PROGRAMACIÓN ORIENTADA A OBJETOS

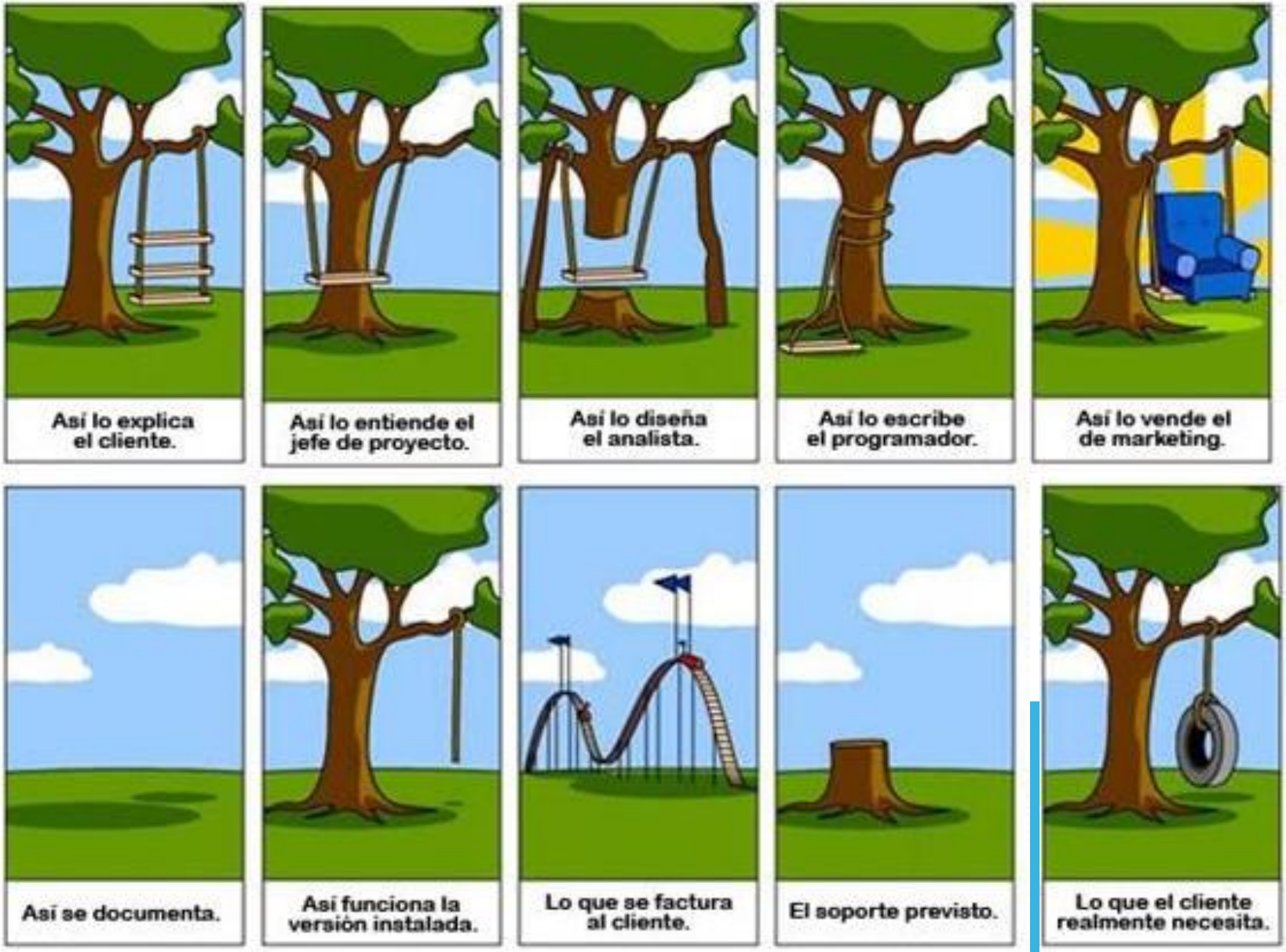
POO

19/01/2015

UNIDAD I:
EL PARADIGMA ORIENTADO A OBJETOS.

- *1.1 Introducción, surgimiento y desarrollo del paradigma Orientado a Objetos*
- *1.2 Clases y Objetos*
- *1.3 Encapsulamiento*
- *1.4 Herencia*
- *1.5 Polimorfismo*
- *1.6 Sobrecarga de Operadores y Funciones*

CRISIS DEL SOFTWARE



CRISIS DEL SOFTWARE

Englobó a una serie de sucesos que se venían observando en los proyectos de desarrollo de software:

Los proyectos no terminaban en plazo.

Los proyectos no se ajustaban al presupuesto inicial.

Baja calidad del software generado.

Software que no cumplía las especificaciones.

Código inmantenible que dificultaba la gestión y evolución del proyecto.

1.1.- INTRODUCCIÓN, SURGIMIENTOS Y DESARROLLO DEL PARADIGMA ORIENTADO A OBJETO

Lenguaje = Idioma Artificial

La palabra programación se define como el proceso de creación de un programa de computadora, mediante la aplicación de procedimientos lógicos, a través de los siguientes pasos:

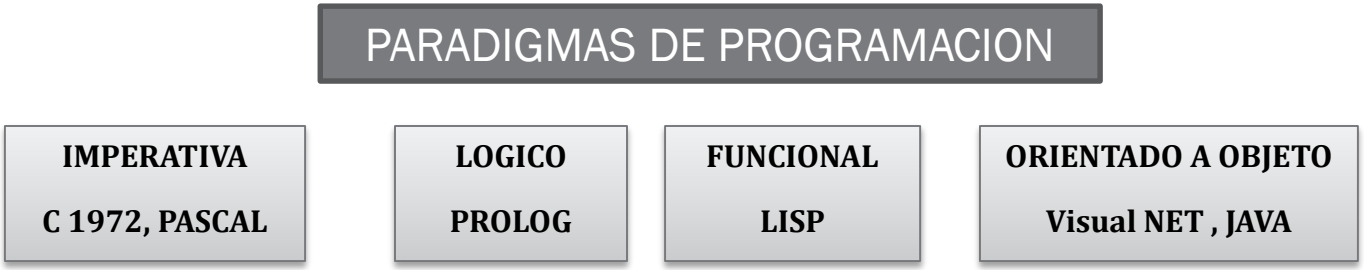
- ✓El desarrollo lógico del programa para resolver un problema en particular.
- ✓Escritura de la lógica del programa empleando un lenguaje de programación específico (codificación del programa).
- ✓Ensamblaje o compilación del programa hasta convertirlo en lenguaje de máquina.
- ✓Prueba y depuración del programa.
- ✓Desarrollo de la documentación.

PARADIGMAS DE PROGRAMACION

IMPERATIVA	LOGICO	FUNCIONAL	ORIENTADO A OBJETO
C 1972, PASCAL	PROLOG	LISP	Visual NET , JAVA

Qué es un paradigma de programación?

Paradigma un conjunto de teorías, estándar y métodos que juntos representan un medio de organización del conocimiento: es decir un medio de visualizar el mundo.



EL PARADIGMA IMPERATIVO

Este paradigma duro lo suficiente hasta que las compañías y empresas tuvieron una explosión de sus actividades y pasaron de ser locales a ser regionales y nacionales, aumentado su grado de complejidad y manejo de información, este se agudizo cuando las necesidades pasaron las fronteras de los países y las empresas nacionales se trasformaron en compañías internacionales.

