



## **INSTITUCIÓN**

Escuela Superior Politécnica del Litoral

## **MATERIA**

CCPH1042: Diseño de Software

## **TEMA**

TALLER #8 - REFACTORING

## **AUTORES**

Víctor García

Rommel Zamora

## **PROFESOR**

Msc. David Jurado

## Code Smells

- Inappropriate intimacy.

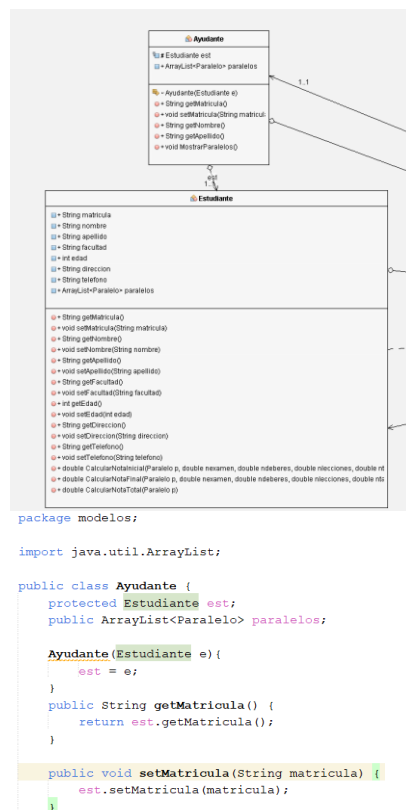
Problemas:

- No es fácil mantenerla, ni reutilizar.

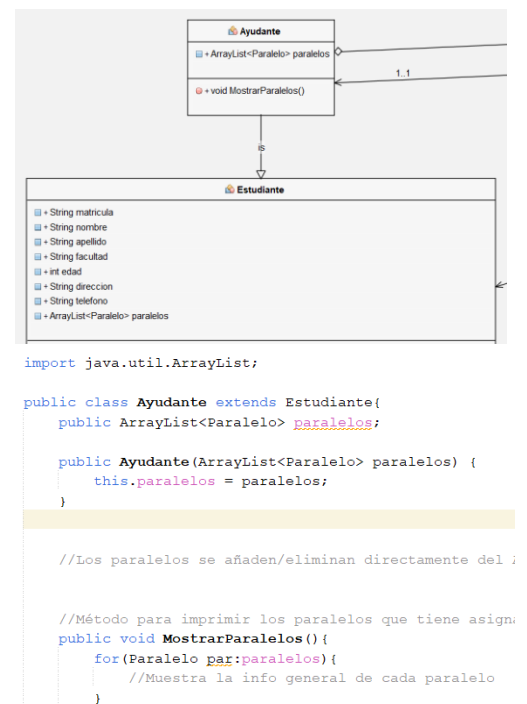
Tratamiento

- Usar *Replace delegation with Herence*:
  - Mejora la organización del código.
  - Simplifica la reutilización del código.
  - Reducción de recursos, debido que la longitud del código decrece.

Code bad:



Code Good:



- Inline Class.

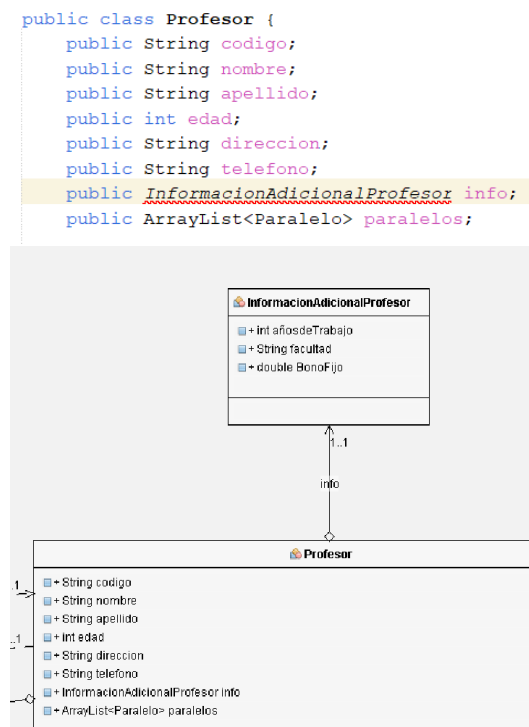
Problemas:

- La class *InformacionAdicionalProfesor* no tiene responsabilidades, solo guarda información de la class *Profesor*.
- Hacer cambios en la class principal, requiere que haga pequeños cambios en clases diferente.

Tratamiento:

- Shotgun Surgery:
  - Mejor Organización
  - Menos codigo duplicado
  - Fácil entendimiento

Code bad:



Code Good:



- Large Class.

