Ejercicios de Especificación Recursos Compartidos

Guillermo Román
guillermo.roman@upm.es

Concurrencia
GRADO EN INGENIERÍA INFORMÁTICA/ GRADO EN MATEMÁTICAS E INFORMÁTICA/
DOBLE GRADO EN ING. INFORMÁTICA Y ADE
Universidad Politécnica de Madrid
`http://babel.upm.es/teaching/concurrencia`

Abril 2020

Ejemplo: lectores/escritores

Con acceso explícito a los elementos del tipo

C-TAD GestorLE

OPERACIONES

ACCIÓN IL, FL, IE, FE:

SEMÁNTICA

DOMINIO:

TIPO: *GestorLE* = $(l : \mathbb{N} \times e : \mathbb{N})$

INICIAL: $\text{self}.l = 0 \land \text{self}.e = 0$

INVARIANTE: $(\text{self}.l > 0 \Rightarrow \text{self}.e = 0) \land (\text{self}.e > 0 \Rightarrow \text{self}.e = 1 \land \text{self}.l = 0)$

CPRE: $\text{self}.e = 0$

IL

POST: $\text{self}.e = \text{self}^{pre}.e \land \text{self}.l = \text{self}^{pre}.l + 1$

CPRE: Cierto

FL

POST: $\text{self} = \text{self}^{pre} \setminus \text{self}.l = \text{self}^{pre}.l - 1$

CPRE: $\text{self}.e = 0 \land \text{self}.l = 0$

IE

POST: $\text{self}.l = 0 \land \text{self}.e = 1$

CPRE: Cierto

FE

POST: $\text{self}.l = 0 \land \text{self}.e = 0$

POLITÉCNICA

Ejemplo: lectores/escritores

Usando *pattern-matching* para acceder al tipo

C-TAD GestorLE

OPERACIONES

ACCIÓN IL, FL, IE, FE:

SEMÁNTICA

DOMINIO:

TIPO: *GestorLE* = $(l : \mathbb{N} \times e : \mathbb{N})$

INICIAL: $\text{self} = (0, 0)$

INVARIANTE: $(\text{self}.l > 0 \Rightarrow \text{self}.e = 0) \wedge (\text{self}.e > 0 \Rightarrow \text{self}.e = 1 \wedge \text{self}.l = 0)$

CPRE: $\text{self}.e = 0$

IL

POST: $\text{self}^{pre} = (lec, esc) \wedge \text{self} = (lec + 1, esc)$

CPRE: Cierto

FL

POST: $\text{self}^{pre} = (lec, esc) \wedge \text{self} = (lec - 1, esc)$

CPRE: $\text{self}.e = 0 \wedge \text{self}.l = 0$

IE

POST: $\text{self} = (0, 1)$

CPRE: Cierto

FE

POST: $\text{self} = (0, 0)$

POLITÉCNICA

Ejercicio: Los nómadas que cantan

- Se está desarrollando una IA para competir en el campeonato del mundo del popular juego de mesa *Los nómadas que cantan*. Los desarrolladores han ideado un recurso compartido para representar las materias primas que tiene el jugador, siendo estas materias primas cereal, agua y madera.
- Hay una operación no bloqueante por cada materia prima para que el jugador coja esa materia prima: **cargarCereal**, **cargarAgua** y **cargarMadera**. El jugador puede llevar a lo sumo una unidad de cada materia prima pero no puede llevar cereal y madera a la vez, si carga una pierde la otra.
- Por otro lado, hay dos operaciones que consumen materias primas: **avanzar** (consume cereal y agua) y **reparar** (consume agua y madera). Estas operaciones son bloqueantes hasta que se disponga de las materias primas necesarias.
- **Se pide:** especificar el recurso compartido teniendo en cuenta que bastaría un booleano por cada materia prima para indicar que el jugador tiene dicha materia (porque la ha cargado) o no tiene dicha materia (porque acaba de empezar el juego o la ha consumido avanzando o reparando).

POLITÉCNICA

Ejercicio: Los nómadas que cantan

C-TAD MateriasPrimas
OPERACIONES
ACCIÓN cargarCereal, cargarAgua, cargarMadera, avanzar, reparar:
SEMÁNTICA
DOMINIO:
TIPO: *MateriasPrimas =*
INICIAL:
INVARIANTE:
CPRE:
cargarCereal
POST:
CPRE:
cargarAgua
POST:
CPRE:
cargarMadera
POST:
CPRE:
avanzar
POST:
CPRE:
reparar
POST:

POLITÉCNICA

Ejercicio: Los nómadas que cantan

C-TAD MateriasPrimas
OPERACIONES
ACCIÓN cargarCereal, cargarAgua, cargarMadera, avanzar, reparar:
SEMÁNTICA
DOMINIO:
TIPO: $\mathit{MateriasPrimas} = (\mathit{cereal} : \mathbb{B} \times \mathit{agua} : \mathbb{B} \times \mathit{madera} : \mathbb{B})$
INICIAL:
INVARIANTE:
CPRE:
cargarCereal
POST:
CPRE:
cargarAgua
POST:
CPRE:
cargarMadera
POST:
CPRE:
avanzar
POST:
CPRE:
reparar
POST:

POLITÉCNICA

Ejercicio: Los nómadas que cantan

C-TAD MateriasPrimas
OPERACIONES
ACCIÓN cargarCereal, cargarAgua, cargarMadera, avanzar, reparar:
SEMÁNTICA
DOMINIO:
TIPO: *MateriasPrimas* = $(cereal : \mathbb{B} \times agua : \mathbb{B} \times madera : \mathbb{B})$
INICIAL: $\text{self} = (\text{Falso}, \text{Falso}, \text{Falso})$
INVARIANTE:
CPRE:
cargarCereal
POST:
CPRE:
cargarAgua
POST:
CPRE:
cargarMadera
POST:
CPRE:
avanzar
POST:
CPRE:
reparar
POST:

