# EJEMPLO DE PRUEBAS UNITARIAS Y DE INTEGRACIÓN.

**Escuela Superior de Informática (UCLM)**

**Ingeniería Técnica en Informática de Sistemas.**

**Ingeniería del Software.**

Se desean realizar las pruebas unitarias y de integración de las 3 clases cuyo código se ofrece a continuación:

**Cliente.java:**

```
import java.util.Vector;

public class Cliente {
    String mNIF, mNombre;
    Vector mFacturas;

    public Cliente(String nif, String nombre) {
        mNIF=nif; mNombre=nombre; mFacturas=new Vector();
    }

    public void add(Factura f) {
        mFacturas.addElement(f);
    }

    public void show() {
        System.out.println("Facturas del cliente " + mNombre + ":");
        for (int i=0; i<mFacturas.size(); i++) {
            System.out.println("Factura " + (i+1));
            ((Factura) mFacturas.elementAt(i)).show();
            System.out.println("-------------------\n\n");
        }
    }
}
```

**Factura.java:**

```
import java.util.Vector;

public class Factura implements Euro {
```

```
    String mNumero, mFecha;
    Linea mLineas[];

    public Factura(String n, String f) {
        mNumero=n; mFecha=f;
        mLineas=new Linea[10];
    }

    public void add(Linea l) {
        int i=0;
        for (i=0; mLineas[i]!=null; i++) ;
        mLineas[i]=l;
    }

    public void quitar(int i) {
        mLineas[i].mArticulo=null;
        mLineas[i].mPrecio=0;
    }

    public void show() {
        double total=0;
        System.out.println(mNumero + "; " + mFecha);
        for (int i=0; mLineas[i]!=null; i++) {
            mLineas[i].show();
            total+=mLineas[i].mPrecio;
        }
        System.out.println("\tTotal .... " + total + " pts.");
        System.out.println("\t          " + (total/kCambio) + " euros");
    }
}
```

**Linea.java:**

```
public class Linea {
    String mArticulo;
    double mPrecio;

    public Linea(String a, double p) {
        mArticulo=a;
        mPrecio=p;
    }

    public void show() {
        System.out.println("\t" + mArticulo + " ..... " + mPrecio + " pts");
    }
}
```

**Euro.java:**

```
interface Euro {
    final double kCambio = 166.386;
}
```

**MATRICES DE USO DE MIEMBROS DE DATOS.**

| **Cliente** | Cliente | add | show |
|---|---|---|---|
| mNIF | t | | |
| mNombre | t | | o |
| mFacturas | t | t | o |

| **Factura** | Factura | add | quitar | show |
|---|---|---|---|---|
| mNumero | t | | | o |
| mFecha | t | | | o |
| mLineas | t | t | t | o |

| **Línea** | Linea | show |
|---|---|---|
| mArticulo | t | o |
| mPrecio | t | o |

## PRUEBAS UNITARIAS

### 1. Pruebas unitarias de Cliente:

1.1 Corte de mNIF:

El único método que hay en este corte es el constructor. Para comprobar el funcionamiento de este constructor podemos hacer dos cosas:

a) Añadir un método getNIF() para ver que devueleva mNIF

b) Construir una clase con permiso para acceder a los atributos de Cliente, de manera que muestre el valor de mNIF.

Si optamos por la 2ª opción, dicha clase podría ser la siguiente:

```
public class Principal {
    public static void main(String args[]) throws Exception {
        Cliente c=new Cliente(null, null);
        System.out.println("NIF: " + c.mNIF);
        System.in.read();
    }
}
```

- Caso de prueba **111**: El resultado de la ejecución del fragmento de código anterior, que ejecuta el caso de prueba en el que mNIF=null, es el de la siguiente figura:


- Caso de prueba **112**: Otra prueba puede ser la siguiente:

Cliente c=new Cliente(new String(), null); ← Nótese que el valor del 2º parámetro no me importa, ya que estoy en el corte correspondiente a mNIF, al que se le da valor en el primer parámetro del constructor.

1.2. Corte de mNombre:

Cliente(String, String)

show()

Como en show() se utiliza el atributo mFacturas, que es de un tipo no básico, lo obviamos en este tipo de pruebas.

