Exámenes teóricos y prácticos de CSP y Statecharts

10 de marzo de 2020

Índice

1. Ejercicios teóricos

1. Para cada uno de los procesos siguientes, encuentre el proceso secuencial equivalente justificando cada paso de su calculo.

   a) $P = a \to b \to STOP \parallel (a \to a \to STOP \sqcap a \to c \to STOP)$.

   b) $Q = (b \to b \to STOP \Box a \to b \to STOP) \parallel a \to c \to STOP$.

   c) $R = c \to a \to STOP \parallel (a \to b \to STOP \Box b \to b \to STOP)$.

Soluciones

a) $a \to b \to STOP_{\{a,b\}} \parallel \big(a \to a \to STOP_{\{a\}} \sqcap a \to c \to STOP_{\{a,c\}}\big)$
$\equiv \langle e \to P \sqcap e \to Q = e \to (P \sqcap Q) \rangle$
$a \to b \to STOP_{\{a,b\}} \parallel a \to \big(a \to STOP_{\{a\}} \sqcap c \to STOP_{\{a,c\}}\big)$
$\equiv \langle Sincronization \rangle$
$a \to \big(b \to STOP_{\{a,b\}} \parallel \big(a \to STOP_{\{a\}} \sqcap c \to STOP_{\{a,c\}}\big)\big)$
$\equiv \langle P \parallel (Q \sqcap R) = (P \parallel Q) \sqcap (P \parallel R) \rangle$
$a \to ((BSTOP \parallel ASTOP) \sqcap (BSTOP \parallel CSTOP))$
$\equiv \langle Interleaving \rangle$
$a \to \big(\big(b \to \big(STOP_{\{a,b\}} \parallel ASTOP\big) \Box a \to \big(BSTOP \parallel STOP_{\{a\}}\big)\big) \sqcap (BSTOP \parallel CSTOP)\big)$
$\equiv \langle \{a,b\} \cap \alpha ASTOP \neq \emptyset \rangle$
$a \to \big(\big(b \to STOP \Box a \to \big(BSTOP \parallel STOP_{\{a\}}\big)\big) \sqcap (BSTOP \parallel CSTOP)\big)$
$\equiv \langle \alpha BSTOP \cap \{a\} \neq \emptyset \rangle$
$a \to ((b \to STOP \Box a \to STOP) \sqcap (BSTOP \parallel CSTOP))$
$\equiv \langle Interleaving \rangle$
$a \to \big((b \to STOP \Box a \to STOP) \sqcap \big(b \to \big(STOP_{\{a,b\}} \parallel CSTOP\big) \Box c \to \big(BSTOP \parallel STOP_{\{a,c\}}\big)\big)\big)$
$\equiv \langle \{a,b\} \cap \alpha CSTOP \neq \emptyset \rangle$
$a \to \big((b \to STOP \Box a \to STOP) \sqcap \big(b \to STOP \Box c \to \big(BSTOP \parallel STOP_{\{a,c\}}\big)\big)\big)$
$\equiv \langle \alpha BSTOP \cap \{a,c\} \neq \emptyset \rangle$
$a \to ((b \to STOP \Box a \to STOP) \sqcap (b \to STOP \Box c \to STOP))$

b) COMPLETAR.

c) COMPLETAR.

2. Modele un proceso CSP que debe recibir cien señales en cualquier orden y, una vez que las ha recibido a todas, debe comportarse como el proceso W.

   Solución Sea α el conjunto de todas las señales que puede recibir dicho proceso. Si definimos $P(n) = \alpha \to P(n+1) \lceil n \neq 100 \rceil W$, entonces el proceso que estamos buscando es: $P(0)$.

3. Explique en pocas líneas el concepto semántico utilizado en CSP que permite formalizar la idea de no determinismo.

   Solución Recordemos que $\boldsymbol{t}(P \Box Q) = \boldsymbol{t}(P \sqcap Q)$. Este concepto formaliza la idea de cuales son los comportamientos que puede tener un proceso, y evidentemente ambos procesos pueden comportarse como P o Q.

   Por el contrario el no determinismo esta relacionado con la idea de los comportamientos que un proceso podría hacer pero finalmente no hace. Por ejemplo en el caso $a \to P \sqcap b \to Q$ si el entorno ofrece el evento a el proceso podría rechazarlo pues prefiere comportarse como $b \to Q$ y viceversa; en cambio si ofrece ambos eventos el proceso solo puede rechazar uno de ellos.

   Los rechazos de un proceso son el concepto que nos permite formalizar el no determinismo y diferenciar entre $\Box$ y $\sqcap$.

4. Explique las diferencias y semejanzas entre los operadores CSP: $|$, $\Box$, $\sqcap$.

   Solución

   - El operador $|$ (llamado alternativa etiquetada) espera que el entorno indique como debe progresar el proceso: si el entorno ofrece el primer evento del proceso a la izquierda entonces se comportara como tal, y si ofrece el de la derecha hará lo propio.

     A ambos lados del operador debe estar explícitamente indicado cual es el primer evento, es decir, no es un termino valido escribir $P \mid Q$, deberá en dicho caso escribirse $a \to P' \mid b \to Q'$. Ademas

dicho primer evento de cada proceso debe ser diferente, es decir que no se admiten términos de la forma $a \to P \mid a \to Q$.

- El operador $\Box$ (llamado selección externa) se comporta de forma idéntica a la alternativa etiquetada, solo que este admite los términos que el otro no, es decir que es valido escribir términos de la forma $P \mid Q$ o $a \to P \mid a \to Q$.
- El operador $\sqcap$ (llamado selección interna) representa el comportamiento no determinístico respecto del entorno. Es decir que el entorno no puede saber si $P \sqcap Q$ se comportará como P o como Q.

5. Explique con la debida precisión la semántica del modelo de concurrencia de CSP.

   Solución COMPLETAR.