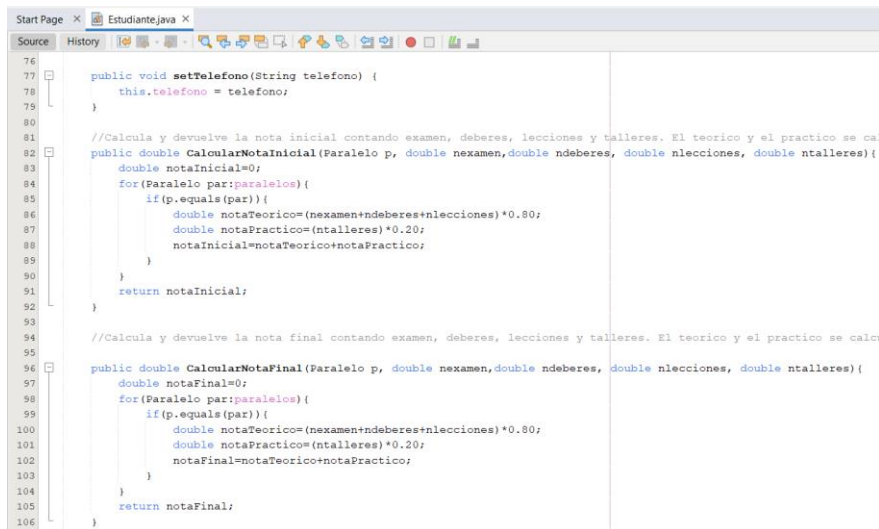
#### Problemas:

- La clase Estudiante es un poco mas larga de lo que debería ser
- Estudiante tiene métodos que no necesitan implementarse dentro de ella

#### Tratamientos:

- Extract class:
  - Los métodos `CalcularNotaFinal()` y `CalcularNotaInicial()` pueden moverse a una clase encargada de los cálculos, como lo es la clase “`CalcularSueldoProfesor`” y se la renombra a “`Calculos`”.

#### Bad Code:

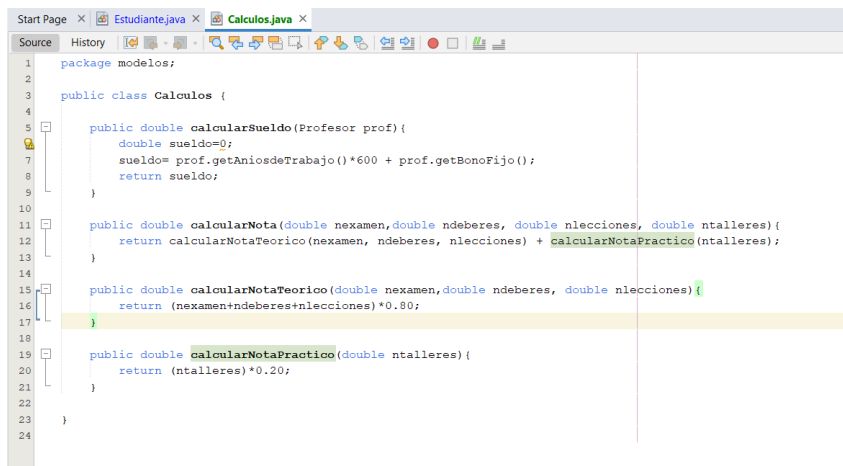


```

76
77 public void setTelefono(String telefono) {
78     this.telefono = telefono;
79 }
80
81 //Calcula y devuelve la nota inicial contando examen, deberes, lecciones y talleres. El teorico y el practico se cal
82 public double CalcularNotaInicial(Paralelo p, double nexamen, double ndeberes, double nlecciones, double ntalleres){
83     double notaInicial=0;
84     for(Paralelo par:paralelos){
85         if(p.equals(par)){
86             double notaTeorico=(nexamen+ndeberes+nlecciones)*0.80;
87             double notaPractico=(ntalleres)*0.20;
88             notaInicial=notaTeorico+notaPractico;
89         }
90     }
91     return notaInicial;
92 }
93
94 //Calcula y devuelve la nota final contando examen, deberes, lecciones y talleres. El teorico y el practico se calcu
95
96 public double CalcularNotaFinal(Paralelo p, double nexamen, double ndeberes, double nlecciones, double ntalleres){
97     double notaFinal=0;
98     for(Paralelo par:paralelos){
99         if(p.equals(par)){
100             double notaTeorico=(nexamen+ndeberes+nlecciones)*0.80;
101             double notaPractico=(ntalleres)*0.20;
102             notaFinal=notaTeorico+notaPractico;
103         }
104     }
105     return notaFinal;
106 }

