METODOLOGÍAS DE PROGRAMACIÓN I

Patrones de diseño

Dr. Lic. Waldo Hasperué

Temario

- Patrones de diseño
  - ¿Qué son?
  - ¿Para que se usan?
  - ¿Por qué son importantes?
  - Clasificación y catálogo de patrones

Patrones de diseño

- Al comenzar con el desarrollo de un sistema en el paradigma de la programación orientada a objetos, una de las primeras tareas a realizar es el de determinar quienes serán los objetos que intervengan:
  - Clases con su estado y comportamiento ¿Cuántas? ¿Cuáles? ¿Qué hace cada una? ¿Qué conoce cada una?
  - Relaciones ¿Quién conoce a quién?
  - Jerarquías ¿Cómo armar la jerarquía? ¿Quién es subclase de quién? ¿Hay clases abstractas?

- El error más frecuente es comenzar el sistema desde cero.

Patrones de diseño

- No es para nada fácil desarrollar un sistema que sea sencillo de mantener y ampliar
- El software evoluciona. Muchas veces para realizar una modificación hay que hacer grandes reestructuraciones en el código, aparecen nuevas clases, nuevas relaciones, lo que hacía una clase ahora lo tiene que hacer otra, etc.
- Toda modificación en el código debe hacerse con mucho cuidado, el más mínimo cambio puede hacer que algo deje de funcionar como corresponde.
- ¿Cómo pensamos en el diseño de un sistema tal que sea fácil de mantener y ampliar?
- ¿Cómo hacemos para no empezar de cero?

Patrones de diseño

- Todo problema que necesites resolver, alguien ya lo resolvió antes. Eso significa que ya **existe** la solución a tu problema.
- Al diseñar software es muy común hacerse uno mismo la pregunta "*Si esto ya lo hice una vez ¿por qué lo estoy haciendo de nuevo?*"
- El desafío entonces, no es empezar de cero sino saber utilizar aquellas soluciones que ya existen y que funcionan.
- Esas soluciones se conocen como patrones de diseño.

¿Qué es un patrón de diseño?

Un patrón de diseño describe un problema que ocurre una y otra vez, así como también su solución. De modo tal que se pueda aplicar esa solución un millón de veces sin hacer lo mismo dos veces.

Christopher Alexander

Características de un patrón de diseño

- Todo patrón de diseño tiene:
  - Un nombre
    - Describe brevemente el problema y la solución
  - El problema que intenta resolver
    - Cuando aplicar el patrón. Describe el problema y su contexto
  - La solución al problema
    - Elementos que constituyen la solución. Describe las clases, las relaciones entre ellas y la responsabilidad de cada una.
  - Consecuencias
    - Resultados de aplicar el patrón. Describe las ventajas y consecuencias de utilizarlo.

Ventajas de usar patrones de diseño

- Tenemos la solución a un montón de problemas
- Aplicamos en el desarrollo del software un diseño que es conocido por la comunidad y que es fácil de mantener y de extender su funcionalidad
- Permiten ampliar el lenguaje entre los diseñadores y programadores de sistemas
  - "… para ese módulo usemos un *Strategy*…"
  - "… te conviene usar un *Proxy*…"
- Son independientes del lenguaje de programación: Java, C++, C#, Smalltalk, Ruby, Python y cualquier otro lenguaje orientado a objetos
  - (en algunos lenguajes es más fácil de implementar que en otros)

Clasificación de patrones

- Los patrones se dividen en tres categorías
  - **Creacionales**: determinan como es el proceso de creación de las instancias en un problema determinado
  - **Estructurales**: definen como es la estructura y composición de clases
  - De **comportamiento**: definen el modo en que interactúan los objetos y como se reparte la responsabilidad

Clasificación de patrones

Ámbito	Patrones de creación	Patrones estructurales	Patrones de comportamiento
Clase	Factory method	Adapter	Interpreter Template method
Instancia	Abstract factory Builder Prototype Singleton	Adapter Bridge Composite Decorator Facade Flyweight Proxy	Chain of responsability Command Iterator Mediator Memento Observer State Strategy Visitor

Patrones de creación

- Estos patrones abstraen el proceso de creación de instancias

- Facilitan el diseño ya que el proceso de creación es independiente al resto del sistema.

Patrones estructurales

- Estos patrones se ocupan de como se combinan las clases para formar estructuras de objetos más grandes.

- Los patrones estructurales describen formas de componer objetos para obtener nueva funcionalidad.

Patrones de comportamiento

- Estos patrones están relacionados con los algoritmos y las responsabilidades que tiene cada objeto.

- Los patrones de comportamiento también describen la comunicación entre los diferentes objetos.