Consejos, conceptos y vocabulario sobre requerimientos y especificaciones

Designaciones sin tipo

- Las reglas de reconocimiento nunca brindan información sobre el tipo del término designado, sin importar el vocabulario que se use.
  - p es una pieza $\approx Pieza(p)$
  - La pieza p tiene temperatura $t \approx TempP(p, t)$
  - $\neg\ (TempP(p,t) \Rightarrow Pieza(p))$
  - $\forall p,t \cdot Pieza(p) \wedge Temp(t) \wedge TempP(p,t) \Rightarrow t > 0$

"Todos" peligrosos (1)

- En general las sentencias *indicativas cuantificadas universalmente* no son ciertas.
  - "Cada auto tiene una única patente"
  - ¿Qué pasa si por una u otra razón:
    - un auto tiene más de una patente
    - a un auto no se le ha asignado una patente
    - una persona reporta un número de patente incorrecto
    - una persona reporta el número de patente de otro auto?

"Todos" peligrosos (2)

- Como las sentencias indicativas pertenecen al DK, forman parte de lo que el IS asume a la hora de construir el sistema.
- Por lo tanto, si se asume una sentencia indicativa cuantificada universalmente como cierta cuando en realidad no lo es, el sistema será incapaz de cumplir con su especificación cada vez que se le presente un caso para el cual la cuantificación es falsa.

"Todos" peligrosos (3)

- En consecuencia, el IS debe dudar y someter a experimentos, pruebas, etc. cualquier sentencia indicativa cuantificada universalmente que el cliente le presente como cierta.
- Por otro lado, las sentencias *optativas* cuantificadas universalmente son razonables y deseables.
  - El sistema de patentes debe responder correctamente a cada uno de los casos raros mencionados.

Especificar: abstraer + describir

- Aprenda a especificar:
  - trate de NO pensar como un programador;
  - trate de NO pensar computacionalmente, olvídese de la computadora... olvídese que existen;
  - describa sólo los fenómenos esenciales de la interfaz entre el entorno y el sistema.
- Sea claro: haga que las verdades sean obvias.
- Utilice el principio de *separación de conceptos* para *modularizar* la especificación.

Especificar: construir modelos

- En muchos casos la especificación constituye un *modelo analógico* de la interacción entre el sistema y el entorno.
- Estos modelos son *analogías* de las cosas que modelan, es decir, establecen relaciones entre los *individuos* que viven en el modelo y aquellos que viven en la realidad.
- Comparten ciertas propiedades y estructura con el mundo real.

Propiedades de las variables de estado

- La más importante es el *tipo*.
- Para variables cuyo tipo es “estructurado” (por ejemplo, listas) se debe tener en cuenta:
  - Orden, ¿es necesario resaltar algún orden?
  - Duplicados, ¿puede haber elementos duplicados?
  - Cotas, ¿están o no acotados los elementos del tipo?
  - Acceso, ¿los elementos son recuperados por un índice o una clave?
  - Forma, ¿estructura lineal, jerárquica, acíclica, etc.?

Propiedades de las... (2)

- Para variables cuyo tipo es relacional/funcional:
  - ¿Función o relación?
  - ¿Total o parcial?
  - ¿Suryectiva, inyectiva o biyectiva?
  - ¿Finita o infinita?
  - ¿Es idempotente?
  - ¿Es reflexiva, simétrica o transitiva?
  - ¿Es conmutativa, asociativa, distributiva?

Invariantes

- Propiedad que no cambia a medida que el sistema pasa de un estado a otro.
  - Es simplemente un predicado en el cual intervienen sólo las variables de estado.
  - Si *True* es el invariante más fuerte, revise la especificación.
  - Pensar largamente sobre el invariante puede llevar a cambiar radicalmente el sistema. ¡Hágalo!

Precondiciones

- Supuestos sobre el entorno efectuados por las operaciones que realizan las transiciones de estado en una máquina de estados.
- La operación no está en control de tales supuestos; el entorno puede verificarlos o no.
- ¿Qué sucede si una precondición no se cumple?
  - Depende del significado que cada lenguaje asigna a las precondiciones.

Invariante y precondición (1)

- Existen al menos dos formas de registrar el invariante en una especificación:
  - restringiendo, por definición, los estados de la máquina a aquellos que verifican el invariante; o
  - especificando las operaciones de la máquina de forma tal que preserven el invariante.
- La primera forma es el estilo tradicional en Z, la segunda es más común en TLA.

Invariante y precondición (2)

- En el primer caso la precondición de cada operación puede quedar implícita y hay que analizarla para documentarla adecuadamente; el IS puede ahorrarse algunas precondiciones.
- En el segundo, hay que elegir muy bien la precondición de cada operación y demostrar que así preserva el invariante.

Postcondiciones

- Supuestos sobre la interfaz entorno/máquina efectuados por el entorno luego de una transición de estado.
- La operación que realizó el cambio de estado (más el DK apropiado) es quien controla que tales supuestos se verifiquen, si su precondición estaba satisfecha.
- La postcondición debe describir el efecto observable sobre las variables de estado y de salida producto de una transición de estado.

Usar no-determinismo

- La introducción de no-determinismo es una técnica efectiva de abstracción.

Errores/excepciones/fallas

- Es tan importante especificar el comportamiento erróneo o excepcional como el normal.
- Existe una relación muy estrecha entre precondiciones y comportamiento erróneo: cada precondición no satisfecha puede llevar al sistema a comportarse de forma anómala.
- Por lo tanto, es necesario analizar las precondiciones para especificar los errores.

Operaciones atómicas

- Debe quedar claramente documentado cuáles de las operaciones son consideradas atómicas a nivel de la especificación pues luego deben ser implementadas de tal forma.

¿Qué podría suceder si se supuso que *Extraer* era atómica pero no se implementó de tal forma?