

Laboratorio de Computación III

Clase XXI

Andrés Fortier

Excepciones

- Una excepción indica una condición en particular que altera el flujo normal de la ejecución de un programa.
- En la mayoría de los casos indican errores que el sistema no puede resolver en ese punto.
- Ejemplo:

```
| numeros |  
numeros := #(33 25) .  
numeros at: 3.
```

Ejemplo 1

```
Banco>>obtenerCuenta: numeroDeCuenta
```


```
  ^self cuentas detect:  
    [:cuenta | cuenta numero = numeroDeCuenta].
```



```
Cajero>>depositar: unMonto enCuentaNumero: numeroDeCuenta
```

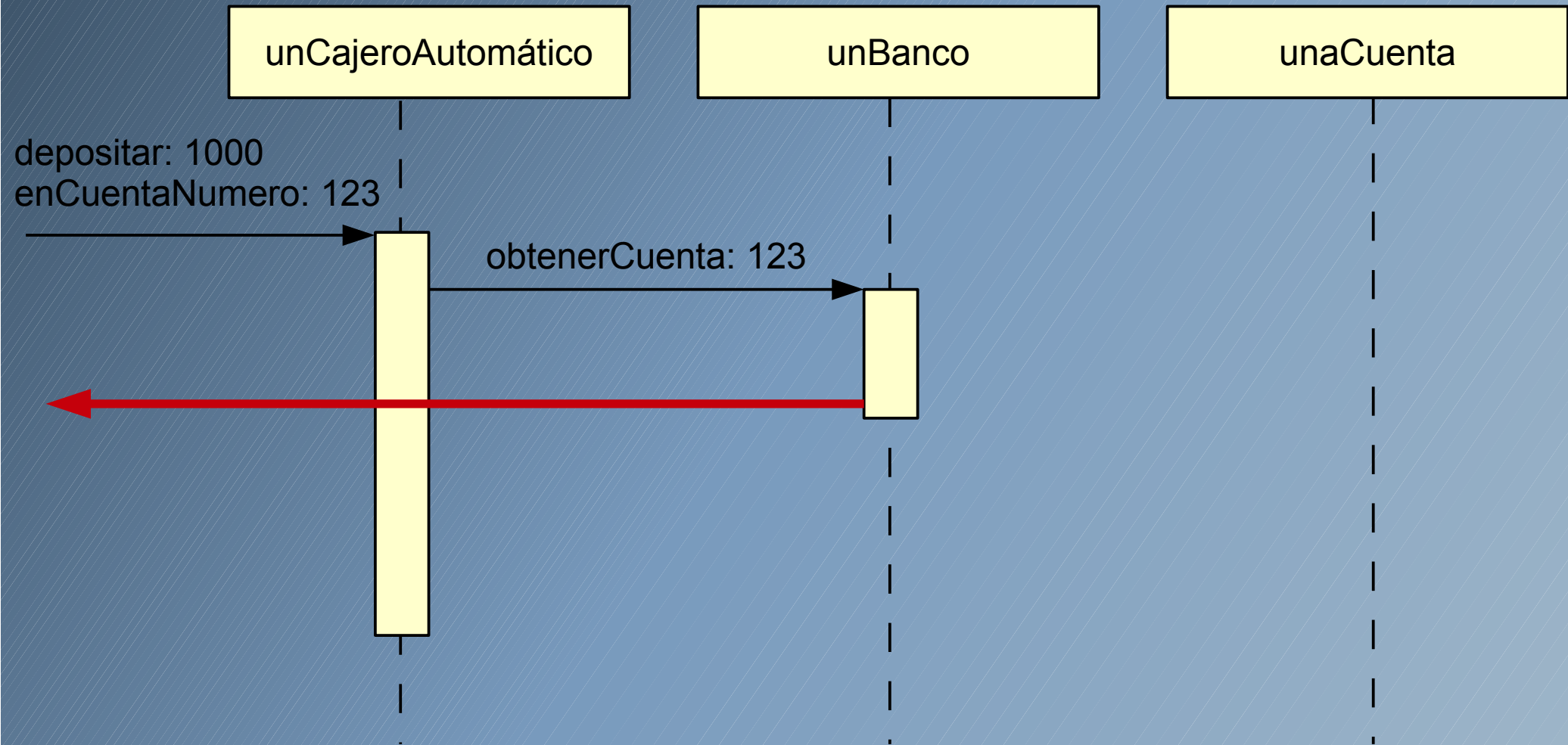
```
  | cuenta |
```

```
  cuenta := self banco obtenerCuenta: numeroDeCuenta  
  cuenta depositar: unMonto.
```



- ¿Qué sucede si el banco no tiene la cuenta buscada?
 - Se genera una excepción (NotFound).
 - La excepción se propaga buscando alguien que la maneje.

