Programación Orientada a Objetos

IESA

Carrera: Tecnicatura Superior en Análisis de Sistemas

Cátedra: Programación Científica

Tema: Clases y Objetos

Docente: Lic. Del Rosario Gabriel Dario

Tabla de Contenidos

Conceptos

Objeto

Informalmente

Entidad del mundo real que puede ser real o abstracto, clasificándose en las siguientes entidades:

- Físicas: vehículo, casa, mueble, etc.
- Conceptuales: proceso químico, transacción bancaria, etc.
- De software: lista enlazada, interfaz gráfica, etc.

Según James Rumbaugh

Concepto, abstracción o cosa con un significado y límites claros en el problema en cuestión.

Según Grady Booch

Aquel que posee:

- Estado: lo que sabe, siendo que:
  - Es una de las posibles condiciones en que el objeto puede existir.
  - Normalmente cambia en el transcurso del tiempo.
  - Es implementado por un conjunto de propiedades o atributos, más las conexiones que puede tener con otros objetos
- Comportamiento: lo que puede hacer, considerando que:
  - Determina cómo éste actúa y reacciona frente a las peticiones de otros objetos.
  - Es modelado por un conjunto de mensajes a los que el objeto puede responder (operaciones que puede realizar).
  - Se implementa mediante métodos.
- Identidad: lo que le da unicidad, incluso si su estado es idéntico al de otro objeto.

Clase

Descripción de un grupo de objetos. Se compone de:

- Atributos: propiedades en común.
- Operaciones: comportamiento similar.
- Relaciones: la misma forma de vincularse con otros objetos.
- Semántica compartida: significan lo mismo.

Programación orientada a objetos

Forma de organizar el software como una colección de módulos que constan de datos y de comportamiento. Un sistema (el software) se concibe como un conjunto de objetos que se comunican entre sí mediante mensajes.

Sigue con frecuencia el mismo método que se aplica en la resolución de problemas de la vida diaria, simulando un modelo del universo (el entorno en que el sistema se basa).

Programa orientado a objetos

Consta de un conjunto de objetos que se crean de forma dinámica, interesando más qué se puede hacer con los objetos antes que cómo se hace. Cada objeto es una instancia de una clase, siendo responsable de unas tareas. Sus clases se pueden organizar en una jerarquía de herencia.

Lenguaje de modelado unificado

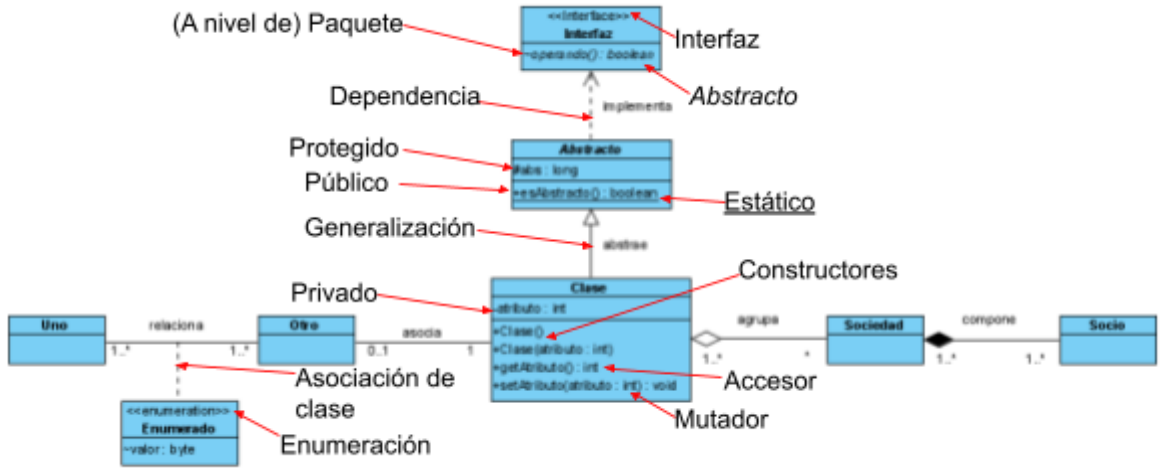
Especificación que define un lenguaje gráfico para visualizar, especificar, construir y documentar los artefactos de los sistemas de objetos distribuidos.

Diagrama de clase

A menudo se describen en la literatura UML como “representaciones de estructura estática”. En otras palabras, se entiende y se repite con frecuencia que un modelo de clase no refleja un comportamiento dinámico. Tomando en cuenta que los diagramas de clase, en efecto, son estáticos (al igual que las reglas cotidianas); el comportamiento, la inteligencia, la resiliencia y la confiabilidad (en resumen, la personalidad dinámica) de su software está definida y restringida por una base de reglas de aplicación. Cada aplicación tiene una y es bueno cristalizar la base de reglas de una manera independiente de la plataforma. Pocos desarrolladores aprecian el grado en que la notación del diagrama de clases puede moldear y restringir el comportamiento de un sistema complejo.

Notación UML

Diagrama de clases



Modificadores de acceso

- Privado: el elemento sólo puede accederse dentro de la clase.
~ (A nivel de) Paquete: el elemento puede accederse dentro del mismo paquete.
# Protegido: el elemento puede accederse dentro del mismo paquete y los hijos de la clase.
+ Público: el elemento puede ser accedido desde cualquier parte.

Tipo de elemento

atributo : tipo
Constructor(parámetros)
método(parámetros) : tipo

Característica del elemento

Normal: se accede a nivel de objeto y tiene implementación.
Abstracto: no tiene implementación.
Estático: se accede a nivel de clase.

Tipos de relaciones

Asociación: ——una clase se comunica con otra.

Agregación: ◇—una clase (a la que apunta el rombo) es propietaria de otra, pero esta última puede existir de forma independiente a la primera.

Composición: ◆—una clase (a la que apunta el rombo) es propietaria de otra, a tal punto esta última no puede existir sin la primera.

Cardinalidad de relaciones

Se define como ‘cantidad’ o ‘mínimo..máximo’, donde cantidad, mínimo y máximo son números o el caracter * (no puede ser el mínimo).
Si aparece sólo * (como cantidad), implica que el mínimo es 0 y el máximo es indefinido (infinito). Si aparece * en máximo, implica que este último es indefinido (infinito).

Generalización (o especialización)

 Se aplica entre una clase normal y una abstracta, donde la flecha apunta a la abstracta.

Dependencia

 Se aplica entre una interfaz y una clase, donde la flecha apunta a la interfaz.