6. Escriba en CSP y explique cada una de las siguientes leyes: «Interleaving», «Deadlock», «Sincronization» y «IEOF».

   Solución

   - Interleaving:

     $$a \notin \alpha Q \wedge b \notin \alpha Q \Rightarrow a \to P \parallel b \to Q = a \to (P \parallel b \to Q) \,\Box b \to (a \to P \parallel Q)$$

     La ley de interleaving dice que si dos procesos paralelos no su primer evento, deben entonces considerarse todas las posibilidades, es decir que el entorno decidirá si ocurre a o b. Si primero se consume a entonces a continuación puede consumirse el resto de P o bien b; por el contrario si primero se consume b a continuación puede consumirse el resto de Q o bien a.
   - Deadlock: $a \in \alpha Q \wedge b \in \alpha P \Rightarrow a \to P \parallel b \to Q = STOP$.
     En CSP si dos procesos comparten un evento, ambos deben estar dispuestos a consumirlo al mismo tiempo. Precisamente como a y b son eventos compartidos, el modelo sincrónico indica que cualquiera de los procesos debe esperar a que el otro esté dispuesto a comunicar el evento que el primero está esperando, pero como ambos están en la misma situación el progreso es imposible

- Sincronization: $a \to P \parallel a \to Q = a \to (P \parallel Q)$.

  La ley de sincronización dice que si dos procesos deben interactuar a través de un evento en común, entonces ambos consumen el evento al mismo tiempo.
- IEOF: $a \notin \alpha Q \land b \in \alpha P \Rightarrow a \to P \parallel b \to Q = a \to (P \parallel b \to Q)$.

  Por ultimo la ley «los eventos independientes ocurren primero» nos indica justamente eso: si un evento (como a) no es compartido por dos procesos paralelos, entonces este debe consumirse primero.

7. Determine si la siguiente proposición es correcta o no (es decir si la igualdad es correcta o no) y justifique su respuesta:

$$c?x \to P; Q = c?x \to (P; Q)$$

Solución La igualdad no es correcta, y para verlo daremos un contraejemplo. Sean $P = SKIP$ y $Q = c!x \to STOP$.

El proceso al lado derecho de la igualdad recibe un mensaje x y luego pasa a comportarse como un proceso que primero termina exitosamente y a continuación reenviá ese mismo mensaje antes de finalizar.

Por el contrario, en el proceso a la izquierda de la igualdad, el mensaje recibido solo esta dentro del alcance de P por lo que Q no enviará dicho mensaje ya que P ha terminado.

8. Describa el modelo de fallas y divergencias de CSP, explicando y definiendo los conceptos de traza, rechazo, falla y divergencia.

Solución COMPLETAR.

9. Enuncie las tres razones por las cuales, según Jackson, R no puede ser S.

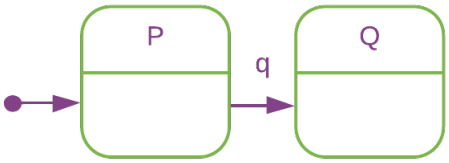
Solución

- Puede haber fenómenos no compartidos.
- Puede haber referencias a futuro.
- COMPLETAR.

10. Explique la diferencia entre los operadores $\Box$ y ∇. Luego muestre como representar ∇ en Statecharts.

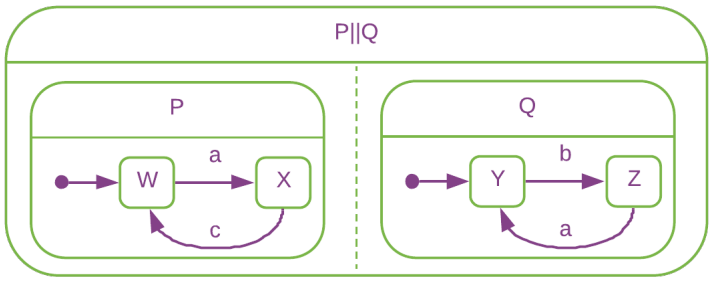
Solución El proceso $P \Box Q$ es un proceso que espera a que el entorno provea o bien el primer evento de P o bien el primero de Q; y pasa a comportarse como el proceso correspondiente.

El proceso $P \nabla Q$ se comporta como P hasta tanto ocurra el primer evento de Q, en cuyo momento pasa a comportarse como este.



11. Muestre un ejemplo sencillo donde se pueda apreciar la diferencia de los modelos de concurrencia de Statecharts y CSP.

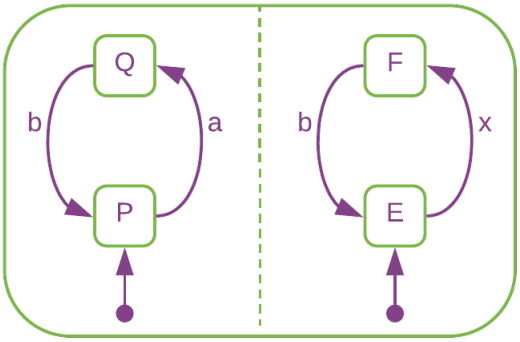
Solución



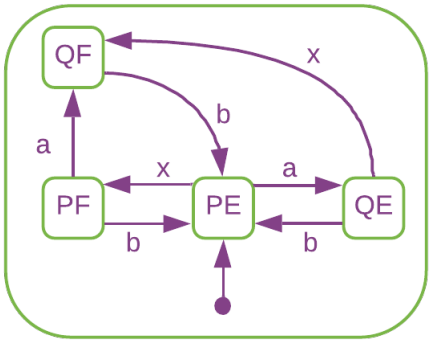
En este Statechart, partiendo del estado inicial (W, Y), si ocurre el evento a se transiciona al estado (X, Y).

En cambio en el proceso CSP $a \to c \to P \parallel b \to a \to Q$, de ocurrir el mismo evento la transición no se produce pues dicho evento forma parte del alfabeto de los dos procesos y para que la transición ocurra, ambos procesos deben estar dispuestos a recibir el evento al mismo tiempo.

12. Muestre un Statechart equivalente al que se muestra en la figura que no haga uso de los estados AND.



Solución



13. Traduzca el Statechart de la figura anterior en un proceso CSP equivalente. Si no puede hacerlo, justifique su respuesta.

    Solución

    - $PE = a \to QE | x \to PF$.
    - $PF = b \to PE | a \to QF$.
    - $QE = b \to PE | x \to QF$.
    - $QF = b \to PE$.

14. Un estado temporizado puede tener cotas temporales «$< T$» y «$t <$». Explique el significado de dichas cotas.

    Solución COMPLETAR.

