MARINA BELMONTE
B-6253/7

[Module Cuentas

EXPORTS obtenerSaldo(): int
obtenerNombre(): String
obtenerDescripcion(): String
obtenerFecha(): String
Actualizar(i Cuentas)
Eliminar(i Cuentas)
obtenerApunte(i int): Cuentas
EjecutarOperacion(i operacion)]

[Module GrupoDeCuentas INHERITS FROM Cuentas]

[Module CuentaRegular INHERITS FROM Cuentas]

[Module ASIENTO INHERITS FROM Cuentas

EXPORTS CuentaOrigen(): CuentaRegular
CuentaDestino(): CuentaDestino]

→ Debe/Haber()

[Module Operacion
IMPORTS CuentaRegular, GrupoDeCuentas, Asiento
EXPORTS EnCuentaRegular(i CuentaRegular)
EnGrupoDeCuentas(i GrupoDeCuentas)
EnAsiento(i Asiento)]

[Module VisualizarOperacion INHERITS FROM Operacion]

[Module GuardarOperacion INHERITS FROM Operacion
IMPORTS BD]

[Module RecuperarOperacion INHERITS FROM Operacion
IMPORTS BD]

[Module FormularioS
IMPORTS FECHA, Remito, Cheque
EXPORTS CrearFECHA(): FECHA
CrearRemito(): Remito
CrearCheque(): Cheque]

[Module FormArgentina INHERITS FROM FormularioS]

[Module FormPeru INHERITS FROM FormularioS]

[Module FormItalia INHERITS FROM FormularioS]

~~Modulo~~

NOTA

[module FECHA
exports monto(): INT
descripcion(): string
origen(): string
destino(): string
...]

[module Cheque
exports monto(): INT
origen(): string
destino(): string
descripcion(): string
...]

[module Remito
exports monto(): INT
descripcion(): string
origen(): string
destino(): string
...]

[module FECHAAR inherits from FECHA]
[module FECHAPER inherits from FECHA]
[module FECHAITZLI inherits from FECHA]
[module ChequeAR inherits from Cheque]
[module ChequePER inherits from Cheque]
[module ChequeITZLI inherits from Cheque]
[module RemitoAR inherits from Remito]
[module RemitoPER inherits from Remito]
[module RemitoITZLI inherits from Remito]

[module PlazoDeCuentas
imports Cuentas
exports crearplazo(i string)
eliminarplazo(i string)
AñadirGrupo(i GrupoDeCuentas)]

[module Sistema
imports Formulario
exports crearCliente(i string)
FECHA(i cliente, i INT, i string)
EMITIRCheque(i cliente, i INT, i string)
EMITIRRemito(i cliente, i INT, i string)]

MARINA BELMONTE
B-6253/7

Pattern — Plan de cuentas

based on — Composite

because
- Un plan de cuentas tiene estructura parte-todo
- permite tratar de manera uniforme las cuentas y asientos

where

Cliente IS PlanDeCuentas

Componente IS Cuentas
operacion() IS obtenerSaldo()
operacion() IS obtenerNombre()
operacion() IS obtenerDescripcion()
operacion() IS obtenerFecha()
Añadir(iComponente) IS Añadir(i Cuentas)
Eliminar(iComponente) IS Eliminar(i Cuentas)
obtenerHijo(i INT) IS obtenerHijo(i INT)

Compuesto IS GrupoDeCuentas
operacion() IS obtenerSaldo()
: : :
obtenerHijo(i INT) IS obtenerHijo(i INT)

Compuesto IS CuentaReal
operacion() IS obtenerSaldo()
:
obtenerHijo(i INT) IS obtenerHijo(i INT)

Hoja IS Asiento
operacion() IS obtenerSaldo()
" IS obtenerNombre()
" IS obtenerDescripcion()
" IS obtenerFecha()
" IS CuentaOrigen()
" IS CuentaDestino()

NOTA

Pattern operaciones

based on visitor

because - se pueden agregar operaciones sin modificar la estructura de las clases.

where ESTRUCTURA de objetos IS: plaza de toros

Elemento IS CLASES
ACEPTAR() IS EJECUTAR()

ElementoConcretoA IS CLASEReal
ACEPTAR() IS EJECUTAR()

ElementoConcretoB IS GRUPOdeCLASES
ACEPTAR() IS EJECUTAR()

ElementoConcretoC IS ASIENTO
ACEPTAR IS EJECUTAR()

