Ingeniería de sistemas

PROGRAMACIÓN ORIENTADA A OBJETOS EN JAVA

Buenas prácticas

Miguel Hernández Bejarano
Luis Eduardo Baquero Rey

edU

Informática

PROGRAMACIÓN ORIENTADA A OBJETOS EN JAVA

Buenas prácticas

Miguel Hernández Bejarano
Luis Eduardo Baquero Rey

edU®
Conocimiento a su alcance
Bogotá - México, D.F.

Hernández Bejarano, Miguel, *et al.*

Programación orientada a objetos en Java/ Miguel Hernández Bejarano y Luis Eduardo Baquero Rey -- 1a. Edición. Bogotá : Ediciones de la U, 2023.
316 p. ; 24 cm.
ISBN 978-958-792-463-3 e-ISBN 978-958-792-464-0
1. Informática 2. Lenguajes de programación 3. Programación orientada a objetos en Java I. Tít.
621.39 ed.

Área: Informática
Primera edición: Bogotá, Colombia, enero de 2023
ISBN. 978-958-792-463-3

Coordinación editorial: Adriana Gutiérrez M.
Carátula: Ediciones de la U
Impresión: DGP Editores SAS
Calle 63 No. 70 D - 34, Pbx. (571) 7217756

Impreso y hecho en Colombia
Printed and made in Colombia

Dedicatoria

A Dios, a mi esposa Flor Ángela,
a mis hijos Jefferson, Julieth y Óscar
y a mi nieto Alejandro
Miguel

A Cristal, Isabela y Rocío
Eduardo

Contenido

Índice de figuras

Índice de tablas

Prólogo

En el quehacer diario de la docencia, y más concretamente en el área de la programación, surgen muchos temas, talleres y ejercicios para trabajar con los estudiantes. En tal sentido, este libro es el resultado de varios años de experiencias donde se retroalimentan carencias y éxitos en busca de buenas prácticas de la Programación Orientada a Objetos (POO); para tal efecto, se utiliza Java como lenguaje de programación de alto nivel. Está diseñado para cualquier estudiante que desea hacer inmersión en la programación de computadores, independientemente del área o disciplina del conocimiento, pero, eso sí, cambiando el paradigma de la programación estructurada a la de los objetos, como en la vida real.

Se tienen en cuenta elementos de buenas prácticas de la programación con la finalidad de hacer fácil la lectura del código, así como su mantenimiento, facilitando la escalabilidad del mismo, la reutilización y la integración de manera homogénea, estandarizando el desarrollo, basado en convenios y reglas sencillas, y aportando higiene en la codificación de los programas fuente.

Con el propósito de lograr lo anterior, esta obra se divide en catorce capítulos, cada uno distribuido de la siguiente manera: relación general de la temática a desarrollar, una introducción al tema central donde se complementa con un mapa mental, el desarrollo de la temática respectiva incluyendo ejemplos, se recomiendan unas lecturas que posibilitan ampliar el tema, unas preguntas de revisión de conceptos, ejercicios propuestos y, finalmente, se propone una bibliografía complementaria.

En el capítulo 1, se tratan temas fundamentales de este paradigma, como el de la programación orientada a objetos, los principios de la POO y los conceptos de clase, objeto, atributo y método.

En el capítulo 2, se estudian los componentes de un programa, donde se involucran identificadores, tipos de datos, palabras reservadas, manejo de comentarios, variables, constantes, las expresiones, la estructura de un programa de Java, la entrada y la salida de datos y los tipos de operadores.

En el capítulo 3, se estudian las herramientas necesarias para el desarrollo de la programación orientada a objetos, como los entornos integrados de desarrollo, herramientas de modelamiento y el lenguaje de programación Java.

En el capítulo 4, se hace referencia a los métodos, qué son los métodos constructores, los métodos set y get y los métodos modificadores.

En el capítulo 5, se estudia todo lo relacionado con los condicionales, en su orden, las estructuras de control, los condicionales simple, compuesto y anidado, los operadores lógicos, selección múltiple y el operador condicional ?.

En el capítulo 6, se estudia el tema relacionado con el manejo de cadenas en Java, incluyendo los métodos asociados a la clase String.

En el capítulo 7, se hace referencia a los ciclos o instrucciones repetitivas que se pueden trabajar en un programa Java.

En el capítulo 8, se estudian las diferentes relaciones entre clases, como la cardinalidad, asociación, agregación, composición, generalización y dependencia.

