UNIVERSIDAD
Popular del cesar

Ingeniería de Sistemas

ESPECIALIZACION EN INGENIERIA DE SOFTWARE
MODULO PATRONES DE DISEÑO DE SOFTWARE

EL DOCENTE

JAIRO FRANCISCO SEOANES LEON

jairoseoanes@unicesar.edu.co
(300) 600 06 70

Educación formal

- ✓ **Ingeniero de sistemas**, Universidad Popular del Cesar sede Valledupar, Feb 2002 – Jun 2009.
- ✓ **MsC en Ingeniería de Sistemas y Computación**, Universidad Nacional de Colombia, Bogotá, Feb 2011 – Mar 2015
- ✓ **PhD Ciencia, Tecnologia e innovación, Urbe, Venezuela, Mayo 2024**

Formación complementaria

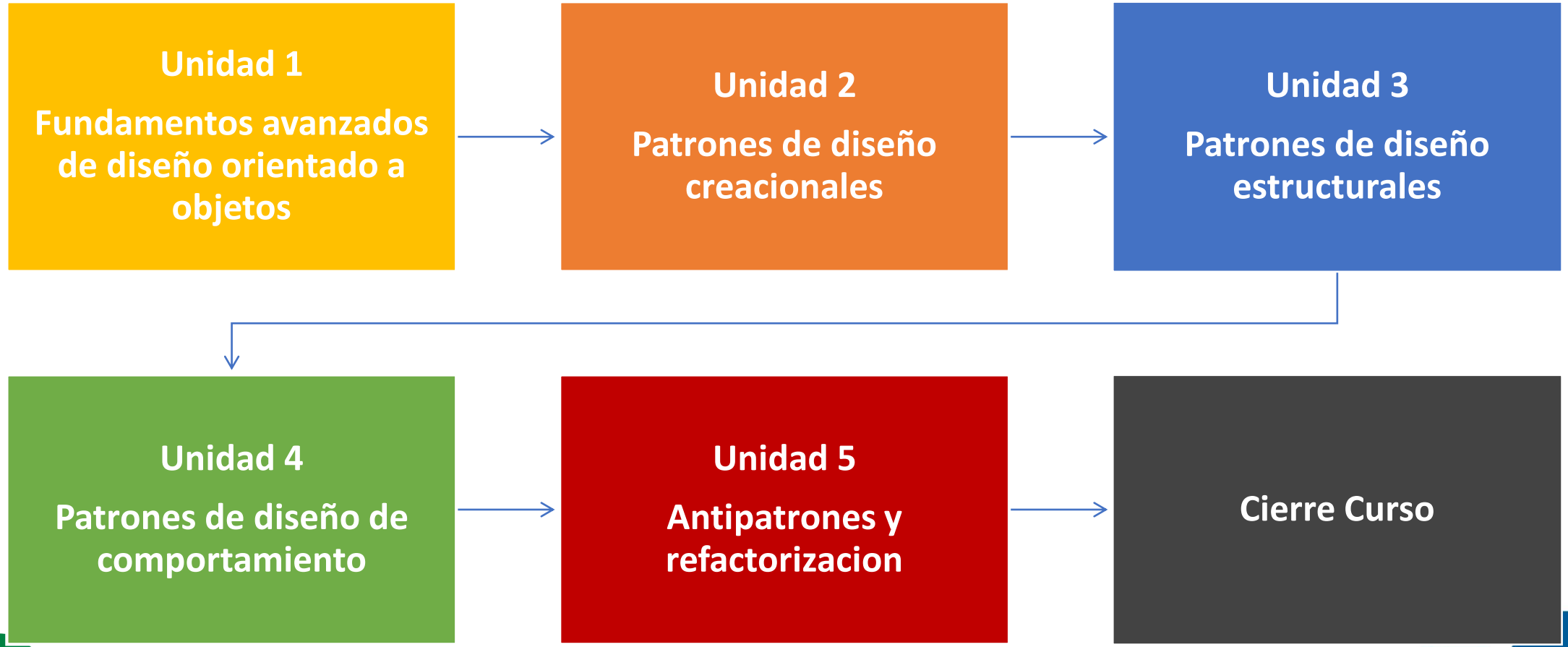
- ✓ **AWS Academy Graduate -** AWS Academy Cloud Foundations, 2022
  https://www.credly.com/go/p3Uwht36
- ✓ **Associate Cloud Engineer Path -** Google Cloud Academy, 2022
  https://www.cloudskillsboost.google/public_profiles/c7e7936c-3e37-4bad-b822-74d40c49d0db
- ✓ **Fundamentos De Programación Con Énfasis En Cloud Computing –** AWS Academy y Misión Tic 2022
- ✓ **Google Cloud Computing Foundations –** Google Academy, 2022
- ✓ **Aplicación de cloud: retos y oportunidades de mejora para las empresas de software gestionando la computación en la nube –** Fedesoft, 2023
- ✓ **Desarrollo De Aplicaciones Web En Angular, Para El Nivel Frontend** – Universidad EAFIT, 2023
- ✓ **Microsoft Scrum Foundations –** Intelligent Training - MinTic , 2023

