Programación

Orientada a Objetos

Prof. Angela Di Serio

Universidad Simón Bolívar
Especialización en Telemática

Agenda Clase 2

Qué es Orientado a Objetos?

Conceptos: objeto, clase, instancias, mensajes

Propiedades de OO

UML

Diagrama de Clases

Programación **convencional**

procedimientos y datos

Estructuras de datos como variables o parámetros que se tratan separadamente de los procedimientos

Programación **Orientada a Objetos**

Idea: Mundo lleno de **objetos**

Resolución de problema en términos de objetos

Los objetos **no** son entes **aislados**

Forman parte de una **organización jerárquica**

Elemento básico

objeto

contiene **datos** que describen su estado y las **operaciones** que operan sobre esos datos

Los datos y funciones se **encapsulan** en una única entidad

Estructura de un **objeto**

Relaciones

Propiedades

Métodos

Estructura de un objeto

Relaciones

permiten que el objeto se inserte en la organización

formado esencialmente por apuntadores a otros objetos

Estructura de un **objeto**

Relaciones

Propiedades

Métodos

Estructura de un objeto

Propiedades

distingue un objeto del resto

propiedades pueden ser heredadas a sus descendientes

Estructura de un **objeto**

Relaciones

Propiedades

Métodos

Estructura de un objeto

Métodos

operaciones que pueden realizarse sobre el objeto

Clase tipo definido por el usuario que determina las estructuras de datos y las operaciones asociadas con ese tipo

un **objeto** de una **clase** corresponde a una ***instancia*** de esa clase.

La **comunicación** con el objeto se realiza a través del **paso de mensajes**

Vainilla

Perro

clase

Perro es una generalización de Vainilla

Vainilla

Objeto o instancia

Componentes de un mensaje

identidad del objeto receptor

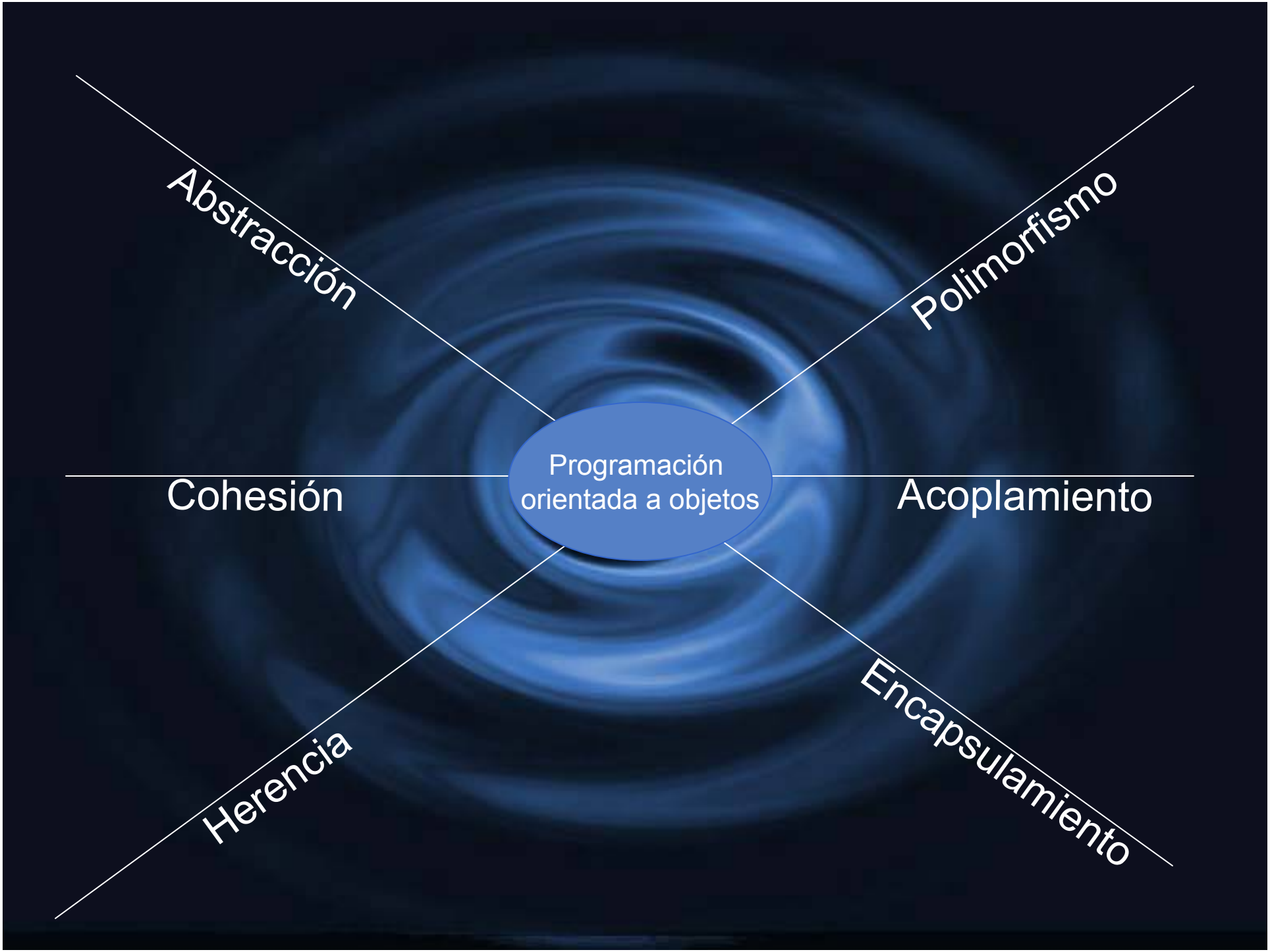
función miembro cuya ejecución se está solicitando