2. Ejercicios prácticos

2.1. Electroencefalogramas

2.1.1. Requerimientos

Un sistema debe controlar un aparato para efectuar electroencefalogramas simples. El análisis consiste en estudiar el voltaje que emiten 10 electrodos que permiten conocer la actividad bioeléctrica cerebral (cada uno comunica un valor al sistema). Es necesario tomar 5 muestras por segundo, espaciadas uniformemente. En cada una de las 5 muestras se lee el valor de los 10 electrodos. Notar que si el cerebro del paciente no presenta actividad en las cercanías de un electrodo, este no emitirá señal alguna. Por lo tanto, el sistema no puede esperar indefinidamente por la señal del electrodo. Finalmente, el sistema debe enviar secuencialmente a una impresora el valor obtenido en cada electrodo (que haya retornado uno o nada en caso de que no se haya registrado ninguno).

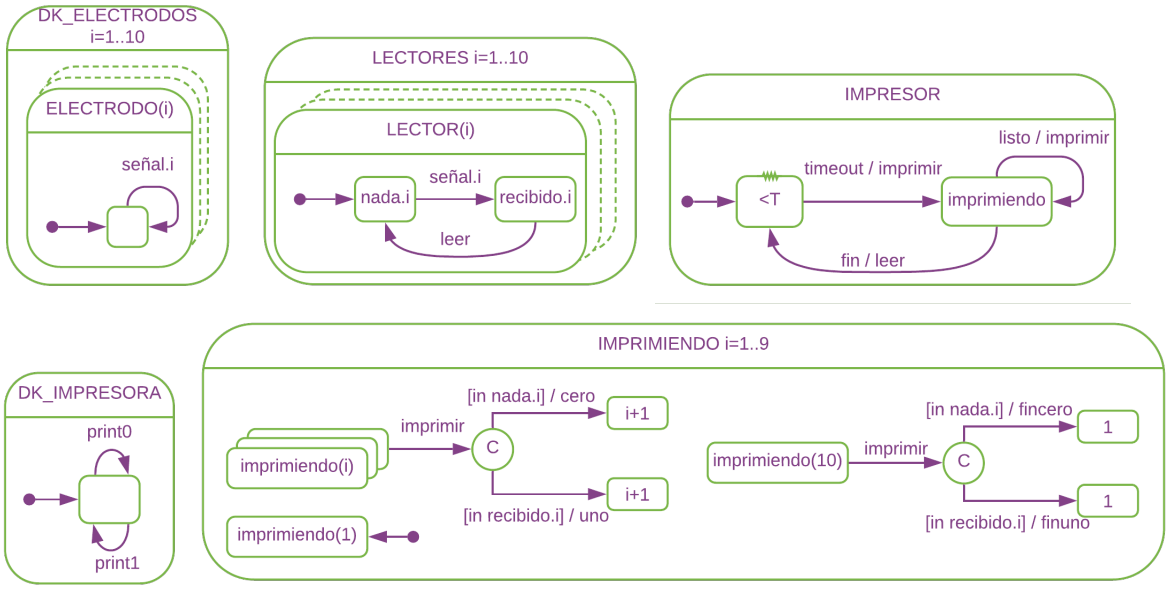
2.1.2. Designaciones

- El electrodo i emite una señal $\approx$ *señal.i*.
- Se toma una muestra de los 10 electrodos $\approx$ *leer*.
- Se terminó de imprimir los 10 electrodos $\approx$ *fin*.
- Se imprime el electrodo actual $\approx$ *imprimir*.
- Se terminó de imprimir el electrodo actual $\approx$ *listo*.
- La impresora imprime un cero $\approx$ *print*0.
- La impresora imprime un uno $\approx$ *print*1.

2.1.3. Tabla de control

Evento	Control	Compartido
señal.i	*EC*	☑
leer	*MC*	☒
*print*0	*MC*	☑
*print*1	*MC*	☑

2.1.4. Statecharts



- $cero = print0 \wedge listo.$
- $fincero = print0 \wedge fin.$
- $uno = print1 \wedge listo.$
- $finuno = print1 \wedge fin.$

$SISTEMA = DK\_ELECTRODOS \parallel DK\_IMPRESORA \parallel LECTORES \parallel IMPRESOR \parallel IMPRIMIENDO$

2.1.5. CSP (COMPLETAR)

2.2. Computación distribuida (COMPLETAR)

2.2.1. Requerimientos

Un sistema distribuido consta de N computadoras cada una de las cuales ejecuta el mismo programa; estas computadoras pueden estar encendidas o apagadas. Este programa recibe dos eventos (en cada computadora) desde el exterior: a y b.

- Cuando el programa en una computadora recibe a, debe comunicarles a todas las otras computadoras por medio de un evento interno, que lo ha recibido.

  Luego de comunicar esta situacion esa computadora debe esperar a que todas le respondan. Cada computadora encendida responde con un evento diferente pero solo si esta en su estado inicial. Como puede haber computadoras apagadas o que estén haciendo otra cosa, el programa esperara un tiempo B cada respuesta.

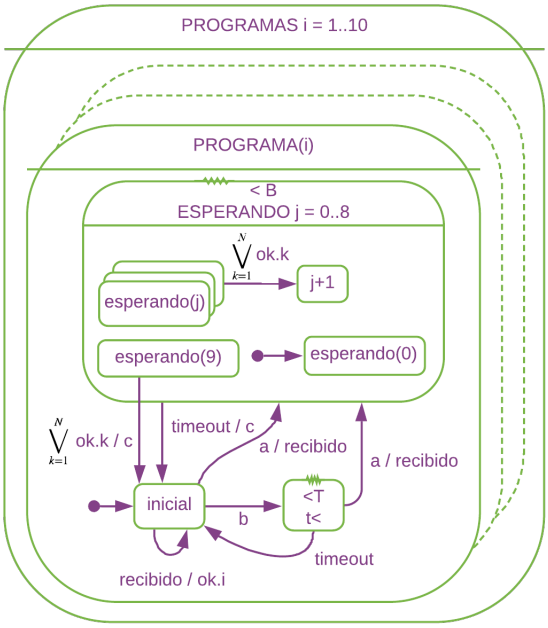
  Cuando recibe todas las respuestas o han transcurrido B unidades de tiempo, debe emitir el evento c y volver al estado inicial.