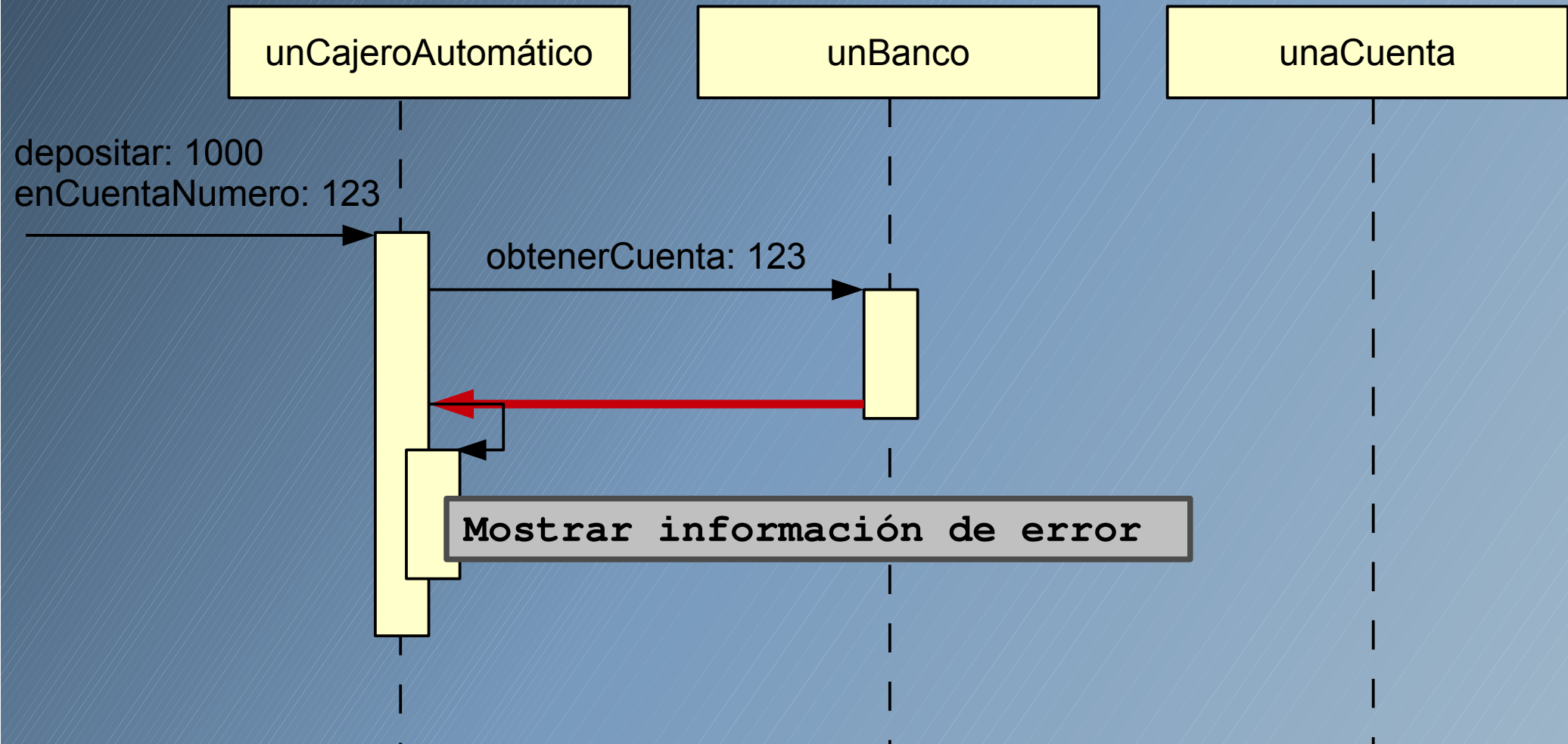
Ejemplo 1



Manejando Excepciones

```
Cajero>>depositar: unMonto enCuentaNumero: numeroDeCuenta  
  
[ | cuenta |  
  cuenta := self banco obtenerCuenta: numeroDeCuenta.  
  cuenta depositar: unMonto.  
]  
  on: NotFound  
  do: [:exception | "Mostrar información de error"]
```

