Juan Esteban Amaya

miércoles, 2 de diciembre de 2020 2:13 p. m.

1)

a) Significa que es mejor depender de una interfaz que de una clase para evitar que haya una índice alto de acoplamiento entre las clases y los módulos, y para cumplir con el principio solid de open/close, ya que puedo mejorar el código sin modificar el que ya realice. Un ejemplo seria ya tener una clase servicios, pero resulta que después yo le tengo que adicionar más servicios que los originalmente pautados, para esto lo mejor es crear una interfaz de serviviosnuevos para cumplir con la nueva pauta y no modificar el código ya creado.

b) Que la aplicación que yo cree está mal. El acoplamiento hace referencia a que tanto un componente sabe o requiere de otro componente. En un mundo perfecto lo ideal sería que no exista ningún tipo de acoplamiento es decir, que los dos componentes sean totalmente independientes el uno del otro, por esta razón lo que se busca es que haya un bajo acoplamiento(lo que según el ingeniero, no es mi caso). Y la cohesión hace referencia a que tan relacionadas están la operaciones de un módulo o componente. Lo que se busca es que cada uno de los componentes tengan funciones que estén lo más relacionadas entre sí y a esto se le conoce como alta cohesión(lo que según el ingeniero tampoco es mi caso)

c) Para poder utilizar un objeto, como si ese objeto tuviera diferentes tipos, dependiendo de cómo se invoque el objeto.

d) Esta el método de instancia y el método de clase. El primer tipo como su nombre lo dice, son métodos que puedo usar sobre la instancia, es decir en la clase en la que estoy trabajando. Y en el segundo tipo, a diferencia de los métodos de instancia, dentro de la clase también puedo tener otros métodos que no dependen de la instancia.

e) Por definición la clase abstracta, es la base de diferentes clases y puede tener métodos abstractos, a diferencia de la interfaz que tiene métodos específicos de otra clase. Pero la clase abstracta puede proporcionar la implementación de una interfaz, mientras que la interfaz no puede proporcionar la implementación de una clase abstracta

f) Un problema de la herencia es que tienen alto acoplamiento, es decir que de las dos clases (clase base y superclase) dependen una de la otra, porque están estrechamente unidas. El segundo problema es que no se pueden realizar cambios en la clase base, debido a que los cambios también deben realizarse en las clases secundarias.

g)Básicamente lo que nos dice este principio es toda clase que es hija de otra clase, debería poder utilizarse como si fuera la misma clase padre, nadie que necesite utilizar la clase padre tiene que comportarse diferente si interactúa con cualquiera de sus clases hijas.

2)

a)