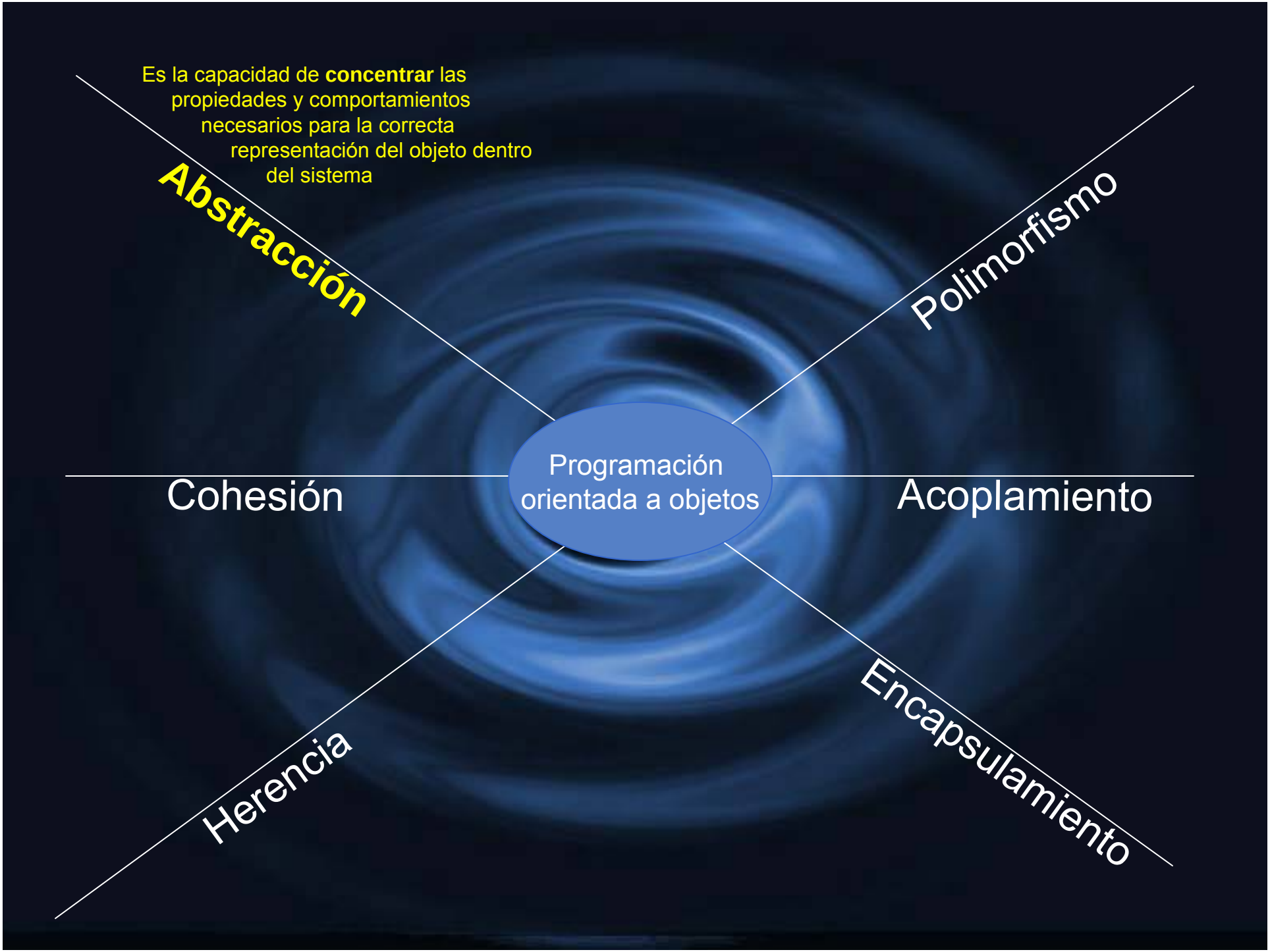
información adicional que necesite el método

