**Interface Segregation Principle**

ISP states "many client-specific interfaces are better than one general-purpose interface."

ISP splits interfaces that are very large into smaller and more specific ones so that clients will only have to know about the methods that are of interest to them. Such shrunken interfaces are also called **role interfaces**.

ISP is intended to keep a system decoupled and thus easier to refactor, change, and redeploy.

ISP is similar to the High Cohesion Principle of GRASP.

# Importance in object-oriented design

Within object-oriented design, interfaces provide layers of abstraction that facilitate conceptual explanation of the code and create a barrier preventing coupling to dependencies.

Using interfaces to further describe the intent of the software is often a good idea.

A system may become so coupled at multiple levels that it is no longer possible to make a change in one place without necessitating many additional changes. Using an interface or an abstract class can prevent this side effect.

# Example

class PrintTasks {

          bool PrintContent(string content) = 0;

          bool ScanContent(string content) = 0;

          bool FaxContent(string content) = 0;

          bool PhotocopyContent(string content) = 0;

          bool PrintDuplexContent(string content) = 0;

};

This is a single interface for a printer. But if there is a printer which can not perform fax and print duplex will not be able to use this interface.

So according to ISP principle we need to segregate this interface into smaller interface by making separate interface for fax and print duplex.

class PrintTasks {

          bool PrintContent(string content) = 0;

          bool ScanContent(string content) = 0;

          bool PhotocopyContent(string content) = 0;

};

class FaxContent {

          bool FaxContent(string content) = 0;

};

class PrintDuplexContent {

          bool PrintDuplexContent(string content) = 0;

};

# END