Casos de prueba:

- **121:** Cliente c=new Cliente(null, null); c.show()

```
Facturas del cliente null:
Factura 1
1/2001; 21-05-2001
        Agua ..... 25.0 pts
        Leche ..... 85.0 pts
        Total .... 110.0 pts.
                   0.6611133148221604 euros
--------------------


Factura 2
2/2001; 21-05-2001
        Pan ..... 50.0 pts
        Mistol ..... 240.0 pts
        Gel ..... 195.0 pts
        Total .... 485.0 pts.
                   2.9149087062613437 euros
--------------------
```

- **122:** Cliente c=new Cliente(new String(), new String()); c.show();

```
Facturas del cliente :
Factura 1
1/2001; 21-05-2001
        Agua ..... 25.0 pts
        Leche ..... 85.0 pts
        Total .... 110.0 pts.
                   0.6611133148221604 euros
--------------------


Factura 2
2/2001; 21-05-2001
        Pan ..... 50.0 pts
        Mistol ..... 240.0 pts
        Gel ..... 195.0 pts
        Total .... 485.0 pts.
                   2.9149087062613437 euros
--------------------
```

- **123:** Cliente c=new Cliente("5.655.999J", "Paco Pil"); c.show();

```
Facturas del cliente Paco Pil:
Factura 1
1/2001; 21-05-2001
        Agua ..... 25.0 pts
        Leche ..... 85.0 pts
        Total .... 110.0 pts.
                   0.6611133148221604 euros
--------------------


Factura 2
2/2001; 21-05-2001
        Pan ..... 50.0 pts
        Mistol ..... 240.0 pts
        Gel ..... 195.0 pts
        Total .... 485.0 pts.
                   2.9149087062613437 euros
--------------------
```

1.3. Corte de mFacturas.

Puesto que éste es un atributo que representa un objeto agregado, y que además no es de un tipo básico, y que además no ha sido probado, no probamos este corte.

## 2. Pruebas unitarias de Factura.

2.1 Corte de mNumero.

Influyen aquí Factura(String, String) y show().

- Caso de prueba **211**.

Factura f=new Factura(null, null); f.show();

```
Symantec Java! JustInTime Com
Copyright (C) 1996-99 Symante

null; null
        Total .... 0.0 pts.
                   0.0 euros
_
```

- Caso de prueba **212**.

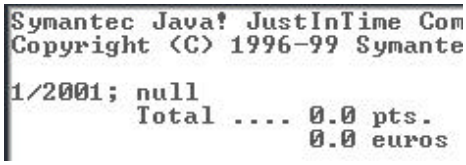
Factura f=new Factura(new String(), null); f.show();

```
Symantec Java! JustInTime Com
Copyright (C) 1996-99 Symantec

; null
        Total .... 0.0 pts.
                   0.0 euros
_
```

- Caso de prueba **213**.

Factura f=new Factura("1/2001", null); f.show();




2.2. Corte de mFecha.

Los casos de prueba del corte de mNumero pueden probarse para este corte con resultados satisfactorios.

2.3. Corte de mLineas.

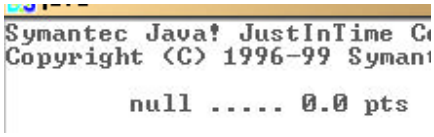
Como este atributo representa una colección de objetos agregados, que no son de un tipo básico (int, float...) y no ha sido probado, no realizamos pruebas de unidad para este corte.

## 3. Pruebas de unitarias de Linea.

3.1. Corte de mArticulo

- Caso de prueba **311**.

Linea l=new Linea(null, 0); l.show();




- Caso de prueba **312**.

Linea l=new Linea(new String(), 0); l.show();


- Caseo de prueba **313**.

Linea l=new Linea("Agua fresca", 0); l.show();


3.2. Corte de mPrecio.

- Caso de prueba **321**.

Sirve el caso de prueba 311, en el que esperamos el valor 0 y obtenemos el valor 0.

