Invariante de Representación y Función de Abstracción

Clase práctica

Algoritmos y Estructuras de Datos II

10 de septiembre de 2019

Diferencias entre especificación y diseño

- En la etapa de especificación:
  - Nos ocupamos del '¿Qué?'
  - Lo explicamos usando TADs
  - Ese '¿Qué?' lo explicamos bajo el paradigma funcional
- En la etapa de diseño:
  - Nos ocupamos del '¿Cómo?'
  - Lo explicamos con *módulos de abstracción*
  - Ese '¿Cómo?' lo explicamos usando el paradigma imperativo
  - Tenemos un *contexto de uso* que nos fuerza a tomar decisiones respecto de la estructura.

Partes de un módulo

- **Interfaz**: las operaciones (servicios) que exporta el módulo, mediante las cuales de interactúa con sus instancias. Incluye las restricciones, complejidades, etc.
- **Representación**: la forma en la que se representa y cómo funciona una instancia internamente, usando otros módulos.
- **Servicios usados**: los servicios exportados por otros módulos que se usan en la representación del actual.

En esta primera parte, nos vamos a centrar en formalismos sobre la **representación** de los módulos. El resto lo veremos más adelante.

Representación: mini ejemplo

Queremos un módulo que implemente el TAD Horario. El módulo se llamará Reloj*, y las instancias del módulo tendrán género `reloj`.

TAD Horario

Ig. obs.: $(\forall H_1, H_2 : hor)(H_1 =_{\text{obs}} H_2 \Leftrightarrow (h(H_1) = h(H_2) \wedge m(H_1) = m(H_2)))$

observadores básicos

$min : \text{hor} \longrightarrow \text{nat}$

$hr : \text{hor} \longrightarrow \text{nat}$

generadores

$nuevoHorario : \text{nat } h \times \text{nat } m \longrightarrow \text{hor} \qquad \{0 \leq h \leq 23 \wedge 0 \leq m \leq 59\}$

axiomas

$min(nuevoHorario(h, m)) \equiv m$

$hr(nuevoHorario(h, m)) \equiv h$

Fin TAD

* No le ponemos de nombre "Horario" para mostrar que no necesariamente debe ser así. De hecho, pueden haber varios módulos que implementen un mismo TAD, y con diferencias considerables.

Representación del Módulo Reloj

Propongo la siguiente representación para una instancia de `reloj`:

`reloj` se representa con `estr`,
donde `estr` es `tupla(nat, nat)`

Noten cómo combinamos otros géneros para construir uno nuevo "más complejo".

- Formalmente, ¿qué tiene que ver este `estr` con nuestro *hor*?
- ¿Cómo podemos hablar de instancias del Módulo Reloj como si fuesen del TAD Horario?

Vinculación especificación y diseño (paréntensis teórico)

Los TADs y los módulos “no viven en el mismo universo”. Una instancia del módulo *Pila(α)* no es lo mismo que una instancia del $\mathrm{TAD}\ \textsc{Pila}(\alpha)$, pero quisiéramos poder establecer alguna equivalencia.

Para eso está la función $\widehat{\bullet}$, que toma (en este caso) una instancia del módulo *Pila(α)*, y nos devuelve su instancia equivalente del $\mathrm{TAD}\ \textsc{Pila}(\alpha)$.

$$\widehat{\texttt{reloj}} \equiv \textit{hor}$$

- `reloj`: una estructura del paradigma imperativo con ciertos valores, más parecido a lo que realmente pasa en la computadora.
- *hor*: una tira de generadores del TAD.

Vinculación especificación y diseño (paréntensis teórico)

Hay géneros de módulos que son, en cierto sentido, *primitivos*; no hay documentación de cómo son internamente, son las unidades atómicas. Su equivalencia en el mundo funcional de los TADs ($\widehat{\bullet}$) la usamos como punto de partida: `nat`, `tupla`, etc.