- Si el programa recibe el evento b debe esperar un tiempo t hasta poder recibir el evento a, a menos que se de otro evento b luego de T unidades de tiempo en cuyo caso debe reiniciar el tiempo de espera.

2.2.2. Designaciones (COMPLETAR)

2.2.3. Tabla de control (COMPLETAR)

2.2.4. Statecharts



2.2.5. CSP (COMPLETAR)

2.3. Sistema de alarma (COMPLETAR)

2.3.1. Requerimientos

Un sistema de alarma hogareño simple consta de un teclado (contiene los 10 dígitos y las dos primeras letras del abecedario), dos sensores de movimiento y dos magnéticos (para aberturas) y una bocina. Todo el sistema será controlado por un programa.

El sistema puede estar armado o no. Armado significa que si se detecta una intrusión debe sonar la bocina hasta que se teclee el código de seguridad. Si no esta armado no se responderá a las intrusiones. Una intrusión se detecta cuando alguno de los sensores envía una señal. Si esta armado, al teclear el código de seguridad el sistema se desarma; si esta desarmado, al teclear el código de seguridad, se arma. El código de seguridad viene de fabrica y consta de 2 dígitos.

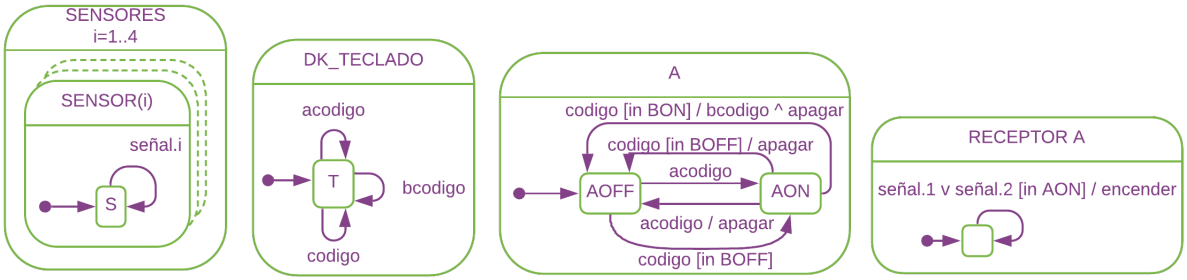
El sistema agrupa un sensor de movimiento y uno magnético en el sector *A* y a los sensores restantes en el sector *B*. Cada sector se puede armar independientemente del otro. Para hacerlo se teclea su letra y el código de seguridad. Si un sector no esta armado no se responderá a las intrusiones en ese sector. Si solo se teclea el código de seguridad se arman todos los sectores; si algún sector esta armado y se teclea el código de seguridad, se desarman todos los sectores.

Abstraiga convenientemente el teclado y el código de seguridad.

2.3.2. Designaciones

2.3.3. Tabla de control

2.3.4. Statecharts



2.3.5. CSP

2.4. Detección de errores (COMPLETAR)

2.4.1. Requerimientos

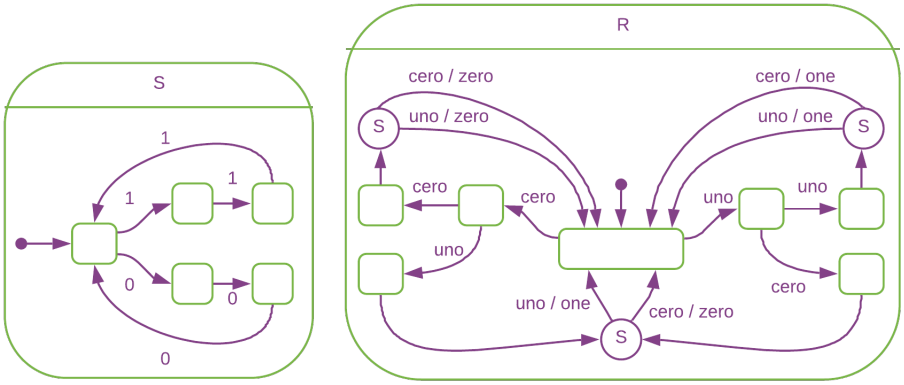
Se cuenta con un proceso S que envía mensajes, un medio E que los transmite y un proceso R que los recibe. E puede almacenar solo un mensaje a la vez, puede corromper a lo sumo uno de cada tres mensajes consecutivos y, por simplicidad, se supone que cada mensaje es un 0 o un 1; E esta fuera del control del sistema, es decir es parte del entorno. Se espera que S y R colaboren para evitar la corrupción producida por E. Para ello utilizaran un mecanismo simple: S enviara cada mensaje por triplicado y R decidirá cual es el valido de acuerdo a una elección por mayoría simple. Especifique los procesos S, E y R descriptos arriba.

2.4.2. Designaciones

- El proceso S envió un $0 \approx 0$.
- El proceso S envió un $1 \approx 1$.
- El medio E envió un $0 \approx cero$.
- El medio E envió un $1 \approx uno$.
- El proceso R determino que el mensaje enviado es $0 \approx zero$.
- El proceso R determino que el mensaje enviado es $1 \approx one$.

2.4.3. Tabla de control

2.4.4. Statecharts



2.4.5. CSP

2.5. Robot industrial (COMPLETAR)

2.5.1. Requerimientos

Se requiere el software de control para un robot industrial con las siguientes características. El robot puede moverse hacia la izquierda o derecha una cierta distancia y consta de un rociador y termómetro. El robot esta ubicado al borde de una cinta transportadora que acarrea piezas metálicas. El requerimiento básico es que si el sensor detecta que la temperatura de la pieza que esta en su cercanía es superior a *maxtemp* , el robot debe abrir el rociador y debe moverse en el sentido de la cinta hasta que la temperatura de la pieza sea inferior a *maxtemp* . Si el robot llega hasta su límite de desplazamiento debe cerrar el rociador. En cualquier caso debe retornar a la posición inicial. Tener en cuenta lo siguiente:

- El termómetro envía una señal con la temperatura medida.
- El motor del robot envía una señal indicando que se ha alcanzado el extremo derecho o izquierdo.
- El software de control puede encender el motor en uno u otro sentido y detenerlo.

- No se puede poner en funcionamiento el motor en el mismo paso en que se recibe una medición de temperatura.