Ejemplo 1



Generando Excepciones

```
CajaDeAhorro>>extraer: unMonto  
  
    (self puedeExtraer: unMonto)  
    ifTrue: [saldo := saldo - unMonto].
```

- ¿Qué sucede si la extracción no se puede realizar?
- Debemos generar una excepción.

Generando Excepciones

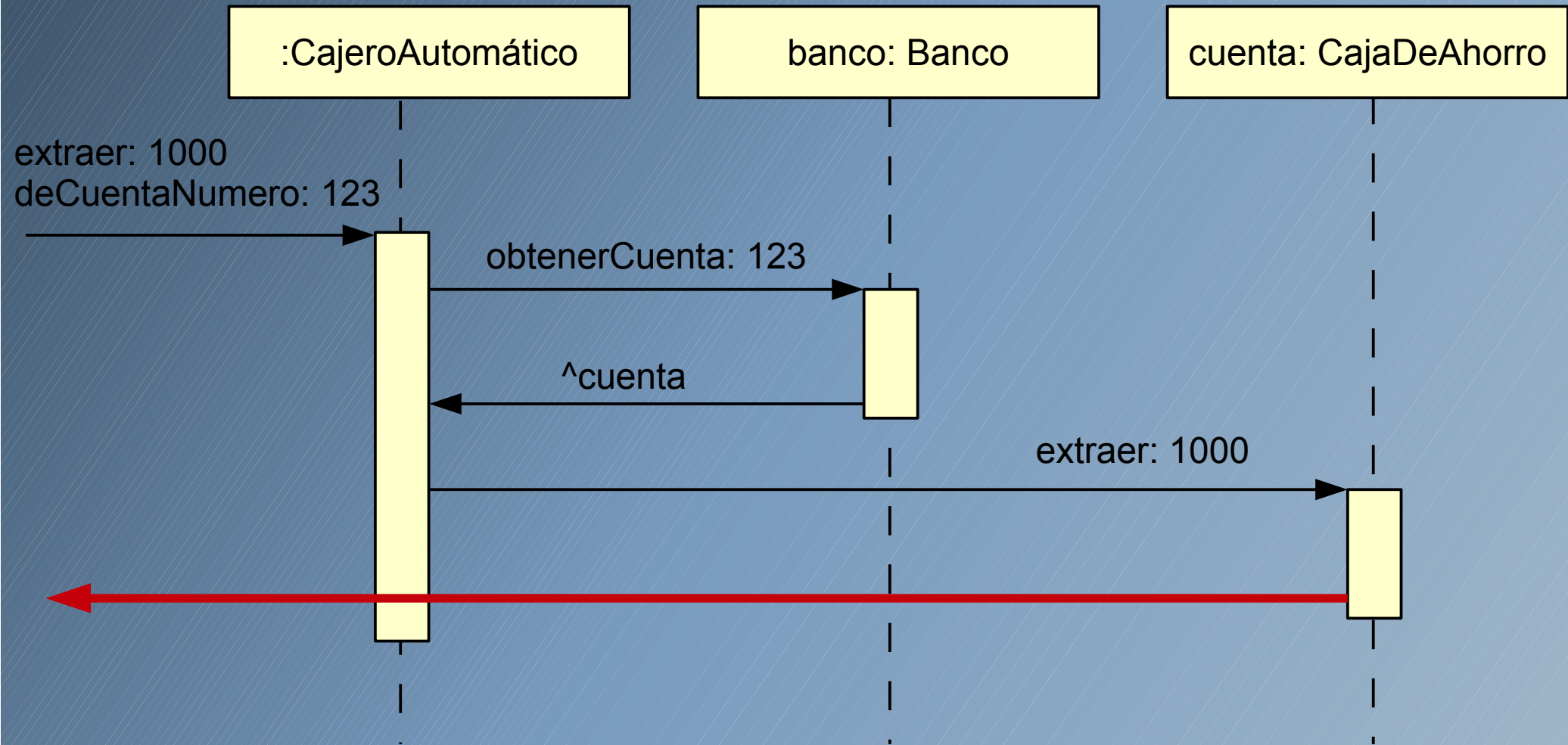
- Primer paso: Crear una clase que represente dicha excepción.

```
Error subclass: #ExtraccionNoRealizable  
  instanceVariableNames: ''  
  classVariableNames: ''  
  category: ''
```