Construimos/representamos nuevos módulos usando otros módulos "más básicos", que ya sabemos interpretar como instancias de TADs. Aca aparecen:

- La función de abstracción:

$$\textbf{Abs: } \widehat{\texttt{estr}}\ e \rightarrow \textit{TAD representado} \qquad \{Rep(e)\}$$

- El invariante de representación:

$$\textbf{Rep: } \widehat{\texttt{estr}} \rightarrow \textit{boolean}$$

Rep y Abs, con ejemplos

`reloj` se representa con `estr`,
donde `estr` es `tupla(nat, nat)`

Invariante de representación
¿Qué forma tendría que tener un `estr` para que podamos interpretarlo como un `reloj`?

$$\text{Rep: } \widehat{estr} \rightarrow Bool$$
$$\text{Rep(e)} \equiv 0 \leq \Pi_0(e) \leq 23 \wedge 0 \leq \Pi_1(e) \leq 59$$

Función de abstracción
¿Cómo interpretamos a un $\widehat{estr}$ como instancia del TAD Horario?

$$\text{Abs: } \widehat{\texttt{estr}}\ e \rightarrow hor \quad \{Rep(e)\}$$
$$\text{Abs(e)} \equiv nuevoHorario(\Pi_0(e), \Pi_1(e))$$

Rep y Abs, con ejemplos

Universo de los módulos

$$
\begin{tikzcd}[column sep=huge, row sep=huge]
\texttt{reloj} \arrow[r, "\text{se representa con}"] \arrow[d, "\text{se explica con } (\ \hat{\ }\ )"'] & \texttt{estr} = \texttt{tupla(nat, nat)} \arrow[d, "\hat{\ }"'] \\
\mathit{hor} & \mathit{tupla}(\mathit{nat}, \mathit{nat}) \arrow[l, "\mathrm{Abs}"]
\end{tikzcd}
$$

Universo de los TADs

Rep y Abs, un ejemplo de verdad

TAD $\textsc{conjRang}$

lg. obs.: ...

observadores básicos

$\bullet \in \bullet \;:\; \text{nat} \times \text{conjran} \longrightarrow \text{bool}$

$\text{low} \;:\; \text{conjran} \longrightarrow \text{nat}$

$\text{up} \;:\; \text{conjran} \longrightarrow \text{nat}$

generadores

$\textit{emptyset} \;:\; \text{nat } \ell \times \text{nat } u \longrightarrow \text{conjran} \qquad \{(\ell \leq u)\}$

$\text{Ag} \;:\; \text{nat } n \times \text{conjran } c \longrightarrow \text{conjran} \qquad \{(\text{low}(c) \leq n \leq \text{up}(c))\}$

axiomas

$\forall c : \text{conjran}, \forall \ell, u, n, n' : \text{nat}$

$$
\begin{aligned}
\text{low}(\emptyset(\ell, u)) &\equiv \ell \\
\text{low}(\text{Ag}(n, c)) &\equiv \text{low}(c) \\
\text{up}(\emptyset(\ell, u)) &\equiv u \\
\text{up}(\text{Ag}(n, c)) &\equiv \text{up}(c) \\
n \in \emptyset(\ell, u) &\equiv \textit{false} \\
n \in \text{Ag}(n', c) &\equiv (n = n') \vee (n \in c)
\end{aligned}
$$

Fin TAD

Rep y Abs, un ejemplo de verdad

Conjunto en rango se representa con *estr*
donde *estr* es
< `low: nat, upper: nat, elems: vector(nat), min: nat` >

Cosas a tener en cuenta a la hora de escribir el Invariante de Representación:

- Restricciones de los generadores (¿qué instancias puedo formar?)
- Decisiones de diseño
- Coherencia en la información redundante

En castellano:

- La cota inferior es menor o igual que la cota superior
- Todos los elementos del conjunto son mayores que la cota inferior y menores que la cota superior.

