



Facultad de Matemática, Astronomía y Física
 Universidad Nacional de Córdoba
 Asignatura: Inginería de Software II
 Primer Cuatrimestre de 2006

Examen Parcial

1. Considere un museo con capacidad para N personas, y con E entradas y S salidas. Cada entrada y cada salida posee una puerta giratoria, cuya habilitación puede ser controlada por un sistema de software. Al abrirse el museo, se habilitan las puertas de entrada y salida. Las puertas de entrada permanecen habilitadas hasta que se alcanza la capacidad máxima del museo, por lo cual es necesario llevar cuenta del número de personas dentro del museo en cada momento. Por otra parte, si el museo se llena en algún momento, a medida que se retiren personas del museo deberán atenderse las puertas de entrada al museo en orden (de 1 a E), permitiendo la entrada de una persona por vez a través de cada una de las puertas en las cuales haya gente esperando. Esta debe ser la política de atención del museo hasta que el número de personas dentro del museo vuelva a V (siendo V un valor fijo menor que N). Asumimos que físicamente el museo admite más de N personas pero que por cuestiones de comodidad, regulatorias y de seguridad se prohíbe que haya más de N al mismo tiempo.

Al llegar el horario de cierre del museo, se cierran las puertas de entrada, pero se dejan habilitadas las puertas de salida del museo, hasta que éste se vacíe. Luego de esto se bloquean todas las puertas y se cierra el museo. El museo puede volver a abrirse sólo luego de que haya sido cerrado por completo.

Especifique el museo y el sistema de control de las puertas de entrada y salida en CSP, considerando las siguientes designaciones y complementando la descripción del conocimiento de dominio provista:

- Se traba la puerta $p \approx cerrar(p)$

Se destraba la puerta $p \approx abrir(p)$

Una persona entra por la puerta $p \approx entra(p)$

Una persona sale por la puerta $p \approx sale(p)$

Cantidad máxima de persona que admite el museo $\approx N$

Cantidad de puertas de entrada $\approx E$

Cantidad de puertas de salida $\approx S$

$PuertaE = \text{abrir} \rightarrow PEAbierta$

$PEAbierta = entra \rightarrow PEAbierta \sqcap cerrar \rightarrow PuertaE$

$PuertaS = \text{abrir} \rightarrow PSAbierta$

$PSAbierta = sale \rightarrow PSAbierta \sqcap cerrar \rightarrow PuertaS$

$DK_Entrada = (\parallel i \in 1..E \bullet i : PuertaE)$

$DK_Entrada = (\parallel i \in 1..S \bullet i : PuertaS)$

$DK = DK_Entrada \parallel DK_Salida$

$PulsarB(i) \triangleq \wedge i \in 2..N$
 $\wedge pisos' = [pisos \text{ EXCEPT } !(i) = [p \mapsto @.p, b \mapsto @.b + 1]]$
 $Next \triangleq \exists i \in Nat : PulsarS(i) \vee PulsarB(i)$
 $Spec \triangleq Init \wedge \square[Next]_{pisos}$

module DK_Botonera

extends DK_Pisos
VARIABLE botonera
 $TypeInv \triangleq botonera \in [1..N \rightarrow Nat]$
 $Init \triangleq botonera \in [i \in 1..N \mapsto 0]$
 $PulsarI(i) \triangleq \wedge i \in 1..N$
 $\wedge botonera' = [botonera \text{ EXCEPT } !(i) = @ + 1]$
 $Spec \triangleq Init \wedge \square[\exists i \in Nat : PulsarI(i)]_{botonera}$

module DK_Ascensor

extends DK_Pisos
VARIABLES status, piso
 $TypeInv \triangleq status \in \{"detenido", "subiendo", "bajando"\} \wedge piso \in 1..N$
 $Init \triangleq status = "detenido" \wedge piso = 1$
 $Subir \triangleq \wedge status = "detenido"$
 $\wedge piso < N$
 $\wedge status' = "subiendo"$
 $\wedge piso' = piso$
 $Bajar \triangleq \wedge status = "detenido"$
 $\wedge piso > 1$
 $\wedge status' = "bajando"$
 $\wedge piso' = piso$
 $Detener \triangleq \wedge status \in \{"subiendo", "bajando"\}$
 $\wedge status' = "detenido"$
 $\wedge piso' = piso$
 $PasaPorPisoS \triangleq \wedge status = "subiendo"$
 $\wedge piso' = piso + 1$
 $\wedge status' = status$
 $PasaPorPisoB \triangleq \wedge status = "bajando"$
 $\wedge piso' = piso - 1$
 $\wedge status' = status$
 $PasaPorPiso \triangleq PasaPorPisoS \vee PasaPorPisoB$
 $Next \triangleq Subir \vee Bajar \vee Detener \vee PasaPorPiso$
 $Spec \triangleq Init \vee \square[Next]_{status, piso}$

2. Considere un ascensor para un edificio de N pisos. Cada piso entre el 2 y el $N - 1$ tiene un par de botones de solicitud del ascensor, para dirigirse a un piso superior o inferior, respectivamente. El primer piso posee sólo un botón de solicitud del ascensor, para dirigirse a pisos superiores. El piso N , en cambio, sólo posee un botón, para dirigirse a pisos inferiores. En el interior del ascensor se cuenta con una botoneña, con botones para cada piso del edificio.

Para asegurar la atención de todos los pisos del edificio, el ascensor tiene ciclos de descenso y ascenso. Si el ascensor está inactivo y se lo solicita en un piso superior inicia un ciclo de ascenso. Durante el ciclo de ascenso sólo atiende requerimientos de pisos que se encuentren entre el piso en que se encuentra y los pisos superiores, y que hayan requerido el ascensor para ascender, con excepción del piso N , que también deberá atender si se encuentra en un ciclo de ascenso. El resto de los requerimientos del ascensor se postergan para su atención en un futuro ciclo de ascenso o descenso, según corresponda. Además, durante el ciclo de ascenso, sólo se atenderán pedidos desde el interior del ascensor a cualquier piso entre el piso en que se encuentra el ascensor y N , pero se ignorarán por completo los pedidos de dirigirse a pisos inferiores (estos requerimientos no se postergan para su futura atención, sino que se ignoran por completo). El funcionamiento del ascensor durante un ciclo de descenso es exactamente el dual del funcionamiento del ascensor en un ciclo de ascenso.

Especifique el sistema de control del ascensor en TLA+, considerando las siguientes designaciones y la descripción del conocimiento de dominio provista:

Se pulsa el botón de ascenso del piso $p \approx PulsarS(p)$

Se pulsa el botón de descenso del piso $p \approx PulsarB(p)$

Se pulsa el botón p dentro del ascensor $\approx PulsarI(p)$

Se le ordena al ascensor que ascienda $\approx Subir$

Se le ordena al ascensor que descienda $\approx Bajar$

Se le ordena al ascensor que se detenga $\approx Detener$

El ascensor cambia de piso debido a que está ascendiendo o descendiendo $\approx PasaPorPiso$

module *DK_Pisos*

```
extends Naturals
CONSTANT N
ASSUME N ∈ Nat ∧ N > 1
```

module *DK_BotonesPisos*

```
extends DK_Pisos
VARIABLE pisos
TypeInvA ≡ p ∈ [1..N → [p, b : Nat]]
InitA ≡ pisos ∈ [i ∈ 1..N ↦ [p ↪ 0, b ↪ 0]]
PulsarS(i) ≡ i ∈ 1..N - 1
          ∧ pisos' = [pisos EXCEPT !(i) = [p ↪ @.p + 1, b ↪ @.b]]
```