- “Levantar” la excepción cuando corresponda.

```
CajaDeAhorro>>extraer: unMonto  
  
  (self puedeExtraer: unMonto)  
    ifTrue: [saldo := saldo - unMonto]  
    ifFalse: [ExtraccionNoRealizable signal].
```


Ejemplo 2



Tests de unidad con excepciones

```
CajaDeAhorroTest>>testExtraer
```

```
| cuenta |
```

```
cuenta := CajaDeAhorro saldo: 100.
```

```
cuenta extraer: 50.
```

```
self assert: cuenta saldo = 50.
```

```
self
```

```
    should: [cuenta extraer: 200]
```

```
    raise: ExtraccionNoRealizable.
```

Las excepciones son objetos

- Al definir una clase de excepción podemos agregar información sobre el error para facilitar el proceso de debugging.
- Veamos como podemos modificar el caso de buscar una cuenta y no encontrarla

```
Banco>>obtenerCuenta: numeroDeCuenta
```

```
^self cuentas detect:
```

```
[:cuenta | cuenta numero = numeroDeCuenta].
```



```
NotFoundError
```

Errores mas claros

```
Error subclass: #CuentaInexistente  
instanceVariableNames: 'numeroDeCuenta'  
classVariableNames: ''  
category: ''
```

```
Banco>>obtenerCuenta: numeroDeCuenta  
  
| filtradas |  
  
filtradas := self cuentas select:  
    [:cuenta | cuenta numero = numeroDeCuenta].  
filtradas isEmpty  
    ifTrue: [ | error |  
  
                error:=CuentaInexistente new.  
                error numeroDeCuenta: numeroDeCuenta.  
                error signal.  
            ]  
    ifFalse: [^filtradas first].
```


Test de unidad

```
BancoTest>>testObtenerCuenta
```

```
| banco cuenta |
```

```
banco := Banco new.
```

```
cuenta := CajaDeAhorro numero: 123.
```

```
banco agregar: cuenta.
```

```
self assert: (banco obtenerCuenta: 123) = cuenta.
```

```
self
```

```
    should: [banco obtenerCuenta: 38978]
```

```
    raise: CuentaInexistente
```

```
    withExceptionDo: [:exception |
```

```
        self assert: exception numeroDeCuenta = 38978]
```

Mejorando el código

```
Banco>>obtenerCuenta: numeroDeCuenta

| filtradas |

filtradas := self cuentas select:
    [:cuenta | cuenta numero = numeroDeCuenta].
filtradas isEmpty
    ifTrue: [ | error |

                error:=CuentaInexistente new.
                error numeroDeCuenta: numeroDeCuenta.
                error signal.
            ]
    ifFalse: [^filtradas first].
```

Mejorando el código

- Las colecciones entienden el mensaje #detect:ifNone:
- Podemos usar eso para mejorar nuestra implementación.

```
Banco>>obtenerCuenta: numeroDeCuenta

^self cuentas
  detect: [:cuenta | cuenta numero = numeroDeCuenta]
  ifNone: [ | error |

              error:=CuentaInexistente new.
              error numeroDeCuenta: numeroDeCuenta.
              error signal.
  ].
```

Mejorando el código

- Podemos definir un constructor en CuentaInexistente

```
CuentaInexistente (class)>>numeroDeCuenta: unNnumero  
  | error |  
  
  error := self new.  
  error numeroDeCuenta: numeroDeCuenta.  
  ^error.
```

```
Banco>>obtenerCuenta: numeroDeCuenta  
  
  ^self cuentas  
    detect: [:cuenta | cuenta numero = numeroDeCuenta]  
    ifNone: [(CuentaInexistente  
              numeroDeCuenta: numeroDeCuenta) signal.  
             ].
```


Excepciones

- ¿Alguna pregunta?