En el capítulo 9, se involucran los temas de abstracción y encapsulamiento, como mecanismos de la programación orientada a objetos.

En el capítulo 10, se estudia el concepto de herencia y la terminología asociada a este concepto, como lo que es una superclase, subclase, protected, extends y super.

En el capítulo 11, se hace referencia al mecanismo de la programación orientada a objetos denominado polimorfismo, incluyendo polimorfismo de subtipado, sobrecarga de métodos y sobreescritura de métodos.

En el capítulo 12, se continúa con el concepto y manejo de interfaz y clases abstractas en Java.

En el capítulo 13, se estudian los aspectos básicos a tener en cuenta a la hora de desarrollar un proyecto o aplicación de software de comienzo a fin, desde el análisis orientado a objetos, el diseño, el modelamiento y la aplicación de software final.

En el capítulo 14, por último, se presentan opciones desde el lenguaje de programación Java para la respectiva documentación de los programas y

aplicaciones, con herramientas con Javadoc, soporte Javadoc para los IDE y la API de Java.

El estudiante interesado en el tema podría orientarlo también a cualquier otro lenguaje de programación de alto nivel, diseñado para trabajar con el paradigma de la programación orientada a objetos.

Capítulo 1

Fundamentos de la Programación Orientada a Objetos (POO)

1.1 Temática a desarrollar

- Programación orientada a objetos
- Principios de la POO
- Clase
- Objeto
- Atributo
- Método

1.2 Introducción

La concepción del mundo en términos de Programación Orientada a Objetos (POO) está integrada por un conjunto de entidades u objetos relacionados entre sí, los cuales pueden ser: personas, árboles, automóviles, casas, teléfonos y computadores; en general, todo elemento que tiene atributos y que pueden ser palpables a través de los sentidos.

El ser humano tiene una apreciación de su ambiente en términos de objetos, por lo cual le resulta simple pensar de la misma manera cuando diseña un modelo para la solución de un problema, puesto que se tiene el concepto o pensamiento de objetos de manera intrínseca.

La Programación Orientada a Objetos (*Object Oriented Programming*) tiene como ejes centrales las clases y los objetos, donde una clase corresponde a la agrupación de datos (atributos) y acciones (métodos). Una clase describe un conjunto de objetos con propiedades y comportamientos similares. La figura 1 muestra, mediante un concepto de mapa mental, este concepto y todo lo que le rodea.

