

**Universidad de las Fuerzas Armadas** **ESPE**

**Departamento de Ciencias de la Computación**

**Análisis y Diseño**

**Integrantes:** Isaac Escobar, Eduardo Mortensen, Diego Ponce

**NRC:** 14571

TRABAJO EN GRUPO (5ptos)

Ejemplo de SRP (Single Responsibility Principle) en Java PATRON SOLID

1. ***¿Qué es SRP?***

SRP SRP significa Single Responsibility Principle (Principio de Responsabilidad Única) que establece que una clase debe tener una sola razón para cambiar, es decir, debe cumplir con una sola responsabilidad o función específica. De esta manera, cualquier modificación afecta únicamente a una parte específica del sistema, facilitando el mantenimiento y reduciendo posibles errores derivados de cambios inesperados.

1. ***Ejemplo que NO cumple SRP***

La siguiente clase mezcla varias responsabilidades claramente diferenciadas:

public class Estudiante {  
 private int id;  
 private String nombre;  
  
 public Estudiante(int id, String nombre) {  
 this.id = id;  
 this.nombre = nombre;  
 }  
  
 public void guardarEnBD() {  
 // Código que guarda en la base de datos  
 System.out.println("Guardando en BD...");  
 }  
  
 public void imprimir() {  
 // Código que imprime en consola  
 System.out.println("Estudiante: " + nombre);  
 }  
}

1. ***Anàlisis del Problema***

Esta clase tiene 3 responsabilidades distintas  
  
1. Modelar los datos del estudiante (id, nombre).

2. Guardar la información en la base de datos (guardarEnBD()).

3. Mostrar o presentar los datos del estudiante en pantalla (imprimir()).

1. ***Ejemplo que SÍ cumple SRP***

Ahora se separan las responsabilidades en clases distintas:

1. Clase Modelo (solo datos):

public class Estudiante {  
 private int id;  
 private String nombre;  
  
 public Estudiante(int id, String nombre) {  
 this.id = id;  
 this.nombre = nombre;  
 }  
  
 // Getters y Setters  
 public int getId() { return id; }  
 public String getNombre() { return nombre; }  
}

1. Clase DAO (persistencia):

public class EstudianteDAO {  
 public void guardar(Estudiante e) {  
 // Código para guardar en BD  
 System.out.println("Guardando estudiante en la BD...");  
 }  
}

1. Clase Vista (presentación):

public class EstudiantePrinter {  
 public void imprimir(Estudiante e) {  
 System.out.println("Estudiante: " + e.getNombre());  
 }  
}

1. ***Ventajas de aplicar SRP***
2. Si cambias cómo se imprimen los datos, no tocas la clase Estudiante ni la clase de persistencia (EstudianteDAO), solo modificas EstudiantePrinter.
3. Si cambias cómo se guardan los datos en la base de datos, no tocas la clase modelo (Estudiante) ni la clase de presentación (EstudiantePrinter), solo modificas EstudianteDAO.
4. Cada clase tiene una sola razón para cambiar, lo que hace que el sistema sea más robusto, flexible, mantenible, y más fácil de extender y probar.

Rúbrica de Evaluation

|  |  |
| --- | --- |
| Criterio | Puntaje Máximo |
| Explica con claridad qué es SRP | 4 puntos |
| Identifica correctamente las responsabilidades mezcladas | 4 puntos |
| Muestra correctamente el ejemplo que sí cumple SRP | 4 puntos |
| Analiza ventajas de aplicar SRP | 4 puntos |
| Presenta el contenido de forma clara y ordenada | 4 puntos |
| TOTAL SOBER 20 PTOS |  |