Advanced Software Engineering Task

DR/ Mahmood sakr

Sohila Ayman Mohamed Lashien

–

Machine Intelligence

—

24-2-2024

- **Differences between Architecture and design pattern.**

Architecture:-

Architecture is the overall structure of software , Developer chooses different design pattern according to the architecture specification and requirement, It's define the Accuracy of the component.

Design pattern:-

Design patterns are concerned with how the components are built , It's about particular solution.

Architecture comes in Designing phase	Design Patterns comes in Building phase.
Architectural pattern is like a blue print.	design pattern is actual implementation.
Architecture is base which everything else adhere to	design pattern is a way to structure classes to solve common problems.
All **Architecture is design pattern**	all **design pattern can not be architecture.**
MVC (Model View Controller)	Singleton

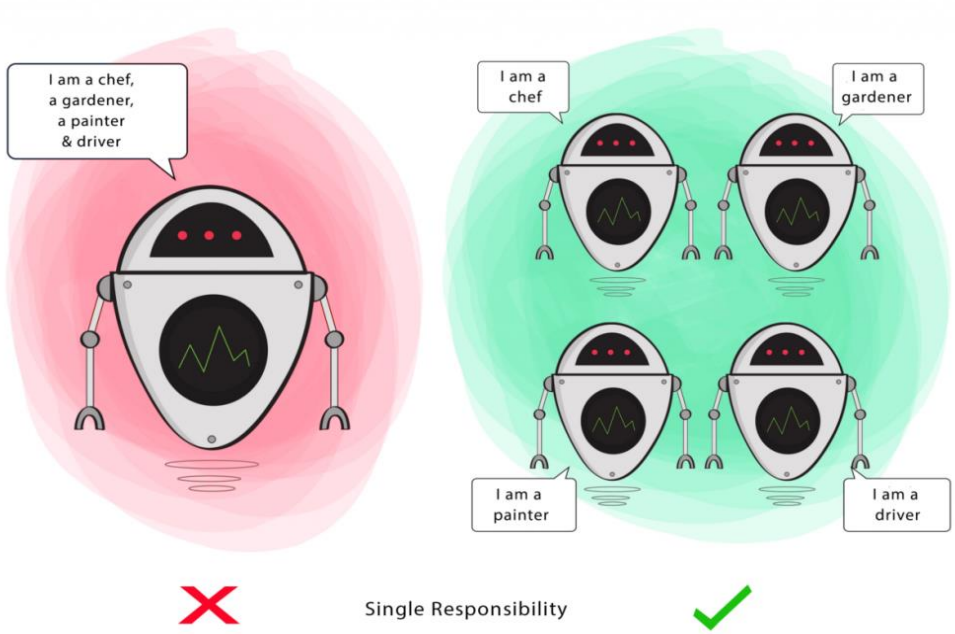
- **what is solid principles?**

SOLID stands for:-

S - Single-responsiblity Principle
O - Open-closed Principle
L - Liskov Substitution Principle
I - Interface Segregation Principle
D - Dependency Inversion Principle

Single-responsibility Principle (SRP) states:

A class should have one and only one reason to change, meaning that a class should have only one job.

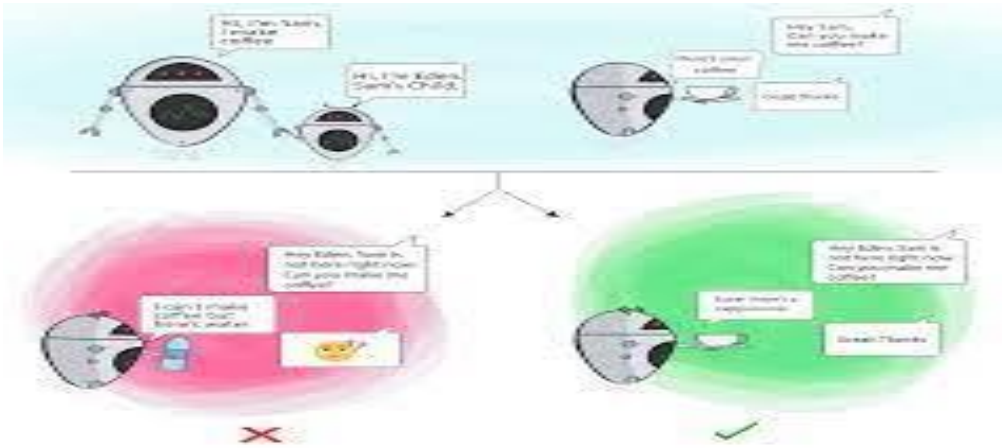


Open-closed Principle (OCP) states:

Objects or entities should be open for extension but closed for modification.