POLITÉCNICA

Ejercicio: Los nómadas que cantan

C-TAD MateriasPrimas
OPERACIONES
ACCIÓN cargarCereal, cargarAgua, cargarMadera, avanzar, reparar:
SEMÁNTICA
DOMINIO:
TIPO: *MateriasPrimas* = $(cereal : \mathbb{B} \times agua : \mathbb{B} \times madera : \mathbb{B})$
INICIAL: $\text{self} = (\text{Falso}, \text{Falso}, \text{Falso})$
INVARIANTE:
CPRE: Cierto
cargarCereal
POST:
CPRE: Cierto
cargarAgua
POST:
CPRE: Cierto
cargarMadera
POST:
CPRE:
avanzar
POST:
CPRE:
reparar
POST:

POLITÉCNICA

Ejercicio: Los nómadas que cantan

C-TAD MateriasPrimas
OPERACIONES
ACCIÓN cargarCereal, cargarAgua, cargarMadera, avanzar, reparar:
SEMÁNTICA
DOMINIO:
TIPO: *MateriasPrimas* = $(cereal : \mathbb{B} \times agua : \mathbb{B} \times madera : \mathbb{B})$
INICIAL: $\text{self} = (\text{Falso}, \text{Falso}, \text{Falso})$
INVARIANTE:
CPRE: Cierto
cargarCereal
POST:
CPRE: Cierto
cargarAgua
POST:
CPRE: Cierto
cargarMadera
POST:
CPRE: $\text{self}.cereal \wedge \text{self}.agua$
avanzar
POST:
CPRE: $\text{self}.madera \wedge \text{self}.agua$
reparar
POST:

POLITÉCNICA

Ejercicio: Los nómadas que cantan

C-TAD MateriasPrimas
OPERACIONES
ACCIÓN cargarCereal, cargarAgua, cargarMadera, avanzar, reparar:
SEMÁNTICA
DOMINIO:
TIPO: *MateriasPrimas* = $(cereal : \mathbb{B} \times agua : \mathbb{B} \times madera : \mathbb{B})$
INICIAL: $\text{self} = (\text{Falso}, \text{Falso}, \text{Falso})$
INVARIANTE:
CPRE: Cierto
cargarCereal
POST: $\text{self} = (\text{Cierto}, \text{self}^{pre}.agua, \text{Falso})$
CPRE: Cierto
cargarAgua
POST:
CPRE: Cierto
cargarMadera
POST:
CPRE: $\text{self}.cereal \wedge \text{self}.agua$
avanzar
POST:
CPRE: $\text{self}.madera \wedge \text{self}.agua$
reparar
POST:

POLITÉCNICA

Ejercicio: Los nómadas que cantan

C-TAD MateriasPrimas
OPERACIONES
ACCIÓN cargarCereal, cargarAgua, cargarMadera, avanzar, reparar:
SEMÁNTICA
DOMINIO:
TIPO: *MateriasPrimas* = $(\mathit{cereal} : \mathbb{B} \times \mathit{agua} : \mathbb{B} \times \mathit{madera} : \mathbb{B})$
INICIAL: $\text{self} = (\text{Falso}, \text{Falso}, \text{Falso})$
INVARIANTE:
CPRE: Cierto
cargarCereal
POST: $\text{self} = (\text{Cierto}, \text{self}^{\mathit{pre}}.\mathit{agua}, \text{Falso})$
CPRE: Cierto
cargarAgua
POST: $\text{self} = (\text{self}^{\mathit{pre}}.\mathit{cereal}, \text{Cierto}, \text{self}^{\mathit{pre}}.\mathit{madera})$
CPRE: Cierto
cargarMadera
POST:
CPRE: $\text{self}.\mathit{cereal} \wedge \text{self}.\mathit{agua}$
avanzar
POST:
CPRE: $\text{self}.\mathit{madera} \wedge \text{self}.\mathit{agua}$
reparar
POST:

POLITÉCNICA

Ejercicio: Los nómadas que cantan

C-TAD MateriasPrimas
OPERACIONES
ACCIÓN cargarCereal, cargarAgua, cargarMadera, avanzar, reparar:
SEMÁNTICA
DOMINIO:
TIPO: *MateriasPrimas* = $(\mathit{cereal} : \mathbb{B} \times \mathit{agua} : \mathbb{B} \times \mathit{madera} : \mathbb{B})$
INICIAL: $\text{self} = (\text{Falso}, \text{Falso}, \text{Falso})$
INVARIANTE:
CPRE: Cierto
cargarCereal
POST: $\text{self} = (\text{Cierto}, \text{self}^{\mathit{pre}}.\mathit{agua}, \text{Falso})$
CPRE: Cierto
cargarAgua
POST: $\text{self} = (\text{self}^{\mathit{pre}}.\mathit{cereal}, \text{Cierto}, \text{self}^{\mathit{pre}}.\mathit{madera})$
CPRE: Cierto
cargarMadera
POST: $\text{self} = (\text{Falso}, \text{self}^{\mathit{pre}}.\mathit{agua}, \text{Cierto})$
CPRE: $\text{self}.\mathit{cereal} \wedge \text{self}.\mathit{agua}$
avanzar
POST:
CPRE: $\text{self}.\mathit{madera} \wedge \text{self}.\mathit{agua}$
reparar
POST:

POLITÉCNICA

Ejercicio: Los nómadas que cantan

C-TAD MateriasPrimas
OPERACIONES
ACCIÓN cargarCereal, cargarAgua, cargarMadera, avanzar, reparar:
SEMÁNTICA
DOMINIO:
TIPO: $\mathit{MateriasPrimas} = (\mathit{cereal} : \mathbb{B} \times \mathit{agua} : \mathbb{B} \times \mathit{madera} : \mathbb{B})$
INICIAL: $\text{self} = (\text{Falso}, \text{Falso}, \text{Falso})$
INVARIANTE: $\neg\text{self}.\mathit{cereal} \vee \neg\text{self}.\mathit{madera}$
CPRE: Cierto
cargarCereal
POST: $\text{self} = (\text{Cierto}, \text{self}^{\mathit{pre}}.\mathit{agua}, \text{Falso})$
CPRE: Cierto
cargarAgua
POST: $\text{self} = (\text{self}^{\mathit{pre}}.\mathit{cereal}, \text{Cierto}, \text{self}^{\mathit{pre}}.\mathit{madera})$
CPRE: Cierto
cargarMadera
POST: $\text{self} = (\text{Falso}, \text{self}^{\mathit{pre}}.\mathit{agua}, \text{Cierto})$
CPRE: $\text{self}.\mathit{cereal} \wedge \text{self}.\mathit{agua}$
avanzar
POST: $\text{self} = (\text{Falso}, \text{Falso}, \text{self}^{\mathit{pre}}.\mathit{madera})$
CPRE: $\text{self}.\mathit{madera} \wedge \text{self}.\mathit{agua}$
reparar
POST:

POLITÉCNICA

Ejercicio: Los nómadas que cantan

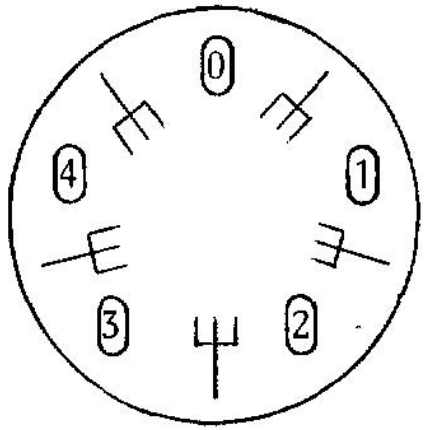
C-TAD MateriasPrimas
OPERACIONES
ACCIÓN cargarCereal, cargarAgua, cargarMadera, avanzar, reparar:
SEMÁNTICA
DOMINIO:
TIPO: $\mathit{MateriasPrimas} = (\mathit{cereal} : \mathbb{B} \times \mathit{agua} : \mathbb{B} \times \mathit{madera} : \mathbb{B})$
INICIAL: $\text{self} = (\text{Falso}, \text{Falso}, \text{Falso})$
INVARIANTE: $\neg\text{self}.\mathit{cereal} \vee \neg\text{self}.\mathit{madera}$
CPRE: Cierto
cargarCereal
POST: $\text{self} = (\text{Cierto}, \text{self}^{\mathit{pre}}.\mathit{agua}, \text{Falso})$
CPRE: Cierto
cargarAgua
POST: $\text{self} = (\text{self}^{\mathit{pre}}.\mathit{cereal}, \text{Cierto}, \text{self}^{\mathit{pre}}.\mathit{madera})$
CPRE: Cierto
cargarMadera
POST: $\text{self} = (\text{Falso}, \text{self}^{\mathit{pre}}.\mathit{agua}, \text{Cierto})$
CPRE: $\text{self}.\mathit{cereal} \wedge \text{self}.\mathit{agua}$
avanzar
POST: $\text{self} = (\text{Falso}, \text{Falso}, \text{self}^{\mathit{pre}}.\mathit{madera})$
CPRE: $\text{self}.\mathit{madera} \wedge \text{self}.\mathit{agua}$
reparar
POST: $\text{self} = (\text{self}^{\mathit{pre}}.\mathit{cereal}, \text{Falso}, \text{Falso})$

POLITÉCNICA

Ejercicio: Dining Philisophers

Lo que veis a continuación es la descripción original de Edsger W. Dijkstra del problema de los *Dining Philosophers*:

> *La vida de un filósofo consiste en alternar pensar y comer en una especie de bucle infinito. N filósofos, numerados del* 0 *al* $N-1$*, viven en una casa con una mesa en la que cada filósofo tiene su sitio asignado:*



> *Su único problema, al margen de los filosóficos, es que su menú consiste en un tipo complicado de espagueti que es necesario comer con dos tenedores. Hay un único tenedor a cada lado de cada plato. Eso no es un problema. Sin embargo, como consecuencia, no puede haber dos filósofos comiendo a la vez uno al lado del otro.*

POLITÉCNICA

Ejercicio: Dining Philisophers

Hemos decidido representar el problema en forma de un recurso compartido *Mesa* con:

- dos operaciones: $mesa.cogerTenedores(i)$ y $mesa.soltarTenedores(i)$
- Esas serán las operaciones que el filósofo i ejecutará antes y después de comer:
  - de forma que tendrá que bloquear en $mesa.cogerTenedores(i)$ cuando alguno de los tenedores a los lados de su plato estén siendo usados (por el filósofo $(i+1) \mod N$ o por el filósofo $(i-1) \mod N$)
- La operación $mesa.soltarTenedores(i)$ no es bloqueante
- NOTA: se sugiere que el dominio del recurso incluya una función parcial que hable de los tenedores libres. Las funciones parciales formalizan las tablas

POLITÉCNICA

Ejercicio: Dining Philisophers

C-TAD Mesa
OPERACIONES
ACCIÓN cogerTenedores: $\mathbb{N}$*[e]*
ACCIÓN soltarTenedores: $\mathbb{N}$*[e]*

SEMÁNTICA
DOMINIO:
TIPO: *Mesa* =
TIPO: *TipoFilosofo* = $\{0, .., N-1\}$

INVARIANTE:

INICIAL:

CPRE:
cogerTenedores(i)
POST:

CPRE:
soltarTenedores(i)
POST:

POLITÉCNICA

Ejercicio: Dining Philisophers

C-TAD Mesa
OPERACIONES
ACCIÓN cogerTenedores: $\mathbb{N}$*[e]*
ACCIÓN soltarTenedores: $\mathbb{N}$*[e]*

SEMÁNTICA
DOMINIO:
TIPO: *Mesa = tenedores* : *TipoFilosofo* $\nrightarrow \mathbb{B}$
TIPO: *TipoFilosofo* = $\{0, .., N-1\}$

INVARIANTE:

INICIAL:

