillToner, b.acquireMachine, b.acquireRefillPaper, b.acquireRefillToner, b.print, b.refillPaper, b.refillToner, b.releaseMachine, b.releaseRefillPaper, b.releaseRefillToner, c.acquireMachine, c.acquireRefillPaper, c.acquireRefillToner, c.print, c.refillPaper, c.refillToner,

	c.releaseMachine, c.releaseRefillPaper, c.releaseRefillToner, terminate }
Number of States	42
Deadlocks (yes/no)	no
Deadlock Trace(s) (If applicable)	

2. FSP "main" Program Code

The code for the parallel composition of all of the sub-processes and the definitions of any constants, ranges & process labelling sets used. (Do not include the code for the individual sub-processes.)

FSP Program:

```
const MAX_SHEETS = 3
range PAPER_RANGE = 0..MAX_SHEETS

const MAX_TICKETS = 3
range TONER_RANGE = 0..MAX_TICKETS

set ACTIONS = {  acquireMachine, print, releaseMachine, acquireRefillPaper, refillPaper, releaseRefillPaper,
               acquireRefillToner, refillToner, releaseRefillToner}

|| TICKET_SYSTEM = (a:PASSENGER(6) || b:TICKET_TECHNICIAN || c:TONER_TECHNICIAN || {a,b,c} :: MACHINE) / {terminate/ {a.terminate, b.terminate, c.terminate}}.
```

3. Combined Sub-processes

(Add rows as necessary.)

Process	Description
MACHINE	MACHINE represents printing ticket system which allows to print tickets, refill papers and refill toners

a :PASSENGER	PASSENGER is the who prints the tickets
b:TICKET_TECHNICIAN	TICKET_TECHNICIAN related to refilling papers
c:TONER_TECHNICIAN	TONER_TECHNICIAN related to refilling toners

4. Analysis of Combined Process Actions

- **Alphabets** of the combined processes, including the final process labelling.
- **Synchronous** actions are performed by at least two sub-process in the combination.
- **Blocked Synchronous** actions cannot be performed, because at least one of the sub-processes can never preform them, because they were added to their alphabet using alphabet extension.
- **Asynchronous** actions are preformed independently by a single sub-process.

Group actions together if appropriate, e.g. if they include indexes in[0], in[1], …, in[5] as in[1..5]. Add rows as necessary.

Processes	**Alphabet** (Use LTSA's **compressed notation**, if alphabet is large.)
MACHINE[p:PAPER_RANGE][t:TONER_RANGE]	{a, b, c}.{acquireMachine, acquireRefillPaper, acquireRefillToner, print, refillPaper, refillToner, releaseMachine, releaseRefillPaper, releaseRefillToner}
PASSENGER[i:0..COUNT]	{a.{acquireMachine, acquireRefillPaper, acquireRefillToner, print, refillPaper, refillToner, releaseMachine, releaseRefillPaper, releaseRefillToner}, terminate}
TICKET_TECHNICIAN	{b.{acquireMachine, acquireRefillPaper, acquireRefillToner, print, refillPaper, refillToner, releaseMachine, releaseRefillPaper, releaseRefillToner}, terminate}
TONER_TECHNICIAN	{c.{acquireMachine, acquireRefillPaper, acquireRefillToner, print, refillPaper, refillToner, releaseMachine, releaseRefillPaper, releaseRefillToner}, terminate}

Synchronous Actions	**Synchronised by Sub-Processes (List)**

Blocked Synchronous Actions	Blocking Processes	Blocked Processes

Sub-Processes	Asynchronous Actions (List)

5. Parallel Composition Structure Diagram

The structure diagram for the parallel composition.