

## Trabajo práctico 2: Diseño “Broker System”

### Normativa

**Límite de entrega:** martes 27 de Mayo de 2014 a las 22:00 hs.

**Normas de entrega:** Ver “Información sobre la cursada” en el sitio Web de la materia.

(<http://www.dc.uba.ar/materias/aed2/2014/1c/informacion>)

**Versión:** 2.2 del 17/5/14 (ver TP2 Changelog)

### Enunciado

El objetivo de este trabajo práctico consiste en realizar el diseño completo del módulo que se explica con el TAD WOLFIE (ver sección Especificación), lo que implica diseñar todos aquellos módulos necesarios para la tarea.

Con respecto al enunciado del TP1 hay algunos cambios:

- Las promesas se ejecutan únicamente ante un cambio de cotización.
- El orden para ejecutar las promesas pendientes ahora está definido y es el siguiente:
  1. Se ejecutan las de venta.
  2. En el momento de ejecutar promesas de compra se le da prioridad a los clientes que mayor cantidad total de acciones poseen.

Estos cambios están especificados como son deseados. Confíen en la axiomatización más que en el español.

### Contexto de uso y complejidades requeridas

Se requiere que las operaciones exportadas de los TADs se mapeen, en lo posible, uno a uno en las interfaces de los módulos correspondientes.

Ademas, las operaciones indicadas a continuación, deberán cumplir las complejidades temporales<sup>1</sup> detalladas.

- **clientes**( $w$ ) es  $O(1)$  y debe devolver un iterador.
- **títulos**( $w$ ) es  $O(1)$  y debe devolver un iterador.
- **promesasDe**( $c, w$ ) es  $O(T \cdot C \cdot |max\_nt|)$  y  $O(1)$  para llamados consecutivos con el mismo  $c$ . Se puede devolver un iterador.
- **accionesPorCliente**( $c, nt, w$ ) es  $O(\log(C) + |nt|)$
- **inaugurarWolfie**( $cs, w$ ) es  $O(\#(cs)^2)$
- **agregarTítulo**( $t, w$ ) es  $O(|nombre(t)| + C)$
- **actualizarCotización**( $nt, cot, w$ ) es  $O(C \cdot |nt| + C \cdot \log(C))$
- **agregarPromesa**( $c, p, w$ ) es  $O(|titulo(p)| + \log(C))$
- **enAlza**( $nt, w$ ) es  $O(|nt|)$

donde  $t$  es un título,  $nt$  es el nombre de algún título y  $|nt|$  su longitud.  $P$  es la cantidad de promesas pendientes del Wolfie  $w$ ,  $C$  es la cantidad de clientes de  $w$ ,  $T$  es la cantidad de títulos de  $w$ , y  $|max\_nt|$  la longitud máxima entre todos los nombres de los títulos de  $w$ .

### Algunas consideraciones

- Hay estructuras que pueden servir para más de una finalidad, sobre todos los contenedores.
- Pueden crear módulos adicionales si así lo necesitan.
- No todas las operaciones exportadas de los TADs deben corresponder operaciones en los módulos. Diseñen sólo lo necesario.

<sup>1</sup>Se desestimarán los costos de eliminación de objetos, con lo cual pueden ignorarlos en el cálculo de complejidades.

- Todas las operaciones deben ser especificadas formalmente según las herramientas vistas en clase. Agregar comentarios necesarios para entender la forma en la cual deben ser utilizadas para su correcto funcionamiento.
- Todos los algoritmos deben tener su desarrollo que justifique los órdenes de complejidad. Si algún paso es no trivial pueden hacer notas a continuación del mismo.
- Cuando se formalicen los invariantes y funciones de abstracción, deben identificar cada parte de la fórmula y comentar en castellano lo que describe, para facilitar su seguimiento y corrección.
- Tener en cuenta que las complejidades son en peor caso. Soluciones más eficientes serán bien recibidas.
- Cuentan con los siguiente TADs y módulos:
  - CHAR que representan los posibles caracteres. Siendo un tipo enumerado de 256 valores. con funciones  $ord$  y  $ord^{-1}$  para la correspondencia de cada *Char* a *Nat*.
  - STRING como sinónimo de  $Vector(Char)$ .
  - Todos los definidos en el apunte de TADs básicos.
  - Todos los definidos en el apunte de módulos básicos.