CPRE:
cogerTenedores(i)
POST:

CPRE:
soltarTenedores(i)
POST:

POLITÉCNICA

Ejercicio: Dining Philisophers

C-TAD Mesa

OPERACIONES

ACCIÓN cogerTenedores: $\mathbb{N}$*[e]*

ACCIÓN soltarTenedores: $\mathbb{N}$*[e]*

SEMÁNTICA

DOMINIO:

TIPO: $Mesa = tenedores : TipoFilosofo \pfun \mathbb{B}$

TIPO: $TipoFilosofo = \{0, .., N - 1\}$

INVARIANTE:

INICIAL: $\forall\, i \in TipoFilosofo \bullet \neg \mathrm{self}.tenedores(i)$

CPRE:

cogerTenedores(i)

POST:

CPRE:

soltarTenedores(i)

POST:

POLITÉCNICA

Ejercicio: Dining Philisophers

C-TAD Mesa
OPERACIONES
ACCIÓN cogerTenedores: $\mathbb{N}$*[e]*
ACCIÓN soltarTenedores: $\mathbb{N}$*[e]*

SEMÁNTICA
DOMINIO:
TIPO: *Mesa = tenedores* : *TipoFilosofo* $\nrightarrow \mathbb{B}$
TIPO: *TipoFilosofo* = $\{0, .., N-1\}$

INVARIANTE:

INICIAL: $\forall\, i \in \mathit{TipoFilosofo} \bullet \neg \mathrm{self}.\mathit{tenedores}(i)$

CPRE: $\neg \mathrm{self}.\mathit{tenedores}(i) \ \wedge \ \neg \mathrm{self}.\mathit{tenedores}(i +_N 1)$
cogerTenedores(i)
POST:

CPRE:
soltarTenedores(i)
POST:

POLITÉCNICA

Ejercicio: Dining Philisophers

C-TAD Mesa
OPERACIONES
ACCIÓN cogerTenedores: $\mathbb{N}$*[e]*
ACCIÓN soltarTenedores: $\mathbb{N}$*[e]*

SEMÁNTICA
DOMINIO:
TIPO: *Mesa = tenedores* : *TipoFilosofo* $\nrightarrow \mathbb{B}$
TIPO: *TipoFilosofo* = $\{0, .., N-1\}$

INVARIANTE:

INICIAL: $\forall\, i \in \mathit{TipoFilosofo} \bullet \neg \mathrm{self}.\mathit{tenedores}(i)$

CPRE: $\neg \mathrm{self}.\mathit{tenedores}(i) \ \wedge \ \neg \mathrm{self}.\mathit{tenedores}(i +_N 1)$
cogerTenedores(i)
POST:

CPRE: Cierto
soltarTenedores(i)
POST:

POLITÉCNICA

Ejercicio: Dining Philisophers

C-TAD Mesa
OPERACIONES
ACCIÓN cogerTenedores: $\mathbb{N}$*[e]*
ACCIÓN soltarTenedores: $\mathbb{N}$*[e]*

SEMÁNTICA
DOMINIO:
TIPO: $Mesa = tenedores : TipoFilosofo \pfun \mathbb{B}$
TIPO: $TipoFilosofo = \{0, .., N - 1\}$

INVARIANTE:

INICIAL: $\forall\, i \in TipoFilosofo \bullet \neg \text{self}.tenedores(i)$

CPRE: $\neg \text{self}.tenedores(i) \land \neg \text{self}.tenedores(i +_N 1)$
cogerTenedores(i)
POST: $\text{self}.tenedores = \text{self}^{pre}.tenedores \oplus \{i \mapsto \text{Cierto}, i +_N 1 \mapsto \text{Cierto}\}$

CPRE: Cierto
soltarTenedores(i)
POST:

POLITÉCNICA

Ejercicio: Dining Philisophers

C-TAD Mesa
OPERACIONES
ACCIÓN cogerTenedores: $\mathbb{N}$*[e]*
ACCIÓN soltarTenedores: $\mathbb{N}$*[e]*

SEMÁNTICA
DOMINIO:
TIPO: *Mesa = tenedores* : *TipoFilosofo* $\nrightarrow \mathbb{B}$
TIPO: *TipoFilosofo* = $\{0, .., N-1\}$

INVARIANTE:

INICIAL: $\forall\, i \in \mathit{TipoFilosofo} \bullet \neg \mathrm{self}.\mathit{tenedores}(i)$

CPRE: $\neg \mathrm{self}.\mathit{tenedores}(i) \ \wedge \ \neg \mathrm{self}.\mathit{tenedores}(i +_N 1)$
cogerTenedores(i)
POST: $\mathrm{self}.\mathit{tenedores} = \mathrm{self}^{\mathit{pre}}.\mathit{tenedores} \oplus \{i \mapsto \mathrm{Cierto}, i +_N 1 \mapsto \mathrm{Cierto}\}$

CPRE: Cierto
soltarTenedores(i)
POST: $\mathrm{self}.\mathit{tenedores} = \mathrm{self}^{\mathit{pre}}.\mathit{tenedores} \oplus \{i \mapsto \mathrm{Falso}, i +_N 1 \mapsto \mathrm{Falso}\}$

POLITÉCNICA

Ejercicio: Dining Philisophers

C-TAD Mesa
OPERACIONES
ACCIÓN cogerTenedores: $\mathbb{N}$*[e]*
ACCIÓN soltarTenedores: $\mathbb{N}$*[e]*

SEMÁNTICA
DOMINIO:
TIPO: $Mesa = tenedores : TipoFilosofo \nrightarrow \mathbb{B}$
TIPO: $TipoFilosofo = \{0, .., N-1\}$

INVARIANTE: $| \{i \in TipoFilosofo \bullet self.tenedores(i)\} | \, mod 2 = 0$

INICIAL: $\forall\, i \in TipoFilosofo \bullet \neg \text{self}.tenedores(i)$

CPRE: $\neg \text{self}.tenedores(i) \ \wedge \ \neg \text{self}.tenedores(i +_N 1)$
cogerTenedores(i)
POST: $\text{self}.tenedores = \text{self}^{pre}.tenedores \oplus \{i \mapsto \text{Cierto}, i +_N 1 \mapsto \text{Cierto}\}$