VISITANTE IS OPERACION
(+) VISITARelementoConcretoA() IS ENCLASEREAL()
VISITARelementoConcretoB() IS ENGRUPODECLASES()
VISITARelementoConcretoC() IS ENASIENTO()

VISITANTEConcreto1 IS VISUALIZAR
(+)

VISITANTEConcreto2 IS GUARDAR
(+)

VISITANTEConcreto3 IS RECUPERAR
(+)

NOTA

MARINA BELMONTE
B-6253/7

Pattern Formularios

based on AbstractFactory

because - se puede cambiar de país en tiempo de ejecución
- permite agregar nuevos formularios de forma sencilla

where

Cliente IS Sistema

FabricaAbstracta IS Formularios
CrearProductoA() IS CrearFactura()
CrearProductoB() IS CrearCheque()
CrearProductoC() IS CrearRemito()

FabricaConcreta1 IS FormArgentina
CrearProductoA() IS CrearFactura()
CrearProductoB() IS CrearCheque()
CrearProductoC() IS CrearRemito()

FabricaConcreta2 IS FormPeru

FabricaConcreta3 IS FormItalia

ProductoA IS Factura

ProductoA1 IS FacturaAR
ProductoA2 IS FacturaPeru
ProductoA3 IS FacturaItalia

ProductoB IS Cheque

ProductoB1 IS ChequeAR
ProductoB2 IS ChequePeru
ProductoB3 IS ChequeItalia

ProductoC IS Remito

ProductoC1 IS RemitoAR
ProductoC2 IS RemitoPeru
ProductoC3 IS RemitoItalia

b)

```
int Asiento.obtenerSaldo() {
    return haber - debe
}

int CuentaReal.obtenerSaldo() {
    i = 0, saldo = 0
    while (obtenerHijo(i) ≠ null) {
    n = obtenerHijo(i).saldo
    saldo = saldo + n
    i = i + 1
    } return saldo }
```

La implementación de GrupoDeCuentas.obtenerSaldo() es igual a la de CuentaReal.

```
String Asiento.obtenerNombre() {
    return nombre
}
```

Los implementadores de GrupoDeCuentas y CuentaReal son iguales a la de Asiento.

Los operadores obtenerDescripción y obtenerFecha son iguales de triviales. Simplemente son return descripción y return fecha, donde éstas son variables de estado.

Además, los operadores de Asiento, cuentaOrigen() y cuentaDestino() también simplemente se devuelve el valor de las variables internas.

NOTA

MARINA BELMONTE
B-6253/7

```
GrupodeCuentas.Añadir (i cuenta) {
    IF (type(cuenta) ≠ ASIENTO) { hijos # [cuenta] }
    else { // lanzar excepción }
}}
cuentaReal.Añadir (i cuenta) {
IF (type(cuenta) == ASIENTO) { hijos # [cuenta] }
else {// lanzar excepción
}}
~~GrupodeCuentas~~
ASIENTO. Añadir (i cuenta) {
   // lanzar excepción
}
```

~~La operación~~
La operación Eliminar es análoga, solo que se sacará el elemento cuenta pasado como parámetro del array de hijos.

No desechable

d)