## Especificación

### Renombres de TADs

TAD CLIENTE es NAT

TAD DINERO es NAT

TAD NOMBRE es STRING

TAD TIPOPROMESA es ENUM {COMPRA, VENTA}

TAD PARPC es TUPLA(PROMESA, CLIENTE)

### TAD PROMESA

#### TAD PROMESA

##### igualdad observacional

$$(\forall p, p' : \text{promesa}) \left( p =_{\text{obs}} p' \iff \left( \begin{array}{l} \text{título}(p) = \text{título}(p') \wedge \text{tipo}(p) = \text{tipo}(p') \wedge \text{límite}(p) = \text{límite}(p') \wedge \text{cantidad}(p) = \text{cantidad}(p') \end{array} \right) \right)$$

**géneros**      promesa

**exporta**      promesa, generadores, observadores, otras operaciones

**usa**            TIPOPROMESA, DINERO, NOMBRE, NAT

##### observadores básicos

título	: promesa	→ nombre
tipo	: promesa	→ tipoPromesa
límite	: promesa	→ dinero
cantidad	: promesa	→ nat

##### generadores

crearPromesa : nombre × tipoPromesa × dinero × nat → promesa

##### otras operaciones

promesaEjecutable	: promesa × dinero × nat	→ bool
compraVenta	: nat × conj(promesa)	→ nat
promesasSobreTítulo	: nombre × tipoPromesa × conj(promesa)	→ conj(promesa)

**axiomas**       $\forall t: \text{nombre} \forall n, m: \text{nat} \forall \text{tipo}: \text{tipoPromesa}$

título(crearPromesa(t, tipo, n, m))	≡ t
tipo(crearPromesa(t, tipo, n, m))	≡ tipo
límite(crearPromesa(t, tipo, n, m))	≡ n
cantidad(crearPromesa(t, tipo, n, m))	≡ m

promesaEjecutable(p, cot, disp) ≡ (tipo(p) = venta ∧ cot < límite(p)) ∨  
(tipo(p) = compra ∧ cot > límite(p) ∧ disp ≥ cantidad(p))

compraVenta(dis, ps) ≡ **if** vacío?(ps) **then**  
     dis  
**else**  
     **if** tipo(dameUno(ps)) = compra **then**  
         compraVenta(dis + cantidad(dameUno(ps)), sinUno(ps))  
     **else**  
         compraVenta(dis - cantidad(dameUno(ps)), sinUno(ps))  
     **fi**  
**fi**

```

promesasSobreTítulo(t1,tp,ps)  ≡  if vacío?(ps) then
    ∅
else
    promesasSobreTítulo(t1, tp, sinUno(ps)) ∪
    (if t1 = título(dameUno(ps)) ∧ tp = tipo(dameUno(ps)) then
        {dameUno(ps)}
    else
        ∅
    fi)
fi

```

**Fin TAD**

## TAD TÍTULO

**TAD TÍTULO**

**igualdad observacional**

$$(\forall t, t' : \text{título}) \left( t =_{\text{obs}} t' \iff \left( \begin{array}{l} \text{nombre}(t) = \text{nombre}(t') \wedge \text{cotización}(t) = \text{cotización}(t') \\ \wedge \# \text{máxAcciones}(t) = \# \text{máxAcciones}(t') \wedge \\ \text{enAlza?}(t) \iff \text{enAlza?}(t') \end{array} \right) \right)$$

**géneros**      título

**exporta**      título, generadores, observadores, otras operaciones

**usa**            NOMBRE, DINERO, NAT

**observadores básicos**

nombre	: título	→ nombre
#máxAcciones	: título	→ nat
cotización	: título	→ dinero
enAlza	: título	→ bool

**generadores**

crearTítulo	: nombre × dinero × nat	→ título
recotizar	: dinero × título	→ título

