ERS PATRÓN BREVE PATRÓN DE USOS TEMPLATE

Análisis y diseño de software
2. Grado en Ingeniería Informática – 2. semestre

Autor: Ibai Rodríguez Ruiz

Contenido

ERS Formato Breve – Patrón de Usos

1. Introducción

Este documento contiene información sobre el patrón de uso llamado Template, que gestiona en este caso el salario de los empleados que contiene la Asociación Afro situado en Vitoria. El objetivo de esta breve especificación es definir de manera clara y precisa la funcionalidad del patrón Template y ver cómo este afecta en el sistema.

a. Significado

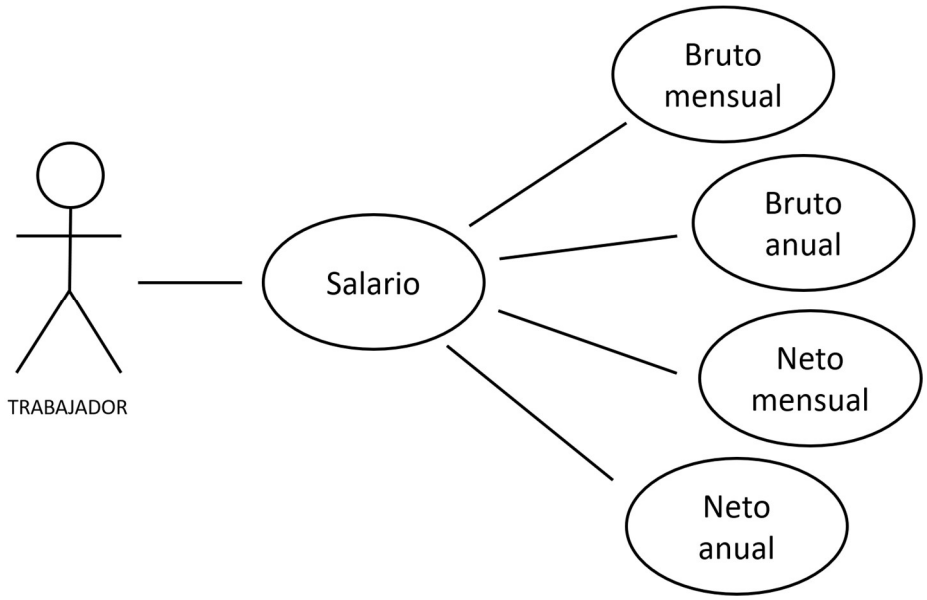
El patrón de método Template o plantilla es un patrón de diseño de comportamiento que define el esqueleto del programa de un algoritmo en un método, llamado método de plantilla, el cual difiere algunos pasos a las subclases. Permite redefinir ciertos pasos seguros de un algoritmo sin cambiar la estructura del algoritmo.

2. Uso del patrón Template

El sistema muestra el salario de un trabajador de la asociación de diferentes maneras.

El usuario que utiliza el sistema tendrá diferentes opciones de visualización del salario de los trabajadores. Bruto y neto de forma mensual o anual.

