# 6SENG002W Concurrent Programming

# FSP Process Composition Analysis & Design Form

| | |
|---|---|
| **Name** | Sarvetha Nadanamainthan |
| **Student ID** | 2018427 |
| **Date** | 06/01/2022 |

## 1. FSP Composition Process Attributes

| Attribute | Value |
|---|---|
| **Name** | BANKING_SYSTEM |
| **Description** | Models a student Bank Account, for a student, the student's grandmother, a loan company and a university. |
| **Alphabet** (Use LTSA's compressed notation, if alphabet is large.) | alphabet(BANKING_SYSTEM) = {grandMother.{calculateBalance[2..6], deposit[1..2], readBalance[4], sendECard, updateAccount[2..6]}, loanCompany.{calculateBalance[2..6], deposit[1..2], readBalance[4], updateAccount[2..6]}, student.{buySamsungPhone, calculateBalance[2..6], readBalance[4], updateAccount[2..6], withdraw[1..2]}, university.{calculateBalance[2..6], readBalance[4], updateAccount[2..6], withdraw[1..2]}} |
| **Sub-processes** (List them.) | STUDENT, GRANDMOTHER, LOANCOMPANY, UNIVERSITY, BANKACCOUNT |
| **Number of States** | 41 |
| **Deadlocks** (yes/no) | No deadlocks / errors |
| **Deadlock Trace(s) (If applicable)** | None |

## 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 other sub-processes.)

**FSP Program:**

```
/* ########## CONSTANTS ########## */
const MIN_TRANSACTION_VALUE = 1
const MAX_TRANSACTION_VALUE = 2

/* ########## RANGES ########## */
range INITIAL_BALANCE = 4..4
range FINAL_BALANCE = 2..6
range TRANSACTION_RANGE =
MIN_TRANSACTION_VALUE..MAX_TRANSACTION_VALUE

/* ########## SETS ########## */
set Users = { student, grandMother, loanCompany, university }

/* ########## BANKING SYSTEM PROCESS ########## */
|| BANKING_SYSTEM = ( student : STUDENT
                      || grandMother : GRANDMOTHER
                      || loanCompany : LOANCOMPANY
                      || university : UNIVERSITY
                      || Users  :: BANKACCOUNT).
```

## 3. Combined Sub-processes

(Add rows as necessary.)

| Process | Description |
|---|---|
| STUDENT | Represents a student who wants to read the current balance, make withdrawal, buy a phone and update the account balance. |
| GRANDMOTHER | Represents a grandmother who wants to read grandchild's current balance, deposit birthday money, update the account and send an ecard. |
| LOANCOMPANY | Represents a loan company who wants to read the current balance, adding the loan amount by updating the account. |
| UNIVERSITY | Represents a university which reads student's current balance, withdraws university fees by updating the account. |
| BANKACCOUNT | Represent a bank account which allows users to read current balance and updates it. |

## 4. Analysis of Combined Process Actions

- **Synchronous** actions are performed by at least two sub-process in the combination.
- **Blocked Synchronous** actions cannot be performed, since at least one of the sub-processes cannot 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, for example if they include indexes,
e.g. in[0], in[1], …, in[5] as in[1..5].

(Add rows as necessary.)

| Synchronous Actions | Synchronised by Sub-Processes (List) |
|---|---|
| readBalance | STUDENT, GRANDMOTHER, LOANCOMPANY, UNIVERSITY, BANKACCOUNT |
| calculateBalance | STUDENT, GRANDMOTHER, LOANCOMPANY, UNIVERSITY, BANKACCOUNT |
| updateAccount | STUDENT, GRANDMOTHER, LOANCOMPANY, UNIVERSITY, BANKACCOUNT |
| deposit | GRANDMOTHER, LOANCOMPANY |
| withdraw | STUDENT, UNIVERSITY |

| Sub-Process | Asynchronous Actions (List) |
|---|---|
| STUDENT | buySamsungPhone |
| GRANDMOTHER | sendECard |

## 5. Parallel Composition Structure Diagram

The structure diagram for the parallel composition.