Conceptos Básicos del Paradigma Orientado a Objetos

Programación II

Abstracción y Programación Orientada a Objetos (POO)

- La programación orientada a objetos, de alguna manera, trata de "modelizar" los elementos del mundo real.
- La idea es desarrollar programas, en los que las personas con algunos clics efectúan una actividad, sin saber que hay una gran complejidad.

Ejemplo

- Todo depende de la capacidad para realizar una abstracción de lo que se quiere modelar, para encontrar la mejor solución que resuelve alguna necesidad.

Peces

Sexo
Color
Edad
Tamaño
Tipo

+Gritar();
+Nadar();
+Comer();

POO

- En lugar de tratar de modelar un problema en algo familiar a la computadora ahora se trata de acercar la computadora al problema.

- Es decir, modelar la realidad del problema a través de entidades independientes pero que interactúan entre sí. Estas entidades serán denominadas objetos.

- Resolver problemas consiste en definir objetos y sus acciones y entonces invocar las acciones enviando mensajes a los objetos que ocultan las características internas de cómo llevan a cabo estas acciones

POO

- La *POO*, es una técnica de programación cuyo soporte fundamental es el ***objeto***.
- Un objeto es una extensión de un *Tipo de Dato Abstracto* (TDA).
- Un TDA es un tipo definido por el usuario, que encapsula un conjunto de datos y las operaciones sobre estos datos.

POO

- A la hora de definir TDA's (u objetos) se usa un concepto que ayuda a representar la realidad, la ***abstracción***.

- La diferencia entre el concepto de TDA y el de ***objeto*** radica en que existen dos procesos con los que se forma el núcleo principal de la programación orientada a objetos, estos son la *herencia* y el *polimorfismo*.

Principios de la POO

- Programar en un lenguaje orientado a objetos es definir **clases** que expresan una determinada funcionalidad la cual es común a todos los individuos de la clase.

- A través de esta funcionalidad los **objetos** dan respuesta a las solicitudes (**mensajes**) que les envían otros objetos.

- Las clases deben ser lo suficientemente CERRADAS como para que cada objeto pueda ocultar información (datos) que lo caracteriza como individuo.

Principios de la POO

- Las clases deben ser lo suficientemente ABIERTAS para permitir la reutilización, adaptación y extensión de las mismas a nuevas funcionalidades sin correr el riesgo de afectar el funcionamiento de lo que ya es correcto.

Principio ABIERTO-CERRADO → piedra angular de la POO

Principios de la POO

- **Principio de Parnas u ocultamiento.** No permitir que los usuarios modifiquen ni la representación de la información ni las operaciones sobre esa información.

- **Principio de compartir comportamiento.** Reutilizar código, no reinventarlo.

- Los mensajes entre los objetos son síncronos(en el mismo instante), al conjunto de mensajes que responde un objeto le llamamos **protocolo.**

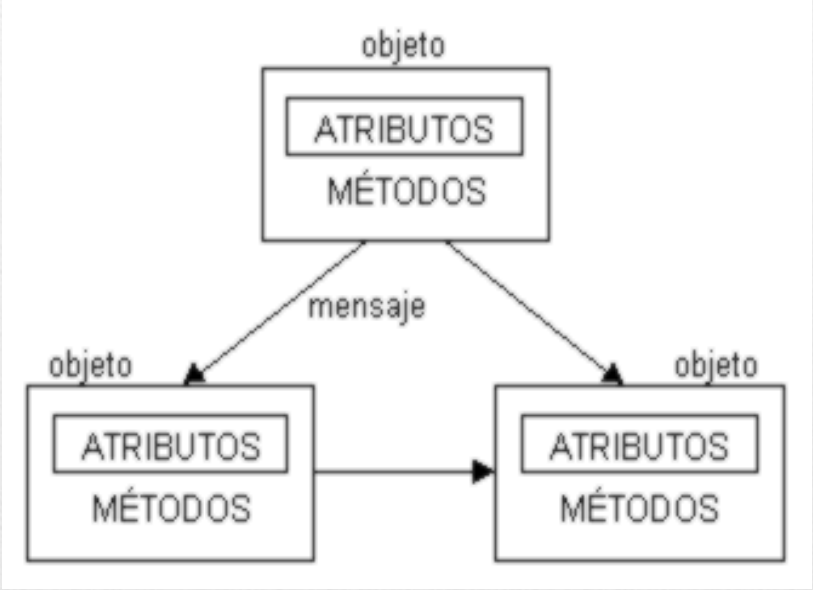
Lenguajes de POO

- **Lenguajes de POO**

- **Lenguajes de POO**
  - **Puros:** Smalltalk, Eiffel, Actor, Java
  - **Híbridos:** C++, Objetive-C, Object-Pascal

POO

- Es un paradigma de programación
- Usa **objetos** en sus interacciones
- Para d**iseñar aplicaciones**
- y **programas** informáticos.



Fuente: http://hacerpaginaswebconphp.com/php-introduccion-a-los-objetos-poo/

POO

- La POO nació en 1969 con el Dr.Kristin Nygaard, él trató de describir el movimiento de los barcos.

Fuente: https://users.dcc.uchile.cl/~rbaeza/inf/nygaard.html

- Observa que era muy difícil simular del mundo real(mareas, formas de líneas de costas y los movimientos de los barcos) con los métodos de programación existentes.