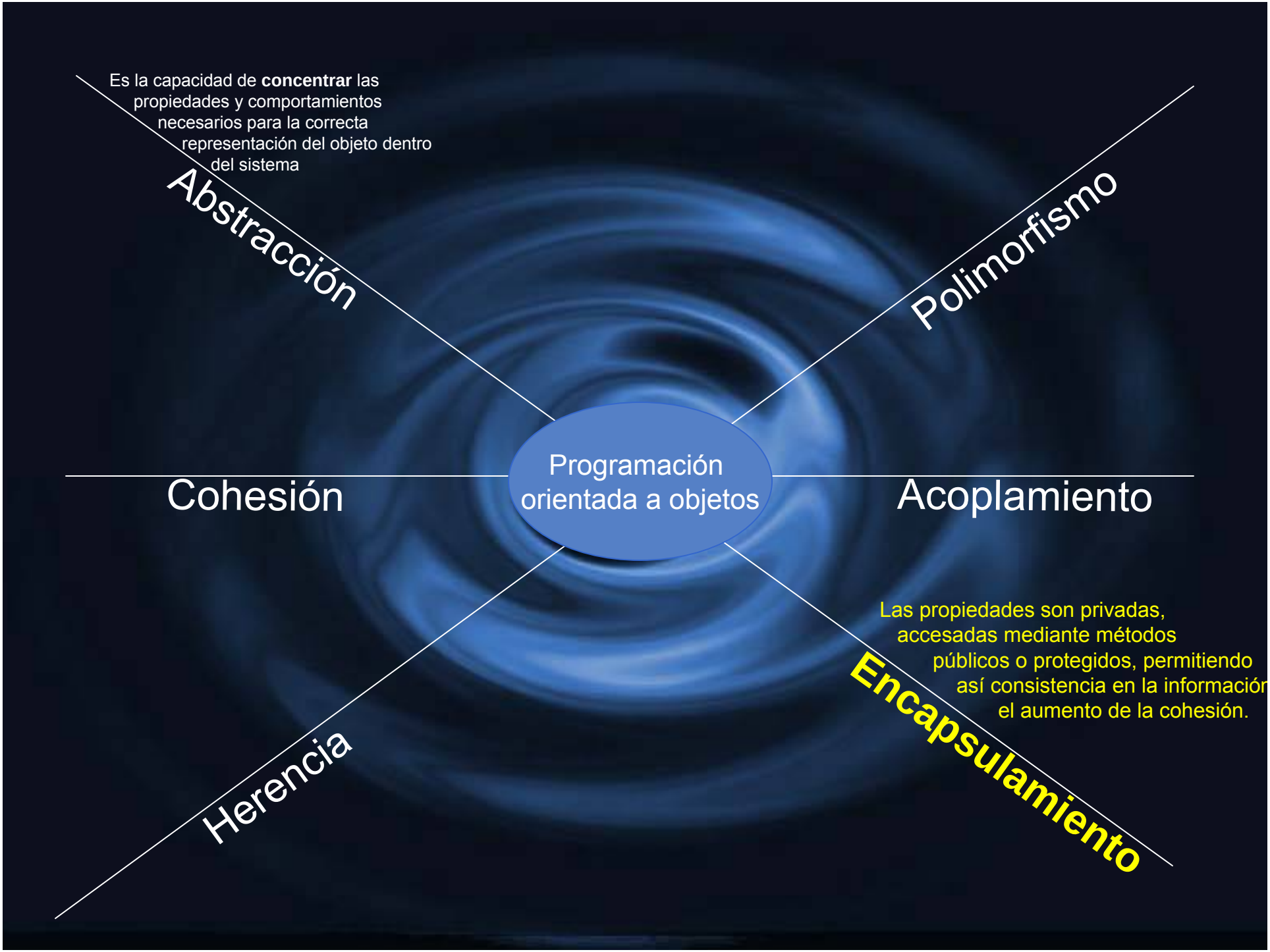
Características de la

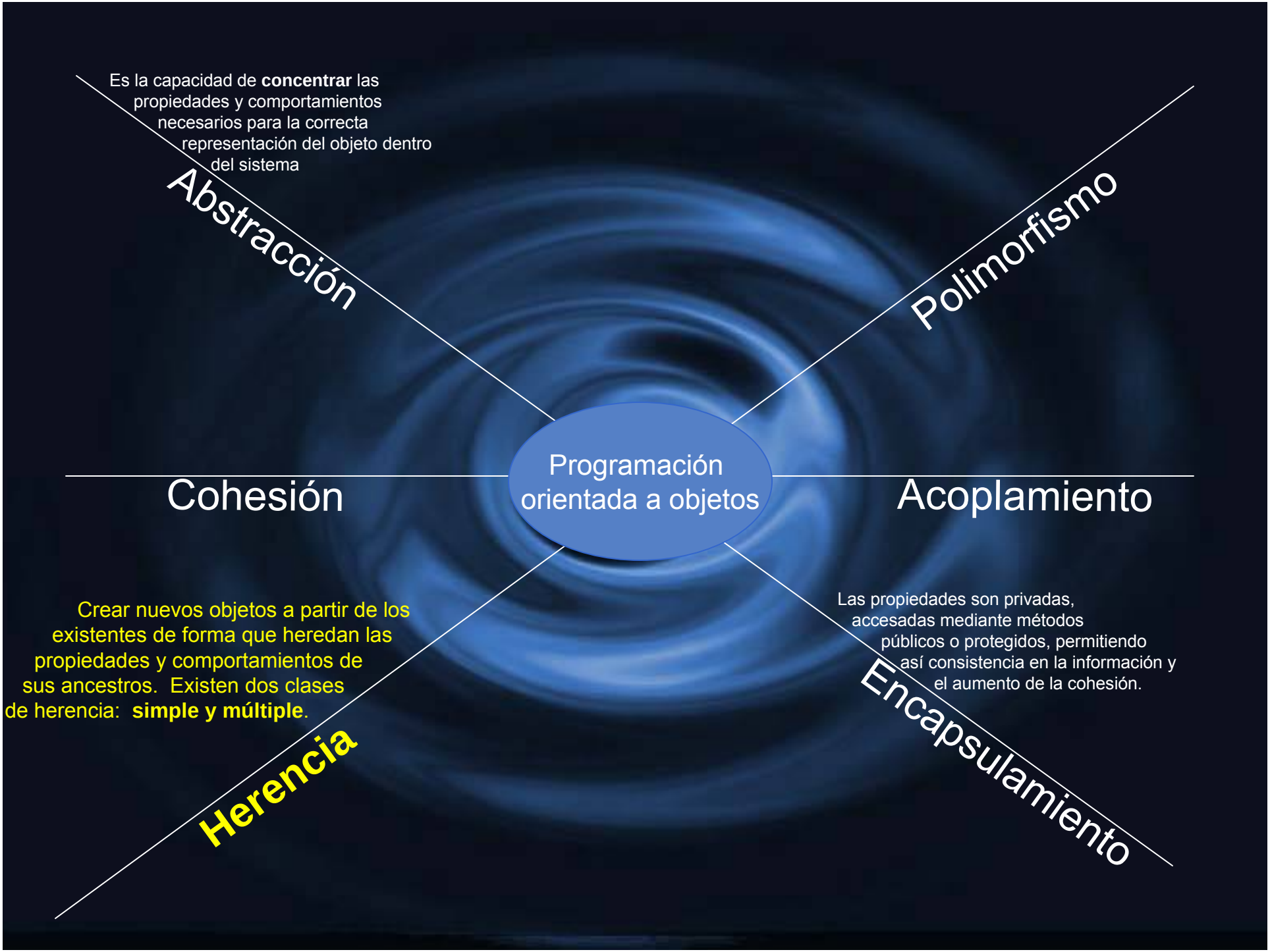
Programación

Orientada a Objetos

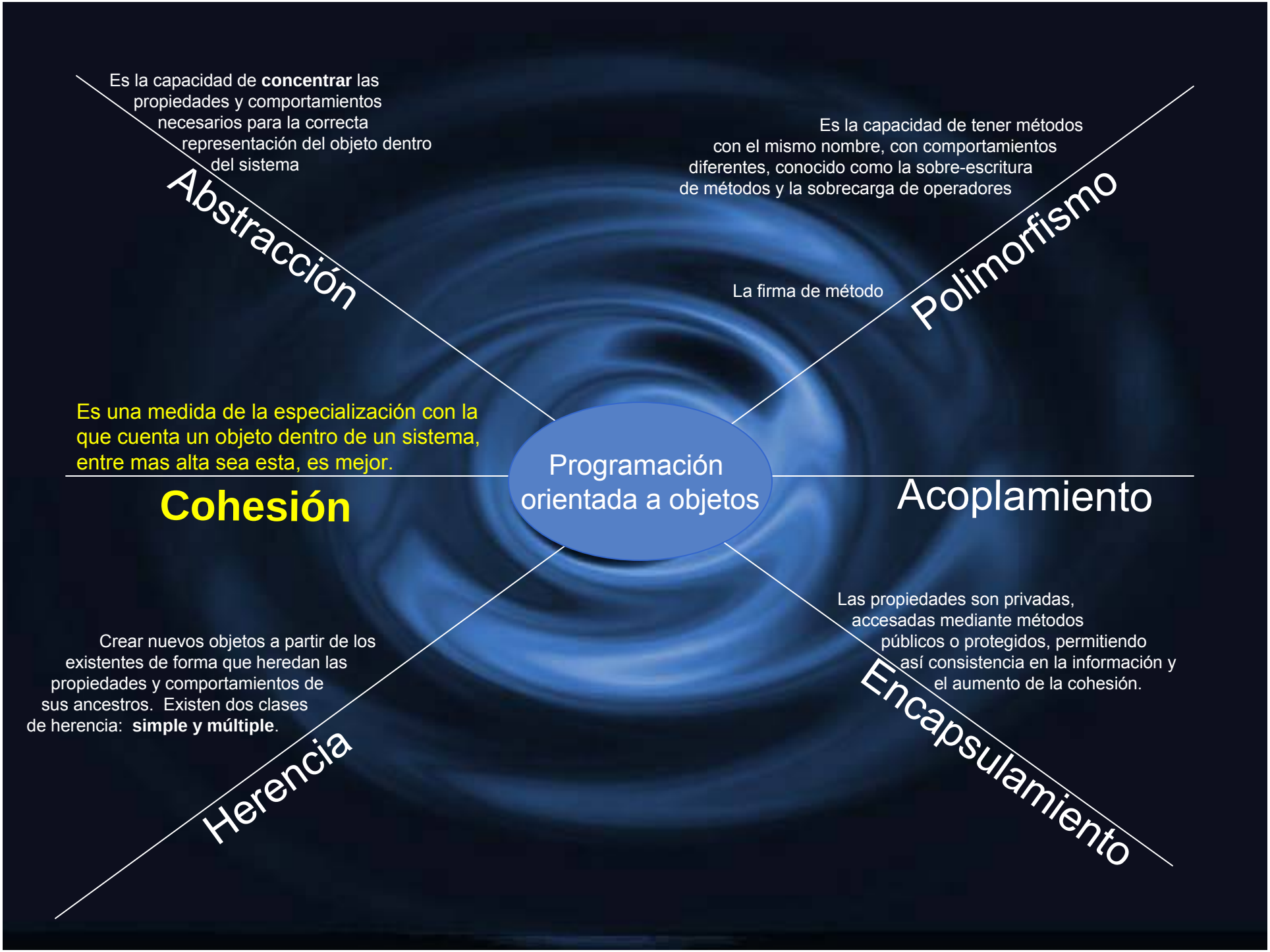

Abstracción
Polimorfismo
Programación orientada a objetos
Cohesión
Acoplamiento
Herencia
Encapsulamiento


Es la capacidad de **concentrar** las propiedades y comportamientos necesarios para la correcta representación del objeto dentro del sistema
Abstracción
Polimorfismo
Programación orientada a objetos