Abstraiga convenientemente las diferentes temperaturas que puede comunicar el termómetro.

2.5.2. Designaciones

2.5.3. Tabla de control

2.5.4. Statecharts

2.5.5. CSP

Asumiremos que la cinta se mueve de izquierda a derecha, y que la posición inicial se encuentra al tope a la izquierda.

$$DK\_TERMOMETRO = DK\_TERMOMETRO1 \Box DK\_TERMOMETRO2$$
$$DK\_TERMOMETRO1 = \overline{superior} \rightarrow \overline{inferior} \rightarrow DK\_TERMOMETRO1$$
$$DK\_TERMOMETRO2 = \overline{inferior} \rightarrow \overline{superior} \rightarrow DK\_TERMOMETRO2$$
$$DK\_SENSOR = \overline{limiteizq} \rightarrow DK\_SENSOR \mid \overline{limiteder} \rightarrow DK\_SENSOR$$
$$DK\_ROCIADOR = abrir \rightarrow DK\_ROCIADOR \mid cerrar \rightarrow DK\_ROCIADOR$$
$$DK\_MOTOR = derecha \rightarrow DK\_ROBOT \mid izquierda \rightarrow DK\_ROBOT \mid detener \rightarrow DK\_ROBOT$$
$$DK\_ROBOT = DK\_TERMOMETRO \parallel DK\_SENSOR \parallel DK\_ROCIADOR \parallel DK\_MOTOR$$
$$SOFTWARE = \overline{cerrar} \rightarrow \overline{izquierda} \rightarrow limiteizq \rightarrow \overline{detener} \rightarrow EMPEZAR$$
$$EMPEZAR = superior \rightarrow \overline{abrir} \rightarrow \overline{derecha} \rightarrow ESPERAR$$
$$ESPERAR = inferior \rightarrow SOFTWARE \mid limiteder \rightarrow SOFTWARE$$

2.6. Productores y consumidores 1 (COMPLETAR)

2.6.1. Requerimientos

Un productor produce de a un mensaje a la vez a razón de 3 segundos cada uno. Existen otros dos productores que se comportan de la misma forma

pero trabajan con tiempos de 1 y 4 segundos, respectivamente. Todos los productores tienen un error en sus relojes de segundos con $0 < \delta < 1$.

Los mensajes enviados por los tres productores son recibidos por un componente que puede recibir de a un mensaje a la vez. El componente cuenta con un «buffer» con capacidad para MB mensajes. Poner o sacar un mensaje del «buffer» le toma 0,1 segundos. Por otro lado, hay dos consumidores de mensajes que pueden consumir de a un mensaje a la vez cada uno, a razón de 2 segundos por mensaje. El componente comunica los mensajes según una política FIFO.

Tener en cuenta que los productores y consumidores son las entidades activas.

Explique claramente (mostrando un ejemplo si es necesario) si el modelo semántico «broadcast» de Statecharts hace que el sistema completo se comporte de forma diferente a como lo hace la especificación CSP, que responde a un modelo semántico sincrónico.

2.6.2. Designaciones

2.6.3. Tabla de control

2.6.4. Statecharts

2.6.5. CSP

2.7. Balanza de camiones (COMPLETAR)

2.7.1. Requerimientos

Una empresa posee una balanza para pesar camiones cargados con materia prima. El camión debe ubicarse mas o menos sobre el centro de la balanza para que la pesada sea correcta. Con este fin la empresa instalo cuatro sensores en los vértices de un rectángulo imaginario de forma tal que cuando detectan que el camión esta dentro de ese rectángulo, se debe bajar una barrera detrás del camión. Si el camión rebasa alguno de los laterales del rectángulo se enciende una (de dos) luz ubicada delante del camión que indica que lado esta rebasado.

Una vez que el camión esta correctamente ubicado y se bajaron las barreras, el chofer debe deslizar una tarjeta magnética que lo identifica. Si la tarjeta es valida, se activa la balanza. Cuando el pesaje finaliza, se debe imprimir un tique con los datos del conductor y el peso. Luego se levantan las

barreras.

Abstraiga adecuadamente la validación de la tarjeta y la impresión del tique con sus datos.

2.7.2. Designaciones

2.7.3. Tabla de control

2.7.4. Statecharts

2.7.5. CSP

2.8. Sistema de ventilación (COMPLETAR)

2.8.1. Requerimientos

Una habitación posee varias puertas de ingreso/egreso. En cada puerta se han instalado dos sensores ópticos uno al lado del otro (cada uno es semejante a los que se instalan en las puertas de los ascensores para evitar que se cierren si hay alguien entrando o saliendo). Estos sensores envían una señal cada vez que algo interrumpe el haz de luz que se establece entre cada extremo. Como hay dos por puerta es posible determinar si una persona ingresa a la habitación o sale de ella. Notar que esta inteligencia *no* es parte de la funcionalidad de los sensores.

La habitación cuenta con un sistema de ventilación electrónico. Es posible aumentar o disminuir discretamente la cantidad de aire que el sistema hace circular.

Se requiere un programa que aumente/disminuya paulatinamente la circulación de aire desde cero hasta el máximo posible, cada vez que ingresan/egresan tres personas a la sala. Es decir la potencia de circulación debe ser la misma, por ejemplo, si hay 3, 4 o 5 personas en la sala. Ademas, si la habitación permanece vaca por mas de 12 horas se debe encender el sistema de ventilación por 1 hora.

2.8.2. Designaciones

2.8.3. Tabla de control

2.8.4. Statecharts

2.8.5. CSP

2.9. Sistema de ascensores (COMPLETAR)

2.9.1. Requerimientos

Un edificio contará con varios ascensores controlados desde un programa. No puede haber mas de dos ascensores moviéndose al mismo tiempo. Las puertas de los ascensores solo deben abrirse cuando estos están detenidos en un piso. Cada ascensor puede ser llamado desde cada piso mediante un botón. Dentro de cada ascensor hay una botonera para dirigirlo hacia el piso al que se desee ir. En cada piso y en cada túnel hay un sensor que avisa el paso del ascensor por ese piso. Las puertas son manuales.

No tener en cuenta los botones de los pisos.