Experiencia profesional

- ✓ **Docente Universitario**, Universidad Popular del Cesar sede Valledupar, marzo del 2013.
- ✓ **Técnico de Sistemas Grado 11**, Rama judicial Seccional Cesar, SRPA Valledupar, Junio del 2009

UNIVERSIDAD Popular del cesar

MODULO DE PATRONES DE DISEÑO DE SOFTWARE



UNIVERSIDAD Popular del cesar

MODULO DE PATRONES DE DISEÑO DE SOFTWARE

UNIVERSIDAD Popular del cesar

Unidad 2. Patrones de diseño creacionales

2.1 Factory Method
2.2 Abstract Factory
2.3 Singleton
2.4 Builder
2.5 Prototype
2.6 Inyeccion de dependencias
2.7 Uso de patrones en Spring y otros frameworks

Definición de Patrón
Clasificación de patrones de diseño.
Ventajas de los patrones de diseño.
Tipos de Patrones de Diseño

- Patrones de Creación
- Patrones Estructurales.
- Patrones de Comportamiento.

UNIVERSIDAD Popular del cesar

Patrón de diseño - Definición

*"Los **patrones de diseño** son soluciones habituales a problemas que ocurren con frecuencia en el diseño de software"*

Alexander Shvets

"Soluciones conocidas y probadas cuyo diseño proviene de la experiencia de programadores ."

Lauren Debrauwer

"Es la solución a un problema de diseño, comprobado, reutilizable, que deben poder utilizar para resolver problemas parecidos en contextos diferentes."

Orcar Blancarte

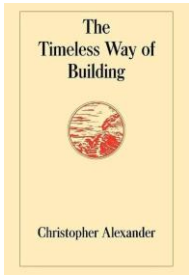
UNIVERSIDAD Popular del cesar

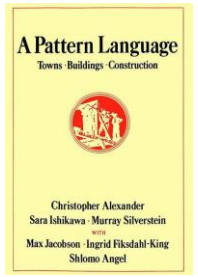
Patrón de diseño - ¿Qué no es un patrón?

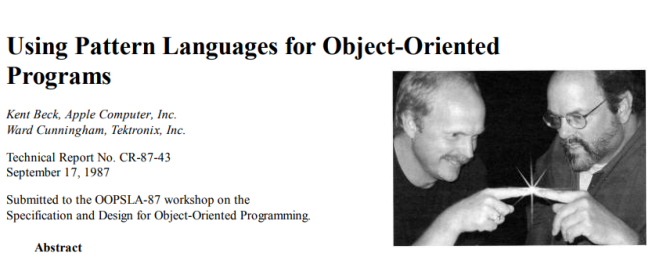
- No son algoritmos
- No son funciones o bibliotecas
- No es un fragmento de código
- No existe un aspecto teórico en los patrones, en particular no existe una formalización

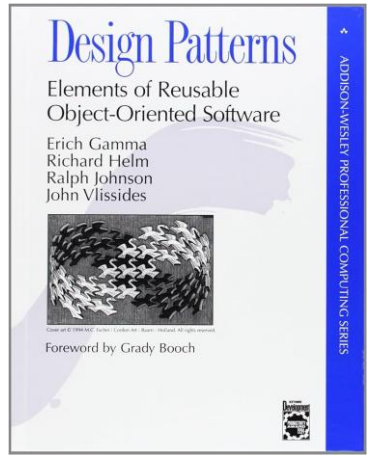
UNIVERSIDAD Popular del cesar

Patrón de diseño - ¿Un poco de historia?









Patrones construir edificios

- Timeles Way of Building
- Cristopher Alexander (1979)

Patrones arquitectónicos

- A Pattern Languaje
- Cristopher Alexander y otros

Buena arquitectura de POO

- Using Pattern Languajes for OO Programans
- Ward Cunningham Y Kent Beck (1987)
- 5 patrones hombre-maquina

Design Patterns

- Gang of Four (1990)
- 23 patrones de diseño

UNIVERSIDAD Popular del cesar

Patrón de diseño - ¿Importancia de los patrones?

