Ingeniería del Software II
Simulacro de Parcial

Ejercicio 1. Una agencia de viajes ofrece paquetes turísticos que incluyen la reserva de una habitación de hotel, la compra de un pasaje de avión y el alquiler de un auto. Para lograrlo interactúa con web-services para la adquisición de cada servicio. Cada web-service provee su propio protocolo. Ante un pedido de un cliente la agencia debe interactuar con los servicios para comprarlo cuando es posible y reportar la falla cuando no (e.g. por falta de disponibilidad).

Modele usando FSP:

a. Un proceso Hotel, que permita consultar la disponibilidad de una habitación (evento *query* más *query.yes* y *query.no* para las respuestas). Luego de una respuesta positiva permite concretar la reserva (evento *reserve*) o cancelarla (evento *cancel*).

b. Un proceso Flight, que permita consultar la disponibilidad de un vuelo a un destino, que puede tener hasta dos opciones por una o dos escalas (evento *query* más eventos *query.yes* y *query.no* para las respuestas). Luego de una respuesta positiva permite seleccionar las escalas (eventos *select.1* y *select.2*). Y finalmente concretar la reserva (evento *reserve*) o cancelarla (evento *cancel*).

c. Un proceso Car, que en cualquier momento permita intentar alquilar un auto (evento *rent*). El intento puede ser exitoso o no (eventos *rent.ok* y *rent.fail*). En caso de éxito el servicio queda pago y sin posibilidad de reembolso.

d. Un proceso Agency, que ante un evento *request* interactúa con los tres servicios para armar un paquete turístico completo (incluyendo un servicio de cada tipo) cuando sea posible (*request.yes*). Además el paquete debe incluir el vuelo con la menor cantidad de escalas posibles. En caso de no ser posible por la falta de disponibilidad de alguno de los tres servicios, se debe reportar una falla (evento *request.no*). La agencia sólo puede hacer una consulta a cada web-service por *request* recibido (i.e., no debe reintentar ante una falla). Es importante que ante una falla no se pague por ningún servicio en el paquete incompleto. Es decir, no ocurran los eventos *reserve* o *rent*.

e. La composición paralela de todos los anteriores indicando claramente prefijos, renombres y priorización de eventos.