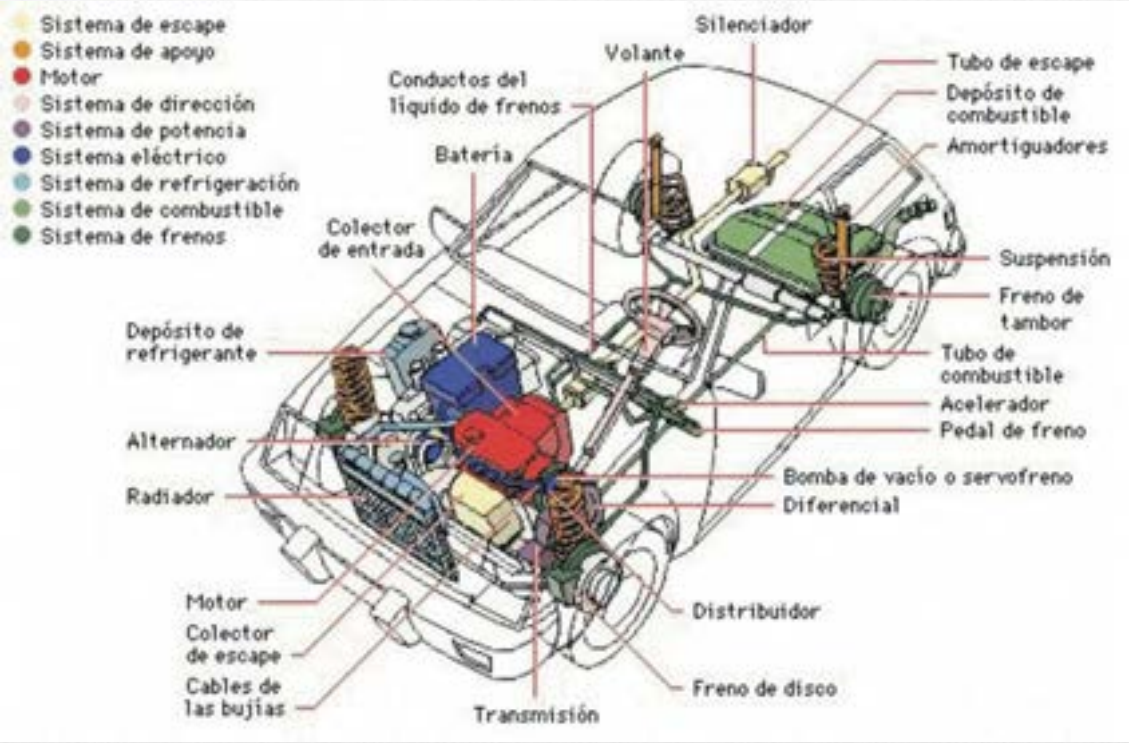
POO

- Era mas fácil adecuar el mundo real a la computadora que al revés.
- El primer lenguaje OO fue Simula 67

Fuente: https://sp.depositphotos.com/36971793/stock-photo-motor-boat-rio-yachts-best.html

Ejemplo

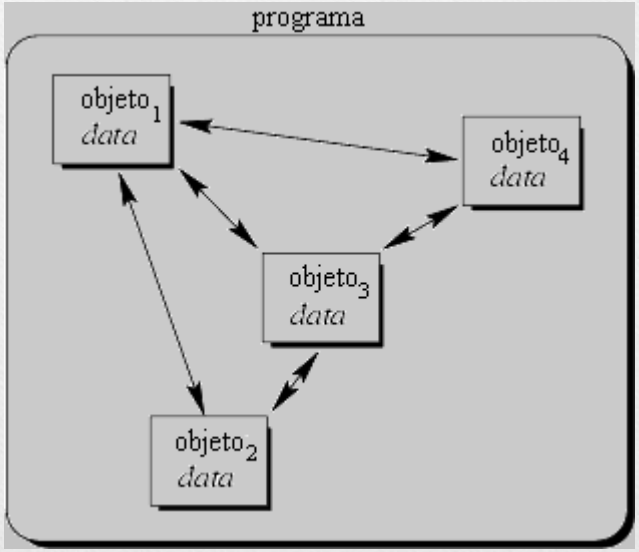
- Las partes de un coche pueden ser internamente muy complejas pero lo importante es como interactúan entre si



Fuente: https://www.atikoestudio.com/disenador/industrial/automovil/index.htm

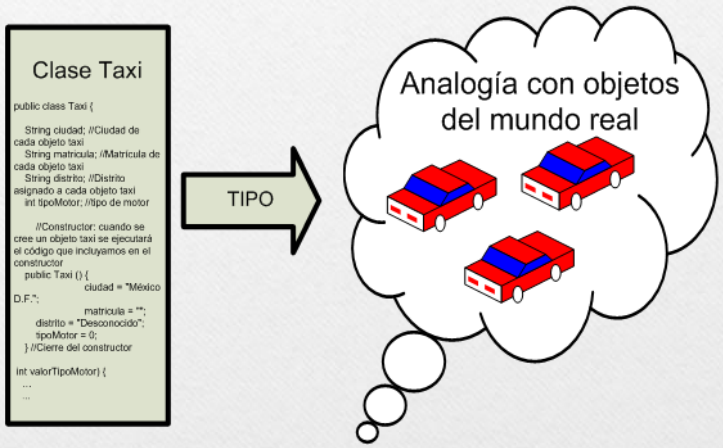
POO

- Un programa OO se forma de muchos componentes independientes y diferentes, llamados **objetos**.
- Cada uno con funcionamiento específico
- Se **comunican** o dan respuesta a solicitudes (**mensajes**) de los demás