- Demuestran la madurez de un programador
- Evita tener que reinventar la rueda
- Lenguaje estándar entre desarrolladores
- Facilita el aprendizaje a nuevas generaciones

- Imponer alternativas de diseño frente a otras
- Solución definitiva a un problema
- Eliminar creatividad
- No siempre son aplicables

UNIVERSIDAD Popular del cesar

Patrón de diseño - ¿Tipos de patrones ?

Idioms: Un idiom es patrón de bajo nivel, específico de un determinado lenguaje de programación. Describen como implementar aspectos particulares de los componentes, o de las relaciones entre ellos, utilizando las características de un determinado lenguaje.

Ejemplo en C++ : while (*destino++ = *src++); // copiar cadenas de caracteres

Patrones arquitectónicos: Patrones de alto nivel, que expresan una organización estructural fundamental para un sistema software. Normalmente expresa un conjunto de subsistemas predefinidos, especifica sus responsabilidades e incluye reglas y guías para organizar las relaciones entre ellos.

UNIVERSIDAD Popular del cesar

Patrón de diseño - ¿Tipos de patrones ?

Patrones de Diseño: Ofrecen esquemas para refinar subsistemas y componentes de un sistema software, o las relaciones entre ellos. Describe normalmente una estructura de comunicación recurrente entre componentes, que sirve para resolver un problema general de diseño dentro de un contexto particular.

Patrones Creacionales
Creación o construcción de objetos

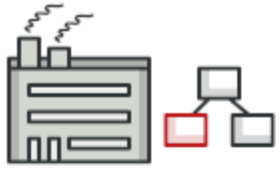
Patrones Estructurales
Como ensamblar objetos y clases en estructuras más grandes

Patrones De Comportamiento
Comunicación efectiva y la asignación de responsabilidades entre objetos

UNIVERSIDAD Popular del cesar

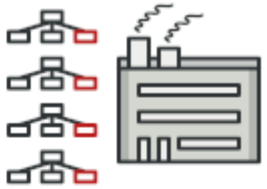
Patrón de diseño – Catalogo de patrones de diseño

Patrones Creacionales



Factory Method

Proporciona una interfaz para la creación de objetos en una superclase, mientras permite a las subclases alterar el tipo de objetos que se crearán.

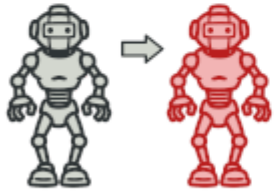


Abstract Factory

Permite producir familias de objetos relacionados sin especificar sus clases concretas.

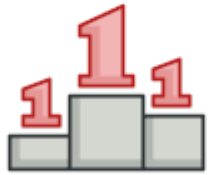
Builder

Permite construir objetos complejos paso a paso. Este patrón nos permite producir distintos tipos y representaciones de un objeto empleando el mismo código de construcción.



Prototype

Permite copiar objetos existentes sin que el código dependa de sus clases.



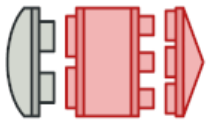
Singleton

Permite asegurarnos de que una clase tenga una única instancia, a la vez que proporciona un punto de acceso global a dicha instancia.

UNIVERSIDAD Popular del cesar

Patrón de diseño – Catalogo de patrones de diseño

Patrones Estructurales



Adapter

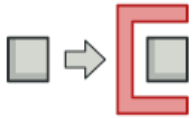
Permite la colaboración entre objetos con interfaces incompatibles.

Bridge

Permite dividir una clase grande o un grupo de clases estrechamente relacionadas, en dos jerarquías separadas (abstracción e implementación) que pueden desarrollarse independientemente la una de la otra.

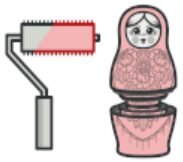
Composite

Permite componer objetos en estructuras de árbol y trabajar con esas estructuras como si fueran objetos individuales.



Proxy

Permite proporcionar un sustituto o marcador de posición para otro objeto. Un proxy controla el acceso al objeto original, permitiéndote hacer algo antes o después de que la solicitud llegue al objeto original.

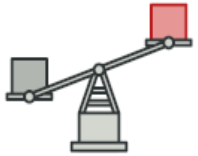


Decorator

Permite añadir funcionalidades a objetos colocando estos objetos dentro de objetos encapsuladores especiales que contienen estas funcionalidades.

Facade

Proporciona una interfaz simplificada a una biblioteca, un framework o cualquier otro grupo complejo de clases.



Flyweight

Permite mantener más objetos dentro de la cantidad disponible de memoria RAM compartiendo las partes comunes del estado entre varios objetos en lugar de mantener toda la información en cada objeto.

