|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  | **10** |
|  | TEMA |

Diseño basado en patrones

[10.1] Patrones de diseño

[10.2] Diseño de *software* basado en patrones

[10.3] Patrones arquitectónicos

[10.4] Patrones de diseño de componentes

[10.5] Patrones de diseño de interfaz de usuario

[10.6] Referencias

Esquema



Ideas clave

10.1. Patrones de diseño

Un patrón de diseño contiene la **solución probada a un problema** en un determinado contexto.

Aquellos patrones que describen un contexto y un problema, pero no ofrecen una solución clara se les denomina **patrones no generativos**.

Los patrones pueden dividirse en:

* **Patrones de creación**: encargados de la creación, composición y representación de objetos. Ejemplos de patrones: Abstract factory, Factory method, Builder, Prototype, Singleton, MVC.
* **Patrones estructurales**: relacionados con el modo en el que se organizan e integran las clases y objetos para construir una estructura más grande. Ejemplos de patrones: Wrapper, Bridge, Composite, Decorator, Facade, Flyweight, Proxy.
* **Patrones de conducta**: enfocados a la asignación de responsabilidad entre los objetos y al modo en el que se comunican. Ejemplos de patrones: Chain of resposability, Command, Interpreter, Iterator, Memento, Observer, Template method, Visitor.

Un **lenguaje de patrón** agrupa un conjunto de patrones descritos con un formato estandarizado e interrelacionado, que muestra cómo colaboran los patrones para resolver problemas en un dominio de aplicación

10.2. Diseño de *software* basado en patrones

El **diseño de *software* basado en patrones** requiere la identificación de patrones en los requisitos de la aplicación que se desea construir, a continuación, hay que buscar si ya existe una solución contrastada propuesta al patrón y, por último, el patrón localizado se usa como guía para el diseño.

10.3. Patrones arquitectónicos

Los **patrones arquitectónicos** describen problemas de diseño que se resuelven con un enfoque estructural. Algunos ejemplos de patrones arquitectónicos son: control de acceso, concurrencia, distribución, persistencia

10.4. Patrones de diseño de componentes

Los **patrones de diseño de componentes** ofrecen soluciones probadas a uno o más subproblemas extraídos del modelo de requisitos.

10.5. Patrones de diseño de interfaz de usuario

Los **patrones de diseño de interfaz de usuario** dan soluciones contrastadas a requisitos de la interfaz de usuario de una aplicación.

Los patrones de este tipo pertenecen a alguna de las siguientes categorías: interfaz de usuario completa, distribución de la página, formatos y datos de entrada, tablas, manipulación directa de datos, navegación, búsqueda, elementos de página o comercio electrónico.

10.6. Referencias

Pressman, R. (2010). *Ingeniería del software* (7ª ed.). México: McGrawHill.

Lo + recomendado

No dejes de leer…

**Patrones de diseño de interacción**

Este artículo se centra en los patrones de diseño desde el punto de vista de la Interacción Persona-Ordenador.

Accede al artículo desde el aula virtual o a través de la siguiente dirección web:

<http://www.uxlumen.com/patrones-de-diseno-de-interaccion/>

No dejes de ver…

**Entrevista a Erich Gamma**

Entrevista a Erich Gamma, coautor del libro más importante en diseño de patrones. Durante la segunda parte de la entrevista (a partir del minuto 13) trata el tema del diseño de patrones.



Accede al vídeo desde el aula virtual o a través de la siguiente dirección web:

<http://www.infoq.com/interviews/gamma-jazz-eclipse-junit-design-patterns>

+ Información

A fondo

**Patrones de diseño**

Gamma, E. (2002). *Patrones de diseño*. Madrid: Prentice Hall.

Este es el libro de referencia en el mundo de los patrones de diseño. Los autores comienzan describiendo los principales patrones y cómo ayudan a la hora de diseñar aplicaciones orientadas a objetos. Posteriormente, va recorriendo los diferentes patrones explicándolos y mostrando ejemplos de cómo se implementan en código, que facilitan su comprensión.



**Antipatrones**

Al contrario que los patrones de diseño, los antipatrones son soluciones que se dan a problemas comunes que conducen a una mala resolución del problema. En esta página se pude encontrar una extensa relación de antipatrones de desarrollo *software* y organizacionales.

Accede al artículo desde el aula virtual o a través de la siguiente dirección web:

<http://es.wikipedia.org/wiki/Antipatr%C3%B3n_de_dise%C3%B1o>

Enlaces relacionados

Catálogo de Patrones JavaEE

JavaEE es la plataforma Java para el desarrollo de aplicaciones web. En la siguiente web encontrarás un completo catálogo de patrones de diseño para esta tecnología en el que realiza un análisis del patrón con diagramas de clases y de secuencia UML.



Accede a la página desde el aula virtual o a través de la siguiente dirección web:

<http://corej2eepatterns.com>

Grupo Hillside

El grupo Hillside es un proyecto educativo sin ánimo de lucro que organiza diferentes conferencias y ha publicado una serie de libros relacionados con el diseño de lenguajes de programación con patrones.



Accede a la página desde el aula virtual o a través de la siguiente dirección web:

<http://www.hillside.net>

Test

**1.** Indica qué aporta un patrón de diseño:

A. Un rol de dirección para todas las tareas de diseño.

B. Un rol de dirección solo para las tareas de diseño derivadas de una fase de análisis previo.

C. La solución probada a un problema en un determinado contexto.

D. Un método de generación automática de código a partir de un problema detectado previamente.

**2.** Los patrones no generativos son aquellos que:

A. No han sido previamente identificados.

B. No son mantenidos y han quedado obsoletos.

C. No cumplen al 100% con los requisitos iniciales.

D. No ofrecen una solución al problema.

**3.** La clasificación habitual de patrones es:

A. Funcionales, arquitectónicos y organizativos.

B. De creación, estructurales y de conducta.

C. De búsqueda, de consulta y de archivado.

D. De planificación, de análisis y de diseño.

**4.** Un lenguaje de patrón:

A. Generalmente está basado en orientación a objetos.

B. Es interpretado pero no compilado.

C. Utiliza una máquina virtual para independizarse de la plataforma.

D. Describe con formato estándar a un conjunto de patrones y sus interrelaciones.

**5.** El diseño de *software* basado en patrones requiere:

A. La identificación de patrones en los requisitos del *software*.

B. La búsqueda de soluciones contrastadas al patrón identificado.

C. El uso de los patrones como guía para el diseño del *software*.

D. Todas son correctas.

**6.** Las soluciones propuestas en los patrones arquitectónicos:

A. Hacen referencia a la estructura del sistema.

B. No son fiables al diferir en gran medida las arquitecturas de los diferentes sistemas.

C. Se construyen de abajo hacia arriba.

D. Son dependientes del sistema operativo en el que se implante la solución.

**7.** Los patrones de diseño de componentes:

A. Tienen la capacidad de autoorganizarse.

B. Deben tener una interfaz de acceso bien definida.

C. Se despliegan en servidores independientes para garantizar su reutilización.

D. Presentan soluciones a problemas del modelo de requisitos.

**8.** Todos los patrones de diseño de interfaz de usuario:

A. Deben ser accesibles.

B. Están orientados a la integración de personas con discapacidad.

C. Dan soluciones a requisitos de la interfaz de usuario de una aplicación.

D. Todas las anteriores son correctas.