Cohesión
Acoplamiento
Herencia
Encapsulamiento

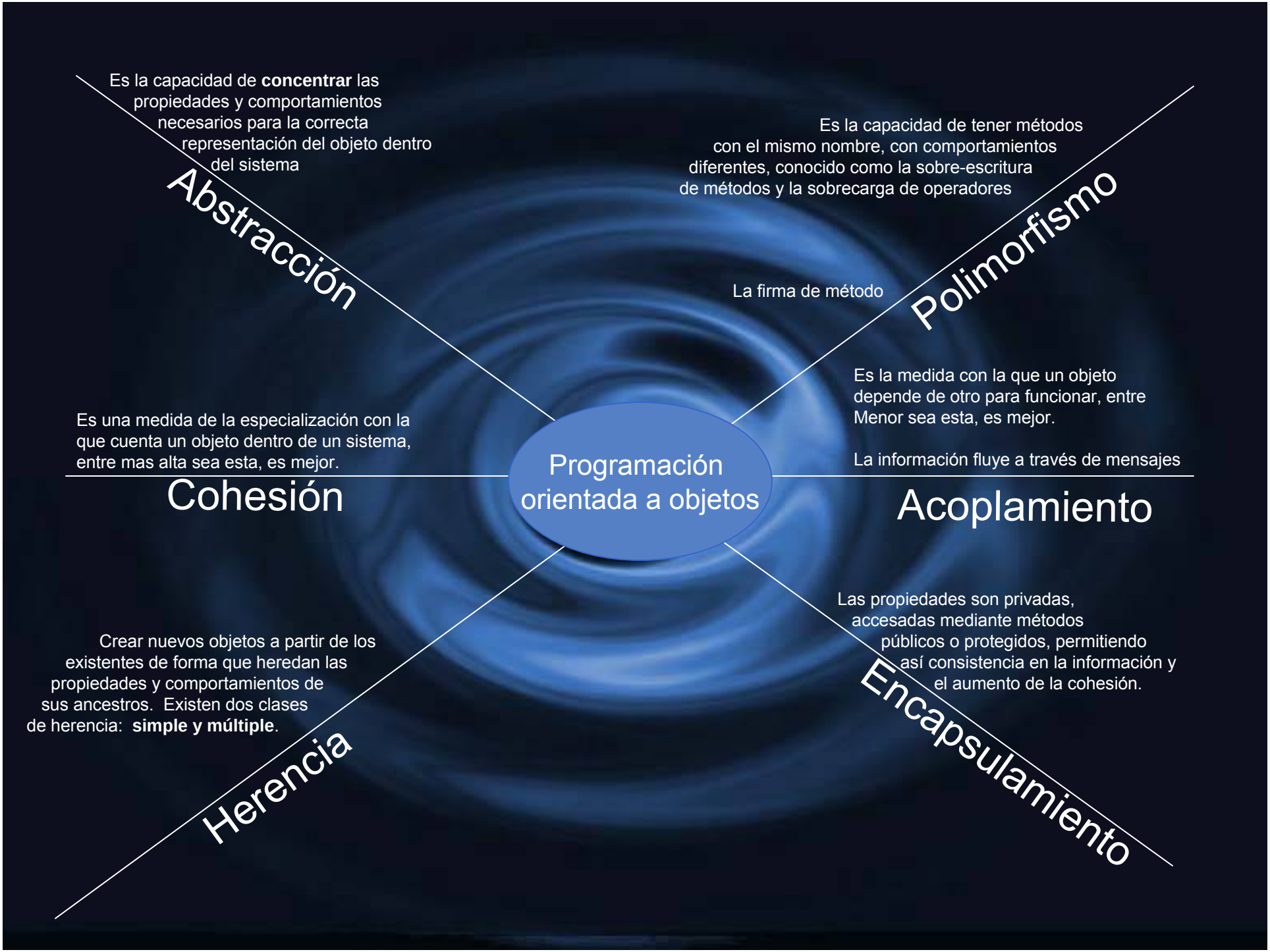

Es la capacidad de **concentrar** las propiedades y comportamientos necesarios para la correcta representación del objeto dentro del sistema
Abstracción
Polimorfismo
Programación orientada a objetos
Cohesión
Acoplamiento
Las propiedades son privadas, accesadas mediante métodos públicos o protegidos, permitiendo así consistencia en la información el aumento de la cohesión.
Encapsulamiento
Herencia