Problema

- Queremos modelar un sistema de archivos
- Dos tipos de componentes
 - Archivos. Tienen un nombre y un tamaño en bytes.
 - Directorios. Tienen un nombre y su tamaño es la suma del tamaño de su contenido + 4096 bytes.
- Nos interesa poder calcular el tamaño de cualquier directorio.

Ejemplo

```
[andres@CrazyGoat:~/Smalltalks/pharo3.0] tree -h --du
```

```
.
├── [4.0M] bin
│   ├── [ 70K] libB3DAcceleratorPlugin.so
│   ├── [ 35K] libFT2Plugin.so
│   ├── [2.2M] libgit2.so.0.20.0
│   ├── [ 12K] libInternetConfigPlugin.so
│   ├── [ 16K] libJPEGReaderPlugin.so
│   ├── [161K] libJPEGReadWriter2Plugin.so
│   ├── [ 50K] libRePlugin.so
│   ├── [ 36K] libSqueakFFIPrims.so
│   ├── [ 22K] libSqueakSSL.so
│   ├── [208K] libssh2.so.1.0.1
│   ├── [1.1M] pharo
│   ├── [ 14K] vm-display-null
│   ├── [118K] vm-display-X11
│   ├── [ 28K] vm-sound-ALSA
│   └── [7.6K] vm-sound-null
├── [706K] icons
│   ├── [263K] Pharo.icns
│   ├── [345K] Pharo.ico
│   └── [ 93K] Pharo.png
├── [ 862] pharo
└── [ 285] README.txt
```

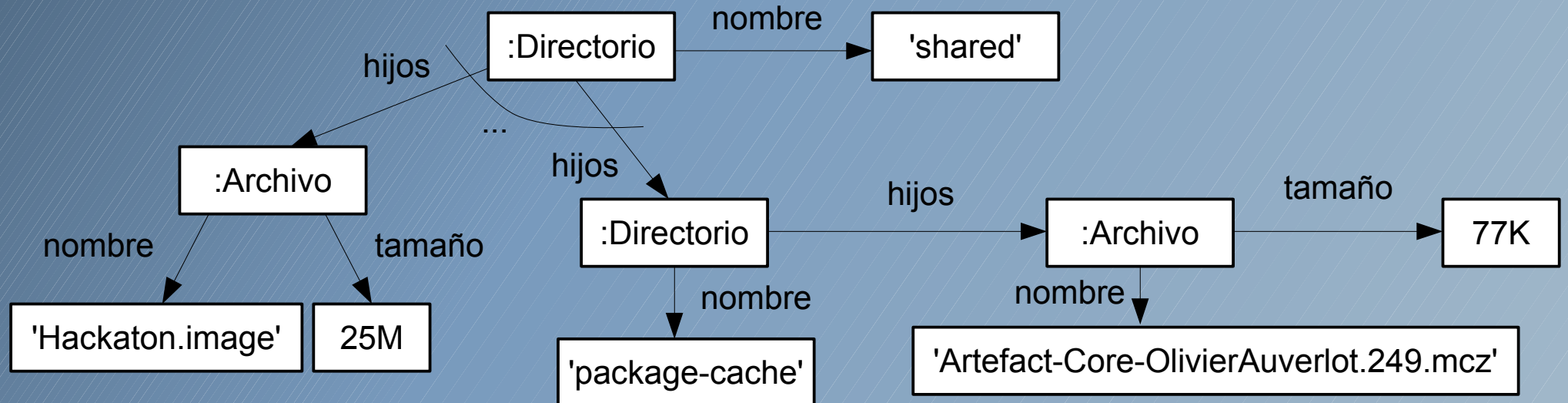
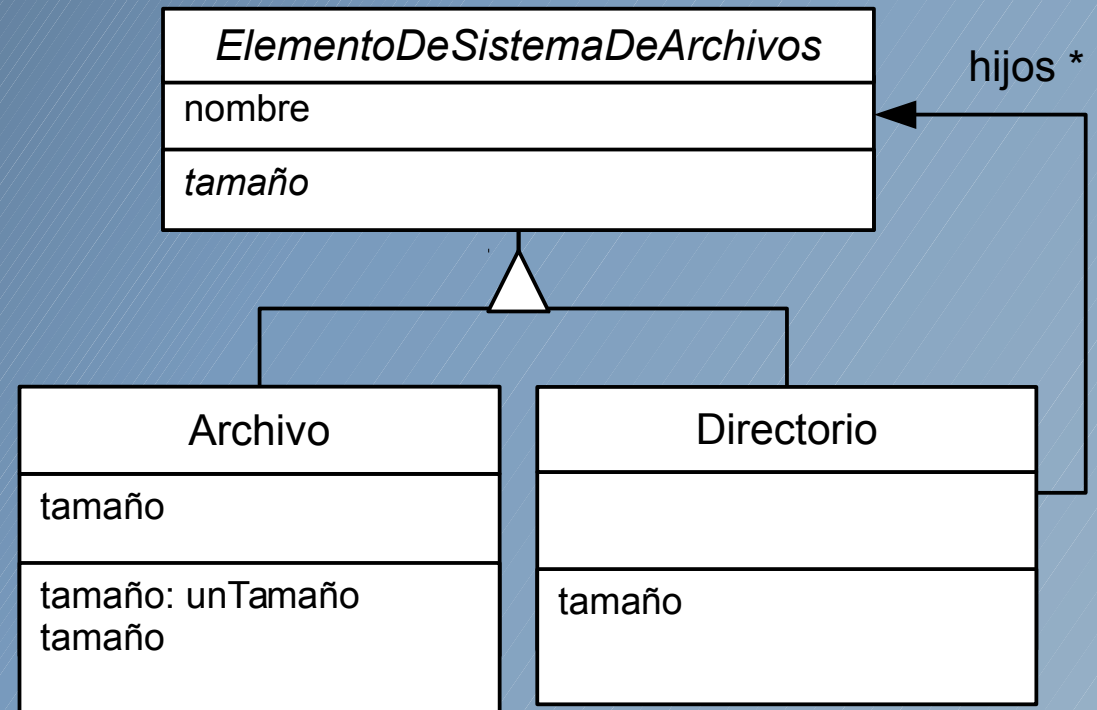
Ejemplo