Programación Imperativa, los primeros lenguajes imperativo fueron los lenguajes de máquina

EL PARADIGMA LÓGICO

Orientado a la Inteligencia Artificial

La programación lógica comprende dos paradigmas de programación: la programación declarativa y la programación funcional.

- La programación declarativa gira en torno al concepto de predicado, o relación entre elementos.

le_gusta_a(juan,maria).

valioso(oro).

Tiene(juan,libro).

da(juan,libro,maria).

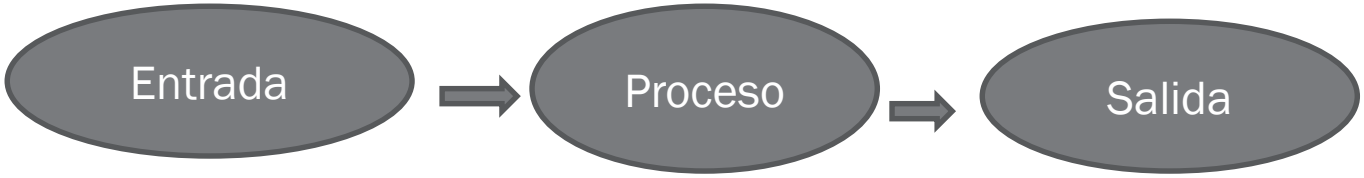
- La programación funcional se basa en el concepto de función (que no es más que una evolución de los predicados), de corte más matemático.

En cambio, la lógica matemática es la manera más sencilla, para el intelecto humano, de expresar formalmente problemas complejos y de resolverlos mediante la aplicación de reglas, hipótesis y teoremas. De ahí que el concepto de "programación lógica" resulte atractivo en diversos campos donde la programación tradicional es un fracaso

Campos de Aplicación

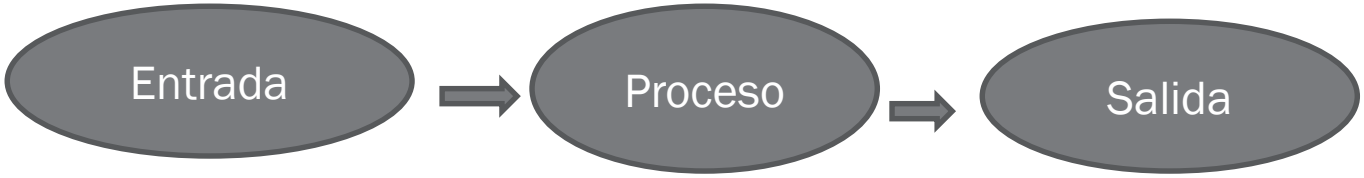
- ✓ Sistemas expertos, donde un sistema de información imita las recomendaciones de un experto sobre algún dominio de conocimiento.
- ✓ Demostración automática de teoremas, donde un programa genera nuevos teoremas sobre una teoría existente.
- ✓ Reconocimiento de lenguaje natural, donde un programa es capaz de comprender (con limitaciones) la información contenida en una expresión lingüística humana.

EL PARADIGMA FUNCIONAL



- ✓ Los matemáticos desde hace un buen tiempo están resolviendo problemas usando el concepto de función. Una función convierte ciertos datos en resultados
- ✓ Lisp el más antiguo a mediado de los 50

EL PARADIGMA FUNCIONAL



- ✓ Lisp el más antiguo a mediado de los 50

EL PARADIGMA ORIENTADO A OBJETOS

- ✓ Podemos afirmar que el paradigma orientado a objeto reina en las metodologías actuales y pasaran varios años para que resulte un nuevo paradigma para que la reemplace totalmente. La programación orientada a objeto vino para quedarse y mejorar nuestras vidas.
- ✓ En POO, las entidades centrales son los objetos, que son tipos de datos que encapsulan con el mismo nombre estructuras de datos y las operaciones o algoritmos que manipulan esos datos.

✓En filosofía un objeto es aquello que puede ser observado, estudiado y aprendido.
✓El término objeto tiene el mismo significado que un nombre o una frase nominal. Es una persona, lugar o cosa.
✓Un objeto en POO representa alguna entidad de la vida real, es decir, alguno de los objetos que pertenecen al negocio con que estamos trabajando o al problema con el que nos estamos enfrentando, y con los que podemos interactuar

En síntesis los objetos conocen cosas y hacen cosas

- Las cosas que un objeto conoce son sus atributos.
- Las cosas que hace un objetos es su comportamiento.

Otros objetos

Telefonía BETTY!
DE BEATRIZ GUTIGUTI
JUJUY N° 461 - ☎(0388) 4232633
4600 S. S. DE JUJUY - PROV. DE JUJUY
RESPONSABLE MONOTRIBUTO

C Nº 0001- 00 000718

CUIT 27-17060411-3
Ing. Brutos A-1-30483
Inicio de Act. 03/01/07

Fecha 17 11 08

FACTURA

Sr.(es): Segundo Cuarta S.A.
Domicilio: Alberdi n.º 467 (P. A.) - Colón, E. Ríos

IVA — No Resp. ☐ Exento ☐ Cons. Final ☐ Resp. Monotrib. ☐ Responsable Inscripto ☒ Responsable No Inscripto ☐
CUIT 30-99887766-5

CONDICIONES DE VENTA — Contado ☒ Cuenta Corriente ☐
REMITO

TARJETA: Nº CUPON: Nº CUOTAS

Cant.	DETALLE	P. Unit.	TOTALES
1	Teléfono de línea Picu 2000	300,-	300,-
	Menos: Descuento por pago en efectivo		(30,-)
			270,-

RETIRADO POR EL CLIENTE EL 17/11/08

TOTAL $ 270,-

Imprese por: Artes Gráficas Noal de Manuel Chanuel - Tel. (0388) 4259699 Juan B. Justo 25 - CUIT 20-24324780-0 (N. M. Prov A2519/01
USO DIDÁCTICO
Fecha de Imp: ENERO/2008 Nº 00000551 al 00000800 Orig. Blanco Dup. Color

LOS OBJETOS SEGÚN SHLAER, MELLOR Y COAD/YOURDON PUEDEN CAER DENTRO DE LAS CATEGORÍAS:

- ✓ Cosas tangibles (camión, bicicleta, batería, motor, libro)
- ✓ Roles o papeles jugados o representados por personas (gerentes, operarios, usuarios cliente)
- ✓ Organizaciones o instituciones (Empresas división, departamento, equipo)
- ✓ Procesos u acciones que representan un suceso (evento) u ocurrencias, tales como llamada a un servicio, juego, acción determinada, pueden ser física o real.
- ✓ Interacciones Generalmente implican un contrato, o transacción y relacionan dos o más objetos del problema en cuestión
- ✓ Especificaciones muestran aplicaciones o plantillas de artículos, productos industriales
- ✓ Lugares oficinas, sala de estudio, aula, muelle de embarque.
- ✓ Cuando hemos determinados los posibles objetos pasamos a identificar los posibles atributos y las operaciones sobre ellos

¿QUÉ CLASE DE COSAS PUEDE SER OBJETOS EN UN PROGRAMA OO? LA RESPUESTA SOLO SE LIMITA A LA IMAGINACIÓN. EJEMPLOS TÍPICOS:

Objetos Físicos

Aviones en un sistema de trafico aéreo.