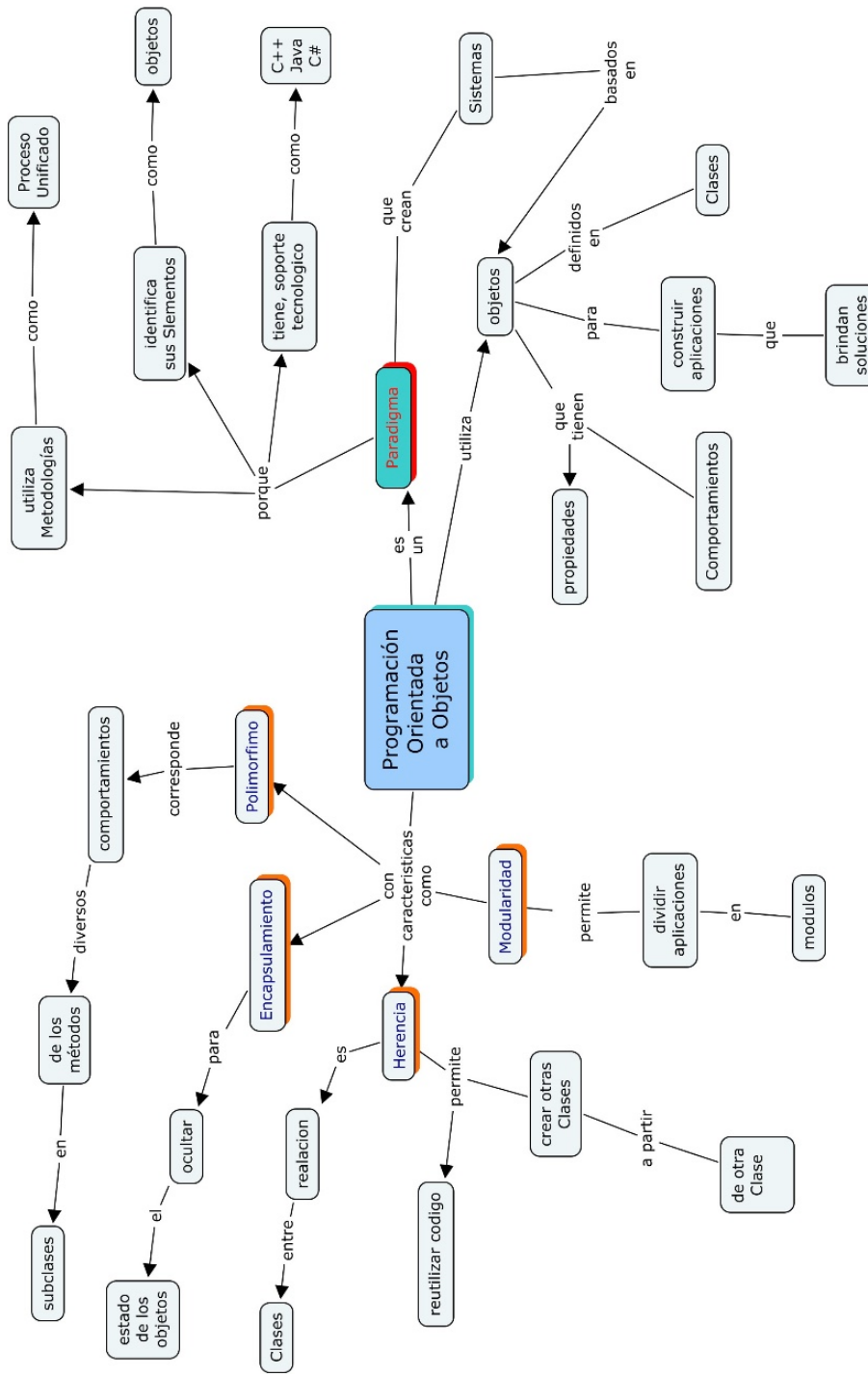


Figura 1
Mapa mental de programación orientada a objetos

El buen uso de este paradigma de programación se convierte en una buena práctica que arrojaría como resultados la construcción de programas de calidad, facilitar su mantenimiento y la reutilización de programas, entre otros aspectos. El tomar objetos de un problema del mundo real y extraer sus características y comportamientos, y representarlos formalmente en diagramas UML, mediante los diagramas de clases y diagramas de objetos, son de las primeras actividades inmersas en este estilo de programación.

1.3 Programación orientada a objetos

La Programación Orientada a Objetos (POO) es un paradigma *o modelo de programación;* esto indica que no es un lenguaje de programación específico o una tecnología, sino una *forma de programar* donde lo que se intenta es llevar o modelar el mundo real a un conjunto de entidades u objetos que están relacionados y se comunican entre ellos como elementos constitutivos del software, dando solución a un problema en términos de programación.

En síntesis, el elemento básico de este paradigma es el objeto, el cual contiene toda la información necesaria que se abstrae del mundo del problema, los datos que describen su estado y las operaciones que pueden modificar dicho estado, determinando las capacidades del objeto.

1.4 Principios de la programación orientada a objetos

Abstracción: es la capacidad de determinar los detalles fundamentales de un objeto mientras se ignora el entorno; por ejemplo, la información de un cliente en una factura requiere el documento, el nombre, la dirección y el teléfono.

Encapsulamiento: es la capacidad de agrupar en una sola unidad datos y métodos. Por ejemplo, para calcular el área de un triángulo, se deben tener la base y la altura; o cuando se ve la televisión, no hay preocupación por el modo cómo este funciona, lo único que se realiza es manipular el control y disfrutar del programa, película, entre otros ejemplos.

Herencia: es el proceso en el que un objeto adquiere las propiedades de otro a partir de una clase base, de la cual se heredan todas sus propiedades, métodos y eventos; estos, a su vez, pueden o no ser implementados y/o modificados;

por ejemplo, la herencia de bienes que se transmite de padre a hijo, pero el padre tiene la potestad de desheredar.

Polimorfismo: es una propiedad que permite enviar el mismo mensaje a objetos de diferentes clases de forma que cada uno de ellos responde al mismo mensaje de diferente modo; por ejemplo, el acto de comer para proveer nutrientes al organismo es diferente en los siguientes animales: el león (carnívoro), el canario (alpiste) y el perro (todo tipo de alimento).