- Caso de prueba **322.**

Linea l=new Linea("Agua fresca", null); l.show();

```
Symantec Java! JustInTime Compiler Ver
Copyright (C) 1996-99 Symantec Corpora

        Agua fresca ..... 100.0 pts
```

**Conclusiones sobre los resultados de las pruebas unitarias:**

No se ha realizado interpretación de los resultados de las pruebas porque carecemos de requisitos funcionales y no sabemos, por tanto, cómo debe comportarse este sistema ante los valores 0, null, etc.

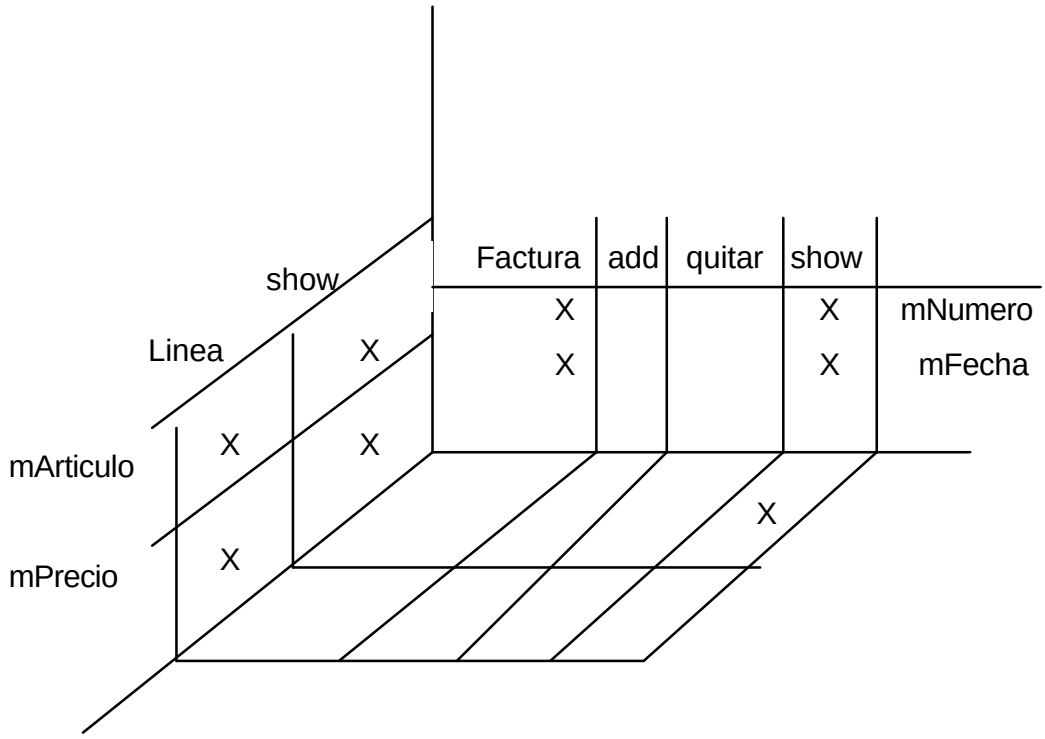
Podemos suponer que el cliente nos ha pedido como único requisito que no haya errores en tiempo de ejecución, con lo que todos los casos de prueba han resultado satisfactorios.

## PRUEBAS DE INTEGRACIÓN

A la hora de realizar estas pruebas, debemos tener en cuenta el nivel de complejidad de cada integración, en lo que influye mucho el grado de interdependencia.

En nuestro sencillo ejemplo, parece claro que debemos comenzar probando la integración de Factura con Linea.

