UDS

Mi Universidad

MAPA CONCEPTUAL

Sofia Pereyra Orantes
Modelos de Implementación
Unidad 4
Ingeniería en Software
Reyes Molina Andres Alejandro
Ingeniería en Sistemas Computacionales
Cuatrimestre 8

Comitan de Dominguez, Chiapas a 5 de Abril de 2025

MODELO DE IMPLEMENTACIÓN

MODELO DE IMPLEMENTACIÓN

artefacto de diseño que describe cómo se asignan recursos a los componentes de un sistema

puede ser

enfoque de desarrollo de sistemas que divide un proyecto en fases.

DIAGRAMAS DE COMPONENTES

son

muestran la estructura de los componentes de la arquitectura de un sistema y cómo están conectados e interactúan

MODELOS DE PRUEBA

enfoques estructurados para evaluar la calidad de un software o sistema

sirven para

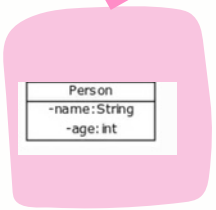
Permiten identificar y eliminar defectos, y garantizar que el software cumpla con los requisitos de los usuarios.

Ayudan a

- Permiten representar el comportamiento deseado de un sistema bajo prueba
- Permiten representar estrategias de prueba y un entorno de prueba
- Proporcionan un marco para identificar y eliminar defectos
- Garantizan que el software cumpla con las expectativas de los usuarios
- Se basan en el diseño basado en modelos

IMPLEMENTACION EN JAVA DE LOS DIAGRAMAS DE CLASE

Una clase se representa mediante un rectángulo con su nombre escrito encima. Una línea debajo del nombre de la clase separa este de la lista de atributos (nombres y tipos de las variables de clase). Los atributos se escriben uno por línea.




```
public class Person {
    private String name;
    private int age;

    public Person(String initialName) {
        this.name = initialName;
        this.age = 0;
    }
}
```

CONSTRUCTOR Y DESTRUCTOR (DECLARACIÓN , USO Y APLICACIONES)

es

CONSTRUCTOR

Un método constructor de una clase es un método especial que:

- tiene el mismo nombre que la clase y
- no tiene tipo de retorno.
- inicializa el estado de un objeto.

es

DESTRUCTOR

el destructor elimina el vínculo y libera el espacio de memoria de un objeto, para que pueda ser ocupado nuevamente.

DIAGRAMAS DE USO/DIAGRAMAS DE INTERACCIÓN

Se utilizan para modelar el comportamiento de un sistema.

es

DIAGRAMA DE USO

- Se usan para describir cómo los usuarios interactúan con un sistema
- Se crean en la etapa inicial del desarrollo de un sistema
- Se utilizan para especificar el contexto del sistema

también

DIAGRAMA DE INTERACCIÓN

- Se utilizan para modelar un sistema como una secuencia de eventos ordenados por tiempo
- Se utilizan para hacer ingeniería inversa o directa de un sistema o un proceso
- Se utilizan para organizar la estructura de diversos eventos interactivos

ELEMENTOS DE UN DIAGRAMA DE COMPONENTES

- Elemento componente: Una parte de un sistema, como un módulo, se denomina entidad.
- Icono de componente: Representación gráfica de la funcionalidad o propósito de un componente.
- Elemento de notación de componente: Elemento gráfico utilizado para mostrar un componente en un diagrama.
- Símbolo de componente: Símbolo que muestra un componente como una entidad separada.
- Relación componente-subcomponente: Representa la relación jerárquica entre componentes y subcomponentes.
- Componentes internos: Funcionalidad dentro de un componente padre.

INTERFACES

son

- Interfaces proporcionadas: Servicios que el componente ofrece a sus clientes.
- Interfaces requeridas: Servicios que el componente necesita de otros componentes.

CONECTORES

son

- Tipo de conector: El conector es la forma en que dos componentes de software se comunican en un diagrama. El tipo de conector puede ser protocolos, interfaces, señales o componentes.
- Puentes de conector: muestra cómo diferentes componentes trabajan juntos en el diagrama.
- Líneas de conector: la línea representa la comunicación entre componentes.
- Conector recto: Un conector que conecta dos cosas en el mismo diagrama.
- Conectores de ensamblaje: muestra una relación entre dos componentes durante el tiempo de ejecución.

su

```
Ejemplo:
class Producto
{
private int clave;
private double precio;
public Producto( int c, double p)
{
clave = c;
precio = p;
}
public double daPrecio( )
{
return precio;
}
}
```

Son dos tipos de métodos especiales que se ejecutan, respectivamente, al crear un nuevo objeto y cuando el recolector de basura detecta que ya no lo estamos utilizando y es posible eliminarlo de la memoria.

tiene

APLICACIONES

1.- clases que sólo poseen el constructor predeterminado ,

2.- clases con un constructor definido por el programador y

3.- clases con un constructor definido por el programador y el predefinido (que deberá ser reescrito por el programador).

El constructor predeterminado es incluído automáticamente.

Cuando el programador define un constructor, el predeterminado se anula.

su

```
Ejemplo:
// Destructores.cs : Maneja tres destructores de clases encadenadas.
using System;
using C = System.Console;
class Primera
{
~ Primera( )
{
C.WriteLine("Se invocó al destructor de Primera...");
}
}
class Segunda : Primera
{
~ Segunda( )
{
C.WriteLine("Se invocó al destructor de Segunda...");
}
}
class Tercera : Segunda
{
~ Tercera( )
{
C.WriteLine("Se invocó al destructor de Tercera...");
}
}
public class Principal
{
public static void Main( )
{
Tercera t = new Tercera ( );
}
}
```

ARQUITECTURA LÓGICA CON PATRONES: SEPARACIÓN MODELO-VISTA.

es

Principio arquitectónico que separa la lógica de negocio de la interfaz de usuario en una aplicación

en

MVC

- Divide la aplicación en tres componentes: Modelo, Vista y Controlador
- El Modelo representa la lógica de negocio y los datos
- La Vista muestra los datos al usuario y se encarga de la interfaz de usuario
- El Controlador procesa las solicitudes del usuario y actualiza el Modelo y la Vista

en

BASE DE DATOS

El modelo, que se encarga de los datos de la app, consulta la base de datos

APLICACIONES WEB

Se han desarrollado multitud de frameworks, comerciales y no comerciales, que implementan este patrón

cuando

El usuario interactúa con la interfaz de usuario de alguna forma

El controlador accede al modelo, actualizándolo, posiblemente modificándolo de forma adecuada a la acción solicitada por el usuario