3. Caso de uso



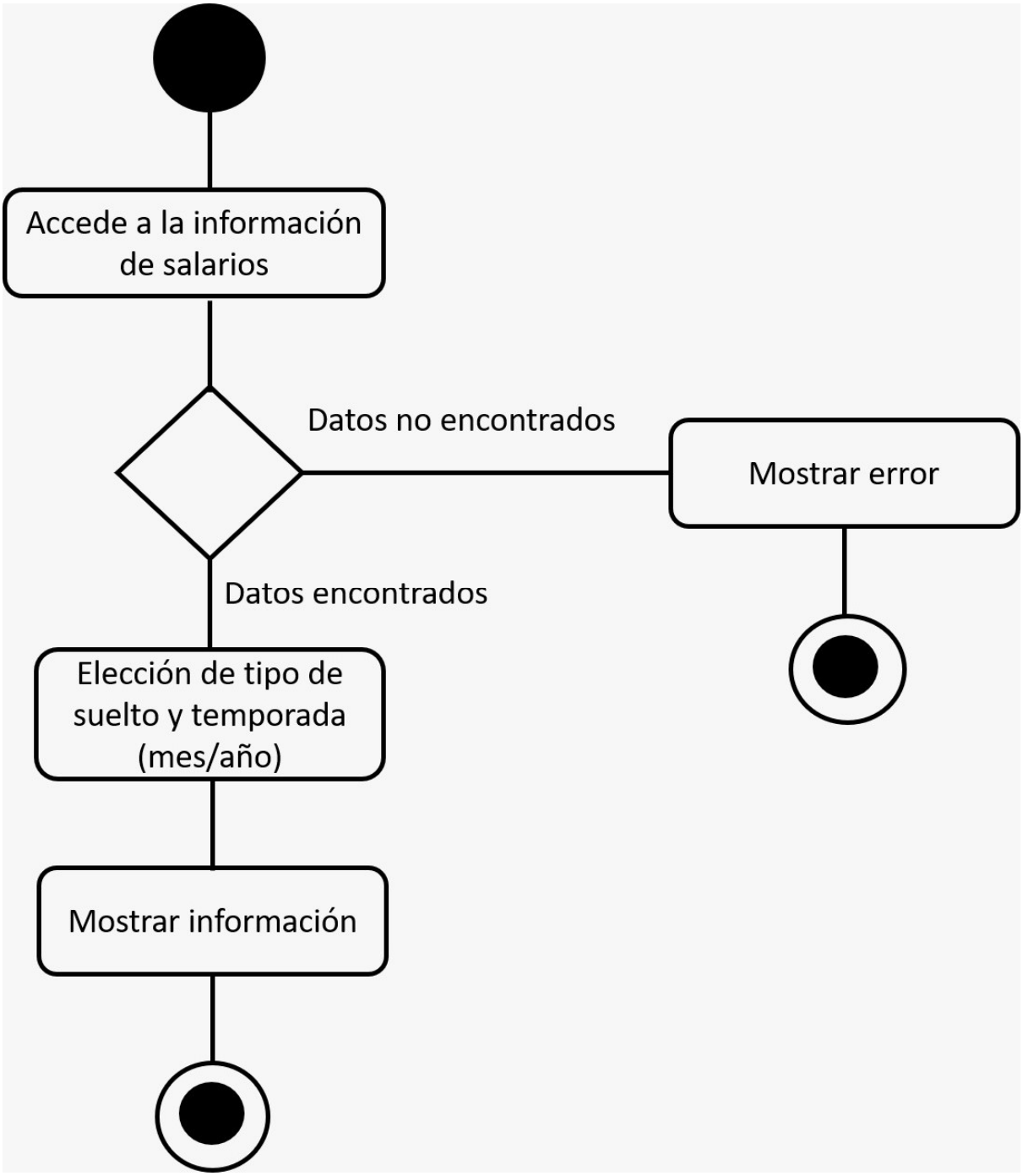
Este patrón se activará en el momento que el trabajador/voluntario quiera acceder a la información del salario. En él tendrá diferentes opciones, donde deberá elegir el tipo de salario bruto/neto y mensual/anual.

4. Escenario

Caso de uso: Información salarios	
Actor: Trabajador	
Curso normal	**Curso alternativo**
1.Se inicia el sistema	
2. El usuario accede a la información de salarios	
3. El usuario elige opción de tipo de salario y temporada (mes/año)	3.1 El usuario no elige la opción
4. El usuario puede visualizar la opción elegida	