Modularidad: es la propiedad que permite dividir una aplicación de software en unidades o partes más pequeñas, denominadas módulos, donde cada una de ellas puede ser independiente de la aplicación y de los restantes módulos. Por ejemplo, el software institucional de una empresa que puede ser dividido en subprogramas como contabilidad, presupuesto, facturación, entre otros.

1.5 Ventajas del uso de la programación orientada a objetos

- Simula el sistema del mundo real.
- Facilita el mantenimiento del software.
- Facilita el trabajo en equipo.
- Modela procesos de la realidad.
- Genera independencia e interoperabilidad de la tecnología.
- Incrementa la reutilización del software.

1.6 Desventajas del uso de la programación orientada a objetos

- Complejidad para adaptarse.
- Mayor cantidad de código (aunque se tiene la posibilidad de volver, de ser reutilizable).

1.7 Lenguajes de programación orientada a objetos

Permite el desarrollo de aplicativos de software que interactúan con los programas como conjuntos de objetos que se ayudan entre ellos para realizar acciones.

Ejemplo

- C++
- ADA
- Java
- SmallTalk
- C#
- Php versión 5.0 en adelante

1.8 Clases y objetos

Las palabras "clase" y "objeto" son utilizadas en la programación orientada a objetos, la cual planea simular u organizar datos en conjuntos modulares de elementos de información del mundo real.

1.8.1 Clases

Asumida como un molde o plantilla empleada para crear objetos, lo que permite considerar que las clases agrupan a un conjunto de objetos similares.