Automóviles en un sistema de control de trafico terrestre.

Casas.

Elementos de interfaces gráficos de usuario

Ventanas

Menús

Objetos gráficos (cuadrados, triángulos, etc.)

Teclado

Cuadros de dialogo

Ratón.

Animales

Vertebrados

Invertebrados

Pescados

Tipos de datos definidos por el usuario

Datos complejos

Puntos de un sistema de coordenadas.

Alimentos

Carnes

Frutas

Pescados

Verduras

Pasteles

OBJETO DEL MUNDO REAL

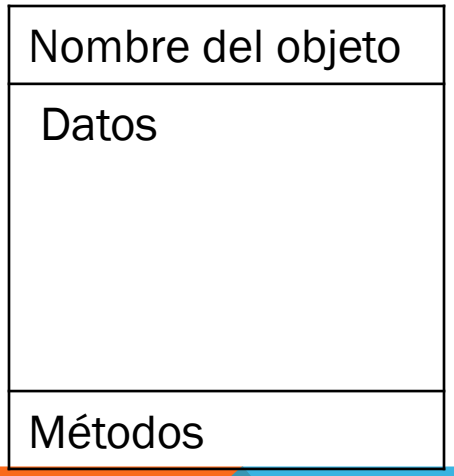
- ✓ **Objeto:** Persona
- ✓ **Atributos:** Características que lo describen (nombre, edad raza, fecha de nacimiento.)
- ✓ **Comportamiento:** el cojunto de cosas que puede hacer un objeto, matar, llorar, trabajar caminar.

Cuando hemos determinados los posibles objetos pasamos a identificar los posibles atributos y las operaciones sobre ellos

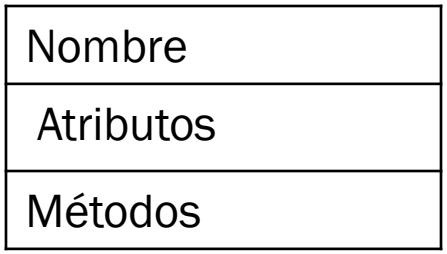
Objeto: Empleado
Atributos:
Nombre
Dirección
Ubicación
Nombre del departamento
Nivel de estudio

Objeto: Película
Atributos: Nombre
Director
Genero
Duración

Dentro del contexto de un Lenguaje OO (LOO), un objeto encapsula datos y los procedimientos/funciones (métodos) que manejan esos datos. La notación grafica varia de una metodología a otras.



Notación grafica Yourdon / Coad



Notación Grafica OMT

Atributos Datos o variables que caracterizan o describen el estado del objeto.
Métodos Procedimientos o acciones que cambian el estado de un objeto. Describen el comportamiento asociado a un objeto.

Ejemplo.

Coche 1
Matricula Marca Precio Año compra
Calcular_Precio_Actual

- ***Los principios básicos orientados a objetos son:***

Objetos *como* instancia de una clase | Atributos | Métodos | Mensaje

Las clases se organizan en las siguiente relaciones:

Herencia | Agregación | Asociación | Uso

- ***Los cuatro elementos (propiedades) que ayudan a definir un objeto son***

Abstracción | Encapsulamiento | Modularidad | Polimorfismo | Jerarquía

ABSTRACCIÓN

- ✓ *Las características esenciales de un objeto, sin preocuparse de las restantes características (no esenciales)*
- ✓ *Centrado en la vista externa de un objeto, de modo que sirva para separar el comportamiento esencial de un objeto de su implementación.*
- ✓ *Con énfasis al ¿Qué hace? Más al ¿Cómo lo hace?*
- ✓ *Permite disponer de las características de un objeto.*
- ✓ *Por ejemplo para un sistemas de matrícula del objeto persona podemos obtener las características nombre, edad, dirección, estado civil, etc.*
- ✓ *si lo requerimos para el área de biología, dentro de sus atributos quizá tengamos, ADN, Tipo de Sangre, etc.*

...ABSTRACCIÓN

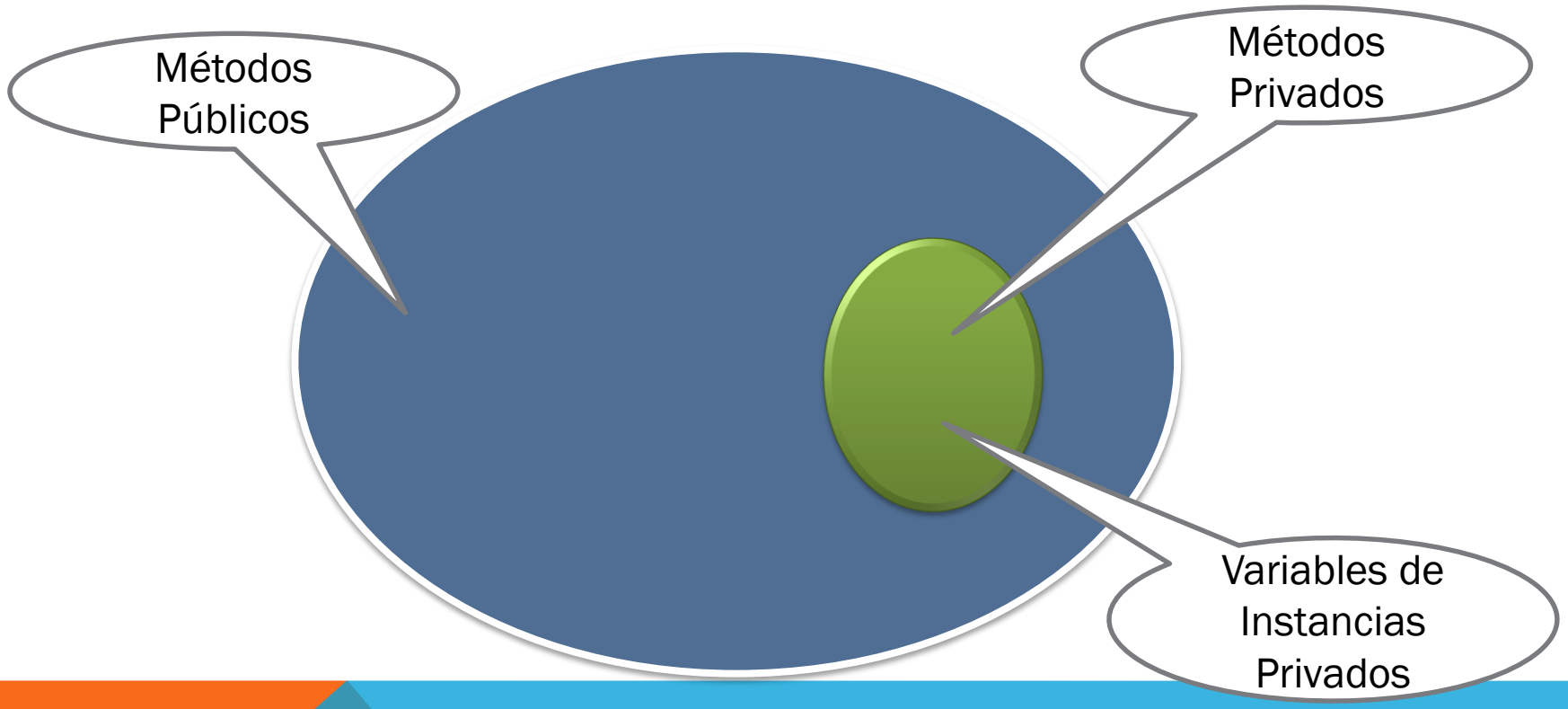
Modelando una fracción, por ejemplo

La fracción será el objeto y tendrá dos propiedades, el numerador y el denominador.