En lógica:

- e.low $\leq$ e.upper
- ($\forall$ n: Nat) está?(n, e.elems) $\Rightarrow$ e.low $\leq$ n $\leq$ e.upper

En castellano:

Función de Abstracción

La función de Abstracción se puede escribir mapeando los observadores del TAD con la información que corresponda de las distintas partes de la estructura:

$$Abs : \widehat{\texttt{estr}}\ e \rightarrow \text{conjran} \qquad \{Rep(\widehat{\texttt{estr}})\}$$

$$Abs(e) = c \ / \ ((low(c) = e.lower) \wedge (up(c) = e.upper) \wedge$$
$$(\forall n : \text{nat})(n \in c \Leftrightarrow esta?(n, e.elems)))$$

O, también, *generando* la instancia correspondiente:

$$Abs : \widehat{\texttt{estr}}\ e \rightarrow \text{conjran} \qquad \{Rep(\widehat{\texttt{estr}})\}$$

$$\begin{aligned}
&\text{Abs(e)} && \equiv \ \text{secuAConjran}(\text{e}.elems, \text{e}.lower, \text{e}.upper)\\
&\text{secuAConjran} : \text{secu(Nat)} \times \text{Nat } l \times \text{Nat } u && \longrightarrow \ \text{conjran} \qquad \{\text{l} \leq \text{u}\}\\
&\text{secuAConjran(s,l,u)} && \equiv \ \textbf{if } \text{vacia?(s)} \ \textbf{ then}\\
& && \qquad emptyset(l, u)\\
& && \ \ \textbf{else}\\
& && \qquad \text{Ag(prim(s), secuAConjran(fin(s),l,u))}\\
& && \ \ \textbf{fi}
\end{aligned}$$

Ejercicio 0: Diccionario

La siguiente es la especificación del Tad Diccionario, restringida a observadores y generadores:

TAD DICCIONARIO(CLAVE, SIG)

observadores básicos

$$\begin{array}{lcllr} \text{def?} & : & \text{clave} \times \text{dicc(clave, sig)} & \longrightarrow \text{bool} & \\ \text{obtener} & : & \text{clave } c \times \text{dicc(clave, sig) } d & \longrightarrow \text{sig} & \{\text{def?}(c, d)\} \end{array}$$

generadores

$$\begin{array}{lcll} \text{vacío} & : & & \longrightarrow \text{dicc(clave, sig)} \\ \text{definir} & : & \text{clave} \times \text{sig} \times \text{dicc(clave, sig)} & \longrightarrow \text{dicc(clave, sig)} \end{array}$$

axiomas $\forall d$: dicc(clave, sig), $\forall c, k$: clave, $\forall s$: sig

$$\begin{array}{lcl} \text{def?}(c, \text{vacío}) & \equiv & \text{false} \\ \text{def?}(c, \text{definir}(k, s, d) & \equiv & c = k \lor \text{def?}(c, d) \\ \text{obtener}(c, \text{definir}(k, s, d)) & \equiv & \textbf{if } c = k \textbf{ then } s \textbf{ else } \text{obtener}(c, d) \textbf{ fi} \end{array}$$

Fin TAD

Ejercicio 0: Diccionario

Para representar el TAD DICCIONARIO se decidió utilizar la siguiente estructura:

diccionario **se representa con** estr, donde

estr es tupla $\langle$ *claves*: lista(α),
significados: lista(β) $\rangle$

Donde cada par $<$clave, significado$>$ se encuentra en el mismo índice de cada una de las listas.

a) Escribir en castellano el invariante de representación.
b) Escribir formalmente el invariante de representación.
c) Escribir formalmente la función de abstracción.

Ejercicio 2: Piratas y Ninjas

Ejercicio de Parcial (1er Cuatrimestre, 2015)

La siguiente especificación modela un castillo donde conviven piratas y ninjas. Con frecuencia arriban al castillo nuevos piratas y ninjas, que nunca mueren ni se van. Por supuesto, cada tanto surgen peleas, que por tradición ancestral son siempre entre un pirata y un ninja. Los piratas y los ninjas se identifican con naturales unívocos: no hay dos piratas, ni dos ninjas, ni un pirata y un ninja que se identifiquen con el mismo número.

