Evaluación Final – 27 de febrero de 2009 Nombre: .....

Ingeniería de Software II

Ejercicio 1.

1. Considere el alfabeto $\Sigma = \{a, b\}$. Determine si la propiedad $(aa + bb)^\omega$ es de *safety*, de *liveness* o ambas. Justifique la respuesta. Si no es de *liveness* extiendala agregando el mínimo conjunto de palabras necesaria de manera que lo sea. Use expresiones ω-regulares. Justifique formalmente que la extensión es de *liveness* y mínima.

2. Dé el autómata de Büchi que representa el lenguaje dado. Es determinista? Justifique.

3. Sean $L, S \subseteq \Sigma^\omega$ propiedades de *liveness* y *safety* respectivamente. Determine si es de

Ejercicio 2. Demuestre que *strong fairness* implica *weak fairness*.

Ejercicio 3. Dada una fórmula LTL ϕ, se dice que un modelo M implementa a otro modelo N bajo una condicón ϕ (notación $M \preceq_\phi N$) si todas las ejecuciones de M que satisfacen ϕ, también son ejecuciones de N. Describa un algoritmo que verifique $M \preceq_\phi N$. Justifique su respuesta. ¿Cual parte cree que sería la más costosa del algoritmo y por qué?

Ejercicio 4. Considere un sistema de telefonía común. En este sistema, el teléfono se modela como sigue:

```
const CANT_TELEFONOS = 4
range NROS = 0..CANT_TELEFONOS-1

TELEFONO = ( descolgar -> LLAMADA_SALIENTE
           | sonar_ring -> LLAMADA_ENTRANTE
           ),
LLAMADA_SALIENTE = ( discar[i:NROS] -> ( ocupado -> colgar -> TELEFONO
                                       | llamando -> ( desconectar -> colgar -> TELEFONO
                                                     | conectar_s -> colgar -> TELEFONO
                                                     )
                                       )
                   | colgar -> TELEFONO
                   ),
LLAMADA_ENTRANTE = ( detener_ring -> TELEFONO
                   | atender -> ( conectar_e -> colgar -> TELEFONO
                                )
                   ).

//
// Notar que que el evento [j].conectar[i] fuerza la sincronizacion
// entre los eventos [i].conectar_e y [j].conectar_s
//
||TELEFONOS = ( [NROS]:TELEFONO )/{[j:NROS].conectar[i:NROS]/{[i].conectar_e,[j].conectar_s}}.
```

Debemos aclarar que los eventos `descolgar` y `colgar` son propios de los teléfonos y por consiguiente no pueden utilizarse para sincronizar.

Notar que también hemos modelado el conjunto de teléfonos que pueden conectarse entre sí a través del evento `[j].conectar[i]` que conecta una llamada saliente del teléfono `j` (evento `[j].conectar_s`) con una llamada entrante en el teléfono `i` (evento `[i].conectar_e`).

1. Modele el controlador del switch `CTRL_SWITCH` que se encarga de establecer la llamada si es posible o de reportar ocupado si corresponde. Una vez establecida la llamada, el contorlador se desliga del resto de la conexión.

   Observaciones: (I) si le parece conveniente, puede modelar la posibilidad de que reporte ocupado aún cuando el telefono de destino no lo esté; (II) note que el usuario llamante tiene la posibilidad de `desconectarse` unilateralmente de la llamada en caso de que el destinatario no atienda.

   Modele también el sistema compuesto y asegúrese de que las componentes sincronicen apropiadamente y de que no haya deadlock.

2. Modele una propiedad de safety que exprese que si el teléfono 0 disca para comunicarse con el teléfono 1, entonces, o bien 1 atiende o bien 0 reporta ocupado. ¿Se cumple esta propiedad en su sistema?

Ejercicio 5. Considere el siguiente problema:

> Indiana Jones, acompañado de su novia, su padre y su suegro, necesita cruzar un puente colgante, un tanto peligroso, de 1 kilómetro de longitud. Está tan oscuro en el lugar, que es imposible cruzar el puente sin una linterna. Además, el puente es tan débil que sólo soporta como máximo a dos personas sobre el puente, y la luz de la linterna es tan débil que cuando dos personas caminan juntas sobre el puente, éstas se ven forzadas a hacerlo a la velocidad del más lento de ellos. Indiana Jones puede cruzar el puente en 5 minutos, su novia en 10, el padre de Indiana en 20 y el suegro en 25. Para que no los atrapen los villanos, deben poder cruzar el puente en una hora. Podrán lograrlo?

Modele el problema en Alloy, de manera tal que, mediante análisis (ya sean *runs* o *asserts* sobre el modelo) uno pueda buscar soluciones al problema.

Ejercicio 6.

(a) Utilizando el algoritmo dado en clase convierta a forma normal conjuntiva la siguiente fórmula:

$$\neg(Q \vee \neg R) \rightarrow (P \wedge \neg Q).$$

(b) Utilice el algoritmo DPLL para verificar si la siguiente fórmula es satisfactible:

$$(P \vee Q \vee R) \wedge (\neg P \vee \neg Q \vee \neg R) \wedge (\neg P \vee Q \vee R) \wedge (\neg Q \vee R) \wedge (Q \vee \neg R) \wedge (\neg P \vee R).$$

Construya el árbol completo. Asegurese de utilizar BCP al menos una vez a lo largo del cálculo.

Ejercicio 7. De acuerdo a la vista de componentes y conectores, ¿qué estilos arquitectónicos conoce? Dé ventajas, desventajas, y ejemplos clásicos para cada uno de ellos.