Luego podría tener varios métodos como simplificarse, sumarse con otra fracción o número, restarse con otra fracción

Fraccion
Numerador Denominador
Simplificar() Sumar() Restar()

ENCAPSULAMIENTO



ENCAPSULAMIENTO

- ✓ Permite asegurar que el contenido de la información de un objeto está oculta al mundo exterior: El contenido del objeto A está oculto al contenido del objeto B.
- ✓ Conocida como la ocultación de la información.
- ✓ La encapsulación permite la división de un programa en módulos. Estos módulos se implementan mediante clases, de forma que una clase representa la encapsulación de una abstracción, en la práctica esto significa que cada clase debe tener dos partes: un interfaz y una implementación.
- ✓ El interfaz de una clase captura solo su vista externa y la implementación contiene la representación de la abstracción, así como los mecanismos que realizan el comportamiento deseado.

ocultamiento del estado, es decir, de los datos miembro, de un objeto de manera que sólo se puede cambiar mediante las operaciones definidas para ese objeto.

Ejemplos

Cuando vemos televisión no nos preocupamos del modo como éste funciona, o lo que hace para cambiar de canal o aumentar de volumen

La encapsulación se encarga de mantener ocultos los procesos internos que necesita para hacer lo que sea que haga, dándole al programador acceso sólo a lo que necesita

Esto da dos ventajas iniciales:

- *Lo que hace el usuario puede ser controlado internamente (incluso sus errores), evitando que todo colapse por una intervención indeseada*

Te imaginas a tu mamá que no tiene ni idea de electrónica desarme el televisor y juegue con los circuitos para cambiar manualmente?? o.O

La segunda ventaja es que, al hacer que la mayor parte del código esté oculto, puedes hacer cambios y/o mejoras sin que eso afecte el modo como los usuarios van a utilizar tu código. Sólo tienes que mantener igual la forma de acceder a él

en el caso del control de la tele, que los botones sigan siendo los mismos y que el botón de “apagado” no cambie el volumen

Estas *puertas de acceso* que das a los usuarios son lo que se conoce como *interfaz*.

El encapsulamiento consiste en unir en la Clase las características y comportamientos, esto es, las variables y métodos.

ENCAPSULAMIENTO

Una de las premisas de POO, es que la mayoría, sino todos, los atributos de un objeto deben ser privados, esto para tener seguridad sobre los valores del objeto, pero entonces…¿cómo acceder a los atributos de una clase?, la respuesta, encapsulamiento, los métodos de encapsulamiento se utilizan sólo cuando es apropiado entregar los datos a otro objeto que solicite la información.

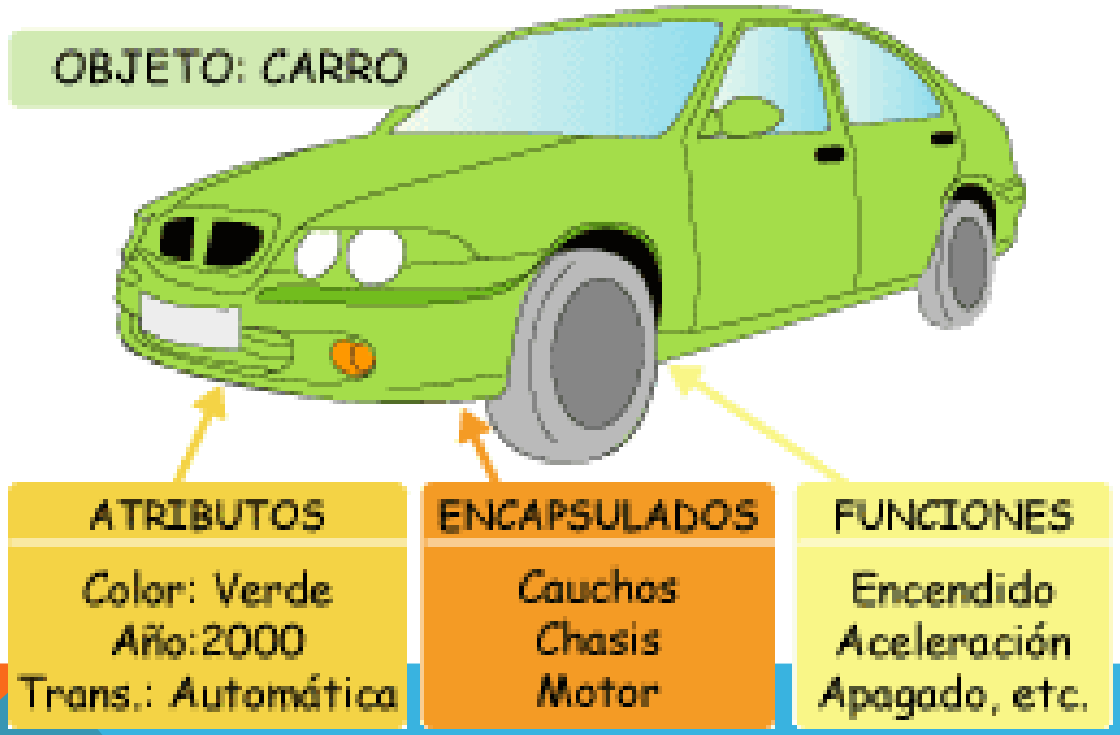
Piensa en tus atributos, tienes nombres, apellidos, etc , todos esos atributos son privados, ¿pero si son privados…porque la gente me llama por mi nombre? fácil, por los métodos de encapsulamiento que permiten al objeto entregar la información que sea solicitada por otro objeto…

…ENCAPSULAMIENTO

(ocultamiento de la información) es la idea de empacar juntas en una **Unidad** de programación una colección de miembros de datos y sus funciones (Operaciones) bien definida.

…ENCAPSULAMIENTO

(ocultamiento de la información) es la idea de empacar juntas en una **Clase** de programación una colección de miembros de datos y sus funciones (Operaciones) bien definida.