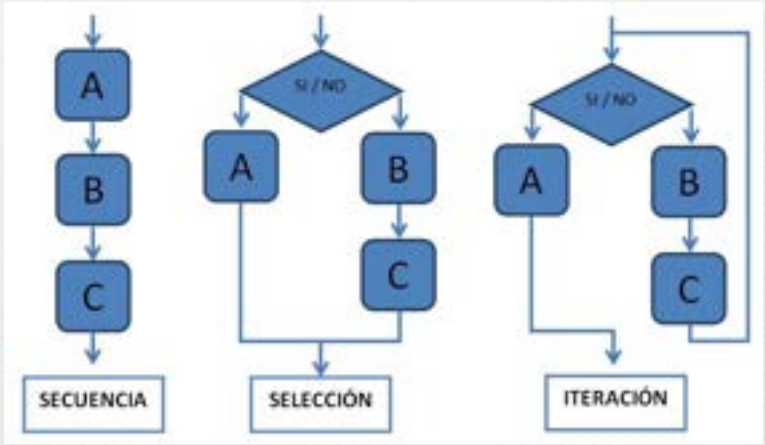


Fuente: https://html.rincondelvago.com/poo_1.html

POO vs programación estructurada

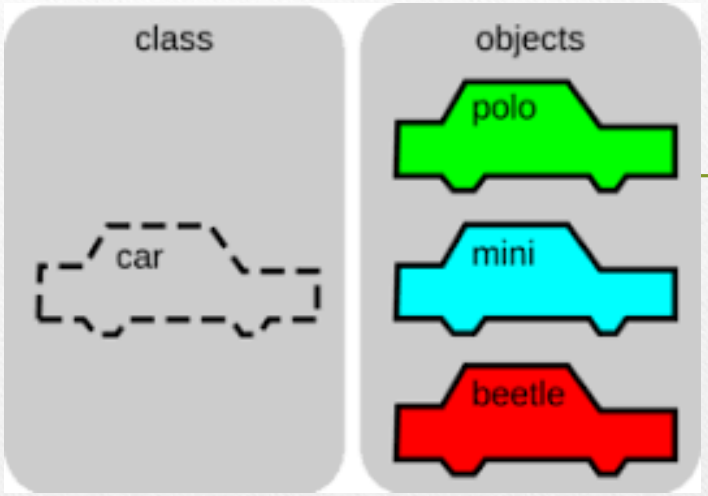


Fuente: https://www.aprenderaprogramar.com/index.php?option=com_content&view=article&id=525:concepto-y-definicion-de-clase-en-java-objetos-del-mundo-real-y-abstractos-ejemplos-y-ejercicio-cu00644b&catid=68&Itemid=188



Fuente: https://medium.com/laboratoria-how-to/programaci%C3%B3n-estructurada-7fe400bae43d

POO Vs Programación estructurada



Fuente: https://www.aluracursos.com/blog/poo-que-es-la-programacion-orientada-a-objetos

- Encapsula datos y métodos

Clase es la unidad de programación

Fuente: https://es.slideshare.net/HenryCentenoMendoza/programacin-estructurada-24456817

- Orientado a acciones.

Función es la unidad de programación

Ventajas de la POO

- *Mantenibilidad:* Facilidad de mantenimiento.
- *Modificabilidad:* Facilidad para modificar los programas.
- *Reusabilidad*: Los objetos, si han sido correctamente diseñados, se pueden usar numerosas veces y en distintos proyectos.
- *Fiabilidad:* Los programas orientados a objetos suelen ser más fiables ya que se basan en el uso de objetos ya definidos que están ampliamente testados.

Ejemplo: Algoritmo en POO

Encender un auto

1. Identificar los objetos

   llave, switch, clutch, palanca de velocidades

2. Relacionarlos de acuerdo a su funcionalidad

   a. poner **palanca de velocidades** en neutral

   b. colocar la **llave** dentro del **switch**

   c. presionar el **clutch** al fondo

   d. girar **llave** a la derecha hasta topar y soltar suavemente

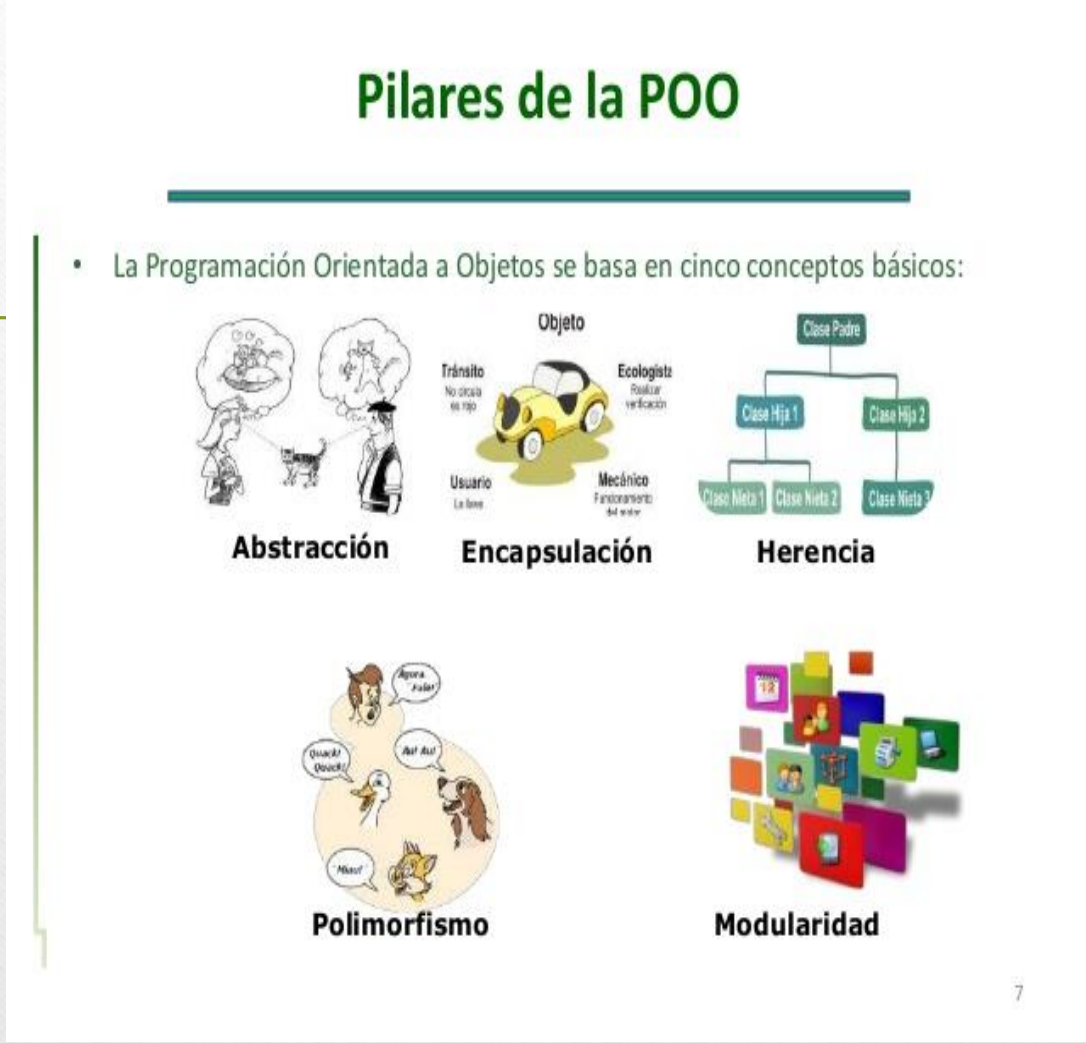
   e. si el auto encendió. Fin del algoritmo