2.9.2. Designaciones

2.9.3. Tabla de control

2.9.4. Statecharts

2.9.5. CSP

2.10. Dispenser de bebidas (COMPLETAR)

2.10.1. Requerimientos

Una maquina expendedora de bebidas funciona de la siguiente manera. La maquina esta inactiva hasta que se libera una traba de seguridad. Los clientes pueden seleccionar una bebida pulsando el botón correspondiente. Hay cinco bebidas diferentes. Cuando la maquina no esta inactiva se pueden insertar monedas mientras se muestra en una pequeña pantalla el total hasta el momento. (Se aceptan monedas de 25 y 50 centavos y de 1 peso. La maquina detecta el valor de la moneda). Si se selecciona una bebida y el total de dinero es igual al precio del producto, entonces la maquina entrega la bebida correspondiente. Cada bebida puede tener un precio diferente. Si el

total es mayor que el precio del producto, se entrega la bebida y la pantalla se actualiza poniendo como nuevo total la diferencia. La maquina no puede ponerse en modo inactivo si el total mostrado no es cero. El usuario puede pulsar un botón para que la maquina le retorne el total mostrado en pantalla, en cuyo caso el total mostrado se pone a cero.

Aunque no es necesario mostrar el total de dinero ingresado por el usuario, el modelo debe contemplar una abstracción adecuada. También abstraiga adecuadamente el tema de los precios de las bebidas.

2.10.2. Designaciones

2.10.3. Tabla de control

2.10.4. Statecharts

2.10.5. CSP

2.11. Temperatura de automóviles (COMPLETAR)

2.11.1. Requerimientos

La temperatura solo puede ser regulada ajustando el funcionamiento del acondicionador de aire (AA); el AA solo se apaga o se prende, no es posible hacer que el aire salga a mayor o menor temperatura. Entonces, cuanto mas funciona el AA mas se enfría el habitáculo. El conductor puede seleccionar la temperatura que desee mantener por medio de una perilla giratoria. La temperatura deseada puede estar entre 18 y 30 C; si la temperatura deseada es de 18 C, el AA no se detiene nunca. El sistema estará en funcionamiento solo si el conductor ha pulsado un botón de activación; el sistema sale de funcionamiento cuando el botón se vuelve a pulsar o el motor se detiene. Existen dos termómetros dentro del habitáculo (uno en la parte delantera y el otro en la parte trasera) que, una vez encendido el sistema, envían señales de incremento o decremento de la temperatura sensada. Los termómetros inicialmente se encuentran a 18 C; los termómetros no envían señales si la temperatura excede su rango de funcionamiento.

2.11.2. Designaciones

2.11.3. Tabla de control

2.11.4. Statecharts

2.11.5. CSP

2.12. Minicomponente musical (COMPLETAR)

2.12.1. Requerimientos

Un software controlara el funcionamiento de un minicomponente musical. El equipo consta de un reproductor de CD y radio AM/FM. El panel de control cuenta con la interfaz de usuario habitual: encendido/apagado, «play», «prev», «next», «stop, «pause», «eject» y selector CD-radio. Los botones como «play», «prev», etc. sirven para todos los componentes dependiendo del selector (por ejemplo, «prev» sirve para mover el sintonizador de la radio hacia las estaciones mas bajas). Si el CD esta seleccionado y se pulsa de forma continua por 2 segundos o mas cualquiera de los botones «prev» o «next», entonces se debe avanzar dentro de la misma canción y no saltar de canción. En la misma situacion si se pulsan dos veces con diferencia de menos de 1 décima de segundo, entonces se pasa a la ultima o primera pista. Ademas, hay un sensor que detecta la presencia de un CD. Si se pulsa el botón de apagado mientras el CD esta en funcionamiento primero se debe apagar el CD y luego el equipo. El minicomponente cuenta con una memoria que permite que al re-encenderse se active el ultimo componente usado, en particular si es la radio se encenderá AM o FM.

Las ordenes recibidas desde los botones, si el CD esta activado, se deben traducir en ordenes al motor (giro normal o giro rápido hacia atrás o adelante, parar, arrancar, expulsar) o el lector laser (encender, apagar, leer).

Las ordenes recibidas desde los botones, si la radio esta activada, se deben traducir en ordenes a un sintonizador (aumentar/disminuir frecuencia y cambiar rango de frecuencia).

2.12.2. Designaciones

2.12.3. Tabla de control

2.12.4. Statecharts

2.12.5. CSP

2.13. Productores y consumidores 2 (COMPLETAR)

2.13.1. Requerimientos

Un proceso B, debe almacenar elementos en dos *buffers* de la misma capacidad finita y conocida N. Por otro lado, existen procesos (llamados productores) que envían datos a B para que este los almacene en los *buffers*, y existen procesos (llamados consumidores) que le piden a B los datos que tiene almacenados (lo que hace que los *buffers* se vayan vaciando).

Cuando un consumidor quiere un dato que B tiene, el proceso le debe indicar el *buffer* del cual lo quiere; en cambio los productores no pueden seleccionar el *buffer* .

Obviamente B no puede poner elementos en un *buffer* lleno y no puede sacar elementos de un *buffer* vacío. Lo que sí debe hacer B es balancear el uso de los *buffers*: cuando almacena datos lo debe hacer en el *buffer* mas vacío y si un *buffer* se esta vaciando mas rápido que el otro, debe pasar elementos del ultimo al primero.

2.13.2. Designaciones

2.13.3. Tabla de control

2.13.4. Statecharts

2.13.5. CSP

$$DK\_PRODUCTORES = send!d \rightarrow DK\_PRODUCTORES$$
$$DK\_CONSUMIDORES = buffer!b \rightarrow receive?d \rightarrow DK\_CONSUMIDORES$$
$$DK = DK\_PRODUCTORES \parallel DK\_CONSUMIDORES$$

$$BUFFER(0, \langle\rangle) = push?x \rightarrow BUFFER(1, \langle x \rangle)$$
$$\mid size!0 \rightarrow BUFFER(0, \langle\rangle)$$
$$BUFFER(n, ys) = NOTFULL(n, ys) \lceil n < N \rceil FULL(ys)$$