**otras operaciones**

cotizaciónActual	: nombre <i>nomTit</i> × conj(título) <i>ts</i>	→ nat	$\{(\exists t: \text{título})(t \in ts \wedge \text{nombre}(t) = \text{nomTit})\}$
cambiarCotización	: nombre <i>nomTit</i> × dinero <i>cot</i> × conj(título) <i>ts</i>	→ conj(título)	$\{(\exists t: \text{título})(t \in ts \wedge \text{nombre}(t) = \text{nomTit})\}$
límiteTenencia	: nombre <i>nomTit</i> × conj(título) <i>ts</i>	→ nat	$\{(\exists t: \text{título})(t \in ts \wedge \text{nombre}(t) = \text{nomTit})\}$
títuloEnAlza	: nombre <i>nomTit</i> × conj(título) <i>ts</i>	→ bool	$\{(\exists t: \text{título})(t \in ts \wedge \text{nombre}(t) = \text{nomTit})\}$

**axiomas**       $\forall s: \text{nombre} \forall n, c, c': \text{nat}$

nombre(crearTítulo(s,c,n))	≡ s
nombre(recotizar(c,t))	≡ nombre(t)
#máxAcciones(crearTítulo(s,c,n))	≡ n
#máxAcciones(recotizar(c,t))	≡ #máxAcciones(t)
cotización(crearTítulo(s,c,n))	≡ c
cotización(recotizar(c,t))	≡ c
enAlza(crearTítulo(s,c,n))	≡ true
enAlza(recotizar(c,t))	≡ c > cotización(t)

```

cotizaciónActual(t1, ts) ≡ if nombre(dameUno(ts)) = t1 then
    cotización(dameUno(ts))
else
    cotizaciónActual(t1, sinUno(ts))
fi

cambiarCotización(t1, cot, ts) ≡ if vacío?(ts) then
    ∅
else
    if nombre(dameUno(ts)) = t1 then
        Ag(recotizar(cot, dameUno(ts)), sinUno(ts))
    else
        Ag(dameUno(ts), cambiarCotización(t1, cot, sinUno(ts)))
    fi
fi

límiteTenencia(t1, ts) ≡ if nombre(dameUno(ts)) = t1 then
    #máxAcciones(dameUno(ts))
else
    límiteTenencia(t1, sinUno(ts))
fi

títuloEnAlza(t1, ts) ≡ if nombre(dameUno(ts)) = t1 then
    enAlza(dameUno(ts))
else
    títuloEnAlza(t1, sinUno(ts))
fi

```

**Fin TAD**

## TAD WOLFIE

**TAD WOLFIE**

**géneros**      wolfie

**exporta**      wolfie, generadores, observadores, enAlza

**usa**          BOOL, NAT, DINERO, PROMESA, CLIENTE, TÍTULO, CONJUNTO( $\alpha$ ), NOMBRE, TIPOPROMESA, PARPC

**igualdad observacional**

$$(\forall w1, w2 : \text{wolfie}) \left( w1 =_{\text{obs}} w2 \iff \left( \begin{array}{l} \text{clientes}(w1) = \text{clientes}(w2) \wedge \text{títulos}(w1) = \text{títulos}(w2) \wedge_L (\forall c:\text{cliente}, t:\text{título}) ((c \in \text{clientes}(w1) \wedge t \in \text{títulos}(w1)) \Rightarrow_L (\text{promesasDe}(c, w1) = \text{promesasDe}(c, w2)) \wedge \text{accionesPorCliente}(c, \text{nombre}(t), w1) = \text{accionesPorCliente}(c, \text{nombre}(t), w2)) \end{array} \right) \right)$$

**observadores básicos**

clientes	: wolfie	→ conj(cliente)	
títulos	: wolfie	→ conj(título)	
promesasDe	: cliente $c \times$ wolfie $w$	→ conj(promesa)	$\{c \in \text{clientes}(w)\}$
accionesPorCliente	: cliente $c \times$ nombre $\text{nomTit} \times$ wolfie $w$	→ nat	$\{c \in \text{clientes}(w) \wedge (\exists t: \text{título})(t \in \text{títulos}(w) \wedge \text{nombre}(t) = \text{nomTit})\}$