CPRE: Cierto
soltarTenedores(i)
POST: $\text{self}.tenedores = \text{self}^{pre}.tenedores \oplus \{i \mapsto \text{Falso}, i +_N 1 \mapsto \text{Falso}\}$

POLITÉCNICA

Ejercicio: Dining Philisophers

C-TAD Mesa
OPERACIONES
ACCIÓN cogerTenedores: $\mathbb{N}$*[e]*
ACCIÓN soltarTenedores: $\mathbb{N}$*[e]*

SEMÁNTICA
DOMINIO:
TIPO: *Mesa = TipoFilosofo* $\nrightarrow \mathbb{B}$
TIPO: *TipoFilosofo* = $\{0, .., N-1\}$

INVARIANTE: $| \{i \in TipoFilosofo \bullet self(i)\} | mod 2 = 0$

INICIAL: $\forall\, i \in TipoFilosofo \bullet \neg \mathrm{self}(i)$

CPRE: $\neg \mathrm{self}(i) \ \wedge \ \neg \mathrm{self}(i +_N 1)$
cogerTenedores(i)
POST: $\mathrm{self} = \mathrm{self}^{pre} \oplus \{i \mapsto \mathrm{Cierto}, i +_N 1 \mapsto \mathrm{Cierto}\}$

CPRE: Cierto
soltarTenedores(i)
POST: $\mathrm{self} = \mathrm{self}^{pre} \oplus \{i \mapsto \mathrm{Falso}, i +_N 1 \mapsto \mathrm{Falso}\}$

POLITÉCNICA

Ejercicios: Examen 18/19 Mayo

Dado el siguiente CTAD:

C-TAD MiCTAD
TIPO: $\mathit{MiCTAD} = \mathit{Indice} \to \mathbb{B}$
TIPO: $\mathit{Indice} = \{0, 1\}$
INICIAL: $\forall\, i \in \mathit{Indice} \bullet \neg\text{self}(i)$
INVARIANTE: $\neg\text{self}(0) \vee \text{self}(1)$
CPRE: $\neg\text{self}(0) \wedge \neg\text{self}(1)$
uno()
POST: $\text{self} = \text{self}^{pre} \oplus \{1 \mapsto \text{Cierto}\}$
CPRE: $\neg\text{self}(0) \wedge \text{self}(1)$
dos()
POST: $\text{self} = \text{self}^{pre} \oplus \{0 \mapsto \text{Cierto}\}$
CPRE: $\text{self}(1)$
tres()
POST: $\text{self} = \text{self}^{pre} \oplus \{0 \mapsto \text{Falso}\} \oplus \{1 \mapsto \text{Falso}\}$

Asumiendo que tenemos tres procesos que invocan repetidamente las operaciones *uno*(), *dos*() y *tres*() del recurso compartido. Se pide marcar la afirmación correcta:

(a) El sistema podría quedarse bloqueado

(b) El sistema nunca quedará bloqueado y siempre se cumplirá la invariante

(c) Podría llegar a violarse la invariante

POLITÉCNICA

Ejercicios: Examen 18/19 Mayo

C-TAD MiCTAD

TIPO: $\mathit{MiCTAD} = \mathit{Indice} \rightarrow \mathbb{B}$

TIPO: $\mathit{Indice} = \{0, 1\}$

INICIAL: $\forall\, i \in \mathit{Indice} \bullet \neg \text{self}(i)$

INVARIANTE: $\neg \text{self}(0) \vee \text{self}(1)$

CPRE: $\neg \text{self}(0) \wedge \neg \text{self}(1)$

uno()

POST: $\text{self} = \text{self}^{pre} \oplus \{1 \mapsto \text{Cierto}\}$

CPRE: $\neg \text{self}(0) \wedge \text{self}(1)$

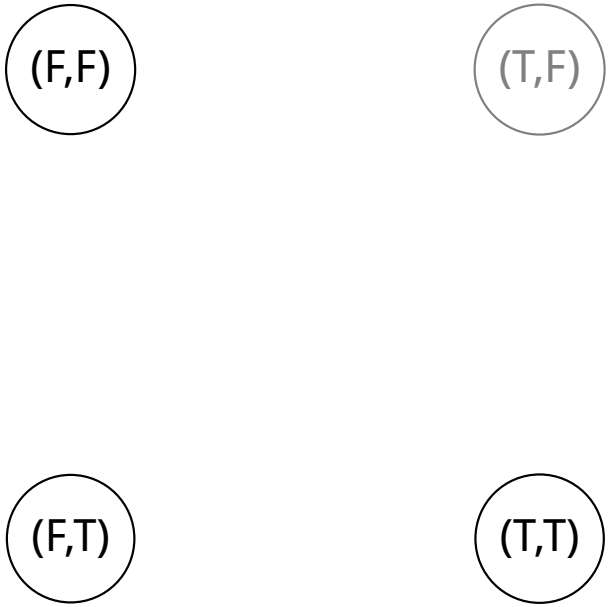
dos()

POST: $\text{self} = \text{self}^{pre} \oplus \{0 \mapsto \text{Cierto}\}$

CPRE: $\text{self}(1)$

tres()

POST: $\text{self} = \text{self}^{pre} \oplus \{0 \mapsto \text{Falso}\} \oplus \{1 \mapsto \text{Falso}\}$



(a) El sistema podría quedarse bloqueado

(b) El sistema nunca quedará bloqueado y siempre se cumplirá la invariante

(c) Podría llegar a violarse la invariante

POLITÉCNICA

Ejercicios: Examen 18/19 Mayo

C-TAD MiCTAD

TIPO: $\mathit{MiCTAD} = \mathit{Indice} \to \mathbb{B}$

TIPO: $\mathit{Indice} = \{0, 1\}$

INICIAL: $\forall\, i \in \mathit{Indice} \bullet \neg\text{self}(i)$

INVARIANTE: $\neg\text{self}(0) \vee \text{self}(1)$

CPRE: $\neg\text{self}(0) \wedge \neg\text{self}(1)$

uno()