```

#### Good Code:



```

1 package modelos;
2
3 public class Calculos {
4
5     public double calcularSueldo(Profesor prof){
6         double sueldo=0;
7         sueldo= prof.getAniosdeTrabajo()*600 + prof.getBonoFijo();
8         return sueldo;
9     }
10
11     public double calcularNota(double nexamen, double ndeberes, double nlecciones, double ntalleres){
12         return calcularNotaTeorico(nexamen, ndeberes, nlecciones) + calcularNotaPractico(ntalleres);
13     }
14
15     public double calcularNotaTeorico(double nexamen, double ndeberes, double nlecciones){
16         return (nexamen+ndeberes+nlecciones)*0.80;
17     }
18
19     public double calcularNotaPractico(double ntalleres){
20         return (ntalleres)*0.20;
21     }
22 }
23
24

```

La siguiente sección de código presenta varios code smells:

```
80
81 //Calcula y devuelve la nota inicial contando examen, deberes, lecciones y talleres. El teorico y el practico se calcula por parcial.
82 public double CalcularNotaInicial(Paralelo p, double nexamen, double ndeberes, double nlecciones, double ntalleres){
83     double notaInicial=0;
84     for(Paralelo par:paralelos){
85         if(p.equals(par)){
86             double notaTeorico=(nexamen+ndeberes+nlecciones)*0.80;
87             double notaPractico=(ntalleres)*0.20;
88             notaInicial=notaTeorico+notaPractico;
89         }
90     }
91     return notaInicial;
92 }
93
94 //Calcula y devuelve la nota final contando examen, deberes, lecciones y talleres. El teorico y el practico se calcula por parcial.
95
96 public double CalcularNotaFinal(Paralelo p, double nexamen, double ndeberes, double nlecciones, double ntalleres){
97     double notaFinal=0;
98     for(Paralelo par:paralelos){
99         if(p.equals(par)){
100             double notaTeorico=(nexamen+ndeberes+nlecciones)*0.80;
101             double notaPractico=(ntalleres)*0.20;
102             notaFinal=notaTeorico+notaPractico;
103         }
104     }
105     return notaFinal;
106 }
```

Code Smells:

- Duplicate code:
  - En la clase Estudiante estos métodos (CalcularNotaInicial() y CalcularNotaFinal()) tienen una función muy similar, ambos calculando una nota a partir de parámetros que tienen el mismo fin.
- Long Parameter List:
  - Los métodos tienen 5 parámetros cada uno.
  - Los hace más difícil de entender
- Speculative Generality:
  - Los métodos tienen “Paralelo p” como parámetro de entrada, pero este no tiene un uso justificable.
  - La inclusión del parámetro objeto Paralelo en el método, o su ausencia no afecta el valor de retorno de este.

Tratamientos posibles:

- Remove Parameter:
  - Se elimina el parámetro paralelo de los métodos ya que este no afecta al resultado final.
  - Con esto también se resuelve el code smell “Long parameter List” ya que pasa de tener 5 a tener 4, algo más aceptable.
- Extract Method:
  - Se crea un método que realice la misma función que los otros 2.
  - Menos código duplicado

Resultado luego de la corrección:

```
1 public double CalcularNota(double nexamen, double ndeberes, double nlecciones, double ntalleres){
2     double notaFinal=0;
3     double notaTeorico=(nexamen+ndeberes+nlecciones)*0.80;
4     double notaPractico=(ntalleres)*0.20;
5     notaFinal=notaTeorico+notaPractico;
6     return notaFinal;
7 }
```

- Temporary Field

Problemas:

- Métodos más largos
- Aumenta complejidad

Tratamientos:

- Replace Temp with Query:
  - Facilita la lectura del código
  - En caso de que se quiera volver a usar la porción de código corregida, evita duplicación.

Code bad:

```

96 public double CalcularNotaFinal(Paralelo p, double nexamen, double ndeberes, double nlecciones, double ntalleres) {
97     double notaFinal=0;
98     for (Paralelo par: paralelos) {
99         if (p.equals(par)) {
100             double notaTeorico=(nexamen+ndeberes+nlecciones)*0.80;
101             double notaPractico=(ntalleres)*0.20;
102             notaFinal=notaTeorico+notaPractico;
103         }
104     }
105     return notaFinal;
106 }
107
108 //Calcula y devuelve la nota inicial contando examen, deberes, lecciones y talleres. Esta nota es solo el promedio
109 public double CalcularNotaTotal(Paralelo p) {
110     double notaTotal=0;
111     for (Paralelo par: paralelos) {
112         if (p.equals(par)) {
113             notaTotal=(p.getMateria().notaInicial+p.getMateria().notaFinal)/2;
114         }
115     }
116     return notaTotal;
117 }