Es la capacidad de **concentrar** las propiedades y comportamientos necesarios para la correcta representación del objeto dentro del sistema
Abstracción
Polimorfismo
Programación orientada a objetos
Cohesión
Acoplamiento
Crear nuevos objetos a partir de los existentes de forma que heredan las propiedades y comportamientos de sus ancestros. Existen dos clases de herencia: **simple y múltiple**.
Herencia
Las propiedades son privadas, accesadas mediante métodos públicos o protegidos, permitiendo así consistencia en la información y el aumento de la cohesión.
Encapsulamiento


Es la capacidad de **concentrar** las propiedades y comportamientos necesarios para la correcta representación del objeto dentro del sistema
Abstracción
Es la capacidad de tener métodos con el mismo nombre, con comportamientos diferentes, conocido como la sobre-escritura de métodos y la sobrecarga de operadores
La firma de método
Polimorfismo
Programación orientada a objetos
Cohesión
Acoplamiento
Crear nuevos objetos a partir de los existentes de forma que heredan las propiedades y comportamientos de sus ancestros. Existen dos clases de herencia: **simple y múltiple**.
Herencia
Las propiedades son privadas, accesadas mediante métodos públicos o protegidos, permitiendo así consistencia en la información y el aumento de la cohesión.
Encapsulamiento


Es la capacidad de **concentrar** las propiedades y comportamientos necesarios para la correcta representación del objeto dentro del sistema
Abstracción
Es la capacidad de tener métodos con el mismo nombre, con comportamientos diferentes, conocido como la sobre-escritura de métodos y la sobrecarga de operadores
La firma de método
Polimorfismo
Es una medida de la especialización con la que cuenta un objeto dentro de un sistema, entre mas alta sea esta, es mejor.
Cohesión
Programación orientada a objetos
Acoplamiento
Crear nuevos objetos a partir de los existentes de forma que heredan las propiedades y comportamientos de sus ancestros. Existen dos clases de herencia: **simple y múltiple**.
Herencia
Las propiedades son privadas, accesadas mediante métodos públicos o protegidos, permitiendo así consistencia en la información y el aumento de la cohesión.
Encapsulamiento

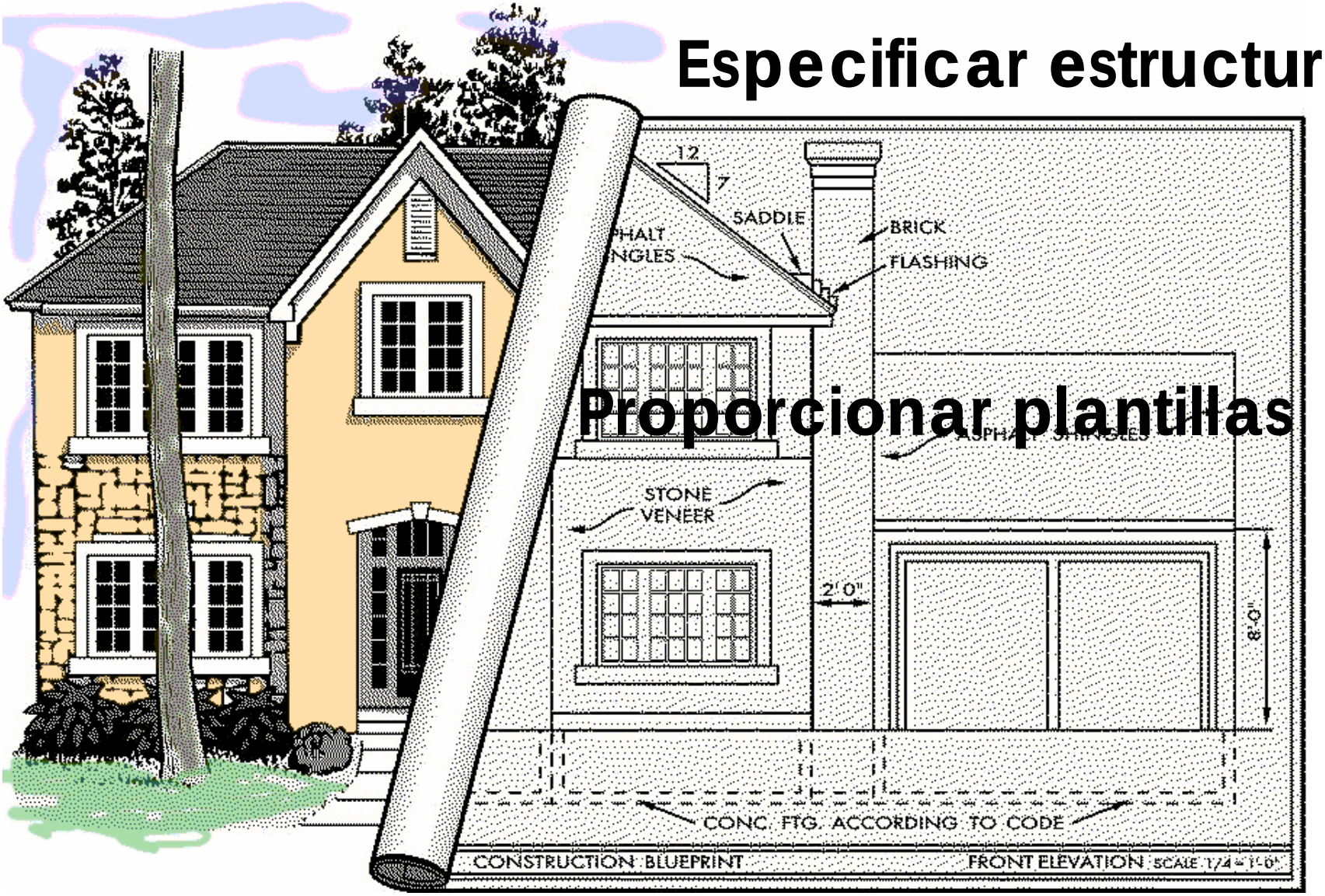

Programación orientada a objetos
Abstracción
Es la capacidad de **concentrar** las propiedades y comportamientos necesarios para la correcta representación del objeto dentro del sistema
Polimorfismo
Es la capacidad de tener métodos con el mismo nombre, con comportamientos diferentes, conocido como la sobre-escritura de métodos y la sobrecarga de operadores
La firma de método
Cohesión
Es una medida de la especialización con la que cuenta un objeto dentro de un sistema, entre mas alta sea esta, es mejor.
Acoplamiento
Es la medida con la que un objeto depende de otro para funcionar, entre Menor sea esta, es mejor.
La información fluye a través de mensajes
Herencia
Crear nuevos objetos a partir de los existentes de forma que heredan las propiedades y comportamientos de sus ancestros. Existen dos clases de herencia: **simple y múltiple**.
Encapsulamiento
Las propiedades son privadas, accesadas mediante métodos públicos o protegidos, permitiendo así consistencia en la información y el aumento de la cohesión.