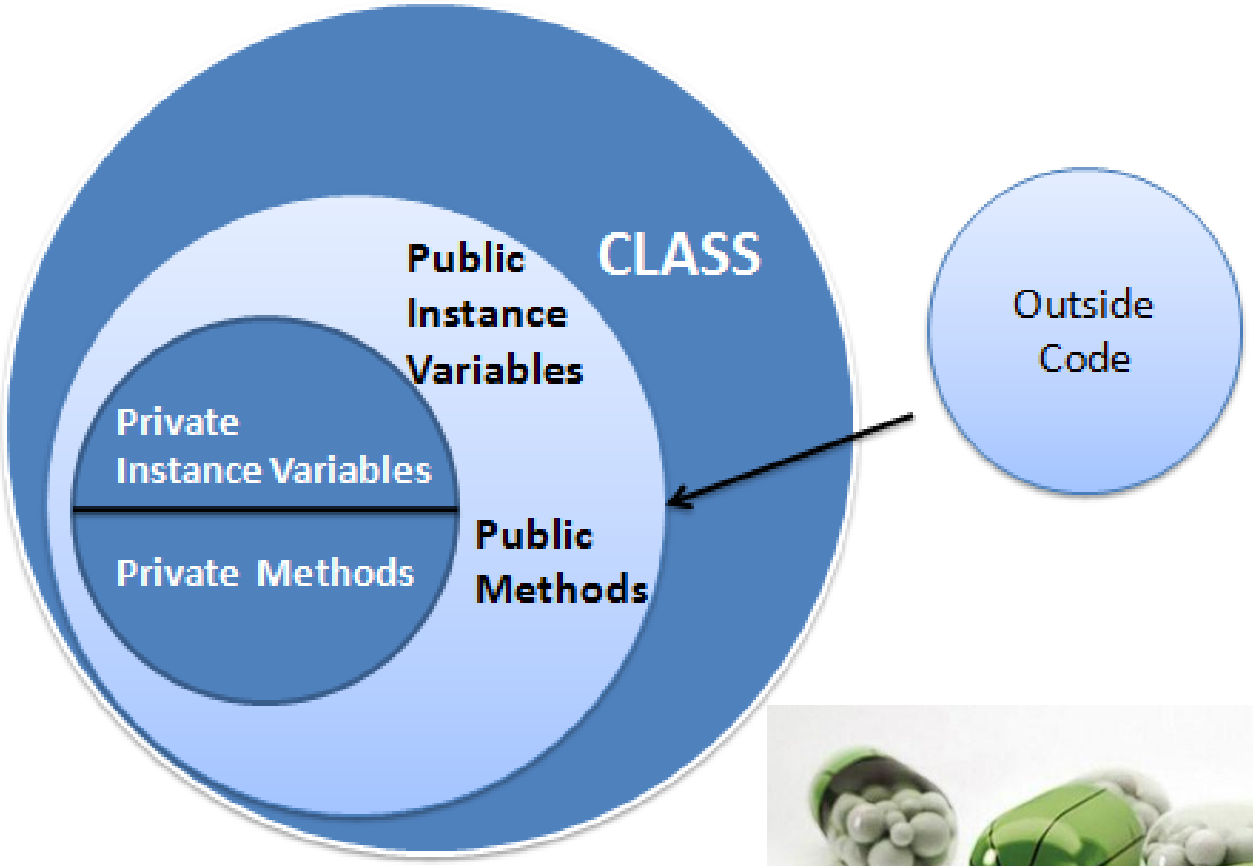

OBJETO: CARRO
ATRIBUTOS
Color: Verde
Año:2000
Trans.: Automática
ENCAPSULADOS
Cauchos
Chasis
Motor
FUNCIONES
Encendido
Aceleración
Apagado, etc.

Existen tres niveles de acceso:

- **público**: funciones de toda clase pueden acceder a los datos o métodos de una clase que se define con el nivel de acceso *público*. Este es el nivel de protección de datos más bajo

- **protegido**: el acceso a los datos está restringido a las funciones de clases heredadas, es decir, las funciones miembro de esa clase y todas las subclases

- **privado**: el acceso a los datos está restringido a los métodos de esa clase en particular. Este es nivel más alto de protección de datos (10)


CLASS
Public
Instance
Variables
Private
Instance Variables
Private Methods
Public
Methods
Outside
Code

...ENCAPSULAMIENTO

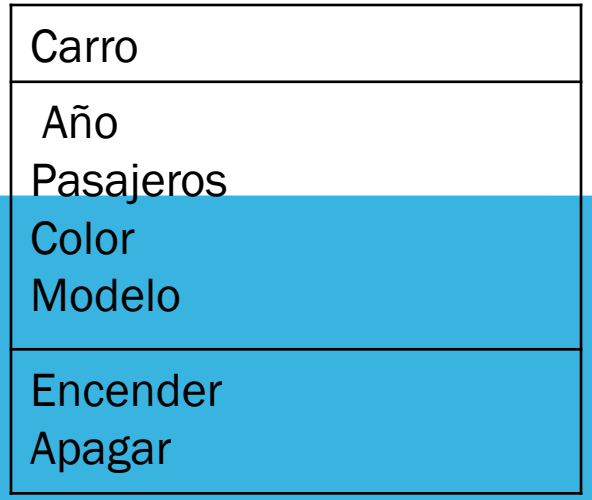
La Clase es, por otro lado, una encapsulación porque constituye una cápsula o saco que encierra y amalgama de forma clara tanto los datos de que constan los objetos como los procedimientos que permiten manipularlos. Las Clases se constituyen, así, en abstracciones encapsuladas

CLASES

Una clase es la descripción de un conjunto de objetos; consta de métodos y datos que resumen características comunes de un conjunto de objetos. Una clase puede definir un numero virtualmente infinito de objetos, y cada objeto es denominado una *instancia de la clase.*

Por consiguiente, los objetos no son mas que instancias de una clase. Una instancia es una variable de tipo objeto.

Ejemplo:



CLASES

Clases

Una descripción abstracta de un conjunto de objetos, cada uno de los cuales se diferencia por su estado específico y por la posibilidad del objeto de realizar una serie de operaciones.

Una clase en términos coloquiales, es el plano de un objeto, es donde se deben definir todas las características de los objetos de la vida real o de objetos abstractos

CLASES

Las clases son una descripción abstracta, ideal de un grupo de objetos, cada uno de los cuales se diferencia por un estado especifico y es capaz de realizar una serie de operaciones.

1.- Carlos Mejía Godoy, Katia Cardenal, José Manuel Espinoza pertenecen a la clase “cantantes de Nicaragüenses”

2.- Orlando Alemán, Joseline González , Geordano Moreno pertenecen a la clase de “Estudiantes del 2M4-Co

3.- Carlos Andrés Pérez, Violeta Barrios, Bill Clinton, pertenecen a la clase “Ex presidentes de América”

CLASES

Clase	Posibles atributos	Posible Comportamiento
Bicicleta	Modelo Velocidades color	Frenar Cambiar velocidades
Naranja	Dulzura Color tipo	Pelar Pudrirse Exprimir
Edificio	Temperatura Número de pisos Número de habitaciones	Abrir Cerrar Calefacción
Estudiante	Nombre Dirección Numero de Carne Año de estudio	Cambiar clases Matricular Graduar Inscribir materias

Clase:
Trabajadores
UNI

...MODULARIDAD

Es la propiedad que permite subdividir una aplicación en partes mas pequeñas (llamadas módulos), cada una las cuales debe ser tan independiente como sea posible de la aplicación en si y de las restantes partes.