**1. Pruebas de integración de Factura con Linea.**

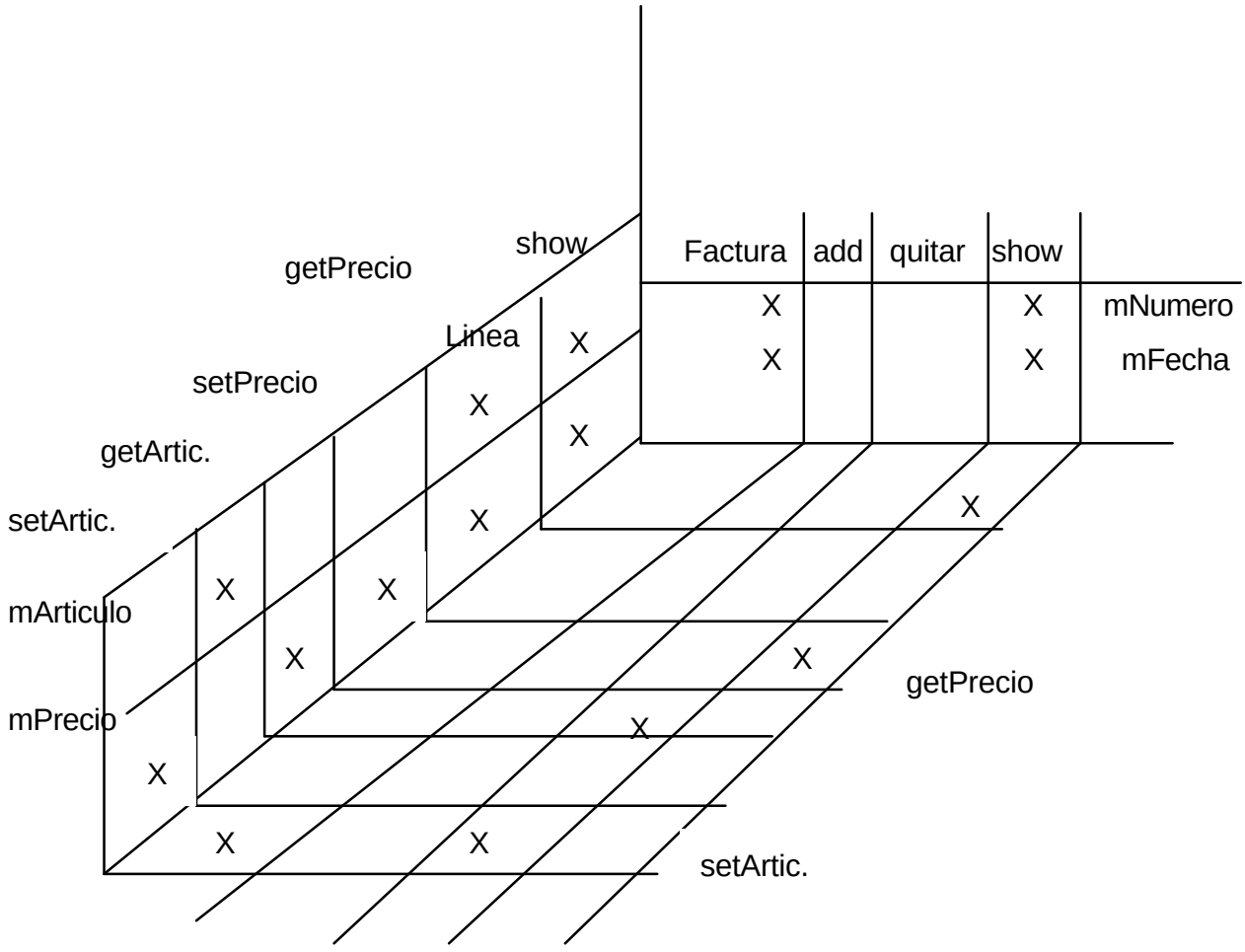



En la matriz anterior notamos cosas raras que impiden seguir ejecutando las pruebas de integración: los métodos Factura(String, String) y quitar(int) acceden a mLineas directamente a los atributos de la clase Linea, lo cual está prohibido. Deberíamos, por tanto, tener métodos get() y set(--) para todos los atributos de Linea.

Los añadimos y volvemos a realizar pruebas de unidad en Linea. Hemos añadido los métodos siguientes:

```
public void setArticulo(String a)  { mArticulo=a; }
    public String getArticulo() { return mArticulo; }
    public void setPrecio(double p) { mPrecio=p; }
    public double getPrecio() { return mPrecio; }
```

Funcionan perfectamente, con lo cual volvemos a las pruebas de integración.




Con la matriz de ahora, sí que podemos continuar las pruebas de integración de Factura con Linea. Siguiendo los pasos que vimos en clase, tras dibujar la “matriz 3D”, determinamos los casos de prueba, que, en principio, serán los mismos que al realizar las pruebas de unidad de la clase Factura.

Igual que con las pruebas de unidad, probamos corte a corte.