Programación orientada a objetos
Abstracción
Es la capacidad de **concentrar** las propiedades y comportamientos necesarios para la correcta representación del objeto dentro del sistema
Polimorfismo
Es la capacidad de tener métodos con el mismo nombre, con comportamientos diferentes, conocido como la sobre-escritura de métodos y la sobrecarga de operadores
La firma de método
Cohesión
Es una medida de la especialización con la que cuenta un objeto dentro de un sistema, entre mas alta sea esta, es mejor.
Acoplamiento
Es la medida con la que un objeto depende de otro para funcionar, entre Menor sea esta, es mejor.
La información fluye a través de mensajes
Herencia
Crear nuevos objetos a partir de los existentes de forma que heredan las propiedades y comportamientos de sus ancestros. Existen dos clases de herencia: **simple y múltiple**.
Encapsulamiento
Las propiedades son privadas, accesadas mediante métodos públicos o protegidos, permitiendo así consistencia en la información y el aumento de la cohesión.

Unified Modeling Language

Visualizar

Especificar estructura



Proporcionar plantillas

Documentar decisiones

Lenguaje de **Modelado** Unificado

basado en notación **gráfica**

permite **especificar**, **construir**, **visualizar** y **documentar** los objetos de un sistema

Componentes

UML

- Vista Estática
- Vista de Casos de Uso
- Vista de Interacción
- Diagrama de Secuencia
- Diagrama de Colaboración
- Vista de la Máquina de Estados
- Vista de Actividades
- Vista Física
- Vista de la Gestión del Modelo
- Constructores de Extensibilidad

Vista	Diagramas	Conceptos Principales
Vista Estática	Diagrama de Clases	Clase, Asociación, Generalización Dependencia, Realización, Interfase
Vista de Casos de Uso	Diagrama de Casos de Uso	Caso de uso, Actor, Asociación, Extensión, Inclusión, Generalización de caso de uso
Vista de Implementación	Diagrama de Componentes	Componente, Interfaz, Dependencia, Realización
Vista del despliegue (deployment)	Diagrama de Despliegue	Nodo, Componente, Dependencia, Locación

Diagrama de Clases

Modela los conceptos del dominio de la aplicación

Permite **visualizar** las **relaciones** entre **las clases** que involucran el sistema

Diagrama de **clases**

Clases: atributos, operaciones
y visibilidad

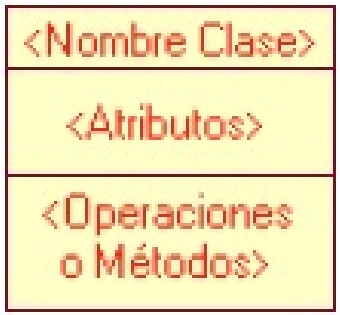
Relaciones: herencia, composición
agregación, asociación y uso