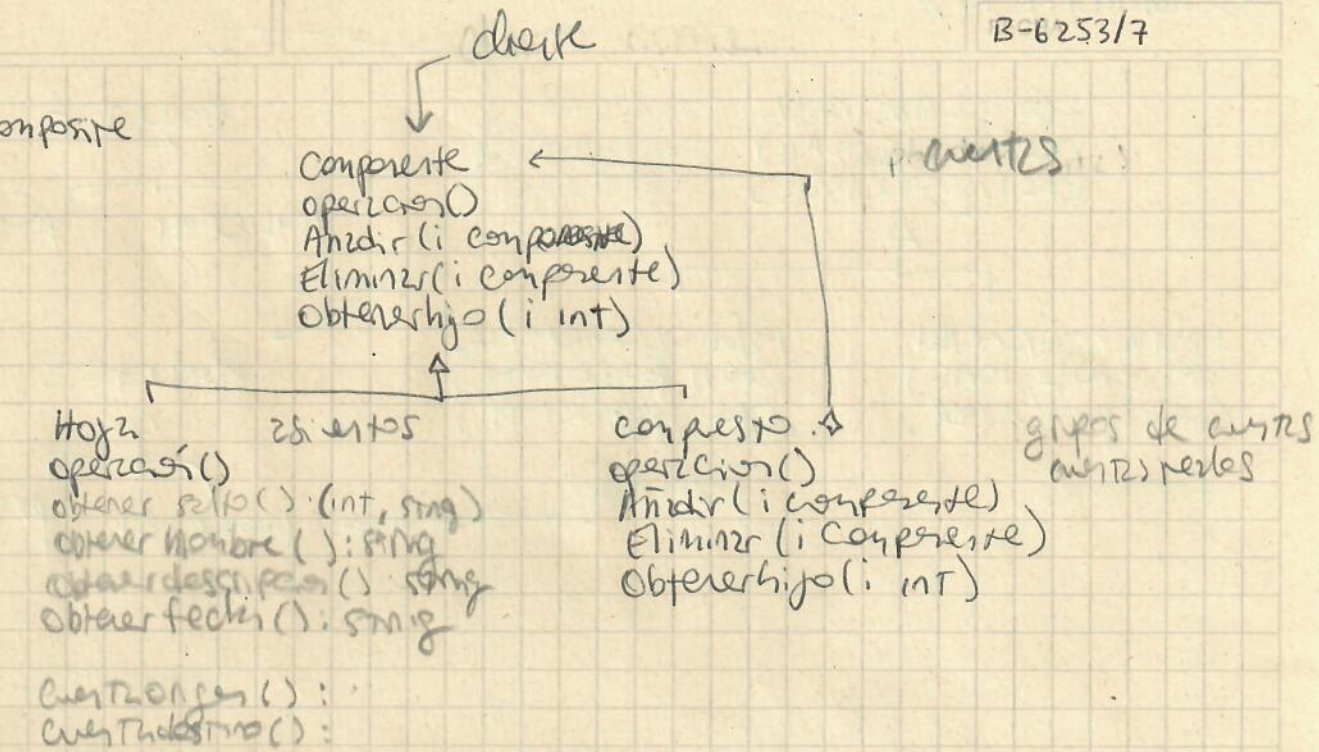
```
Asiento.Ejecutar(i operacion) {
    operacion.EnAsiento(this)
}

CuentaRaiz.Ejecutar(i operacion) {
    operacion.EnCuentaRaiz(this)
}

GrupoDeCuentas.Ejecutar(i operacion) {
    operacion.EnGrupoDeCuentas(this)
}

Guardar.EnAsiento(i Asiento) {
    BD.Guardar(Asiento)
}

Guardar.EnCuentaRaiz(i cuentaRaiz) {
    i=0
    while (cuentaRaiz.obtenerHijo(i) ≠ null) {
        Guardar.EnAsiento(obtenerHijo(i)) // guarda info hijos
        i = i+1
    }
    BD.Guardar(cuentaRaiz)
}

Guardar.EnGrupoDeCuentas(i GrupoDeCuentas) {
    i=0, hijo = GrupoDeCuentas.obtenerHijo(i)
    while (hijo ≠ null) {
        Guardar.EnGrupoDeCuentas(hijo)
        i = i+1, hijo = GrupoDeCuentas.obtenerHijo(i)
    }
    BD.Guardar(GrupoDeCuentas)
}
```