**generadores**

inaugurarWolfie	: conj(cliente) $cs$	→ wolfie	$\{\neg \text{vacío}(cs)\}$
agregarTítulo	: título $t \times$ wolfie $w$	→ wolfie	$\{(\forall t_2: \text{título})(t_2 \in \text{títulos}(w) \Rightarrow \text{nombre}(t) \neq \text{nombre}(t_2))\}$
actualizarCotización	: nombre $\text{nomTit} \times$ nat $cot \times$ wolfie $w$	→ wolfie	$\{(\exists t: \text{título})(t \in \text{títulos}(w) \wedge \text{nombre}(t) = \text{nomTit})\}$
agregarPromesa	: cliente $c \times$ promesa $p \times$ wolfie $w$	→ wolfie	

$$\left\{ \begin{array}{l} (\exists t: \text{título})(t \in \text{títulos}(w) \wedge \text{nombre}(t) = \text{título}(p)) \wedge c \in \text{clientes}(w) \wedge_L (\forall p_2: \text{promesa})(p_2) \\ \in \text{promesasDe}(c, w) \Rightarrow (\text{título}(p) \neq \text{título}(p_2) \vee \text{tipo}(p) \neq \text{tipo}(p_2)) \wedge (\text{tipo}(p) = \text{vender}) \\ \Rightarrow \text{accionesPorCliente}(c, \text{título}(p), w) \geq \text{cantidad}(p) \end{array} \right\}$$

**otras operaciones**

enAlza	: nombre <i>nomTit</i> × wolfie <i>w</i>	→ bool
		{(∃t: título)(t ∈ títulos(w) ∧ nombre(t) = nomTit)}
accDisponibles	: nombre <i>nomTit</i> × wolfie <i>w</i>	→ nat
		{(∃t: título)(t ∈ títulos(w) ∧ nombre(t) = nomTit)}
tenenciasDeTodos	: nombre <i>nomTit</i> × conj(cliente) <i>cs</i> × wolfie <i>w</i>	→ nat
		{cs ⊆ clientes(w) ∧ (∃t: título)(t ∈ títulos(w) ∧ nombre(t) = nomTit)}
promesasAEjecutarPorCliente	: nombre <i>nomTit</i> × dinero <i>cot</i> × cliente <i>c</i> × wolfie <i>w</i>	→ conj(promesa)
		{c ∈ clientes(w) ∧ (∃t: título)(t ∈ títulos(w) ∧ nombre(t) = nomTit)}
filtrarPorCliente	: cliente × conj(parPC)	→ conj(promesa)
ventasAEjecutar	: nombre <i>nomTit</i> × dinero <i>cot</i> × wolfie <i>w</i>	→ conj(parPC)
		{(∃t: título)(t ∈ títulos(w) ∧ nombre(t) = nomTit)}
comprasAEjecutar	: nombre <i>nomTit</i> × dinero <i>cot</i> × wolfie <i>w</i>	→ conj(parPC)
		{(∃t: título)(t ∈ títulos(w) ∧ nombre(t) = nomTit)}
promesasAEjecutar	: dinero <i>cot</i> × nat <i>disp</i> × secu(parPC) <i>pends</i>	→ conj(parPC)
aparearPC	: nombre × tipoPromesa × conj(cliente) × wolfie	→ conj(parPC)
accVendidas	: conj(parPC)	→ nat
promesasOrdenAccClie	: conj(parPC) <i>cs</i> × conj(parPC) <i>vs</i> × wolfie <i>w</i>	→ secu(parPC)
ordenSecu	: conj(parPC) × secu(cliente))	→ secu(parPC)
estaClie?	: cliente <i>c</i> × conj(parPC) <i>ps</i>	→ bool
		{¬ vacío(ps)}
damePromesa	: cliente <i>c</i> × conj(parPC) <i>ps</i>	→ promesa
		{estaClie?(c, ps)}
ordenarPorAcciones	: dicc(cliente, nat)	→ secu(cliente)
máxAcc	: dicc(cliente, nat) <i>dc</i>	→ cliente
		{¬ vacío(dc)}
máxAccAux	: conj(cliente) <i>cs</i> × dicc(cliente, nat) <i>dc</i>	→ tupla(cliente, nat)
		{¬ vacío(dc) ∧ <sub>L</sub> cs ⊆ claves(dc)}
totalAccClie	: conj(parPC) <i>vs</i> × wolfie <i>w</i>	→ dicc(cliente, nat)
totalAccClieAux	: conj(parPC) <i>cs</i> × conj(parPC) <i>vs</i> × wolfie <i>w</i>	→ dicc(cliente, nat)
accTotalClie	: cliente <i>c</i> × wolfie <i>w</i>	→ nat
		{c ∈ clientes(w)}
accTotalClieAux	: cliente <i>c</i> × conj(título) <i>ts</i> × wolfie <i>w</i>	→ nat
		{c ∈ clientes(w) ∧ ts ⊆ títulos(w)}
accVendidasClie	: conj(parPC) <i>ps</i> × cliente <i>c</i>	→ nat

