S.O.L.I.D. Principles

POO - UNNOBA - 2020

Acoplamiento y Cohesión

- Bajo Acoplamiento
- Alta Cohesión

S.O.L.I.D. Principles

- **S: Single responsibility principle** o Principio de responsabilidad única
- **O: Open/closed principle** o Principio de abierto/cerrado
- **L: Liskov substitution principle** o Principio de sustitución de Liskov
- **I: Interface segregation principle** o Principio de segregación de la interfaz
- **D: Dependency inversion principle** o Principio de inversión de dependencia

Single Responsibility Principle

A class should have one, and only one, reason to change

- Gather together those things that change for the same reason and separate those things that change for different reasons.
- Debemos evitar que las clases realicen tareas que no son su responsabilidad.

Open/Closed principle

You should be able to extend a classes behavior, without modifying it.

- Software entities (classes, modules, functions, etc.) must be opened for an extension, but closed for modification.

Liskov Substitution principle

Let q(x) be a property provable about objects of x of type T. Then q(y) should be provable for objects y of type S where S is a subtype of T.

- Derived classes must be substitutable for their base classes
- Una subclase debe ser sustituible por su superclase, y si al hacer esto, el programa falla, estaremos violando este principio.

Interface segregation principle

Make fine grained interfaces that are client specific.

- Many client-specific interfaces are better than one general-purpose interface.
- Este principio establece que los clientes no deberían verse forzados a depender de interfaces que no usan.

Dependency inversion principle

Depend on abstractions, not on concretions.

- Los módulos de alto nivel no deberían depender de módulos de bajo nivel, sino de abstracciones.
- Las abstracciones no deberían depender de los detalles. Los detalles deberían depender de las abstracciones.