UNIVERSIDAD Popular del cesar

Patrón de diseño – Catalogo de patrones de diseño

Patrones de Comportamiento

Chain of Responsibility

Permite pasar solicitudes a lo largo de una cadena de manejadores. Al recibir una solicitud, cada manejador decide si la procesa o si la pasa al siguiente manejador de la cadena.

Command

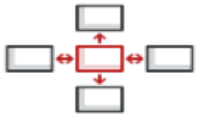
Convierte una solicitud en un objeto independiente que contiene toda la información sobre la solicitud. Esta transformación te permite parametrizar los métodos con diferentes solicitudes, retrasar o poner en cola la ejecución de una solicitud y soportar operaciones que no se pueden realizar.

Iterator

Permite recorrer elementos de una colección sin exponer su representación subyacente (lista, pila, árbol, etc.).

Visitor

Permite separar algoritmos de los objetos sobre los que operan.

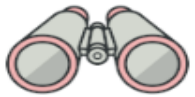


Mediator

Permite reducir las dependencias caóticas entre objetos. El patrón restringe las comunicaciones directas entre los objetos, forzándolos a colaborar únicamente a través de un objeto mediador.

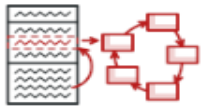
Memento

Permite guardar y restaurar el estado previo de un objeto sin revelar los detalles de su implementación.



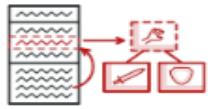
Observer

Permite definir un mecanismo de suscripción para notificar a varios objetos sobre cualquier evento que le suceda al objeto que están observando.



State

Permite a un objeto alterar su comportamiento cuando su estado interno cambia. Parece como si el objeto cambiara su clase.



Strategy

Permite definir una familia de algoritmos, colocar cada uno de ellos en una clase separada y hacer sus objetos intercambiables.

Template Method

Define el esqueleto de un algoritmo en la superclase pero permite que las subclases sobrescriban pasos del algoritmo sin cambiar su estructura.

UNIVERSIDAD Popular del cesar

Patrón de diseño – Descripción de los patrones de diseño

Formatos de descripción de patrones:

- Christopher Alexander
- Formato del GoF
- Portland Pattern Repository
- Formato canónico

Elementos comunes:

- El propósito del patrón (problema y la solución).
- La motivación (detalla el problema y la solución)
- La estructura de las clases (Diagrama UML)
- El ejemplo de código (asimilación de la idea)

Formato del GoF	
Sección	**Descripción**
Nombre del Patrón y Clasificación	Comunica la esencia del patrón.
Propósito	Una corta introducción respondiendo a las preguntas siguientes: ¿Qué hace el patrón?, ¿Cuál es su razón? ¿De qué problema particular de diseño se ocupa?
También Conocido Como	Otros nombres para el patrón.
Motivación	Un escenario que ilustra un problema y cómo las clases y los objetos se estructuran en un patrón para solucionar el problema.
Aplicabilidad	¿Cuáles son las situaciones en las que el patrón puede aplicarse? Ejemplos de casos mal planteados en los que el patrón puede aplicarse. ¿Cómo pueden reconocerse estas situaciones?
Estructura	Representación gráfica de las clases en el patrón (diagramas de clase, diagramas de colaboración...).
Participantes	Las clases y/u objetos que participan en el patrón de diseño y sus responsabilidades.
Colaboraciones	La forma en que los participantes colaboran para llevar a cabo sus responsabilidades.
Consecuencias	¿Cómo el patrón soporta sus objetivos? ¿Cuáles son las concesiones y resultados de usar un patrón?
Implementación	¿Qué trucos o técnicas deben tenerse en cuenta para implementar el patrón? ¿Hay temas propios de un lenguaje específico?
Código de Ejemplo	Fragmentos de código que ilustren como se puede implementar el patrón en C++ u otro lenguaje.
Usos Conocidos	Ejemplos del patrón encontrados en sistemas reales.
Patrones relacionados	¿Qué patrones están relacionados con el patrón que se está describiendo? ¿Cuáles son las principales diferencias? ¿Con qué otros patrones debe usarse?

Tabla 1. Formato del GoF.

UNIVERSIDAD Popular del cesar

Patrón de diseño – Recursos

Libros:

- ***"Patrones de diseño. Elementos de software orientado a objetos reutilizables"***
  Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides

- ***"Sumérgete en los patrones de diseño"***
  Alexander Shvets

- ***"Introducción a los patrones de diseño. Un enfoque practico"***
  Oscar Blancharte

- ***"Patrones de diseño en java. Los 23 modelos de diseño: descripción y soluciones ilustradas "***
  Lauren Debrauwer

UNIVERSIDAD
Popular del cesar