   f. Caso contrario. Girar **llave** a posición inicial, esperar 2 seg. y repetir desde el inciso d.

POO

- La **POO** es un paradigma de programación que usa los **objetos** en sus interacciones, para diseñar programas.
- En POO los TDA's se refieren a las clases
- Está basado en varias técnicas, incluyendo herencia, abstracción, polimorfismo y encapsulamiento.

- **Abstracción**
- Objeto
- **Encapsulamiento**
- Clase
- Jerarquía
- **Herencia**
- **Modularidad**
- **Polimorfismo**



Fuente: http://toopropedeutico.blogspot.com/2015/10/bases-en-que-se-fundamenta-poo.html

Abstracción

- Lo que tenemos son especímenes de "vaca", "hormiga", "cocodrilo“ o "gorrión" pero no "**animal**" en plan general. Es cierto que una vaca es un animal, pero el concepto final, el ejemplar, es de vaca y no animal.
- Por tanto "animal", es un concepto genérico, pero no una concreción. Es decir no puedes tocar “animal” pero si a una “vaca”.

Abstracción

- En términos de POO animal es un concepto abstracto, que se implementa a través de una clase abstracta.
- No se instancian animales como tal en el mundo.
- Si se instancian especímenes de un tipo de animal concreto.

Abstracción

- **Aislar** un elemento de su contexto o del resto de los elementos que lo acompañan, **identificando** sus **características** esenciales que lo distinguen de los demás.
- Capacidad de conceptualizar entidades genéricas de información a partir de cosas concretas.

Abstracción

- Hay muchas aves diferentes pero cuando vemos una en particular la podemos reconocer inmediatamente aunque nunca la hayamos visto antes.

Ave

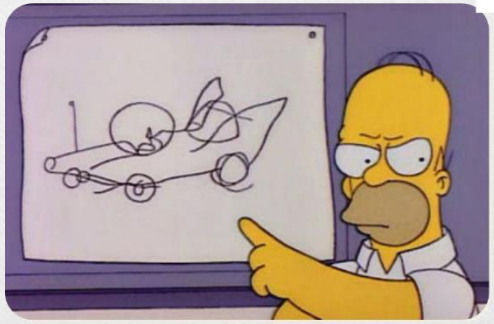
- Vertebrado
- Sangre caliente
- Con pico y alas
- Con plumas
- Se reproduce por huevos
- Respiración pulmonar
- Circulación doble y completa.

Fuente: https://www.anipedia.net/aguilas/

Abstracción

- Se centra en las **características esenciales** de algún objeto, en relación a la **perspectiva** del **observador**