TAD CASTILLO

observadores básicos

$$
\begin{array}{lcll}
\text{piratas} & : & \text{castillo} & \longrightarrow \text{conj(nat)} \\
\text{ninjas} & : & \text{castillo} & \longrightarrow \text{conj(nat)} \\
\text{cantPeleas} & : & \text{castillo } c \times \text{nat } p \times \text{nat } n & \longrightarrow \text{nat}
\end{array}
$$

$$\{p \in \text{piratas}(c) \wedge n \in \text{ninjas}(c)\}$$

generadores

$$
\begin{array}{lcll}
\text{crear} & : & & \longrightarrow \text{castillo} \\
\text{llegaPirata} & : & \text{castillo } c \times \text{nat } p & \longrightarrow \text{castillo}
\end{array}
$$

$$\{p \notin (\text{piratas}(c) \cup \text{ninjas}(c))\}$$

$$
\begin{array}{lcll}
\text{llegaNinja} & : & \text{castillo } c \times \text{nat } n & \longrightarrow \text{castillo}
\end{array}
$$

$$\{n \notin (\text{piratas}(c) \cup \text{ninjas}(c))\}$$

$$
\begin{array}{lcll}
\text{pelean} & : & \text{castillo } c \times \text{nat } p \times \text{nat } n & \longrightarrow \text{castillo}
\end{array}
$$

$$\{p \in \text{piratas}(c) \wedge n \in \text{ninjas}(c)\}$$

Fin TAD

TAD CASTILLO

axiomas

$$
\begin{array}{lll}
\text{piratas(crear)} & \equiv & \emptyset \\
\text{piratas(llegaPirata}(c, p)) & \equiv & \text{Ag}(p, \text{ piratas}(c)) \\
\text{piratas(llegaNinja}(c, n)) & \equiv & \text{piratas}(c) \\
\text{piratas(pelean}(c, p, n)) & \equiv & \text{piratas}(c) \\
\text{ninjas(crear)} & \equiv & \emptyset \\
\text{ninjas(llegaPirata}(c, p)) & \equiv & \text{ninjas}(c) \\
\text{ninjas(llegaNinja}(c, n)) & \equiv & \text{Ag}(n, \text{ ninjas}(c)) \\
\text{ninjas(pelean}(c, p, n)) & \equiv & \text{ninjas}(c) \\
\text{cantPeleas(llegaPirata}(c, p'),\ p, n) & \equiv & \textbf{if } p = p' \textbf{ then } 0 \\
 & & \textbf{else } \text{cantPeleas}(c, p, n) \textbf{ fi} \\
\text{cantPeleas(llegaNinja}(c, n'),\ p, n) & \equiv & \textbf{if } n = n' \textbf{ then } 0 \textbf{ else} \\
 & & \text{cantPeleas}(c, p, n) \textbf{ fi} \\
\text{cantPeleas(pelean}(c, p', n'),\ p, n) & \equiv & \textbf{if } p = p' \wedge n = n' \textbf{ then } 1 \textbf{ else } 0 \textbf{ fi} \\
 & & + \text{ cantPeleas}(c, p, n)
\end{array}
$$

Fin TAD

Ejercicio 2: Piratas y Ninjas

Para representar el TAD CASTILLO se decidió utilizar la siguiente estructura:

castillo **se representa con** estr, donde

estr es tupla $\langle$ *piratas*: conj(nat),
ninjas: conj(nat),
rivalesQueTuvo: dicc(nat, conj(nat)),
historialPeleas: secu(tupla $\langle p$: nat, n : nat$\rangle$) $\rangle$

donde *piratas* y *ninjas* representan los conjuntos de identificadores de piratas y ninjas, respectivamente, *rivalesQueTuvo* asocia a cada peleador (tanto piratas como ninjas, ya que todos los identificadores son distintos) con el conjunto de todos los rivales contra los que peleó al menos una vez, e *historialPeleas* tiene la secuencia de parejas $\langle$pirata, ninja$\rangle$ que se entreveraron en una pelea, en el orden en que éstas sucedieron.

a) Escribir en castellano el invariante de representación.
b) Escribir formalmente el invariante de representación.
c) Escribir formalmente la función de abstracción.