UNIVERSIDAD DON BOSCO
Educación Superior con Estilo Salesiano

FACULTAD DE INGENIERÍA

Asignatura: Ingeniería de Software

Título:

"Trabajo de investigación #01"

Docente:

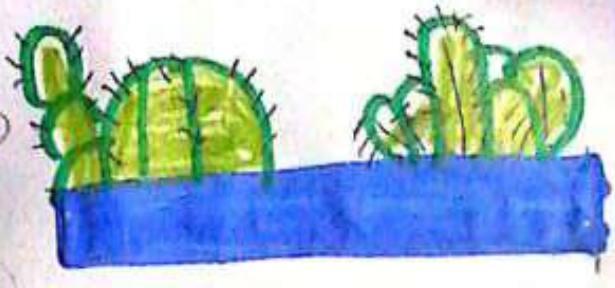
Ing. Alexander Siguenza

Presentado por:

Apellidos, Nombres	Carné	Carrera	Teoría
Katherine Lissette Sánchez Vila,	SV161855	Ing. en Ciencias de la Computación	01T
Salvador Alejandro González Meléndez	GM190689	Ing. en Ciencias de la Computación	01T
Manuel Oswaldo Hernández López	RP142494	Ing. en Ciencias de la Computación	01T
,Gerson Ernesto Lopez Chevez	LC151886	Ing. en Ciencias de la Computación	01T

Soyapango, 18 de Junio de 2021

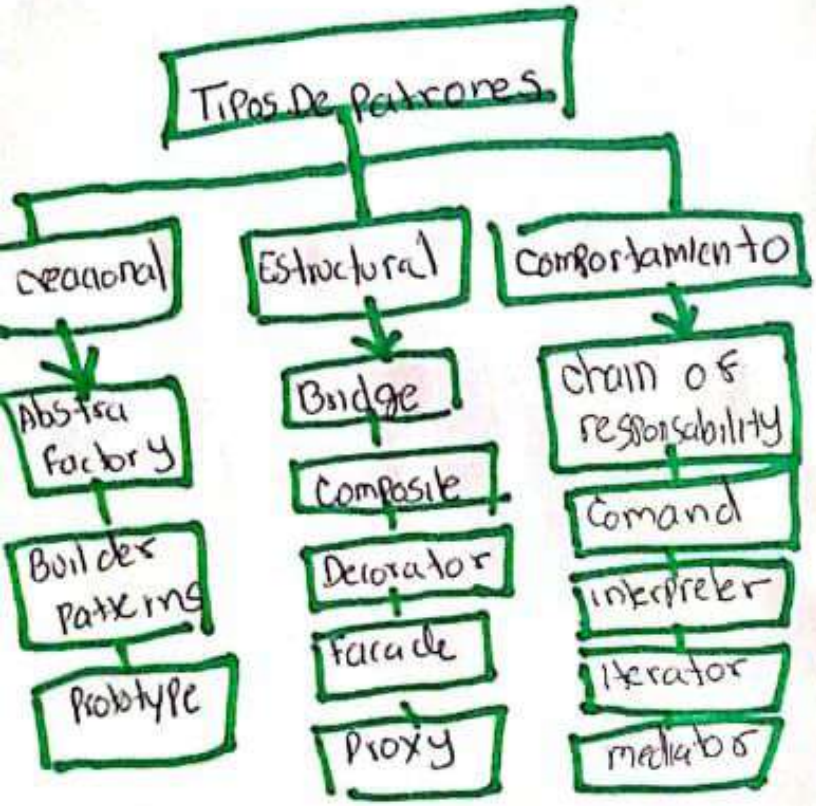
PATRONES DE DISEÑO



¿Que es?

es una solucion general y reutilizable aplicable a diferentes problemas
Se trata de plantillas que identifican problemas en el sistema y proporcionan soluciones apropiadas a problemas generales.

Estos patrones de diseño nos permiten crear nuestro codigo de manera más facil y con estructuras de codigo que ya han sido probadas



CLEAN Architecture

DEFINICION

es un compendio de Principios y Patrones de desarrollo que tienen como objetivo facilitar el proceso de construccion de software asi como su mantenimiento.

BENEFICIOS

- Creacion de aplicaciones desacopladas que son mas faciles de usar.
- Mayor Flexibilidad para añadir o remover Funcionalidades del Software.
- Diseño basandose en componentes con responsabilidades bien definidas.

USOS

Aplicaciones de negocios para proyectos.

Infraestructuras heterogeneas a nivel base de datos, servicios web, etc.

Aplicaciones pensadas para ser extendidas por terceros a traves de plugins.

- enterprise Business rules
- Aplication Business rules
- Interface Adapters
- Frameworks y drivers

SOLID

¿Qué es Solid?

Es el acrónimo que acuño a Michael Feathers, basandose en los principios de Programacion Orientada a objetos

SIGLAS

- **S** ingle responsability Principle
- **O** pen/closed Principle
- **L** iskov Substitucion Principle
- **I** nterface segregation Principle
- **D** ependency Inversion Principle

Aplicar estos principios facilitara mucho el trabajo, tanto propio como ajeno

Ventajas

- Mantenimiento del código más facil y rápido.
- Permite añadir nuevas Funcionalidades de forma más Sencilla.
- Favorece una mayor reusabilidad y calidad de código, asi como la encapsulación.

Principios

Responsabilidad unica
Una clase deberia tener una y solo una razon para cambiar.

Principio Abierto debemos ser capaz de extender el comportamiento de la clase sin tocarla.

Sustitucion de Liskov Las clases derivadas deben poder sustituirse por sus clases base.

Segregacion de la interfaz hacer interfaces que definan pocos metodos.

Dependencias dependen de abstracciones no de clases concretas.