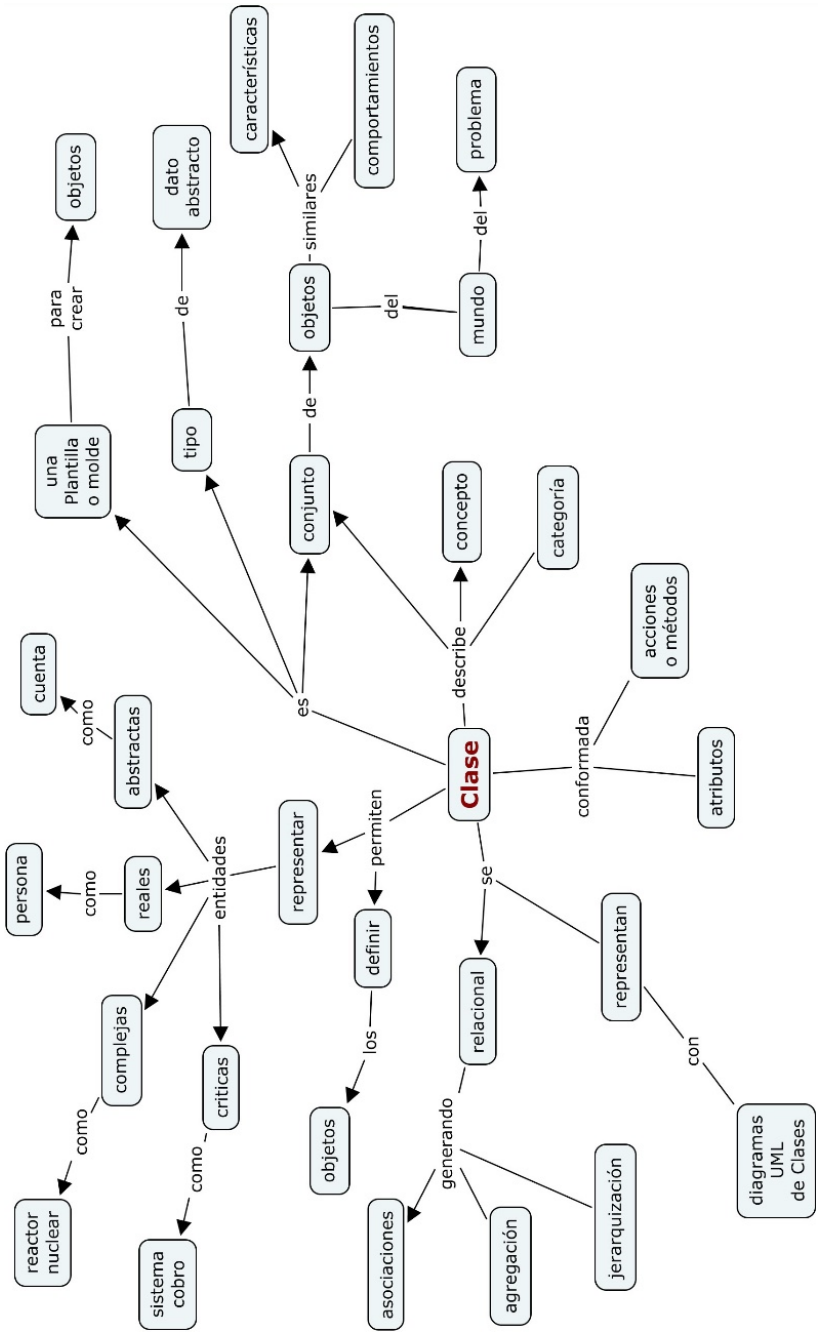
Ejemplo

Tabla 1.
Ejemplos de objetos asociados a su correspondiente clase

Objetos	Clase
Mazda 626 sdk23, rojo Ford, capacidad 32 personas	Auto
• Cra. 15 No. 85 – 19 bloque 8 • Calle 128 # 52 – 90 bloque 12 apto. 402.	Vivienda
39.652.123 Claudia Ramírez	Estudiante
• Garfield (Felino) • Snoopy (Perro) • Simba (Rey León) • Willy (Cetáceo)	Mamífero
• Código 1478 Estructura de Datos • Código 18156 Matemáticas discretas	Asignatura

La figura 2 amplía el concepto de clase.

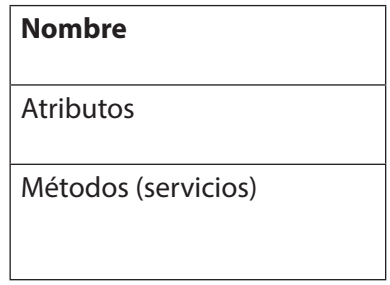
Figura 2.
Mapa mental de clase



1.8.1.1 Representación de las clases

La clase encapsula la información de uno o varios objetos a través de la cual se modela el entorno en estudio. Se representa haciendo uso de los diagramas de clases de UML, conformados por un rectángulo que tiene tres divisiones, como se muestra en la figura 3.