Por su parte Bárbara Liskov establece que "modularización consiste en dividir un programa en módulos que pueden ser compilados de forma separada, pero que tienen conexiones con otros módulos"[2]

MÉTODOS

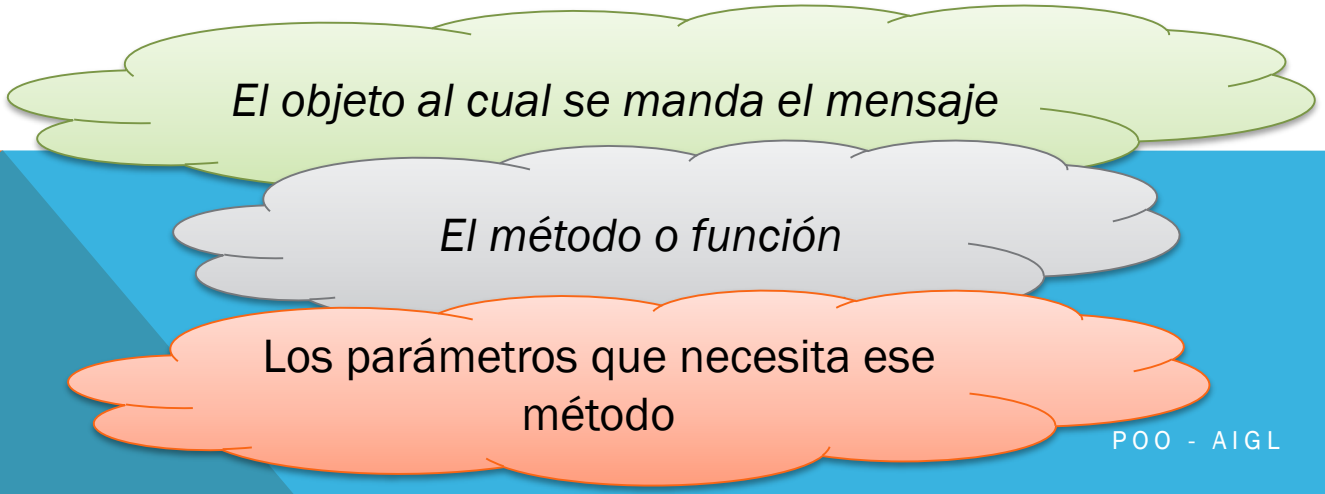
Algoritmo asociado a un objeto (o Clase de objetos), cuya ejecución se desencadena tras la recepción de un mensaje.

Desde el punto de vista del comportamiento, es lo que el objeto puede hacer.

Un método puede producir un cambio en las propiedades del objetos.

MENSAJES

- *Los objetos de un programa interactúan y se comunican entre ellos por medio de mensajes*
- A veces el objeto que recibe el mensaje necesita más información Esta información se pasa junto con el mensaje en forma de parámetro.
- *Partes del mensaje*

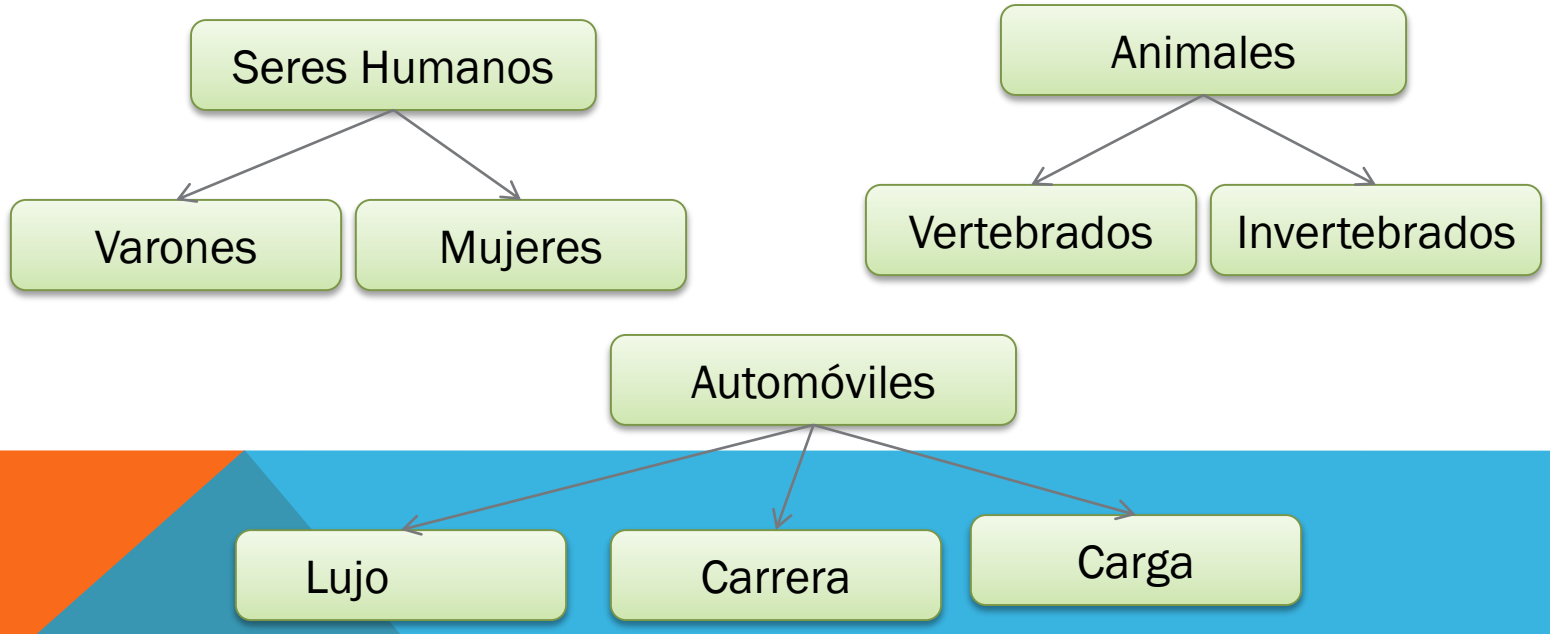


HERENCIA

- La herencia es la propiedad que permite a los objetos ser construidos a partir de otros objetos., es decir la capacidad de un objeto para utilizar las estructuras de datos y los métodos previstos en antepasados o ascendientes. El objetivo final es la **reutilización o reutilizabilidad** es decir reutilizar código anteriormente ya desarrollado.

...HERENCIA

La idea de herencia parte de la experiencia de la vida real las clases básicas o fundamentales se dividen en subclases



...HERENCIA

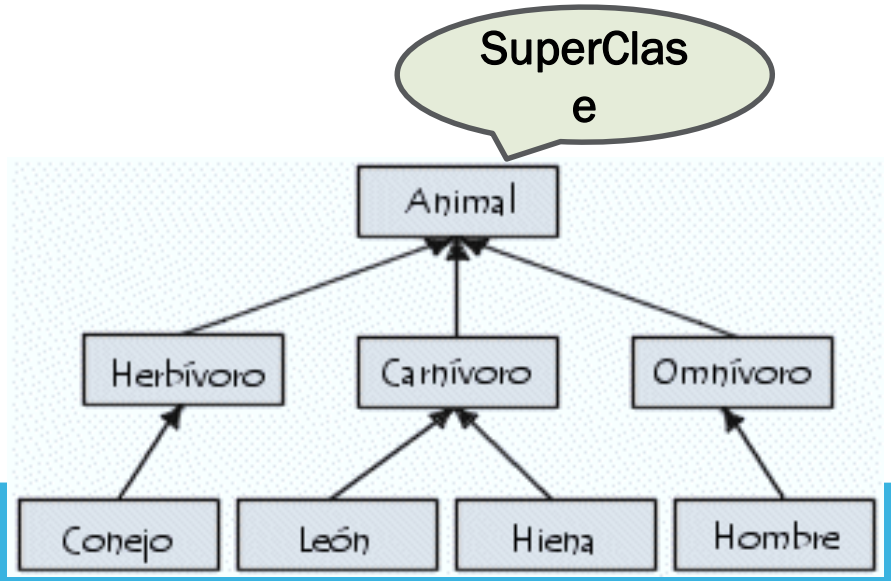
los objetos con propiedades comunes se agrupan en una clase