**axiomas**     $\forall cs: \text{conj}(\text{cliente}) \forall w: \text{wolfie} \forall t: \text{título} \forall t1, t2: \text{nombre} \forall c, cl: \text{cliente} \forall p: \text{promesa} \forall cot: \text{nat} \forall ts:$   
 $\text{conj}(\text{título}) \forall ps: \text{conj}(\text{promesa}) \forall disp: \text{nat} \forall tp: \text{tipoPromesa}$

clientes(inaugurarWolfie(cs))     $\equiv cs$

clientes(agregarTítulo(t, w))     $\equiv \text{clientes}(w)$

clientes(actualizarCotización(t2, cot, w))     $\equiv \text{clientes}(w)$

clientes(agregarPromesa(c, p, w))     $\equiv \text{clientes}(w)$

títulos(inaugurarWolfie(cs))     $\equiv \emptyset$

títulos(agregarTítulo(t, w))     $\equiv \text{Ag}(t, \text{títulos}(w))$

títulos(actualizarCotización(t1, cot, w))     $\equiv \text{cambiarCotización}(t1, c, \text{títulos}(w))$

títulos(agregarPromesa(c, p, w))     $\equiv \text{títulos}(w)$

promesasDe(c, inaugurarWolfie(cs))     $\equiv \emptyset$

promesasDe(c, agregarTítulo(t, w))     $\equiv \text{promesasDe}(c, w)$

promesasDe(c, agregarPromesa(cl, p, w))     $\equiv \text{promesasDe}(c, w) \cup \text{if } c = cl \text{ then } \{p\} \text{ else } \emptyset \text{ fi}$

```

promesasDe(c, actualizarCotización(t1, cot, w)) ≡ promesasDe(c, w) -
                                                    promesasAEjecutarPorCliente(t1, cot, c, w)

enAlza(t1,w)                                     ≡ títuloEnAlza(t1,títulos(w))

accDisponibles(t1,w)                             ≡ límiteTenencia(t1,títulos(w)) - tenenciasDeTodos(t1,clientes(w),w)

tenenciasDeTodos(t1,cs,w) ≡ if vacío?(cs) then
                                0
                                else
                                accionesPorCliente(dameUno(cs),t1,w) +
                                tenenciasDeTodos(t1,sinUno(cs),w)
                                fi

promesasAEjecutarPorCliente(t1,cot,c,w) ≡ filtrarPorCliente(c, (ventasAEjecutar(t1,cot,w) ∪
                                                                    comprasAEjecutar(t1,cot,w)) )

filtrarPorCliente(c, ps) ≡ if vacío?(ps) then
                                ∅
                                else
                                filtrarPorCliente(c,sinUno(ps)) ∪
                                (if  $\Pi_2(\text{dameUno}(ps)) = c$  then  $\{\Pi_1(\text{dameUno}(ps))\}$  else ∅ fi)
                                fi

ventasAEjecutar(t1,cot,w) ≡ promesasAEjecutar(cot,0,conjASecu(aparearPC(t1,venta,clientes(w),w)))

comprasAEjecutar(t1,cot,w) ≡ promesasAEjecutar(cot,
                                                (accDisponibles(t1,w) + accVendidas(ventasAEjecutar(t1,cot,w))),
                                                promesasOrdenAccClie(aparearPC(t1,compra,clientes(w),w),
                                                                      ventasAEjecutar(t1,cot,w),w))

aparearPC(t1,tp,cs,w) ≡ if vacío?(cs) then
                                ∅
                                else
                                aparearPC(t1,tp,sinUno(cs),w) ∪
                                (if vacío?(promesasSobreTítulo(t1, tp, promesasDe(dameUno(cs),w)))
                                then
                                ∅
                                else
                                {<promesasSobreTítulo(t1,tp,promesasDe(dameUno(cs),w)),dameUno(cs)>}
                                fi)
                                fi

promesasAEjecutar(cot,disp,pends) ≡ if vacía?(pends) then
                                ∅
                                else
                                if promesaEjecutable( $\Pi_1(\text{prim}(pends))$ ,cot,disp) then
                                    {prim(pends)} ∪
                                    (if tipo( $\Pi_1(\text{prim}(pends))$ ) = compra then
                                        promesasAEjecutar(cot,
                                                            disp - cantidad( $\Pi_1(\text{prim}(pends))$ ),
                                                            fin(pends))
                                    else
                                        promesasAEjecutar(cot,
                                                            disp + cantidad( $\Pi_1(\text{dameUno}(pends))$ ),
                                                            fin(pends))
                                    fi )
                                else
                                    promesasAEjecutar(cot, disp, fin(pends))
                                fi
                                fi