el visitante deberá saber/conocer Asiento

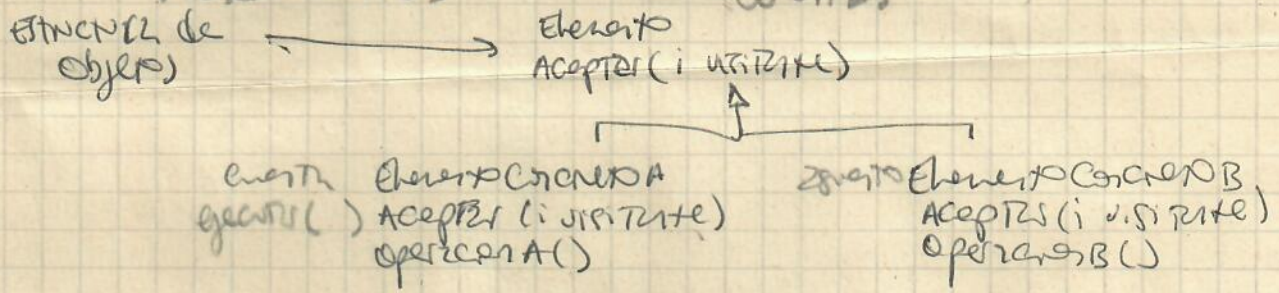
NOTA

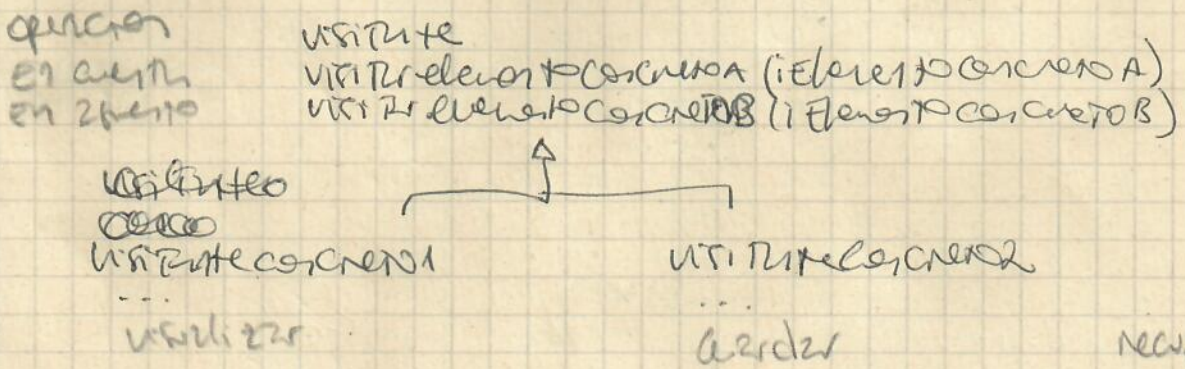
MARINA BELMONTE
B-6253/7

a) composite

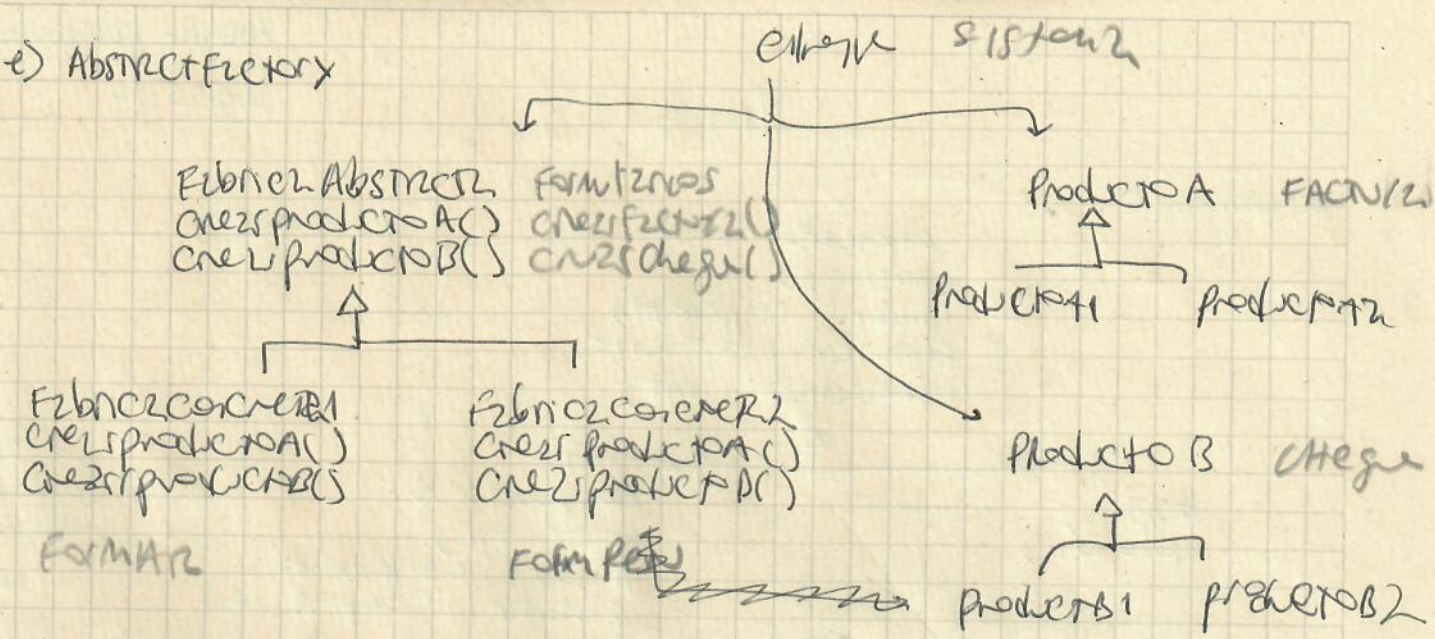


b) visitor





e) ABSTRACT FACTORY



FORMATO

NOTA