Abstracción

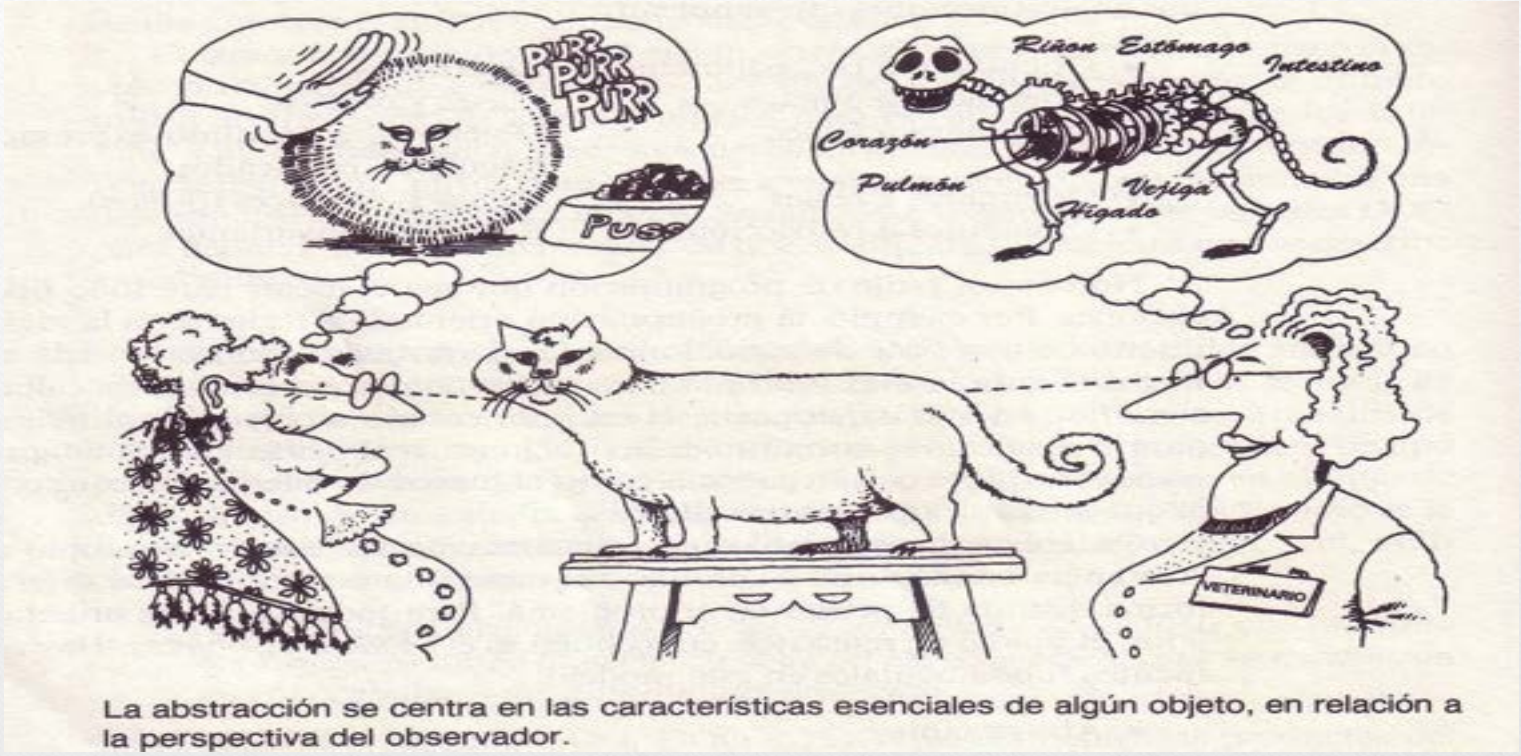


Homero Simpson construyendo el auto de sus sueños

Énfasis en el ¿qué hace? mas que en el ¿cómo lo hace?

Fuente: https://programacion2.webcindario.com/pag3.html

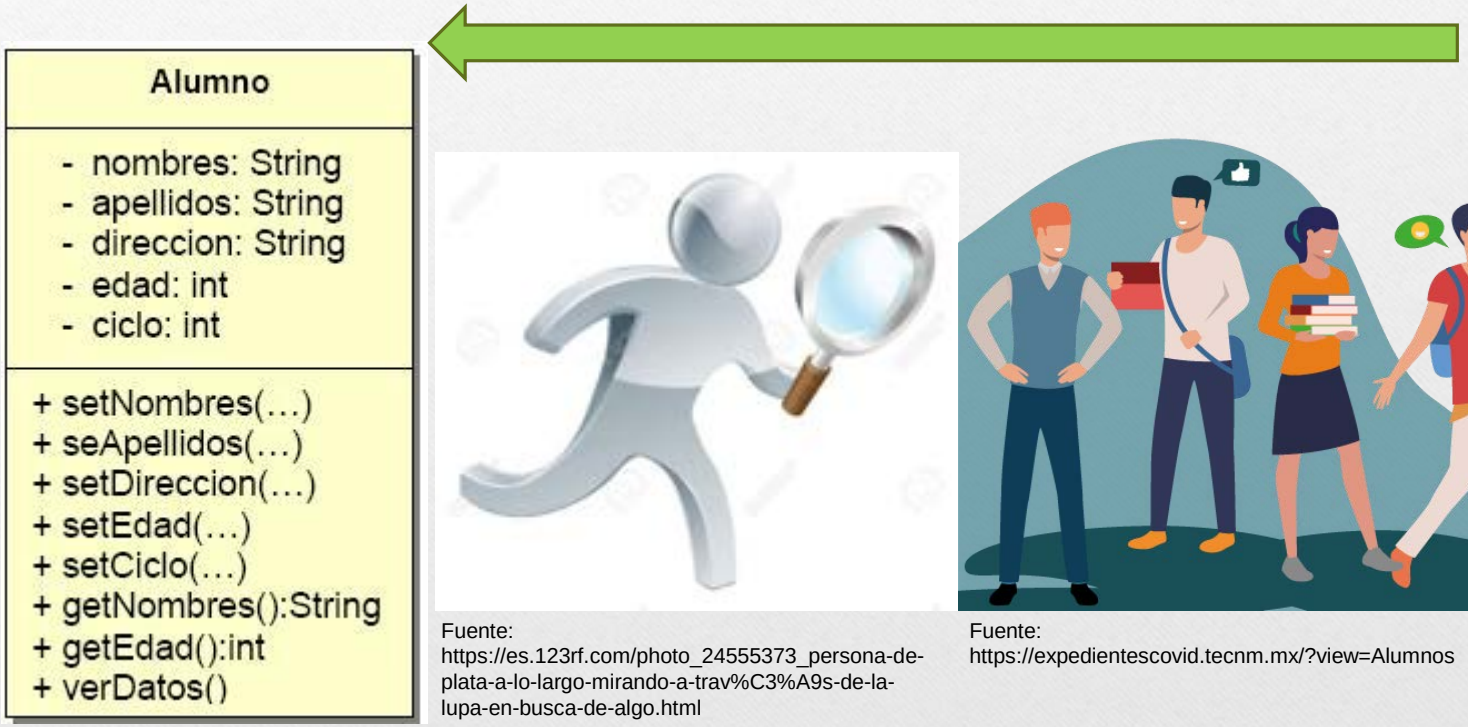
TDA



La abstracción se centra en las características esenciales de algún objeto, en relación a la perspectiva del observador.

