Tema 6: REQUISITOS DE CORRECCIÓN

Ingeniería de Protocolos y Servicios

6.1. Introducción

El lenguaje SDL nos permite desarrollar modelos de validación. Una vez diseñado un modelo, nos podemos preguntar: ¿Es correcto?

Para responder esta pregunta, hay que establecer antes que significa “correcto”.

En este tema se estudiarán distintos criterios de corrección.

DTIC
Área Ingeniería Telemática

francesc.burrull@upct.es

6.2. Razonamientos sobre comportamiento

El comportamiento de un modelo queda definido completamente por el conjunto de todas las posibles secuencias de ejecución.

Una secuencia de ejecución es un conjunto finito y ordenado de estados.

Un estado queda definido por el conjunto de los valores de las variables locales y globales, los puntos de ejecución de los procesos y el contenido de los canales.

6.2. Razonamientos sobre comportamiento

Un modelo de validación puede alcanzar un estado dado ejecutando distintas instrucciones, así como puede ser colocado en un determinado estado asignando los valores adecuados a las variables, los puntos de control de los procesos y los canales.

Ahora bien, una colección de estados cualquiera no tiene porque ser una secuencia de ejecución válida.

6.2. Razonamientos sobre comportamiento

Para que una colección de estados sea una secuencia de ejecución válida del modelo M se debe dar:

• El primer estado de la secuencia (p. ej. 1) es el estado inicial de M, con todas las variables inicializadas a 0, los canales vacíos, y un solo proceso activo y en su estado inicial.

• Si M se encuentra en el estado i, existe al menos una instrucción ejecutable que que puede llevar M del estado i al estado i+1.

6.2. Razonamientos sobre comportamiento

Dentro de las secuencias posibles hay 2 tipos que son de especial interés:

• Terminación: Una secuencia es de terminación si no se repite ningún estado en ella y el modelo M no tiene ninguna instrucción ejecutable cuando se le sitúa en el último estado de la secuencia.

• Cíclica: Una secuencia es cíclica si todos sus estados menos el último son distintos, y el último estado de la secuencia es igual a uno de los anteriores.

Tema 6: REQUISITOS DE CORRECCIÓN

Ingeniería de Protocolos y Servicios

6.2. Razonamientos sobre comportamiento

Todas las secuencias de ejecución de terminación y cíclicas que se pueden generar con un modelo M definen el “comportamiento del sistema” del modelo M.

La unión de todos los estados incluidos en el comportamiento del sistema se llama “conjunto de estados alcanzables” del modelo M.

DTIC
Área Ingeniería Telemática

francesc.burrull@upct.es

Tema 6: REQUISITOS DE CORRECCIÓN

Ingeniería de Protocolos y Servicios

6.3. Aserciones

Los criterios de corrección pueden expresarse como condiciones booleanas que deben ser satisfechas cuando un proceso alcanza un determinado estado.

DTIC
Área Ingeniería Telemática

francesc.burrull@upct.es

6.4. Invariantes de Sistema

Son una generalización de las aserciones. Los invariantes de sistema son, pues, condiciones booleanas que se mantienen durante toda la ejecución del programa, independientemente del camino de ejecución que este siga.

6.5. Bloqueos

El bloqueo es un estado que alcanza el sistema, a partir del cual cualquier ejecución es imposible.

Una secuencia de bloqueo es un caso particular de secuencia de terminación (normalmente no deseada).

6.6. Ciclos indeseados

Un ciclo indeseado o *livelock* es un ciclo de ejecución del programa donde a pesar de que se van ejecutando instrucciones, el programa concurrente no avanza en la tarea que supuestamente debe realizar.

Un ciclo indeseado es un caso particular de secuencia cíclica (normalmente no deseada).

6.7. Proclamaciones temporales

Una proclamación temporal especifica un cierto orden temporal en las propiedades de los estados.

P->Q : Significa cada estado donde la propiedad P es verdadera irá seguido por un estado donde la propiedad Q es verdadera.

DTIC
Área Ingeniería Telemática
francesc.burrull@upct.es

6.8. Ejemplo clásico: El problema de los 5 filósofos.

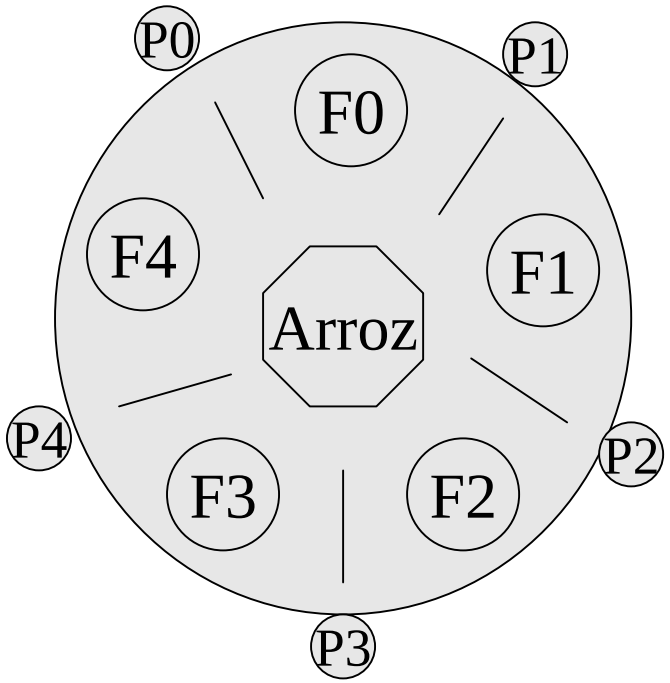
El problema de los 5 filósofos es un problema clásico de la literatura de sistemas concurrentes.

