Laboratorio 3: Paradigma Orientado A Objetos

Rodrigo Mardones Aguilar

8/10/2020

Indice

Introducción

Los Paradigmas de programación son modelos o patrones de trabajo, los cuales nos permiten resolver problemas abordando de distintas manera la ejecución de la solución.

Es preciso señalar que indiferente de los lenguajes nuevos, modernos y sus herramnientas propias asociadas, siempre se regirán por al menos uno o más paradigmas de programación. Pues son estos quienes nos permiten analizar de buena manera

Objectivo General

Para este laboratorio n°3 se pretende construir una pieza de software que permita simular un "controlador de versiones" tipo git, utilizando como requerimiento base el paradigma orientado a objetos y las caracteristicas que rigen a este paradigma.

Objectivos Específicos

Para abordar el problema se plantean lo siguientes objectivos especificos

- Diseño de la solución.
- Aspectos de la implementación.
- Instrucciones de uso.
- Resultados y Autoevalución.
- Conclusiones.

Descripción del problema

La necesidad de mantener y producir software escalable, ordenado y mantenible durante el tiempo es una cuestión importante dentro de la industria. Poder llevar todos los cambios asociados a un proyecto, trabajado por un grupo importante de personas y todo lo que conlleva el manejo de errores es parte de lo que resuelven los software denominados "controladores de versiones".

Para el problema en cuestión se requiere de una pieza de software que permita realizar las siguientes acciones dentro de un directorio de trabajo, independiente del trabajo realizado dentro de este:

- Poder guardar cambios de un proyecto en cuestión, asociados a un espacio de trabajo y a un usuario.
- Poder revisar, listar y mostrar los cambios guardados de un proyecto asociado
- poder enviar y traer los cambios que se tienen guardados a un repositorio externo.

Descripción del paradigma

El "paradigma orientado a objetos" es una paradigma de programación cuya unidad de trabajo son los "objetos", medidas de representación y modelado del mundo real. Estos objetos, pueden relacionarse con otros objetos mediante el uso de mensajes que puedan interpretar.

La definición de un objeto está dada por las "Clases" que son moldes o estructuras definidas para la construcción de objetos semejantes a la clase planteada. Las clases a su vez tienen atributos y metodos. Los atributos son las caracteristicas(variables) que componen a la clase y los metodos(funciones) son los comportamientos definidos para esta misma.

Caracteristicas definidas del paradigma

Como ya mencionamos, su unidad principal son los "objetos", y la construcción o creación de estos se asocia al concepto de "Clase". Otras caracteristicas importantes a destacar son las siguientes:

- Herencia: Es la capacidad de "heredad" comportamientos y caracteristicas de una clase "padre" a otra "hija", ayudando así a la reutilizacion de codigo y el entendimiento logico.
- Polimorfismo: Esta propiedad relacionada con la herencia, hace referencia al comportamiento de una clase hija, la cual puede comportarse tanto como por su definicion y como por la definición de la clase padre que hereda.
- Sobrecarga: La sobrecarga permite redifinir métodos que experimenten comportamientos distintos dependiendo también de los parametros que son enviados a estos. así podemos tener más de una defincion para el mismo método pero con respuestas distintas.
- Sobreescritura: La sobreescritura se entiene como la redefinición completa de un metodo heredado del padre, para así ajustarlo al comportamiento deseado por la clase hija.
- Clase Abstracta: se entiende por "clase abstracta" la definición de una clase en la que no se puede hacer una instancia directa. Otra definión más entendible es la implementación parcial de un TDA(tipo de dato abstracto), pues permite definir de manera parcial el comportamiento deseado de un representación.
- Interfaces: Las interfaces a su vez son representaciones completas de un TDA, definiciones características y comportamientos ordenados.

Analisis del problema

Diseño de solución

Aspectos de la implementación

Instrucciones de uso

Resultados y Autoevaluacion

Conclusiones

Referencias