Fuente: https://aeiua.wordpress.com/2017/11/07/metafora-acerca-de-conceptos-de-la-poo/

Ejemplo: Abstracción de Alumno



Fuente: https://es.123rf.com/photo_24555373_persona-de-plata-a-lo-largo-mirando-a-trav%C3%A9s-de-la-lupa-en-busca-de-algo.html

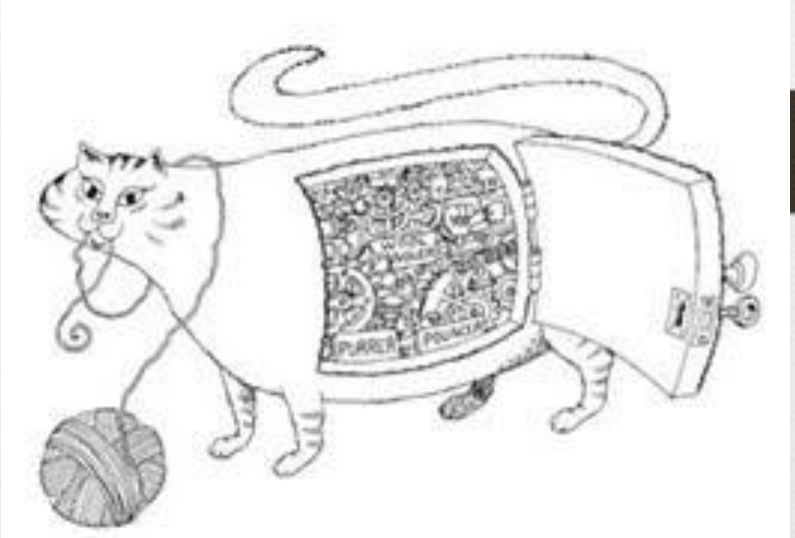
Fuente: https://expedientescovid.tecnm.mx/?view=Alumnos

Encapsulamiento

- Significa **reunir** en cierta **estructura** todos los **elementos** que a determinado nivel de abstracción, se pueden considerar de una **misma entidad**.
- Oculta lo que hace un objeto de lo que hacen otros objetos del mundo exterior

Encapsulamiento

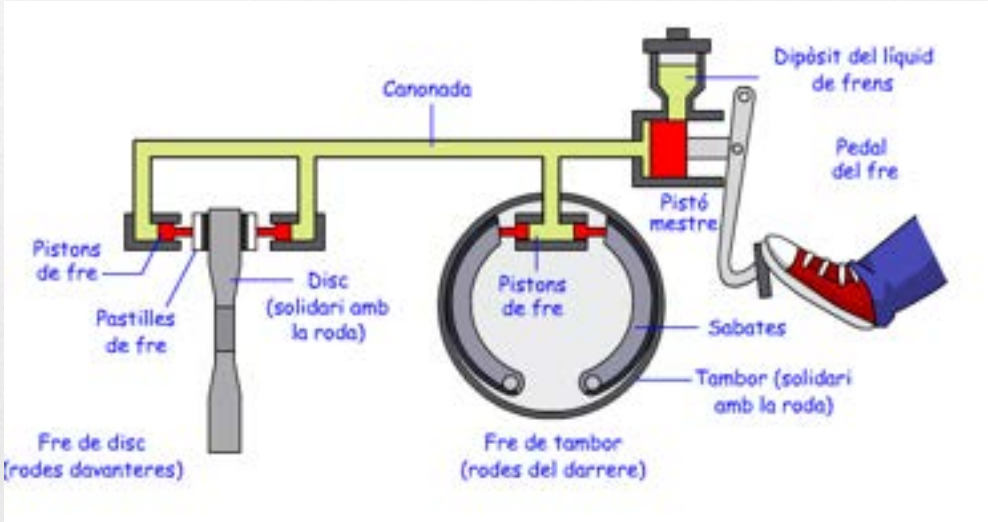
- Esconder detalles de como funciona algo, detrás de una interfaz.
- El usuario no se ve afectado si se cambia el funcionamiento interno del objeto mientras no se cambie la interfaz.



Fuente: https://aeiua.wordpress.com/2017/11/07/metafora-acerca-de-conceptos-de-la-poo/

Ejemplo

- En el objeto vehículo el encapsulamiento significa que el conductor cuando frena, no necesita conocer el sistema de frenado.



Fuente: https://sites.google.com/site/ilummelonatis/4rt-d-eso/primer-trimestre/unidad-2-neumatica/2-6-frenos-neumaticos

Clase y Objeto

- ***Objeto*** es una entidad tanto **tangible** como **intangible**, que se puede imaginar y que tiene un **estado**, un **comportamiento** y una **identidad**.

Fuente: https://www.elmundo.es/cultura/literatura/2021/11/13/618e6be9fc6c83361f8b4598.html

Fuente: https://slideplayer.es/slide/75745/

Ejemplos

Fuente: https://hogaresherso.com.mx/seguro-para-el-hogar/

seguro es un objeto intangible que tiene propiedades tales como **tipo de cobertura, costo, vigencia**, etc. Además tiene cierto comportamiento tal que este puede ser **contratado, renovado, cancelado, modificado,** etc.

Seguro
-Tipo_Cobertura -Costo -Vigencia
+Contratar(..) +Renovar(…) +Cancelar(…) +Modificar(…)

Ejemplo

mesa Objeto tangible, tiene propiedades tales como **color**, **largo**, **ancho**, etc. Y tiene comportamiento tal que puede ser **pintada**, **restaurada**, **modificada**, etc.

Fuente: https://www.cristalamedida.com/vidrio/1/productos-de-cristal

Mesa
-Color -Largo -Ancho
+Pintar(..) +Restaurar(…) +Modificar(…)

Clase

- **Clase** Tipo de molde o **plantilla** que dicta lo que los **objetos son y pueden o no hacer**.
- Es un conjunto de objetos que comparten una estructura y un comportamiento.