Consiste en 5 filósofos chinos sentados alrededor de una mesa circular, intercalando entre cada 2 filósofos un palillo para comer. En el centro de la mesa hay una fuente de arroz. Un filósofo realiza 2 tareas de manera cíclica: Pensar y comer.

Tema 6: REQUISITOS DE CORRECCIÓN

Ingeniería de Protocolos y Servicios

6.8. Ejemplo clásico: El problema de los 5 filósofos.



DTIC
Área Ingeniería Telemática

francesc.burrull@upct.es

6.8. Ejemplo clásico: El problema de los 5 filósofos.
Con distintas soluciones del problema se puede ver:

- Bloqueo: Ningún filósofo -proceso- logra comer -avanzar- (S1).
- *Livelock*: Secuencia donde no se avanza (S2).
- Inanición: Un filósofo -proceso- puede quedarse sin comer si sus vecinos así lo deciden (S2).
- Paso de mensajes: Los filósofos -procesos- pasan a comer enviando palillos -mensajes- (S4, S5) .

6.8. Ejemplo clásico: El problema de los 5 filósofos.

• Sincronización de condiciones (S*): Un filósofo no puede comer hasta tener ambos palillos.

• Exclusión Mutua (S*): Cada palillo solo puede ser utilizado por un filósofo a la vez.

• Variables Compartidas (S*): Un palillo puede verse como una variable que va a ser compartida por 2 filósofos.

• Espera activa (S2): Un filósofo puede quedarse a la espera de un palillo continuamente preguntando.

6.8. Ejemplo clásico: El problema de los 5 filósofos.
Solución 1 (S1), memoria compartida. Con bloqueo.
Cada filósofo se representa con un proceso.
Cada palillo con una variable booleana { libre, ocupado }
Las reglas de procedimiento son coger el palillo de la derecha y luego el de la izquierda para comer. Liberarlos para pensar.

6.8. Ejemplo clásico: El problema de los 5 filósofos.
Solución 2 (S2), memoria compartida. Con *livelock*.
Cada filósofo se representa con un proceso.
Cada palillo con una variable booleana { libre, ocupado }
Las reglas de procedimiento son coger el palillo de la derecha y luego el de la izquierda para comer. Si se ha cogido el palillo derecho y el izquierdo esta ocupado, liberar el derecho y reintentar. Liberarlos para pensar.

6.8. Ejemplo clásico: El problema de los 5 filósofos. Solución 2 (S2), memoria compartida. Con bloqueo. Esta solución puede llevar a inanición a un filósofo, si sus vecinos comen de manera alternada, y nunca pensando de manera concurrente.

DTIC
Área Ingeniería Telemática

francesc.burrull@upct.es

6.8. Ejemplo clásico: El problema de los 5 filósofos.
Solución 3 (S3), memoria compartida. Sin bloqueo.
Cada filósofo se representa con un proceso.
Cada palillo con una variable booleana { libre, ocupado }
Las reglas de procedimiento son coger el palillo de la derecha y luego el de la izquierda para comer, excepto el filósofo 0, que es zurdo. Liberarlos para pensar.

6.8. Ejemplo clásico: El problema de los 5 filósofos. Solución 4 (S4), paso de mensajes. Sin bloqueo. Solución centralizada.

Cada filósofo se representa con un proceso. Hay un proceso que distribuye los palillos (camarero).

Las reglas de procedimiento son mandar un mensaje al camarero para pasar a comer. Éste responderá cuando los palillos del filósofo en cuestión queden libres. De manera equivalente, mandar un mensaje para liberarlos y pensar.

francesc.burrull@upct.es

Tema 6: REQUISITOS DE CORRECCIÓN

Ingeniería de Protocolos y Servicios

6.8. Ejemplo clásico: El problema de los 5 filósofos. Solución 4 (S4), paso de mensajes. Sin bloqueo. Solución distribuida.

DTIC
Área Ingeniería Telemática

francesc.burrull@upct.es

6.9. Resumen

Se han descrito distintos criterios de corrección, relacionándose éstos con subconjuntos del espacio de estados alcanzables.

También se han introducido los conceptos de aserciones, invariantes de sistema, bloqueo, *livelock* y proclamaciones temporales.

Finalmente se ha visto la problemática de los sistemas concurrentes mediante un ejemplo clásico.