POST: $\text{self} = \text{self}^{pre} \oplus \{1 \mapsto \text{Cierto}\}$

CPRE: $\neg\text{self}(0) \wedge \text{self}(1)$

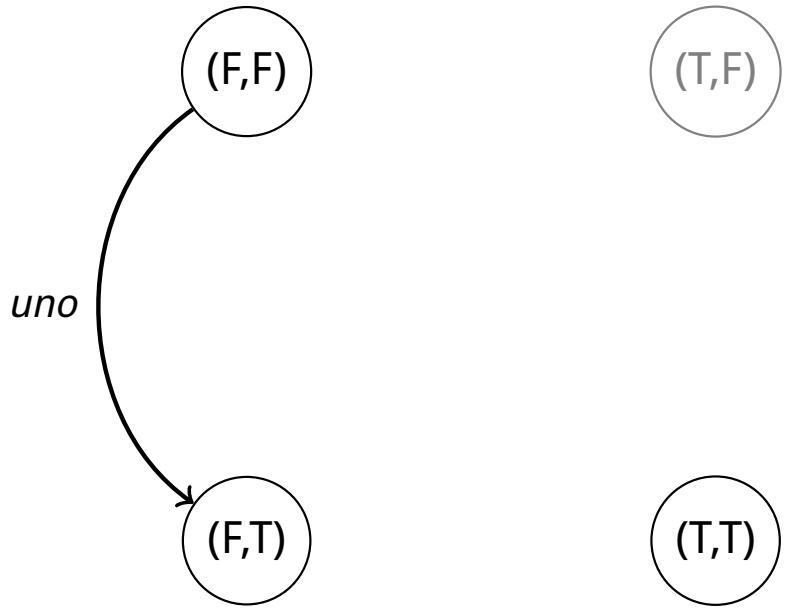
dos()

POST: $\text{self} = \text{self}^{pre} \oplus \{0 \mapsto \text{Cierto}\}$

CPRE: $\text{self}(1)$

tres()

POST: $\text{self} = \text{self}^{pre} \oplus \{0 \mapsto \text{Falso}\} \oplus \{1 \mapsto \text{Falso}\}$



(a) El sistema podría quedarse bloqueado

(b) El sistema nunca quedará bloqueado y siempre se cumplirá la invariante

(c) Podría llegar a violarse la invariante

POLITÉCNICA

Ejercicios: Examen 18/19 Mayo

C-TAD MiCTAD

TIPO: $\mathit{MiCTAD} = \mathit{Indice} \to \mathbb{B}$

TIPO: $\mathit{Indice} = \{0, 1\}$

INICIAL: $\forall\, i \in \mathit{Indice} \bullet \neg \mathrm{self}(i)$

INVARIANTE: $\neg \mathrm{self}(0) \vee \mathrm{self}(1)$

CPRE: $\neg \mathrm{self}(0) \wedge \neg \mathrm{self}(1)$

uno()

POST: $\mathrm{self} = \mathrm{self}^{pre} \oplus \{1 \mapsto \mathrm{Cierto}\}$

CPRE: $\neg \mathrm{self}(0) \wedge \mathrm{self}(1)$

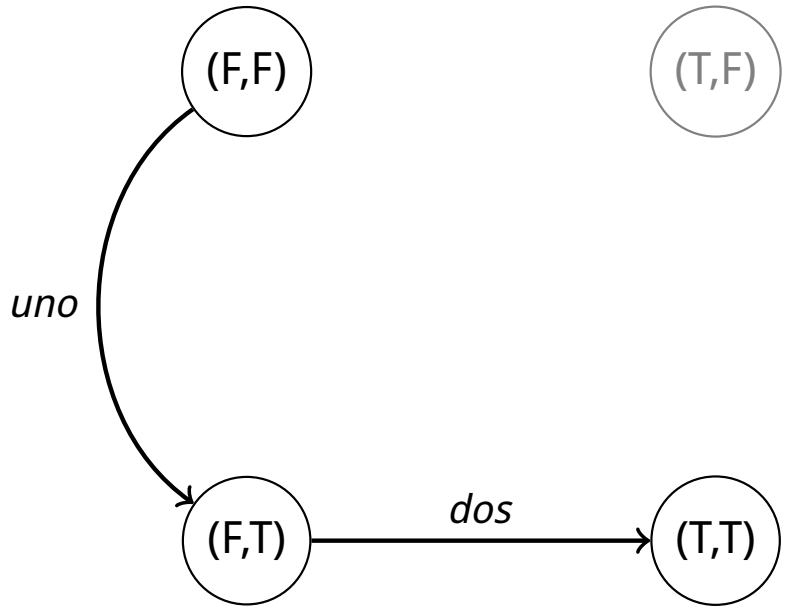
dos()

POST: $\mathrm{self} = \mathrm{self}^{pre} \oplus \{0 \mapsto \mathrm{Cierto}\}$

CPRE: $\mathrm{self}(1)$

tres()

POST: $\mathrm{self} = \mathrm{self}^{pre} \oplus \{0 \mapsto \mathrm{Falso}\} \oplus \{1 \mapsto \mathrm{Falso}\}$



(a) El sistema podría quedarse bloqueado

(b) El sistema nunca quedará bloqueado y siempre se cumplirá la invariante

(c) Podría llegar a violarse la invariante

POLITÉCNICA

Ejercicios: Examen 18/19 Mayo

C-TAD MiCTAD

TIPO: $\mathit{MiCTAD} = \mathit{Indice} \to \mathbb{B}$

TIPO: $\mathit{Indice} = \{0, 1\}$

INICIAL: $\forall\, i \in \mathit{Indice} \bullet \neg \text{self}(i)$

INVARIANTE: $\neg \text{self}(0) \vee \text{self}(1)$

CPRE: $\neg \text{self}(0) \wedge \neg \text{self}(1)$

uno()