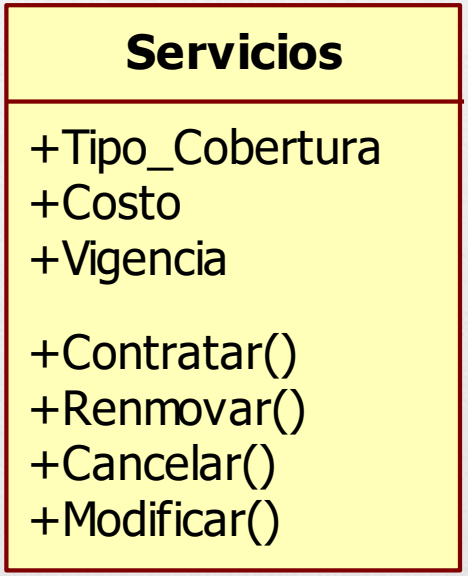
Animales

Fuente: https://www.temasambientales.com/2018/08/tipos-de-animales.html

Ejemplo

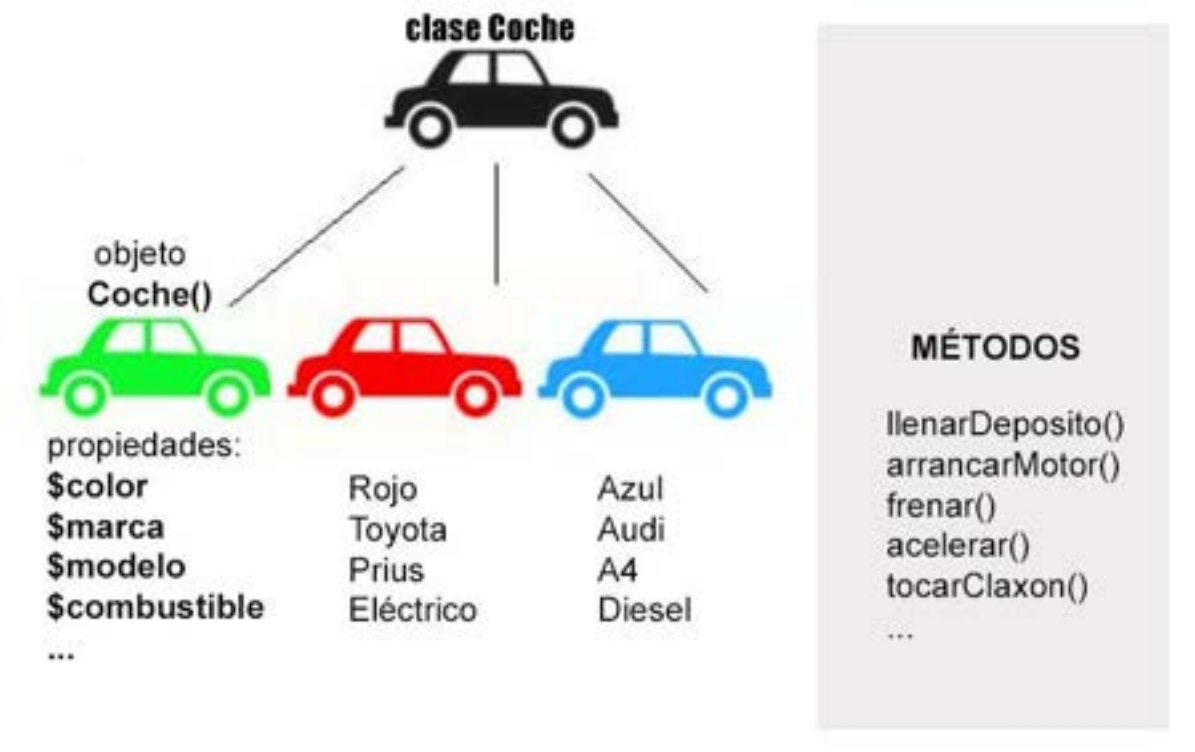
Servicios es una clase que define objetos que tienen propiedades tales como **tipo de cobertura, costo, vigencia**, etc.

Además tienen cierto comportamiento tal que estos puede ser **contratados, renovados, cancelados, modificados**, etc.



Atributos

- **Atributos** son los datos necesarios para describir los objetos creados a partir de alguna clase en particular.



Fuente: https://programandoamimanera.com/programacion-orientada-a-objetos-con-php-i-para-empezar/

Ejemplo

el objeto **Seguro de Juan** es identificado por los atributos: tipo de cobertura, costo, vigencia, etc.

Seguro de Juan
-Tipo_Cobertura -Costo -Vigencia
+Contratar() +Renovar() +Cancelar() +Modificar()

Fuente: https://www.vivendo.co/actualidad/tips-para-comprar-vivienda-nueva/sabes-que-incluye-tu-seguro-de-vivienda

Estado

- Posibles **condiciones** en que un objeto puede existir.
- El estado de un objeto puede **cambiar** durante el tiempo.

Fuente: https://www.amazon.com/-/es/Twitter-Thunder-velocidades-carbono-bicicleta/dp/B08Y788QCG

Fuente: https://es.123rf.com/photo_15303608_vieja-bicicleta-roja-con-fondo-de-hormig%C3%B3n.html

Ejemplo

Seguro de Juan
Amplia 150,000 3 años
Contratar() Renovar() Cancelar() Modificar()

Fuente: https://blog.verti.es/hogar/cuanto-cuesta-seguro-hogar/

Método

- **Método** es una secuencia de **instrucciones** que una clase u objeto sigue para realizar una tarea.
- Es un conjunto de operaciones que **manipulan** a los atributos del objeto.

Fuente: https://twitter.com/Anaro74/status/1487992399590744066

Ejemplo

- En la clase **Seguros** los métodos que manipulan a los atributos de los objetos son: **contratar**, **renovar**, **cancelar**, **modificar**, etc

Fuente: https://www.advans.es/seguro-vida

Práctica 2

- En equipos de 4 resuelve los problemas.

  1. Cambiar un neumático

  2. Pintar una mesa

- Analicen el problema e identifiquen en una lista todos los objetos.
- Usando los objetos de acuerdo a su funcionalidad, escriban un algoritmo para resolver el problema.

Tarea 2

- Resuelve los problemas.