$$
\begin{aligned}
NOTFULL\,(n, y : ys) &= push?x \rightarrow BUFFER\,(n+1, x : y : ys) \\
&\quad \mid pop!y \rightarrow BUFFER\,(n-1, ys) \\
&\quad \mid size!n \rightarrow BUFFER\,(n, y : ys) \\
FULL\,(y : ys) &= pop!y \rightarrow BUFFER\,(N-1, ys) \\
&\quad \mid size!N \rightarrow FULL\,(y : ys) \\
BUFFERS &= \|_{i=1}^{2}\ i : BUFFER\,(0, \langle\rangle)
\end{aligned}
$$

$$
\begin{aligned}
PUSHER &= send?d \rightarrow 1.size?s1 \rightarrow 1.size?s2 \\
&\quad \rightarrow (1.push!d \rightarrow PUSHER\,\lceil s1 < s2 \rceil\ 2.push!d \rightarrow PUSHER) \\
POPER &= buffer?b \rightarrow b.pop?d \rightarrow receive!d \rightarrow BALANCER \\
BALANCER &= 1.size?s1 \rightarrow 1.size?s2 \\
&\quad \rightarrow LEFT\,\lceil s1 - s2 > 1 \rceil\,(RIGHT\,\lceil s2 - s1 > 1 \rceil\ POPER) \\
LEFT &= 1.pop?x \rightarrow 2.push!x \rightarrow BALANCER \\
RIGHT &= 2.pop?x \rightarrow 1.push!x \rightarrow BALANCER \\
B &= PUSHER \parallel POPER
\end{aligned}
$$

$$SISTEMA = DK \parallel B \parallel BUFFERS$$

2.14. Cinta Transportadora (COMPLETAR)

2.14.1. Requerimientos

Una cinta transportadora acarrea ítems de dos tipos (A y B) que luego son capturados por las maquinas M_A y M_B, respectivamente. Un software controla la cinta y ambas maquinas. El software debe encender cada una de las maquinas M_A y M_B, mediante los eventos *turnona* y *turnonb*, respectivamente. Cada maquina necesita r unidades de tiempo para estar operativa.

Cada vez que en la cinta se detecta un ítem de un tipo, el software debe emitir el evento *takea* o *takeb* para que la maquina correspondiente capture el ítem en cuestión, a menos que a continuación se indique otra cosa.

Si entre la aparición de dos o mas ítems transcurren menos de n unidades de tiempo, no se debe emitir la orden de captura correspondiente a los ítems posteriores al primero; se deberá emitir las ordenes de captura cuando aparezca un ítem n unidades de tiempo después del anterior.

Si transcurren mas de t unidades de tiempo entre dos ítems, se debe apagar todo el sistema mediante el evento *abort*.

2.14.2. Designaciones

2.14.3. Tabla de control

2.14.4. Statecharts

2.14.5. CSP

2.15. Protocolo CSMA/CD (COMPLETAR)

2.15.1. Requerimientos

Para transmitir datos entre terminales de trabajo conectadas en red se debe hacer uso de algún protocolo. En algunas redes *broadcast* con un único bus la clave está en como asignar el uso de este cuando varias terminales compiten por él.

Uno de los protocolos que resuelven esta cuestión es el *Carrier Sense, Multiple Access with Collision Detection*, o simplemente CSMA/CD. Una breve descripción del funcionamiento de este protocolo es la siguiente.

Una terminal transmite al sistema (es decir a la implementación del protocolo) un mensaje que debe ser enviado por la red. El sistema, si el bus esta disponible (esto es, no hay otra terminal transmitiendo), comienza a enviar su mensaje. Sin embargo, si detecta que el bus esta ocupado, espera un tiempo aleatorio y vuelve a intentar transmitir el mensaje. Esto lo hará tantas veces como sea necesario hasta que pueda empezar a transmitir.

Aun tomando estas precauciones puede ocurrir que dos terminales usen el bus al mismo tiempo, lo que da lugar a una *colisión*. Cuando una colisión ocurre el bus comunica esta situacion a todas las terminales. Esto implica que todas las terminales abortan inmediatamente las transmisiones y, nuevamente, esperan un tiempo aleatorio para empezar a transmitir de nuevo.

Los mensajes que colisionan se pierden. Una vez que el mensaje ha sido transmitido, la terminal que inició la transmisión es notificada.

Se supone que todos los mensajes (sin importar tamaño) demoran exactamente λ unidades de tiempo en ser transmitidos.

En resumen, en cada terminal corre una implementación del protocolo y todas las terminales comparten el bus. La interfaz que provee el bus consta de: determinar si el bus esta libre o no, enviar un mensaje, comunicar que un mensaje se transmitió, comunicar a todas las terminales que hay colisión.

2.15.2. Designaciones

2.15.3. Tabla de control

2.15.4. Statecharts

2.15.5. CSP

2.16. Teléfono celular (COMPLETAR)

2.16.1. Requerimientos

Se trata del software que controla un teléfono celular muy simple. El teléfono cuenta con un teclado numérico, una tecla «enviar» y otra «cortar». Ademas posee una bocina que puede ser encendida o apagada y una pantalla que puede mostrar una cadena de dígitos; existe una operación que borra el contenido de la pantalla. El teléfono se enciende si se pulsa la tecla «cortar» durante mas de 2 segundos. Una vez encendido el teléfono espera que el usuario teclee un numero o que llegue una llamada.

Los números a los cuales se puede llamar poseen una cantidad de dígitos variable. Cada vez el usuario pulsa un digito el numero correspondiente debe mostrarse en la pantalla. Una vez que el usuario pulsa «enviar» el sistema debe esperar a que se pulse «cortar» (no se modela la comunicación en si). Mientras tanto no se pueden recibir llamadas, es decir se pierden. Cuando se pulsa «cortar» se debe borrar la pantalla.

Si el teléfono puede recibir una llamada y esto ocurre, entonces el sistema debe hacer sonar la bocina de manera intermitente hasta que el usuario pulse «enviar» o «cortar». La intermitencia es: 1 segundo de ruido seguido de un segundo de silencio. Si el usuario pulsa «cortar» el teléfono podrá recibir o hacer nuevas llamadas. Si el usuario pulsa «enviar» no se podrán hacer ni recibir nuevas llamadas hasta que pulse «cortar». Si el numero que se recibe coincide con uno que se mantiene en la agenda del teléfono, entonces se debe mostrar en la pantalla el nombre que le corresponda.