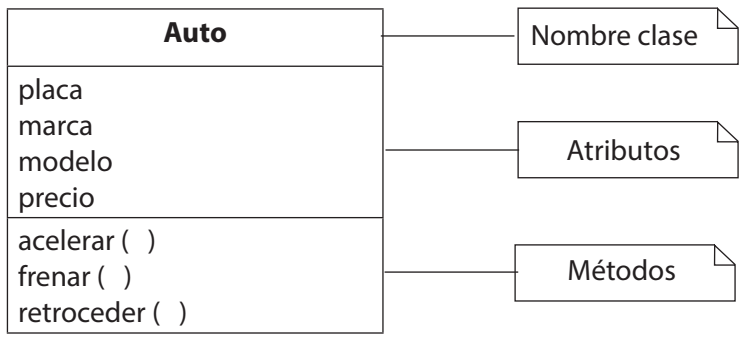
Figura 3.
Partes del diagrama de clase

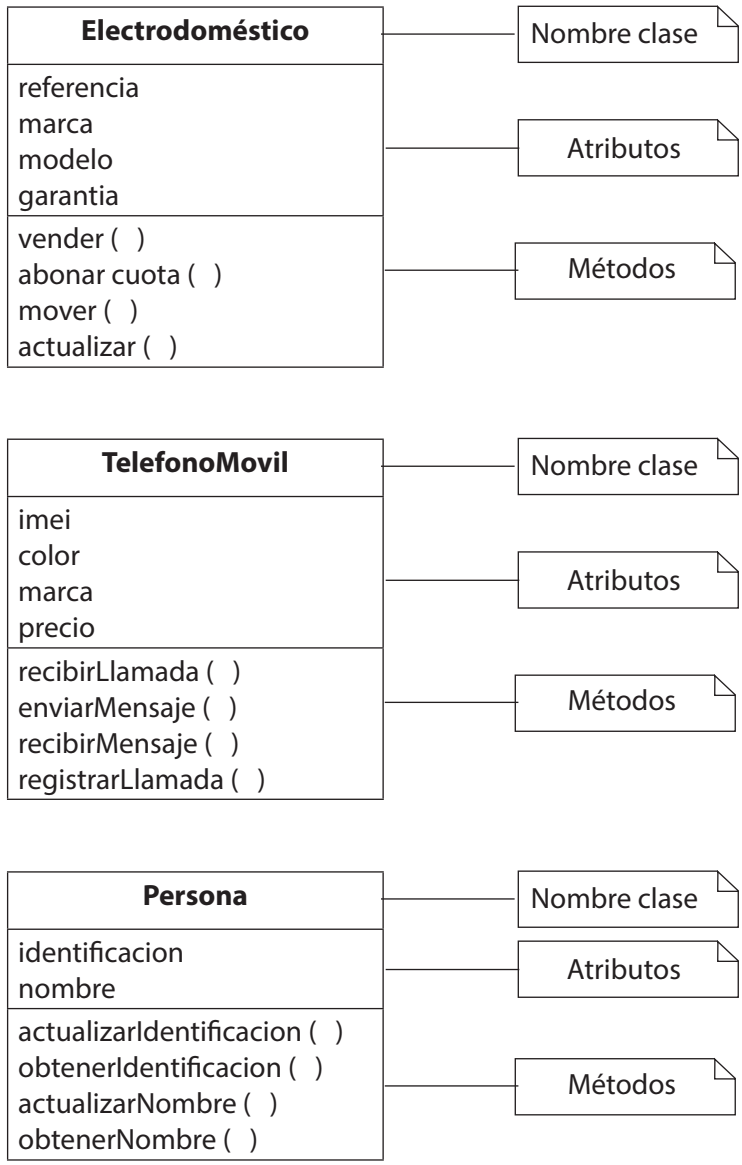


UML (*Unified Modeling Language*) es un lenguaje que permite modelar, construir y documentar los elementos que forman un sistema software orientado a objetos.

Ejemplo

Los siguientes diagramas de clases asocian la información correspondiente de acuerdo con el número del cuadrante, así, en el primer cuadrante se registra el nombre de la clase, en el segundo cuadrante los nombres de los atributos y en el tercer cuadrante los nombres de los métodos, comportamientos o responsabilidades que realizan.





Los ejemplos anteriores reprentan diagramas de clases con sus correspondientes atributos y métodos; los rectángulos de los lados unidos con líneas punteadas corresponden a la documentación.

Durante del desarrollo de la temática, incluyendo los capítulos siguientes, se irá ampliando la información correspondiente al tema de las clases.

1.8.2 Objeto

Es cualquier cosa real (tangible o intangible) del mundo que nos rodea; por ejemplo, un auto, una casa, un árbol, un reloj, una estrella, una cuenta de ahorro. Un objeto se puede considerar como una entidad compleja provista de propiedades o características y de comportamientos o estados. La figura 4 *(véase en pág. siguiente)* detalla el concepto de objeto de una manera más somera.

A un objeto lo identifican las siguientes características:

Estado

Conformado por el conjunto de propiedades o atributos de un objeto correspondiente a los valores que pueden tomar cada uno de esos atributos.

Ejemplo

El objeto cliente tiene las siguientes características: documento, nombres, apellidos, dirección y teléfono.

Tabla 2.
Ejemplo de objeto

Atributo	Estado
Documento:	10.265.254
Nombres:	Luis Eduardo
Apellidos:	Baquero Rey
Teléfono:	320 256 32 18