5. Fin del caso de uso	

5. Diagrama de actividad



6. Implementación del patrón Template

Corresponde a la clase que crea la ventana de Trabajador

```
private Component crearPanelDatos() {
    JPanel panel = new JPanel(new GridLayout(12, 1));

    if (trabajador instanceof Empleado)
        sueldoCambiante = ((Empleado) trabajador).getSalario();
    identificador = new JLabel(String.valueOf(this.trabajador.getId()));
    nombre = new JLabel(this.trabajador.getNombre());
    apellido1 = new JLabel(this.trabajador.getApellido1());
    apellido2 = new JLabel(this.trabajador.getApellido2());
    fnac = new JLabel(this.trabajador.getFnac());
    nacionalidad = new JLabel(this.trabajador.getNacionalidad());
    direccion = new JLabel(this.trabajador.getDireccion());
    documentoIdentidad = new JLabel(this.trabajador.getDocumentoIdentidad());
    telefono = new JLabel(String.valueOf(this.trabajador.getTelefono()));
    genero = new JLabel(String.valueOf(this.trabajador.getGenero()));
    email = new JLabel(this.trabajador.getEmail());
    puesto = new JLabel(this.trabajador.getPuesto());
    username = new JLabel(this.trabajador.getUsername());
    //Crea la parte donde se vera el codigo
    sueldo = new JLabel(String.valueOf(this.sueldoCambiante));
    password = new JLabel(this.trabajador.getPassword());
    //Crea la opcion para elegir el salario
    tipoSueldo = new JComboBox<String>();
    tipoSueldo.addItemListener(this);
    tipoSueldo.setSelectedItem(null);
      puesto = new JLabel(this.trabajador.getPuesto());
      username = new JLabel(this.trabajador.getUsername());
      //Crea la parte donde se vera el codigo
      sueldo = new JLabel(String.valueOf(this.sueldoCambiante));
      password = new JLabel(this.trabajador.getPassword());
      //Crea la opcion para elegir el salario
      tipoSueldo = new JComboBox<String>();
      tipoSueldo.addItemListener(this);
      tipoSueldo.setSelectedItem(null);

      panel.add(crearComponentBorder(identificador, "Identificador"));
      panel.add(crearComponentBorder(nombre, "Nombre"));
      panel.add(crearComponentBorder(apellido1, "Primer Apellido"));
      panel.add(crearComponentBorder(apellido2, "Segundo Apellido"));
      panel.add(crearComponentBorder(genero, "Genero"));
      panel.add(crearComponentBorder(fnac, "Fecha de nacimiento"));
      panel.add(crearComponentBorder(direccion, "Domicilio"));
      panel.add(crearComponentBorder(nacionalidad, "Nacionalidad"));
      panel.add(crearComponentBorder(documentoIdentidad, "Documento de identidad"));
      panel.add(crearComponentBorder(telefono, "Teléfono"));
      panel.add(crearComponentBorder(email, "Email"));
      panel.add(crearComponentBorder(puesto, "Puesto"));
      //Visualiza el tipo de salario
      panel.add(crearComponentBorder(tipoSueldo, "Tipo de vista denSueldo"));
      //Visualiza el salario
      panel.add(crearComponentBorder(sueldo, "Sueldo"));
```

```
    // Funcion espera asta que una de las opciones este elegida, cuando sea asi,
    // añade el valor
    @Override
    public void itemStateChanged(ItemEvent e) {

        // Elegimos Bruto Mensual
        if (tipoSueldo.getSelectedItem().equals("Bruto Mensual")) {
            BrutoMensual bM = new BrutoMensual();
            // Realiza las operaciones
            sueldoCambiante = bM.Operacion(sueldoCambiante);

            // Elegimos Neto Mensual
        } else if (tipoSueldo.getSelectedItem().equals("Neto Mensual")) {
            NetoMensual nM = new NetoMensual();
            // Realiza las operaciones
            sueldoCambiante = nM.Operacion(sueldoCambiante);

            // Elegimos Bruto Anual
        } else if (tipoSueldo.getSelectedItem().equals("Bruto Anual")) {
            BrutoAnual bA = new BrutoAnual();
            // Realiza las operaciones
            sueldoCambiante = bA.Operacion(sueldoCambiante);

            // Elegimos Neto Anual
        } else if (tipoSueldo.getSelectedItem().equals("Neto Anual")) {
            NetoAnual nA = new NetoAnual();
            // Realiza las operaciones
            sueldoCambiante = nA.Operacion(sueldoCambiante);

        }
    }
```

Son las funciones internas de Template

```
package PatronesDeDiseño;

//Clase General para todas las  funciones de Template, extienden de aqui
public abstract class Sueldo {

    //Constructor
    public float obtener(float num) {
        float numero = this.Operacion(num);
        return numero;
    }

    // Funcion Abstracta general
    public abstract float Operacion(float num);

}
```

```java
package PatronesDeDiseño;

//Realizaz la operacion de Neto Anual
public class NetoAnual extends Sueldo
{
    public NetoAnual(){
    }
    // ------------------------------------
    @Override
     public float Operacion( float num )
    {
        num=((num*15)/100);
        return (num*12);
    }
}
```

```java
package PatronesDeDiseño;

//Realiza las operaciones para conseguir el sueldo neto de forma mensual
public class NetoMensual extends Sueldo
{
    public NetoMensual(){
    }
    // ------------------------------------
    @Override
     public float Operacion( float num )
    {
        num=((num*15)/100);
        return num;
    }
}
```

```java
package PatronesDeDiseño;

//Calcula el sueldo Bruto mensual de un empleado
public class BrutoMensual extends Sueldo
{
    public BrutoMensual(){
    }
    // ------------------------------------
    @Override
     public float Operacion( float num )
    {
        return num;
    }
}
```

```java
package PatronesDeDiseño;

//Calcula el sueldo bruto Anual
public class BrutoAnual extends Sueldo
{
    public BrutoAnual(){
    }
    // ------------------------------------
    @Override
     public float Operacion( float num )
    {
        return (num*12);
    }
}
```