- Notar que los directorios se pueden anidar

```
[ 543K] Pharo.tcc
[ 93K] Pharo.png
[ 862] pharo
[ 285] README.txt
[246M] shared
[1.4M] Correcciones.changes
[ 23M] Correcciones.image
[198K] HackathonDelivery.changes
[ 22M] HackathonDelivery.image
[1.5M] Hackaton.changes
[ 25M] Hackaton.image
[493K] package-cache
[ 77K] Artefact-Core-OlivierAuverlot.249.mcz
[ 19K] Artefact-Examples-OlivierAuverlot.2.mcz
[8.6K] Artefact-Tests-OlivierAuverlot.2.mcz
[7.4K] ConfigurationOfArtefact-OlivierAuverlot.14.mcz
[5.1K] ConfigurationOfMinimalConnectors-GuillermoPolito.24.mcz
[ 11K] ConfigurationOfObjectBrowser-ClaraAllende.100.mcz
[ 11K] ConfigurationOfSmallUML-ClaraAllende.88.mcz
[6.6K] ConfigurationOfUnits-StephaneDucasse.12.mcz
[8.3K] MinimalConnectors-ConnectableShapes-GuillermoPolito.21.mcz
```


Diseño

```
[ 345K] Pharo.tcc
[ 93K] Pharo.png
[ 862] pharo
[ 285] README.txt
[246M] shared
[1.4M] Correcciones.changes
[ 23M] Correcciones.image
[198K] HackathonDelivery.changes
[ 22M] HackathonDelivery.image
[1.5M] Hackaton.changes
[ 25M] Hackaton.image
[493K] package-cache
[ 77K] Artefact-Core-OlivierAuv
[ 19K] Artefact-Examples-Olivie
[8.6K] Artefact-Tests-OlivierAu
```