```

```

accVendidas(ps)  ≡ if vacío?(ps) then
    0
else
    cantidad( $\Pi_1$ (dameUno(ps))) + accVendidas(sinUno(ps))
fi

promesasOrdenAccClie(cs,vs,w) ≡ ordenSecu(cs, ordenarPorAcciones(totalAccClie(vs,w)))

ordenSecu(ps,sc) ≡ if vacía?(sc) then
    <>
else
    if estaClie?(prim(sc),ps) then damePromesa(prim(sc),ps) • <> else <> fi
    & ordenSecu(ps,fin(sc))
fi

estaClie?(c,ps) ≡  $\Pi_2$ (dameUno(ps)) = c  $\vee_L$  estaClie?(c,sinUno(ps))

damePromesa(c,ps) ≡ if  $\Pi_2$ (dameUno(ps)) = c then
     $\Pi_1$ (dameUno(ps))
else
    damePromesa(c,sinUno(ps))
fi

ordenarPorAcciones(dc) ≡ if vacío?(dc) then
    vacío
else
    máximoAcc(dc) • ordenarPorAcciones(borrar(máximoAcc(dc),dc))
fi

máximoAcc(dc)      ≡  $\Pi_1$ (máximoAccAux(claves(dc),dc))
máximoAccAux(cs,dc) ≡ if vacío(sinUno(cs)) then
    <dameUno(cs), obtener(dameUno(cs),dc)>
else
    if obtener(dameUno(cs),dc) >  $\Pi_2$ (máximoAccAux(sinUno(cs),dc)) then
        <dameUno(cs), obtener(dameUno(cs),dc)>
    else
        máximoAccAux(sinUno(cs),dc)
    fi
fi

totalAccClie(vs,w)      ≡ totalAccClieAux(clientes(w),vs,w)
totalAccClieAux(cs,vs,w) ≡ if vacío?(cs) then
    vacío?
else
    definir(dameUno(cs),
        (accTotalClie(dameUno(cs),w)+accVendidasClie(vs,dameUno(cs))),
        totalAccClieAux(sinUno(cs),vs,w))
    fi

accTotalClie(c,w)      ≡ accTotalClieAux(c,títulos(w),w)
accTotalClieAux(c,ts,w) ≡ if vacío?(ts) then
    0
else
    accionesPorCliente(c,dameUno(ts),w)+accTotalClieAux(c,sinUno(ts),w)
fi

accVendidasClie(ps,c) ≡ if vacío?(ps) then
    0
else
    accVendidasClie(c,sinUno(ps)) +
    if c =  $\Pi_2$ (dameUno(ps)) then cantidad( $\Pi_1$ (dameUno(ps))) else 0 fi
fi

```



```
accionesPorCliente(c, t1, agregarTítulo(t2, cot, w))    ≡ if t1 = t2 then  
                                                         0  
                                                         else  
                                                         accionesPorCliente(c, t1, w)  
                                                         fi  
accionesPorCliente(c, t1, agregarPromesa(cl, p, w))    ≡ accionesPorCliente(c,t1,w)  
  
accionesPorCliente(c,t1,actualizarCotización(t2,cot,w)) ≡ if t1 = t2 then  
                                                         compraVenta(accionesPorCliente(c,t1,w),  
                                                         promesasAEjecutarPorCliente(t1,cot,c,w))  
                                                         else  
                                                         accionesPorCliente(c,t1,w)  
                                                         fi
```

**Fin TAD**