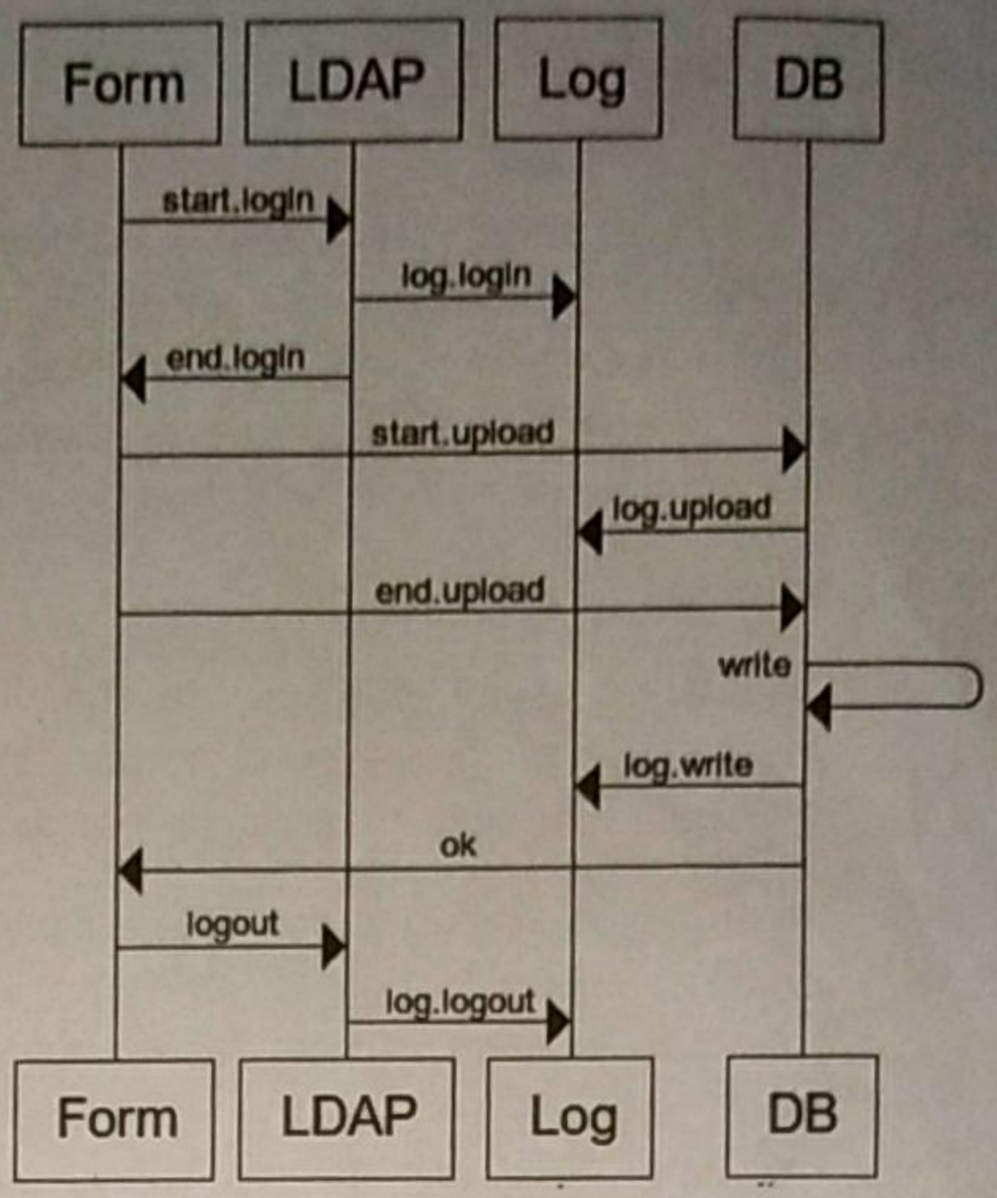
Ejercicio 2. Indique cuáles de los siguientes procesos descritos en FSP son bisimilares o débilmente bisimilares entre sí. Justifique mostrando una relación o un contraejemplo.

a.
```
S0 = (a -> S0 | b -> S1),
S1 = (c -> S1 | d -> S0).
```

b.
```
T0 = (a -> T1 | b -> T2),
T1 = (a -> T0 | b -> T2),
T2 = (c -> T3 | d -> T0),
T3 = (c -> T2 | d -> T0).
```

c.
```
U0 = (a -> U0 | b -> U1),
U1 = (e -> U2 | f -> U3),
U2 = (c -> U1),
U3 = (d -> U0) \ {e,f}.
```

Ejercicio 3. Dado el siguiente diagrama de secuencias que describe un comportamiento deseable de un Sistema:



Escriba:

a. Un proceso FSP que compuesto con el Sistema produzca un deadlock si el comportamiento descrito por el diagrama no es posible.
b. Una propiedad que detecte si las actividades de *login* y *upload* (comprendidas entre los eventos *start* y *end* respectivos) pueden ejecutarse concurrentemente o sin iniciarse.
c. Una propiedad que determine si el evento *start.upload* puede suceder infinita veces y si los eventos *log.upload* o *log.write* (cualquiera) pueden suceder infinitas veces.

Ejercicio 4. Dado el siguiente Kripke (descrito con un modelo NuSMV), que representa un single-player game al que se puede ganar o perder:

```
MODULE Game
VAR
        state : {out, play, win, lose};
ASSIGN
        init(state) := out;
        next(state) :=
                case
                        state = out : play;
                        state = play : {play, win, lose, out};
                        state = win : {play, out};
                        state = lose : {play, out};
                esac;
```

a. Transformar el Kripke a un autómata de Büchi.
b. Dar la fórmula LTL que represente la propiedad: "Eventualmente por siempre se gana".
c. Dada la fórmula del punto anterior, construya el autómata de Büchi correspondiente, necesario para una verificación.
d. Componer los dos Büchi e indicar si el modelo satisface la fórmula del punto b.