```

Corrección luego de aplicar los tratamientos previos:

```

1 public double CalcularNota(double nexamen, double ndeberes, double nlecciones, double ntalleres) {
2     return CalcularNotaTeorico(nexamen, ndeberes, nlecciones) + CalcularNotaPractico(ntalleres);
3 }
4
5 public double CalcularNotaTeorico(double nexamen, double ndeberes, double nlecciones) {
6     return (nexamen+ndeberes+nlecciones)*0.80;
7 }
8
9 public double CalcularNotaPractico(double ntalleres) {
10     return (ntalleres)*0.20;
11 }

```

- Feature Envy

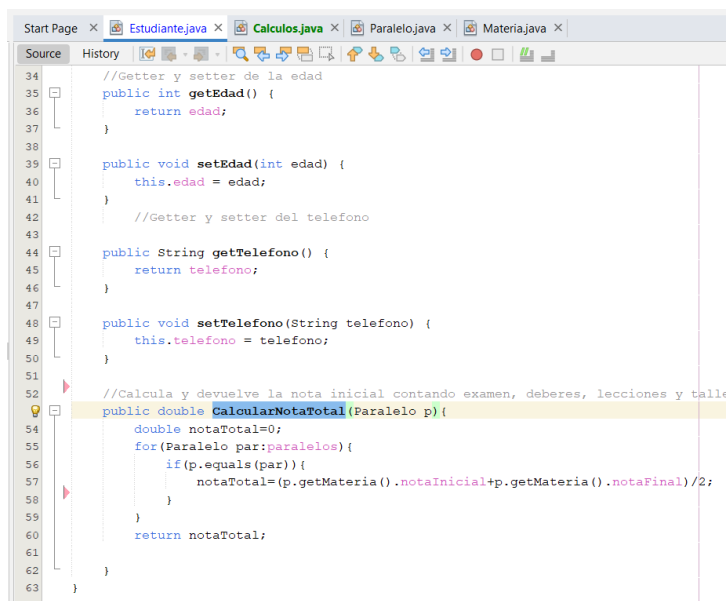
Problema:

- La clase Estudiante tiene el método CalcularNotaTotal(), que no utiliza atributos de la clase que lo implementa.

Tratamiento:

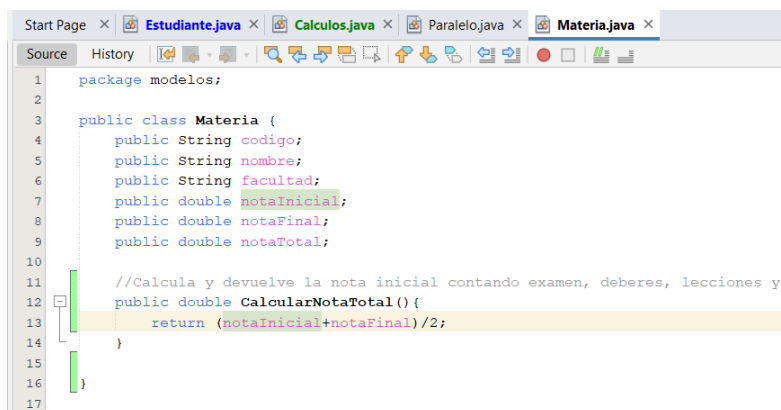
- Move Method:
  - La clase Estudiante no debería tener el método CalcularNotaTotal() ya que este no accede a sus atributos, sino que accede a un atributo de objeto Materia en la clase Paralelo. Por lo que este método debería ser movido a la clase Materia.

Bad Code:



```
34 //Getter y setter de la edad
35 public int getEdad() {
36     return edad;
37 }
38
39 public void setEdad(int edad) {
40     this.edad = edad;
41 }
42 //Getter y setter del telefono
43
44 public String getTelefono() {
45     return telefono;
46 }
47
48 public void setTelefono(String telefono) {
49     this.telefono = telefono;
50 }
51
52 //Calcula y devuelve la nota inicial contando examen, deberes, lecciones y talleres
53 public double CalcularNotaTotal(Paralelo p){
54     double notaTotal=0;
55     for(Paralelo par:paralelos){
56         if(p.equals(par)){
57             notaTotal=(p.getMateria().notaInicial+p.getMateria().notaFinal)/2;
58         }
59     }
60     return notaTotal;
61 }
62
63
64 }
```

Good Code:



```
1 package modelos;
2
3 public class Materia {
4     public String codigo;
5     public String nombre;
6     public String facultad;
7     public double notaInicial;
8     public double notaFinal;
9     public double notaTotal;
10
11 //Calcula y devuelve la nota inicial contando examen, deberes, lecciones y talleres
12 public double CalcularNotaTotal(){
13     return (notaInicial+notaFinal)/2;
14 }
15
16 }
17 }
```