  1. Hacer Tarea

  2. trasladarse a la escuela

- Analiza el problema e identifica en una lista todos los objetos.
- Usando los objetos de acuerdo a su funcionalidad, escribe un algoritmo para resolver el problema.

Introducción al Lenguaje Unificado de Modelado

Introducción

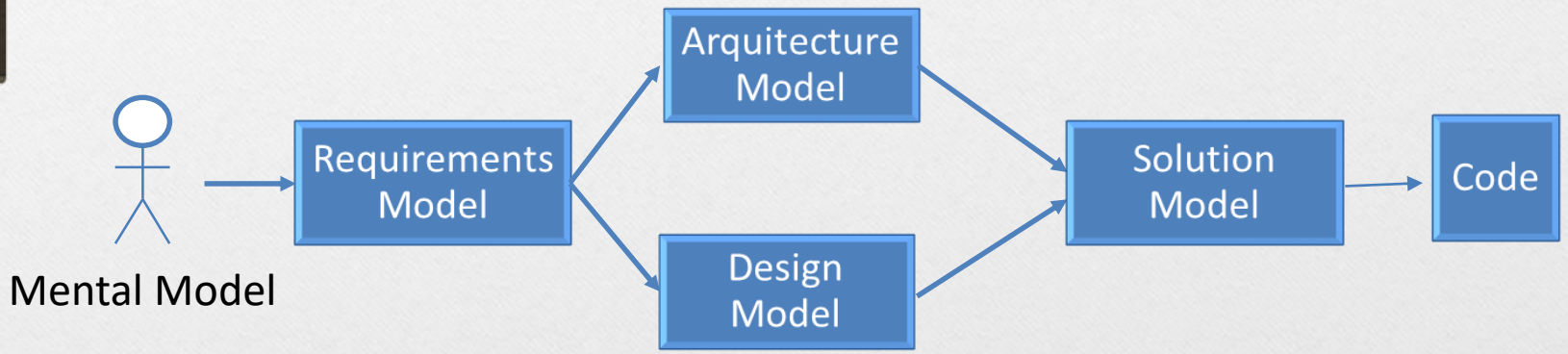
- Es un **lenguaje** con un vocabulario reglas para utilizarlo.
- Además es un lenguaje de modelado pues el vocabulario y las reglas se utilizan para la representación conceptual y física del sistema.
- Es un lenguaje para interpretar grandes sistemas mediante gráficos o mediante texto obteniendo modelos explícitos que ayudan a la comunicación durante el desarrollo.
- Al ser estándar, los modelos podrán ser interpretados por personas que no participaron en su diseño sin ninguna ambigüedad.

Introducción

- Se compone de tres elementos básicos, los bloques de construcción, las reglas y algunos mecanismos comunes.
- Los **bloques de construcción** se dividen en tres partes: **Elementos**, que son las abstracciones de primer nivel, **Relaciones**, que unen a los elementos entre sí, y los **Diagramas**, que son agrupaciones interesantes de elementos.
- Existen cuatro tipos de elementos en UML, dependiendo del uso que se haga de ellos: *elementos estructurales*, *elementos de comportamiento, elementos de agrupación y elementos de anotación.*

Introducción

- El desarrollo de software puede ser visto como una serie de transformaciones del modelo mental inicial al código actual



Bloques de construcción

- Elementos Estructurales:
  - Los elementos estructurales en UML, es su mayoría, son las partes estáticas del modelo y representan cosas que son conceptuales o materiales.

Bloques de construcción

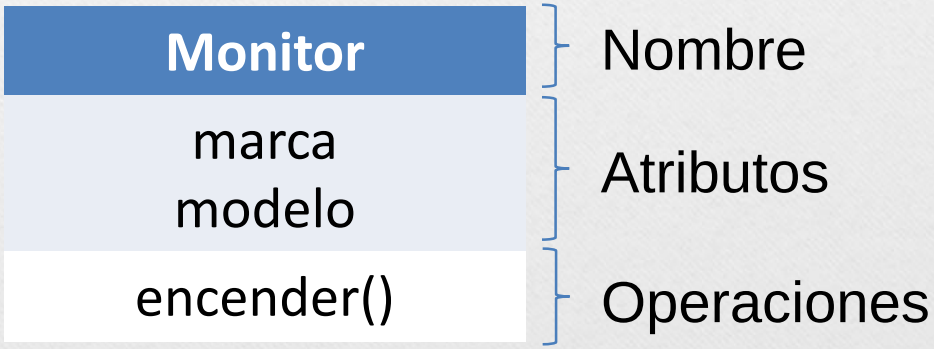
Clases:

- Una clase es una descripción de un conjunto de objetos que comparten los mismos atributos, operaciones, relaciones y semántica.
- Una clase implementa una o más interfaces. Gráficamente se representa como un rectángulo que incluye su nombre, sus atributos y sus operaciones.

Ventana
origen tamaño
abrir() cerrar() mover() dibujar()

Representación de clases

- Una **clase** es una descripción de un conjunto de objetos que comparten los mismos atributos, operaciones, relaciones y semántica.
- Un **objeto** es una instancia individual de una clase.
- Una **clase** se identifica por un nombre que la distingue del resto. Ese nombre sólo se denomina nombre simple; un nombre de camino consta del nombre de la clase precedido del nombre del paquete en el que se encuentra incluida.



Representación de clases

- Un **atributo** es una propiedad de una clase identificada con un nombre.
- Describe un rango de valores que pueden tomar las instancias de la propiedad.
- Los atributos representan propiedades comunes a todos los objetos de una determinada clase. Los atributos son propiedades interesantes de las clases.

Representación de clases

- Una instancia de una determinada clase tendrá valores concretos para sus atributos, mientras que las clases tienen rangos de valores que pueden admitir sus atributos.
- Una **operación** es una implementación de un servicio que puede ser requerido a cualquier objeto de la clase para que muestre su comportamiento.
- Una **operación** representa algo que el objeto puede hacer.