las clases con propiedades comunes se agrupan en una superclase

Las clases que se derivan de una superclase son las **subclases.**

Herencia Simple O Jerárquica

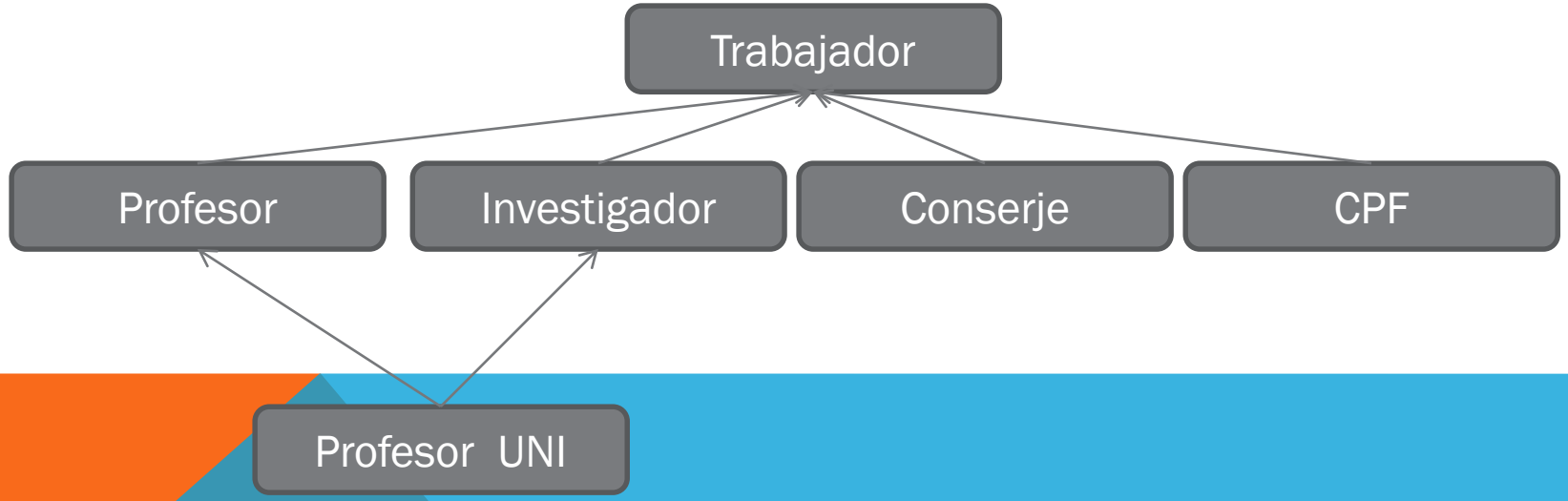
Un objeto (clase) puede tener solo un ascendiente, o dicho de otro modo, una subclase puede heredar datos y métodos de una única clase, así como añadir o quitar comportamientos de la clase base.



...HERENCIA

Herencia Múltiple o en Malla

Es la propiedad de una clase de poder tener más de un ascendiente inmediato, o lo que es igual, adquirir datos y métodos de más de una clase.



RELACIONES ENTRE LAS CLASES

Las relaciones entre clases son tres:

1. Relación de asociación
2. Relación de agregación
3. Relación de generalización.

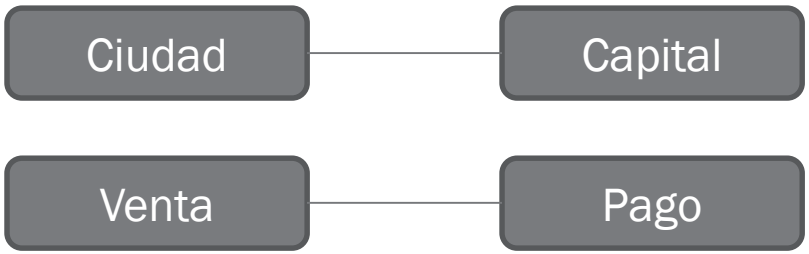
Relaciones de Asociación

Es preciso buscar frase tales como: pertenece a, es miembro de, esta asociado con, trabaja para, etc.

Una propiedad importante intrínseca a la relación de asociación o multiplicidad es la cardinalidad o multiplicidad.