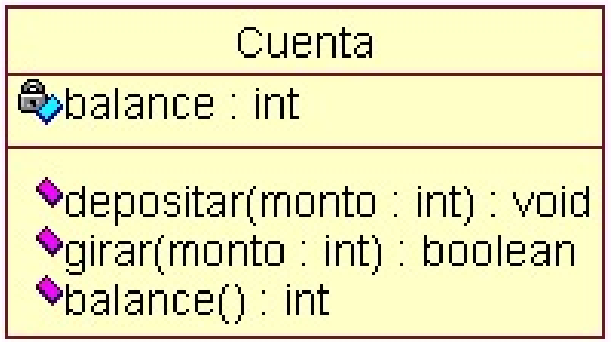
Responsabilidades

Diagrama de **clases**

Elemento **Clases**

Encapsula toda la información de un objeto





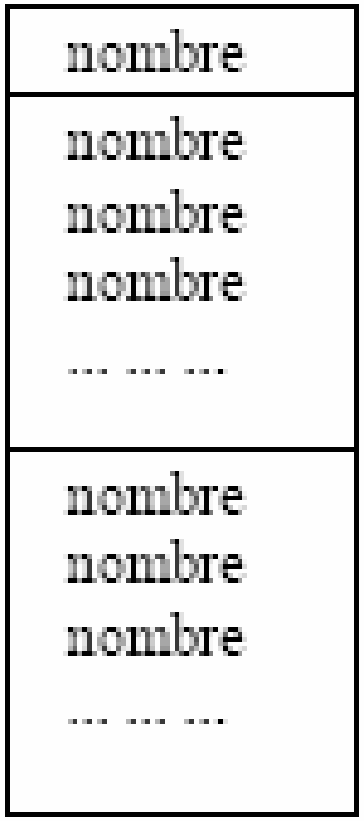
Representación gráfica

una clase

nombre de la clase

atributos
características de la clase
lo que se es o cómo se está

métodos
lo que puede hacerse
con objetos de esta clase:
comportamiento posible



un objeto

nombre del objeto

valores
memoria o estado
particular de cada objeto

todos los objetos de
una clase hacen
exactamente
lo mismo

Diagrama de **clases**

Elementos **Atributo**

Describen la clase

Públicos (+), **privados (-)** o **protegidos (#)**

Diagrama de **clases**

Elementos **Método**

Describen la forma en la cual interactúa la clase con su entorno

Públicos (+), **privados (-)** o **protegidos (#)**

Diagrama de **clases**

Elementos **Relaciones entre clases**

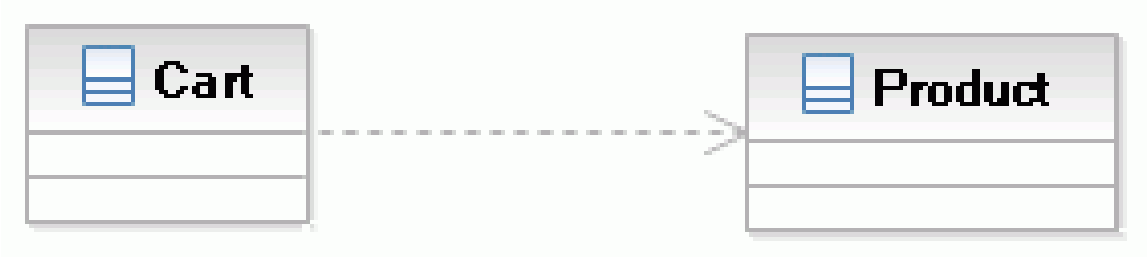
Dependencia

Generalización

Asociación

Elementos Relaciones entre clases

Dependencia

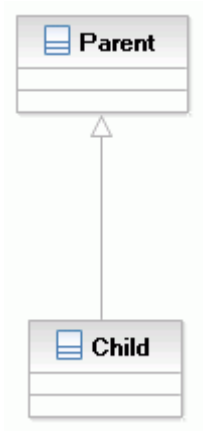


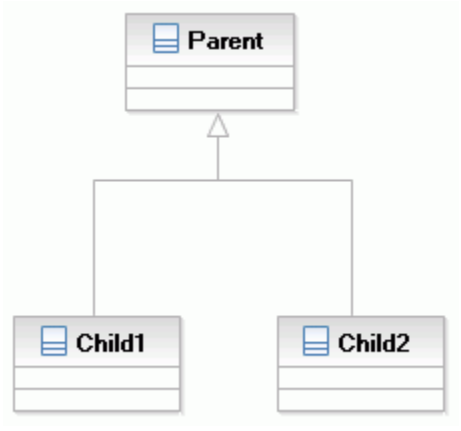
La clase **Cart** usa la clase **Product** como argumento en la operación de agregar al Cart

Un cambio en Product puede requerir un cambio en Cart

Elementos Relaciones entre clases

Generalización

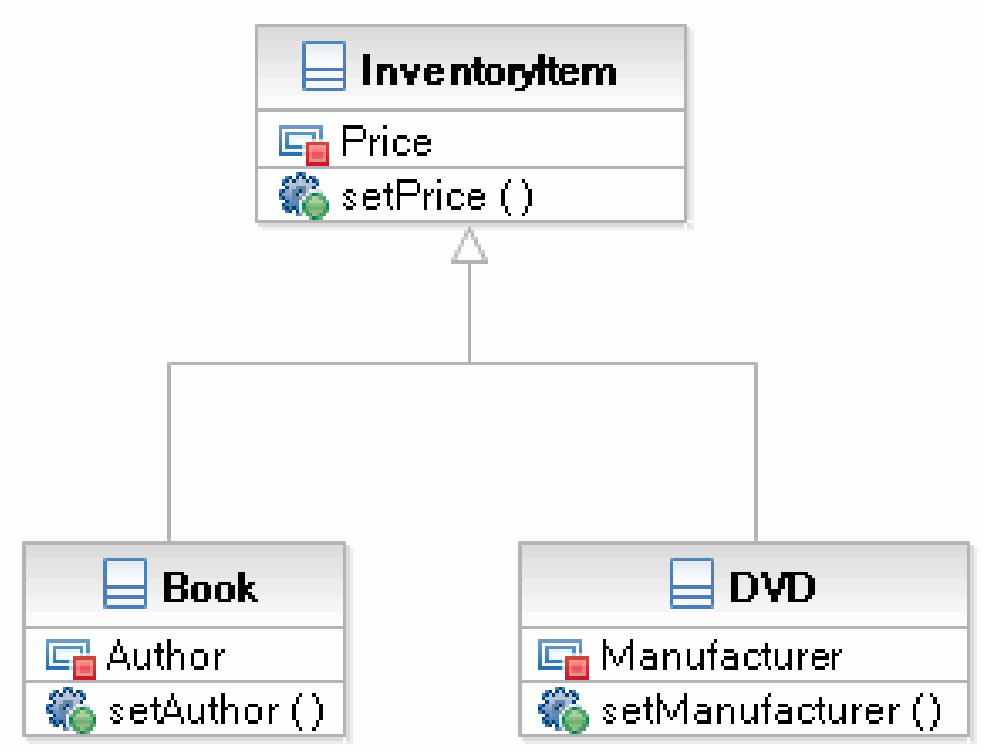




La clase Child (subclase) está basada en la clase Padre (superclase)

Elementos Relaciones entre clases

Generalización



Elementos Relaciones entre clases

Generalización - Polimorfismo

Da lugar al **polimorfismo** entre clases en una jerarquía

Un objeto de la subclase **puede sustituir** a un objeto de la superclase

Una operación de la subclase con **igual firma** sustituye a la operación en la superclase

Elementos Relaciones entre clases

Asociación

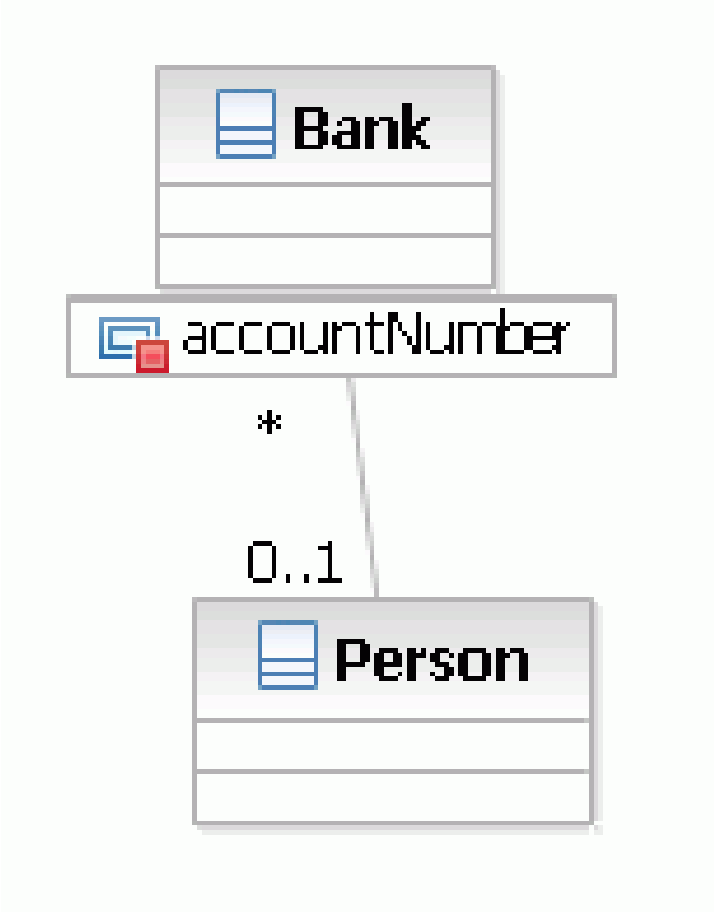
Relación estructural entre clases

Tiene nombre que la describe (verbo)