Implementación

ElementoDeSistemaDeArchivos>>tamaño

```
^self subclassResponsibility.
```

Archivo>>tamaño

```
^tamaño.
```

Directorio>>tamaño

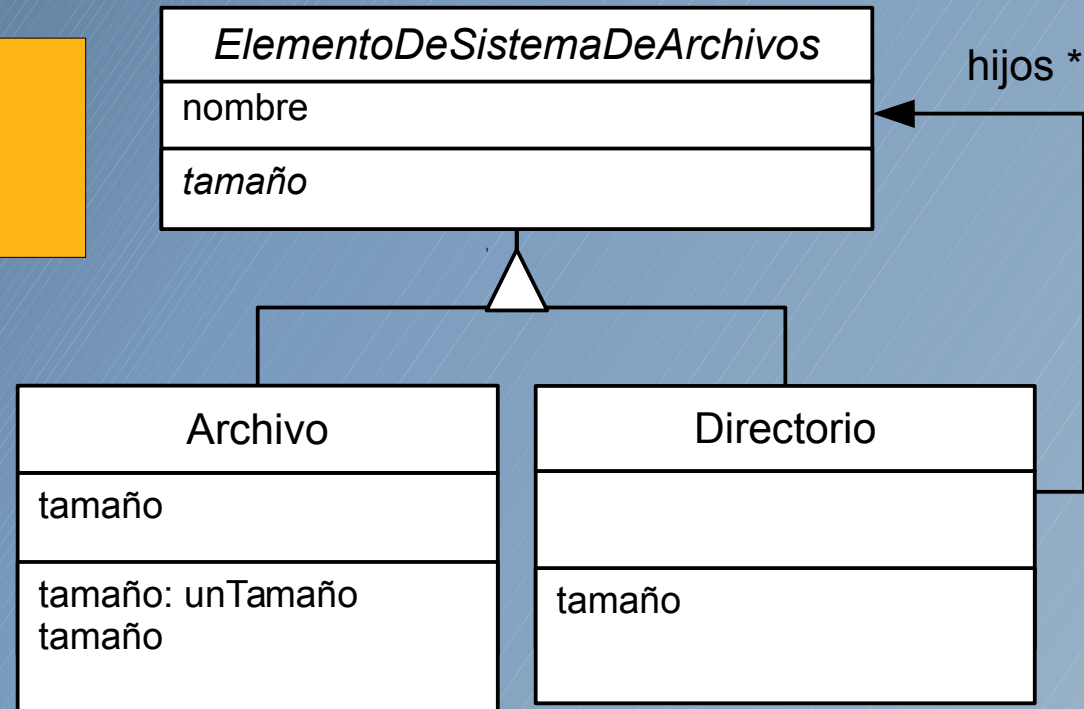
```
| suma |
```

```
suma := self hijos
```

```
inject: 0
```

```
into: [:sum :elemento | sum + elemento tamaño].
```

```
^suma + 4096.
```



Mejorando la implementación - Opción 1

```
Directorio>>tamaño
```

```
| suma |
```

```
suma := self hijos
```

```
    inject: 0
```

```
    Into: [:sum :elemento | sum + elemento tamaño].
```

```
^suma + self tamañoPropio.
```

```
Directorio>>tamañoPropio
```

```
^4096.
```

Mejorando la implementación - Opción 1

```
Directorio>>tamaño
```

```
^self tamañoPropio + self tamañoHijos.
```

```
Directorio>>tamañoPropio
```

```
^4096.
```

```
Directorio>>tamañoHijos
```

```
^self hijos
```

```
  inject: 0
```

```
  Into: [:sum :elemento | sum + elemento tamaño].
```


Mejorando la implementación - Opción 2

```
Directorio>>tamaño
```

```
| suma |
```

```
suma := self hijos
```

```
    inject: 0
```

```
    Into: [:sum :elemento | sum + elemento tamaño].
```

```
^suma + self tamañoPropio.
```

```
Directorio>>tamañoPropio
```

```
^4096.
```

Mejorando la implementación - Opción 2

```
Directorio>>tamaño
```

```
  ^self hijos
```

```
    inject: self tamañoPropio.
```

```
    Into: [:sum :elemento | sum + elemento tamaño].
```

```
Directorio>>tamañoPropio
```

```
  ^4096.
```

Tests

- Caso simple (archivo).
- Caso compuesto (directorio)
 - Vacío.
 - Un elemento.
 - N elementos.

Algunos recordatorios

- Ya está disponible la Práctica 5.
- El 20/10 tienen que entregar el ejercicio de la Práctica 4.
- El 30/10 es la primera fecha del parcial.
- El 10/11 es la entrega de la Práctica 5.