- Una a una
- Una a muchas
- Muchas a muchas

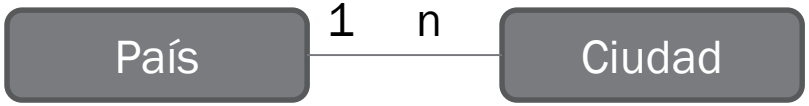
Relaciones de Asociación

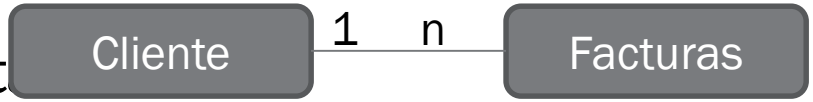


Notacion

_____ Una a Una

Relaciones de Asociación

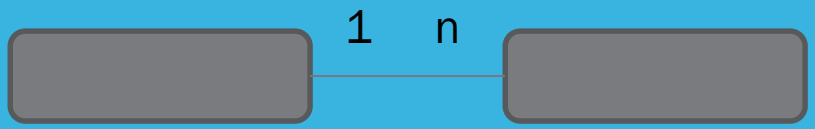




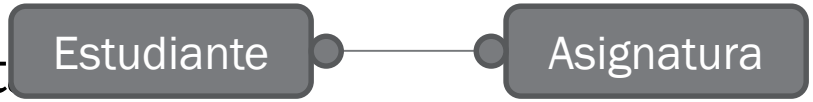
Not

_____ Una a muchas

1..n, 1-m Una a muchas



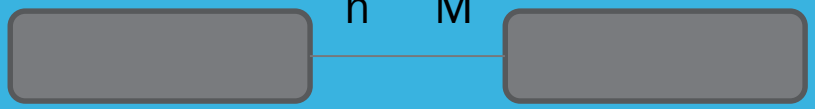
Relaciones de Asociación



Not

_____ Muchas a muchas

m..n, m-n Muchas a muchas



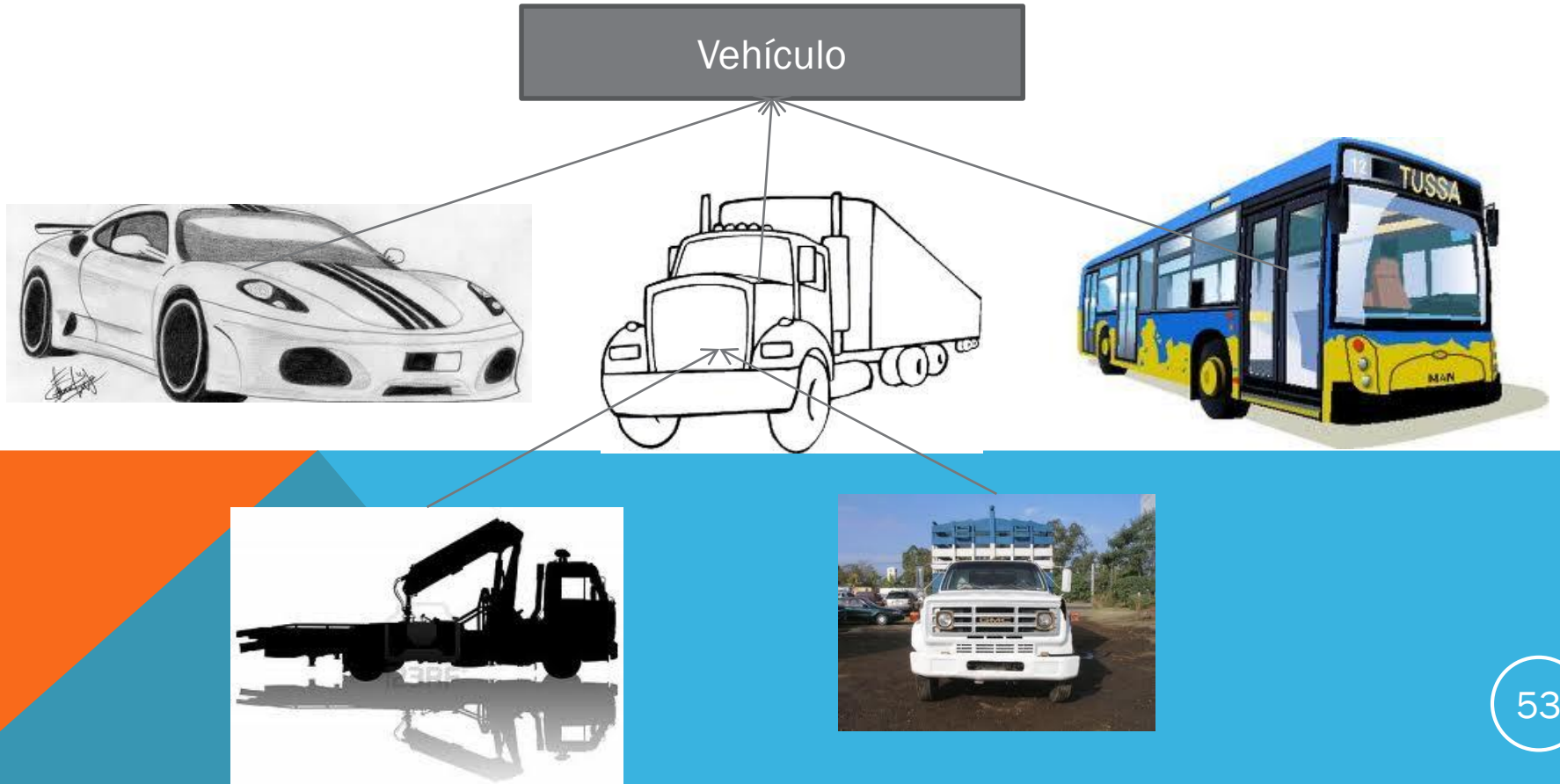
Relaciones de Agregación o Composición

Es un concepto que se utiliza para expresar tipos de relaciones entre objetos: parte-de o tiene-un.

M

Relación de Generalización

Representa una relación “ un tipo de “. Por ejemplo, una rosa es un tipo de flor, significando que una rosa es una subclase especializada de la clase general, flor.



Polimorfismo

Permite referirse a objetos de clases diferentes mediante el mismo elemento de programa y realizar la misma operación de diferentes formas, según sea el objeto que se referencia en ese momento.

Permite que una misma función se comporte de diferente formas según sea la clase sobre la que se aplica.

un mismo identificador puede tener distintas formas (distintos cuerpos de función, distintos comportamientos) dependiendo del contexto en el que se halle.

Polimorfismo y Sobrecarga

La sobrecarga de métodos hace que un mismo nombre pueda representar distintos métodos con distinto tipo y número de parámetros, manejados dentro de la misma clase. En el ámbito de la POO, la sobrecarga de métodos se refiere a la posibilidad de tener dos o más métodos con el mismo nombre pero distinta funcionalidad. Es decir, dos o más métodos con el mismo nombre realizan acciones diferentes y el compilador usará una u otra dependiendo de los parámetros usados. Esto también se aplica a los constructores (de hecho, es la aplicación más habitual de la sobrecarga).

Polimorfismo

```
Public Class Class1

    Private NumLlantas As Integer

    Sub New()
        NumLlantas = 4
    End Sub

    Sub New(ByVal NLlantas As Integer)    'Overloaded Constructor
        NumLlantas = NLlantas + 23
    End Sub

    Sub suma(ByVal N As Integer, ByVal r As String)
        MsgBox("loco")
    End Sub
    Sub suma()
        MsgBox("loco triple")
    End Sub
    Public Property NLlantas()
        Get
            Return NumLlantas
        End Get
        Set(ByVal Value)
            NumLlantas = Value
        End Set
    End Property
End Class
```

Polimorfismo

uso del mismo identificador u operador en distintos contextos y con distintos significados.

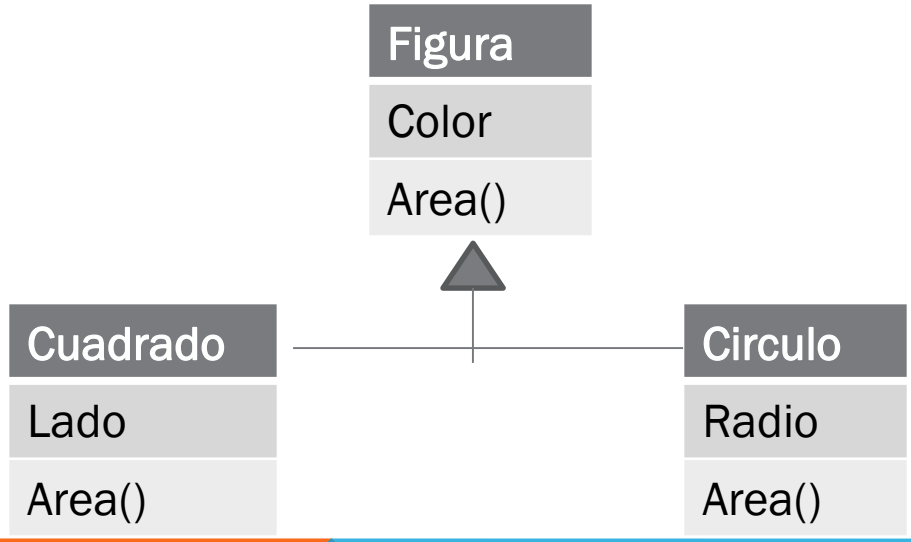
```
public class Articulo {

    private float precio;

    public void setPrecio() {
        precio = 3.50;
    }
    public void setPrecio(float nuevoPrecio) {
        precio = nuevoPrecio;
    }
    public void setPrecio(float costo, int porcentajeGanancia) {
        precio = costo + (costo * porcentajeGanancia);
    }
}
```

Polimorfismo

o a un "triangulo, y el objeto ejecutara el código apropiado dependiendo del tipo específico



Polimorfismo del método Area

CLASE Figura
- int x,y;
- Mover ();
- Área ();

CLASE Círculo Hereda de Figura
- int radio;
- Área (); donde Área = pi * radio * radio

CLASE Cuadrado Hereda de Figura
- int lado;
- Área (); donde Área = lado * lado

Abstracción:	Visión simplificada de la realidad
Objeto:	Modelado del software del mundo real
Clase:	Estructura de software que definen los objetos
Herencia de la Clase:	Heredan los atributos y/o métodos de sus padres (o Jerarquización de clases)
Polimorfismo:	Elemento de programa que realiza operación de distintas formas.