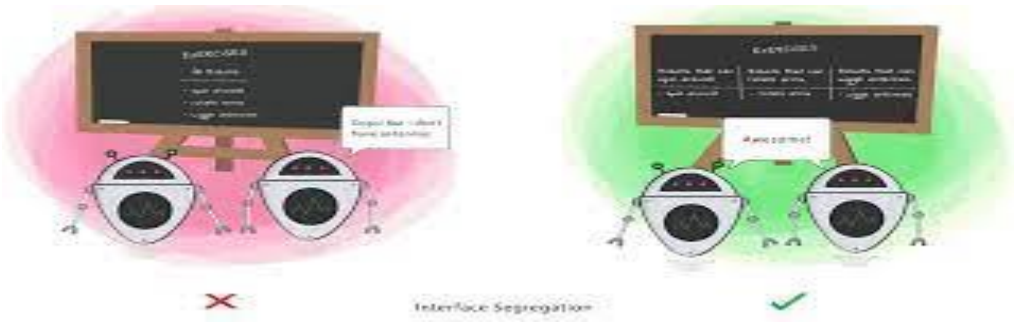
LSP - Liskov Substitution Principle:-

The Liskov Substitution Principle says that **a derived class must be substitutable for its base class without any problem.**



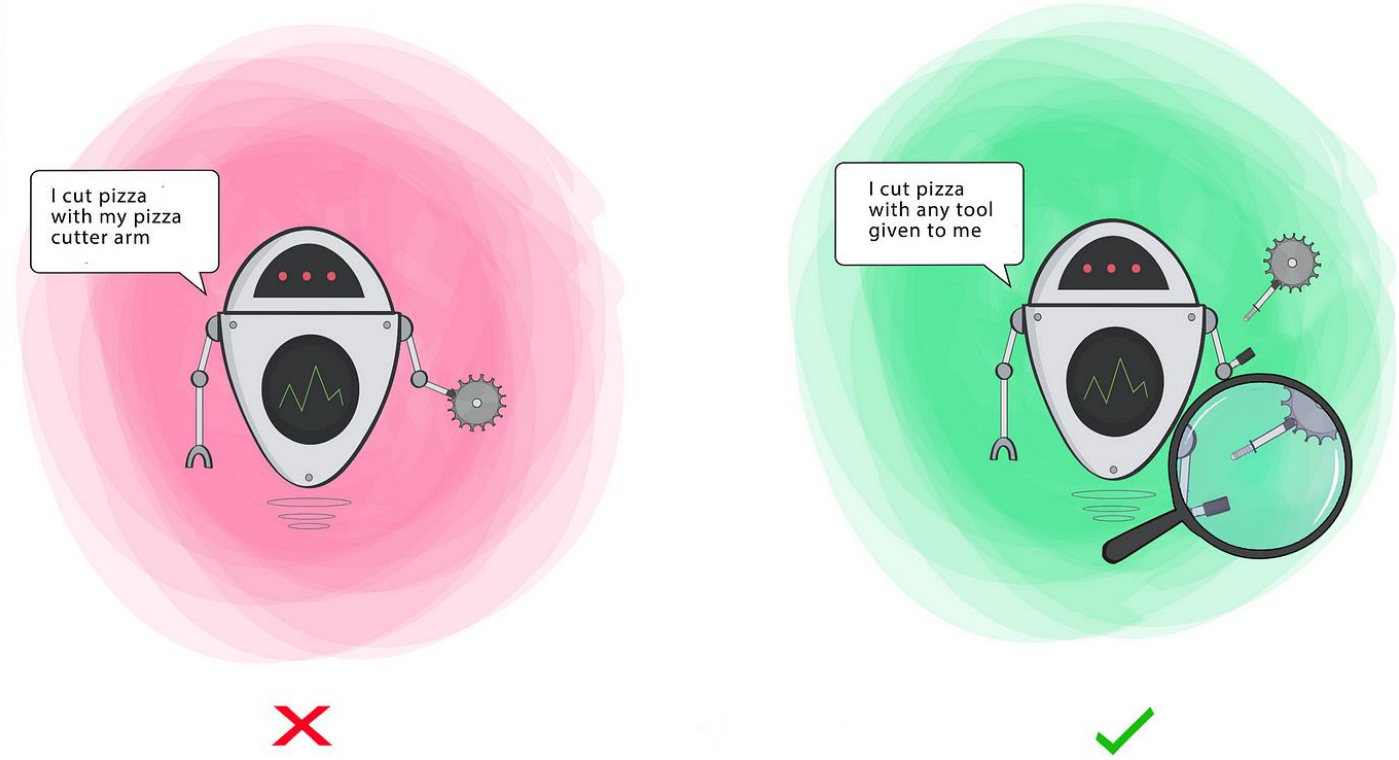
ISP - Interface Segregation Principle:-

This one says that **a class should not be forced to implement interfaces and methods it does not use.** It's better to create more specific interfaces than a big and generic one.



DIP - Dependency Inversion Principle:-

Depend on abstractions and not on implementations.



- **What is UML?**

  UML stands for Unified Modeling Language. It is a standardized modeling language used in software engineering for visualizing, specifying, constructing, and documenting the artifacts of a system.

  UML provides a set of graphical notations and a framework for creating visual models of software systems.

  <u>Some common types of UML diagrams include:</u>

  - Class Diagrams: Represent the static structure of a system, including classes, attributes, and relationships.

  - Use Case Diagrams: Illustrate the interactions between a system and its external actors, focusing on the system's functionality.