Tiene multiplicidad que especifica el número de objetos de la clase opuesta que se relacionan con un solo objeto de dicha clase

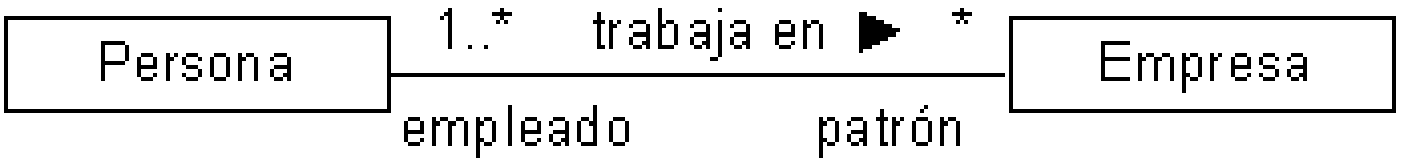
Elementos Relaciones entre clases

Asociación



Elementos Relaciones entre clases

Asociación

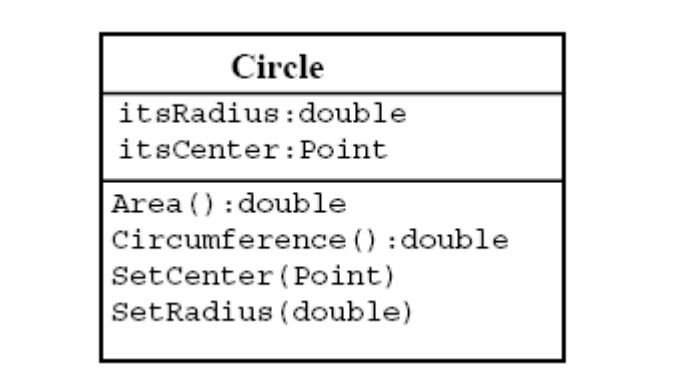


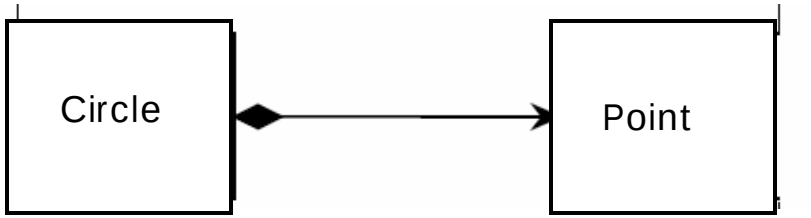
Elementos Relaciones entre clases

Composición

Relación estática en donde el tiempo de vida del objeto incluido está condicionado por el del que lo incluye

El objeto base se construye a partir del objeto incluido, es parte/todo como un parámetro pasado por valor


Circle
itsRadius:double
itsCenter:Point
Area():double
Circumference():double
SetCenter(Point)
SetRadius(double)


Circle
Point

Elementos Relaciones entre clases

Agregación

Relación dinámica, donde el tiempo de vida del objeto incluido es independiente del que lo incluye.

El objeto base utiliza al incluido para su funcionamiento

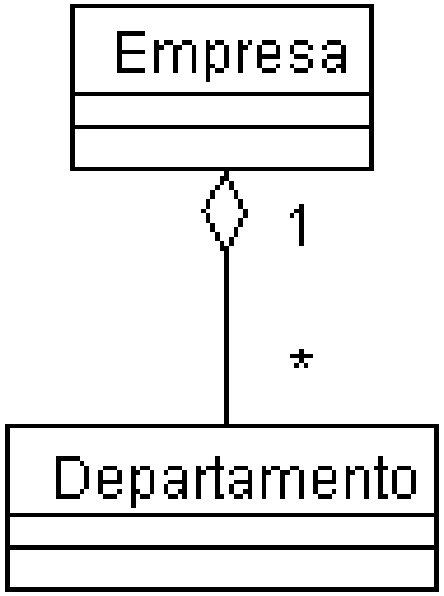


Diagrama de **clases**

Elementos **Responsabilidades**

La distribución de responsabilidades en un sistema, se realiza identificando un conjunto de clases que colaboran entre sí para llevar a cabo algún comportamiento. Luego hay que identificar el conjunto de responsabilidades para cada clase

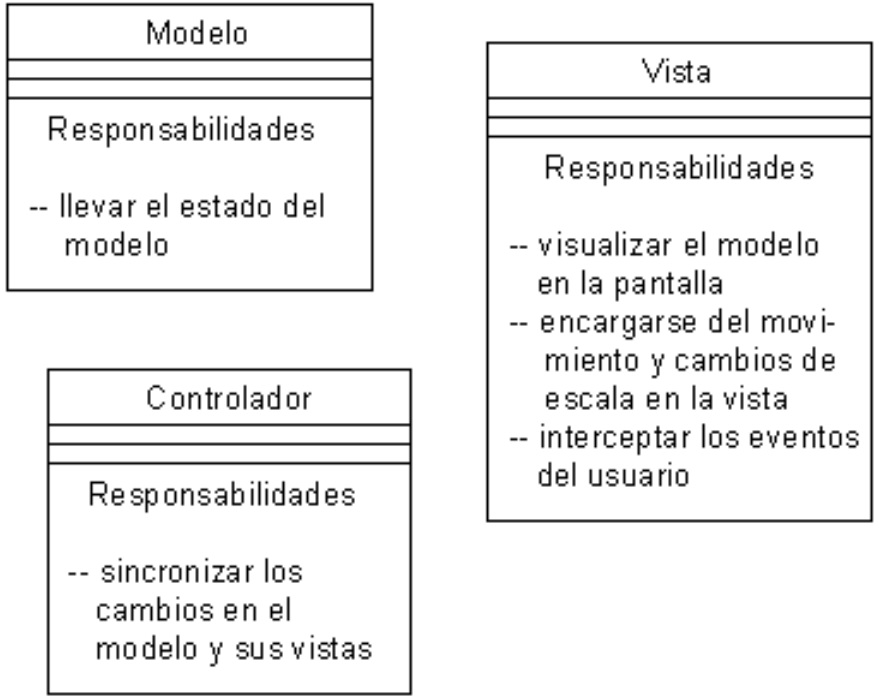


Diagrama de Clases