POST: $\text{self} = \text{self}^{pre} \oplus \{1 \mapsto \text{Cierto}\}$

CPRE: $\neg \text{self}(0) \wedge \text{self}(1)$

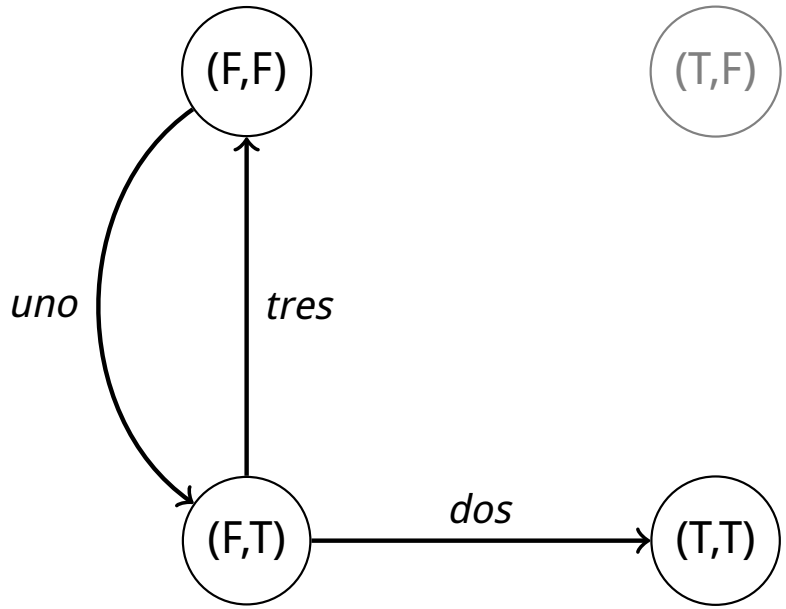
dos()

POST: $\text{self} = \text{self}^{pre} \oplus \{0 \mapsto \text{Cierto}\}$

CPRE: $\text{self}(1)$

tres()

POST: $\text{self} = \text{self}^{pre} \oplus \{0 \mapsto \text{Falso}\} \oplus \{1 \mapsto \text{Falso}\}$



(a) El sistema podría quedarse bloqueado

(b) El sistema nunca quedará bloqueado y siempre se cumplirá la invariante

(c) Podría llegar a violarse la invariante

POLITÉCNICA

Ejercicios: Examen 18/19 Mayo

C-TAD MiCTAD

TIPO: $\mathit{MiCTAD} = \mathit{Indice} \to \mathbb{B}$

TIPO: $\mathit{Indice} = \{0, 1\}$

INICIAL: $\forall\, i \in \mathit{Indice} \bullet \neg\text{self}(i)$

INVARIANTE: $\neg\text{self}(0) \vee \text{self}(1)$

CPRE: $\neg\text{self}(0) \wedge \neg\text{self}(1)$

uno()

POST: $\text{self} = \text{self}^{pre} \oplus \{1 \mapsto \text{Cierto}\}$

CPRE: $\neg\text{self}(0) \wedge \text{self}(1)$

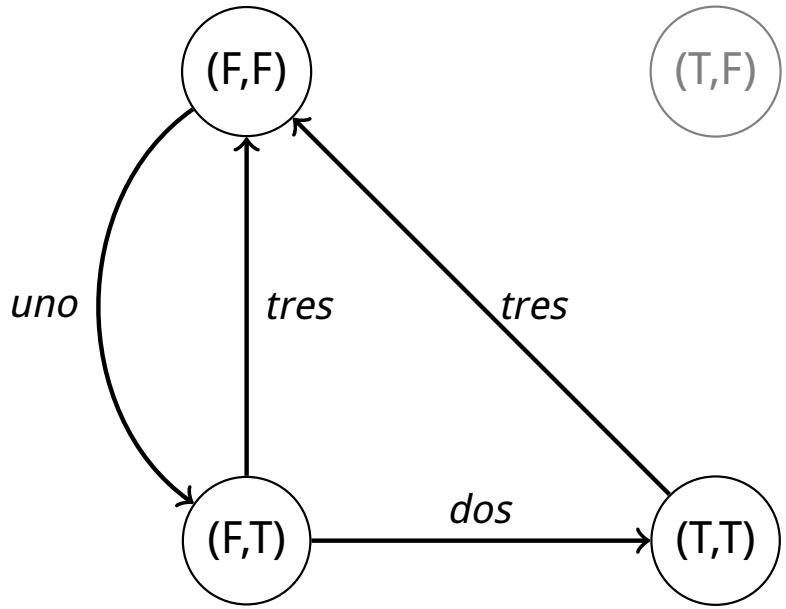
dos()

POST: $\text{self} = \text{self}^{pre} \oplus \{0 \mapsto \text{Cierto}\}$

CPRE: $\text{self}(1)$

tres()

POST: $\text{self} = \text{self}^{pre} \oplus \{0 \mapsto \text{Falso}\} \oplus \{1 \mapsto \text{Falso}\}$



(a) El sistema podría quedarse bloqueado

(b) El sistema nunca quedará bloqueado y siempre se cumplirá la invariante

(c) Podría llegar a violarse la invariante

POLITÉCNICA

Ejercicios: Examen 18/19 Mayo

Dado el siguiente programa

```
class Hilos {
  static class MiHilo extends Thread {
    int n;
    public void run () {
      for (int i=0; i<100; i++) {
        n ++;
      }
    }
  }
```

```
  public void main (String [] args) {
    Thread t = new MiHilo ();
    t.start();
    t.run();
    hacerAlgo(); // hace algo...
    try {t.join();}
    catch (InterruptedException e){}
    System.out.println(t.n);
  }
```

¿Cuál será la salida por consola al ejecutar el `main`? **Se pide** marcar la afirmación correcta.