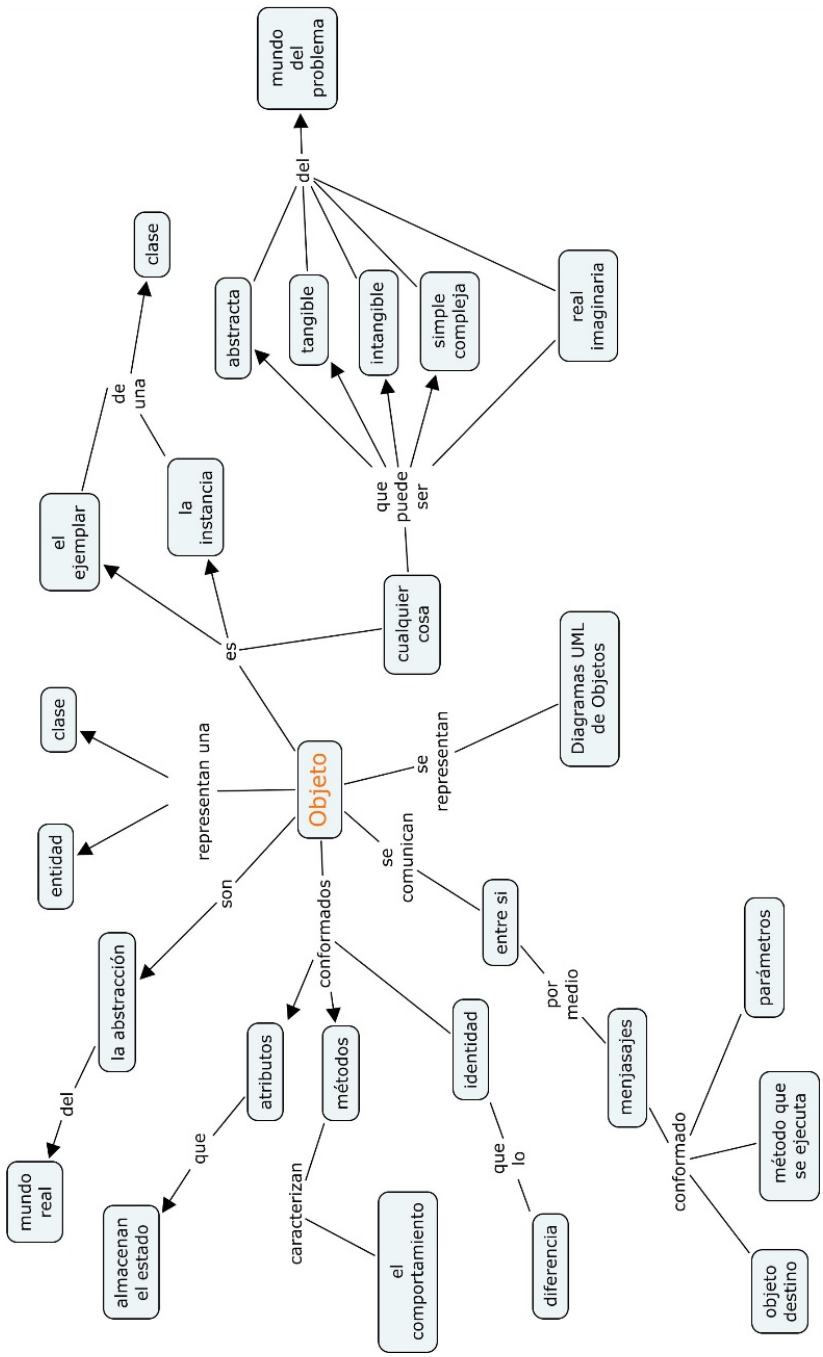


Figura 4.
Mapa mental de objeto

Comportamiento

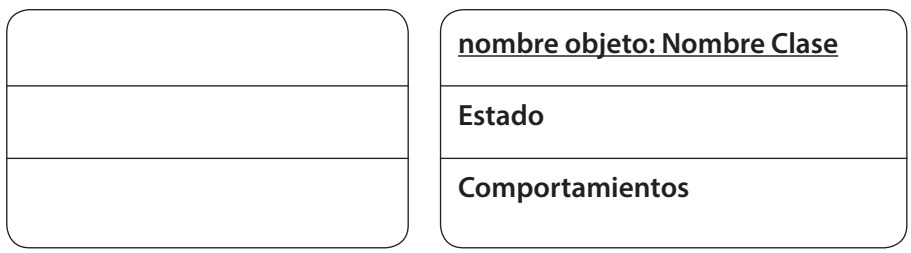
De un objeto, está relacionado con la funcionalidad y determina las operaciones que este puede realizar, responder o ejecutar alguna acción ante mensajes enviados por otros objetos, por ejemplo, actualizar el teléfono de un cliente, realizar la suma de dos números.

Identidad

Corresponde a la propiedad de un objeto que le distingue de todos los demás, como es el caso del ejemplo del cliente Luis Eduardo Baquero Rey, que tiene un número de documento que lo identifica, una dirección y un teléfono.

1.8.2.1 Representación gráfica de los objetos

Los objetos se pueden representar con diagramas UML de objeto, que es un rectángulo con tres divisiones.



Se pueden confundir los términos clase y objeto; en general, una clase es una representación abstracta de algo, mientras que un objeto es una representación de ese algo conformado por la clase.

Ejemplo

Se tienen cuatro objetos correspondientes a relojes (de pared, arena, digital y de pulsera). La clase Reloj tiene las características que agrupa a esa colección de objetos.