2.1. Corte de mNumero.

En las pruebas de unidad, ejecutamos estos casos de prueba:

**211:** Factura f=new Factura(null, null); f.show();

**212:** Factura f=new Factura(new String(), null); f.show();

**213:** Factura f=new Factura("1/2001", null); f.show();

Los tres casos consisten en una llamada al constructor de Factura y una llamada a su método show(). Para continuar, miramos qué métodos de Linea son llamados desde los métodos de Factura que pertenecen a este corte:

- Factura(String, String) llama a setArticulo(double)
- show() llama a show() y a setPrecio(double)

Podríamos volver a ejecutar los casos de prueba 211 a 213. Sin embargo, debemos dar valores de prueba al atributo mLineas, añadiendo objetos de clase Linea mediante el método add(Linea):

- Caso de prueba **I-211-1.**

Factura f=new Factura(null, null);
Linea l=new Linea(null, 0);
f.add(l);
f.show();

```
null; null
        null ..... 0.0 pts
        Total .... 0.0 pts.
                   0.0 euros
```

- Caso de prueba **I-211-2**.

Factura f=new Factura(null, null);
Linea l=new Linea(null, 0); Linea l2=new Linea("Coca cola", 1000);
f.add(l); f.add(l2);
f.show();

```
null; null
        null ..... 0.0 pts
        Coca cola ..... 1000.0 pts
        Total .... 1000.0 pts.
                   6.010121043837822 euros
```

- Caso de prueba **I-211-3**.

Factura f=new Factura(null, null);
Linea l=new Linea(null, 0);
Linea l2=new Linea(null, 0.001);

```
null; null
        null ..... 0.0 pts
        null ..... 0.0010 pts
        Total .... 0.0010 pts.
                   6.010121043837822E-6 euros
```

```
f.add(l); f.add(l2);
f.show();
```

- Caso de prueba **I-211-4.**

```
Linea lineas[]={
  new Linea("Agua", 25), new Linea("Leche", 85),
  new Linea("Pan", 50), new Linea("Mistol", 240),
  new Linea("Gel", 195), new Linea("Tomates", 70),
  new Linea("Agua 2", 25), new Linea("Leche 2", 85),
  new Linea("Pan 2", 50), new Linea("Mistol 2", 240),
  new Linea("Gel 2", 195), new Linea("Tomates 2", 70)
};
Factura f=new Factura(null, null);
try {
  for (int i=0; i<lineas.length; i++)
        f.add(lineas[i]);
        f.show();
}
catch (Exception e) {System.out.println(e);}
```

```
Copyright (C) 1996-99 Symantec Corporation

java.lang.ArrayIndexOutOfBoundsException
```

**APÉNDICE.**

a) Código de Factura tras añadir métodos set y get a Linea:

```
import java.util.Vector;
public class Factura implements Euro {
    String mNumero, mFecha;
    Linea mLineas[];

    public Factura(String n, String f) {
        mNumero=n; mFecha=f;
        mLineas=new Linea[10];
    }

    public void add(Linea l) {
        int i=0;
        for (i=0; mLineas[i]!=null; i++) ;
        mLineas[i]=l;
    }

    public void quitar(int i) {
        mLineas[i].setArticulo(null);
        mLineas[i].setPrecio(0);
    }

    public void show() {
        double total=0;
        System.out.println(mNumero + "; " + mFecha);
        for (int i=0; mLineas[i]!=null; i++) {
            mLineas[i].show();
            total+=mLineas[i].getPrecio();
        }
        System.out.println("\tTotal .... " + total + " pts.");
```

```
        System.out.println("\t          " + (total/kCambio) + " euros");
    }
}
```

b) Código de Linea tras añadirle métodos set y get.

```
public class Linea {
    private String mArticulo;
    private double mPrecio;

    public Linea(String a, double p) {
        mArticulo=a;
        mPrecio=p;
    }

    public void setArticulo(String a)  { mArticulo=a; }

    public String getArticulo() { return mArticulo; }

    public void setPrecio(double p) { mPrecio=p; }

    public double getPrecio() { return mPrecio; }

    public void show() {
        System.out.println("\t" + mArticulo + " ..... " + mPrecio + " pts");
    }
}
```