2.16.2. Designaciones

2.16.3. Tabla de control

2.16.4. Statecharts

2.16.5. CSP

2.17. Sistema de estadísticas (COMPLETAR)

2.17.1. Requerimientos

El dispositivo D debe recibir datos de N sensores conectados en serie, y con esos datos realizar diversas estadísticas. El mecanismo debe ser el siguiente: D debe pedir al primer sensor que le envíe el dato que tenga disponible. Este sensor, a su vez, luego de enviar el dato le «avisa» al siguiente sensor que es su turno, y as sucesivamente hasta el sensor N. D espera un cierto tiempo a que arribe el siguiente dato, tras el cual considera que no existen mas sensores (el hecho de que estén en serie y que se «avisen» entre ellos hace que N sea transparente para D).

Cuando D termina de recibir los valores debe enviarle el promedio de los valores recibidos a otro dispositivo, y comenzar el ciclo nuevamente en cuyo caso se reinicia el arreglo de sensores de manera tal que el primero es el que posiblemente enviara el primer valor en el nuevo ciclo. En caso de no haber recibido ningún valor luego de efectuar el primer pedido (porque N es cero o porque el primer sensor se demoro mas de lo esperado), debe enviarle al otro dispositivo un mensaje de error.

Considerar que la maquina a construir es el software que controla a D.

2.17.2. Designaciones

2.17.3. Tabla de control

2.17.4. Statecharts

2.17.5. CSP

2.18. GUI (COMPLETAR)

2.18.1. Requerimientos

Una interfaz gráfica de usuario (GUI) consta de tres pantallas: A, B y C. El usuario puede seleccionar cualquiera de ellas en cualquier orden, pero si

durante más de N segundos permanece en cualquier pantalla sin interactuar con el sistema, este mostrará la pantalla A.

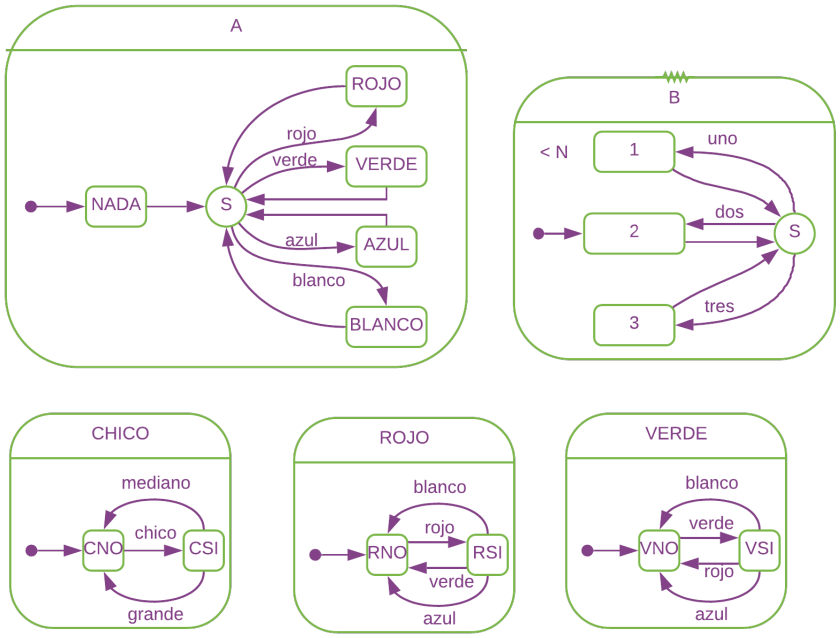
Cada vez que el sistema muestra una pantalla, debe dibujar la última selección del usuario. En cada pantalla el usuario puede hacer lo siguiente:

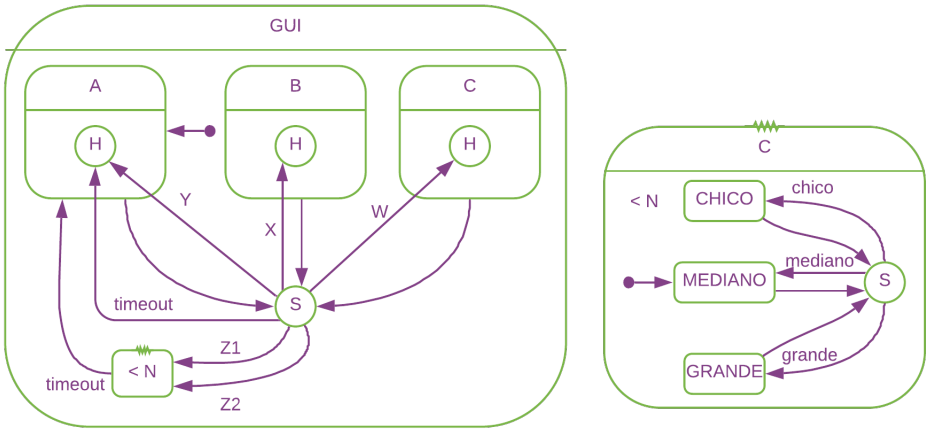
- A: Seleccionar uno de cuatro rectángulos cada uno de un color (rojo, verde, azul y blanco). Inicialmente no hay ninguno seleccionado.
- B: Si el rectángulo seleccionado en A es verde o rojo y el tamaño seleccionado en C es chico, entonces puede seleccionar un numero entre 1 y 3. En cualquier otro caso no puede hacer nada.
- C: Si el rectángulo seleccionado en A es el verde, entonces el usuario no puede seleccionar nada en esta pantalla. En cualquier otro caso puede aumentar o disminuir el tamaño del rectángulo seleccionado en A, eligiendo un elemento de la siguiente escala: chico, mediano o grande.

2.18.2. Designaciones

2.18.3. Tabla de control

2.18.4. Statecharts





- $Y \equiv a/dibujara.$
- $X \equiv b\,[in\,CSI \wedge (VSI \vee RSI)]\,/dibujarb.$
- $W \equiv c\,[in\,VNO]\,/dibujarc.$
- $Z1 \equiv b\,[in\,CNO \vee (VNO \wedge RNO)]\,/dibujarb.$
- $Z2 \equiv c\,[in\,VSI]\,/dibujarc.$
- $S \equiv GUI \parallel CHICO \parallel ROJO \parallel VERDE.$

2.18.5. CSP