(a) 200

(b) No se puede saber porque podría haber condiciones de carrera.

(c) 100

POLITÉCNICA

Ejercicios: Examen 18/19 Mayo

Dado el siguiente programa

```
class Hilos {
  static class MiHilo extends Thread {
    int n;
    public void run () {
      for (int i=0; i<100; i++) {
        n ++;
      }
    }
  }
```

```
  public void main (String [] args) {
    Thread t = new MiHilo ();
    t.start();
    t.run();
    hacerAlgo(); // hace algo...
    try {t.join();}
    catch (InterruptedException e){}
    System.out.println(t.n);
  }
```

¿Cuál será la salida por consola al ejecutar el `main`? **Se pide** marcar la afirmación correcta.

(a) El número máximo de procesos ejecutando a la vez será 2 y el método `main` podrá terminar antes que el thread `t`

(b) El número máximo de procesos ejecutando a la vez será 3 y el método `main` terminará siempre después que el thread `t`

(c) El número máximo de procesos ejecutando a la vez será 2 y el método `main` terminará siempre después que el thread `t`

POLITÉCNICA

Ejercicios: Examen 18/19 Mayo

Dado un programa concurrente en la que tres *threads* instancias de las clases A, B y C comparten una variable n:

```
static volatile int n = 0;
static Semaphore s1 = new Semaphore(1);
static Semaphore s2 = new Semaphore(0);
```

```
class A extends Thread {
  public void run() {
    s2.await();
    n = 2 * n;
    s1.signal();
  }
}
```

```
class B extends Thread {
  public void run() {
    s1.await();
    n = n * n;
    s2.signal();
  }
}
```

```
class C extends Thread {
  public void run() {
    s1.await();
    n = n + 2;
    s2.signal();
  }
}
```

¿Cuál es el valor de n tras terminar los tres threads?

(a) 4

(b) 4 o 16

(c) 2 o 16

Si en el código anterior los semáforos s1 y s2 se inicializan a 1.

(a) No está garantizada la exclusión mutua en el acceso a n

(b) No está garantizada la terminación de las tres tareas

POLITÉCNICA

Ejercicios: Examen 18/19 Mayo

Dada la siguiente implementación de una solución al problema de la exclusión mutua con *espera activa*:

```
static volatile boolean inc_quiere = false;
static volatile boolean dec_quiere = false;
static volatile int cont = 0;
```

```
class Incrementador extends Thread {
  public void run() {
    for (int i = 0; i < N_OPS; i++) {
      inc_quiere = true;
      while (dec_quiere) {}
      cont++;  // SC
      inc_quiere = false;
    }
  }
}
```

```
class Decrementador extends Thread {
  public void run() {
    for (int i = 0; i < N_OPS; i++) {
      dec_quiere = true;
      while (inc_quiere) {}
      cont--; // SC
      dec_quiere = false;
    }
  }
}
```

Suponiendo que tenemos un proceso de tipo `Incrementador` y otro proceso `Decrementador`, **se pide** marcar la afirmación correcta.

(a) El programa no garantiza la exclusión mutua en el acceso a la sección crítica (`cont++` y `cont--`)

(b) El programa no garantiza la propiedad de ausencia de interbloqueo

(c) El programa no garantiza la ausencia de esperas innecesarias

POLITÉCNICA

Ejercicio: Secuencias ordenadas de enteros

El recurso compartido *OrdMezcla* mezcla dos secuencias ordenadas de números enteros para formar una única secuencia ordenada.

- En este recurso interactúan tres procesos: dos productores que van pasando números de sus secuencias de uno en uno y un consumidor que va extrayendo los números en orden
- El recurso será capaz de almacenar, como mucho, un dato de la secuencia que llamaremos “izquierda” y un dato de la secuencia “derecha”
- Cuando hay datos de ambas secuencias la operación *extraerMenor* tomará el menor de ambos y permitirá que se añada un nuevo dato de la secuencia correspondiente
- La operación *insertarIzda(d)* inserta el dato *d* como parte de la secuencia izquierda y bloquea hasta que el hueco para el dato izquierdo está disponible
- La operación *insertarDcha* es análoga por lo que no es necesario implementarla

POLITÉCNICA

Ejercicio: Secuencias Ordenadas de enteros

C-TAD OrdMezcla
OPERACIONES
ACCIÓN insertarIzda: $\mathbb{Z}$*[e]*
ACCIÓN insertarDcha: $\mathbb{Z}$*[e]*
ACCIÓN extraerMenor: $\mathbb{Z}$*[s]*

SEMÁNTICA
DOMINIO:
TIPO: $OrdMezcla = \langle hayDato : Lado \to \mathbb{B} \times dato : Lado \to \mathbb{Z} \rangle$
TIPO: $Lado = Izda \mid Dcha$

INICIAL: $\forall\, i \in Lado \bullet \neg self.hayDato(i)$

CPRE: $\neg \text{self}.hayDato(Izda)$
insertarIzda(d)
POST: $\text{self}.hayDato = \text{self}^{pre}.hayDato \oplus \{Izda \mapsto \text{Cierto}\} \wedge$
$\text{self}.dato = \text{self}^{pre}.dato \oplus \{Izda \mapsto d\}$

CPRE: $self.hayDato(Izda) \wedge self.hayDato(Dcha)$
extraerMenor(min)
POST: $(\text{self}^{pre}.dato(Izda) \leq \text{self}^{pre}.dato(Dcha) \wedge$
$\text{self}.hayDato = \text{self}^{pre}.hayDato \oplus \{Izda \mapsto \text{Falso}\} \wedge min = self^{pre}.dato(Izda)) \vee$
$(\text{self}^{pre}.dato(Dcha) \leq \text{self}^{pre}.dato(Izda) \wedge$
$\text{self}.hayDato = \text{self}^{pre}.hayDato \oplus \{Dcha \mapsto \text{Falso}\} \wedge min = self^{pre}.dato(Dcha))$

POLITÉCNICA