Curso C

Introducción Orientación Objetos

Índice

- Introducción.
- Conceptos básicos.

Situación de partida

- Requisitos de los programas cada vez más complejos
  - Se requiere un aumento en el nivel de abstracción al que se abordan
- Complejidad de desarrollo de sistemas dirigidos por eventos
  - La estructura impuesta por los lenguajes de tercera generación no se adapta bien a situaciones en las que no es la estructura del programa sino las condiciones del entorno las que determinan qué rutinas ejecutar

Solución

- Acercar la programación al modo en que se resuelven problemas en el mundo real
  - Objetos independientes que colaboran y se comunican mediante mensajes
  - Software más modular: la implementación de cada parte es relativamente independiente de la implementación de otras partes del sistema (sólo depende de la interfaz)
  - Ya no es necesario saber de antemano todas las posibles secuencias de acción
- No todos los sistemas son apropiados para desarrollo OO
  - Aplicaciones altamente algorítmicas

¿Por qué la Orientación a Objetos?

- Proximidad de los conceptos de modelado respecto de las entidades del mundo real.
  - Mejora captura y validación de requisitos.
- Facilita construcción, mantenimiento y reutilización.
- Conceptos comunes durante el análisis, diseño e implementación
  - Facilita la transición entre distintas fases
  - Favorece el desarrollo iterativo del sistema.

Sin embargo, existen problemas ...

Problemas en OO

- "…La mayoría de los usuarios de OO **no** utilizan los conceptos de la OO de forma purista, como inicialmente se pretendía. Esta práctica ha sido promovida por muchas herramientas y lenguajes que intentan utilizar los conceptos en diversos grados".

Objetivos

- ↑ Productividad
- ↑ Calidad
  - Minimizar Errores.
  - Facilidad de uso.
  - Portabilidad.
  - Modificabilidad.
- Misma representación en fases del ciclo de vida.
- Reusabilidad
  - Crear nuevos sistemas utilizando los ya existentes
- Extensibilidad
  - Fácil ampliación sin necesidad de retocar módulos

Conceptos básicos de la OO

- Objeto y Clase
- Encapsulación
  - Ocultación Información/Implementación
- Paso de Mensajes
- Generalización/Especialización
- Polimorfismo

Objeto

- Concepto, abstracción o cosa con límites y significado nítidos para el problema en consideración. Los objetos sirven **dos propósitos**: mejorar la comprensión del mundo real y proporcionar una base práctica para la implementación (Rumbaugh 1991)

- Los objetos tienen estado, comportamiento e identidad (Booch 1994) (estado puede determinar comportamiento)

Clase

- Descripción de un conjunto de objetos que comparten los mismos atributos, operaciones, métodos, relaciones y semántica. Su propósito es declarar una colección de métodos, operaciones y atributos que describen completamente la estructura y comportamiento de los objetos (UML 2001)
- Plantilla a partir de la cual se **instancian** los distintos objetos. Cada objeto tendrá la misma estructura y comportamiento que la clase desde la cual se instanció.
- *Una clase es lo que se diseña y programa. Un objeto es lo que se crea EN TIEMPO DE EJECUCIÓN.*

- Zona visible (métodos)
- Zona no visible (atributos o variables instancia)
- Características:
  - Estado: propiedades del objeto. Valores actuales de sus atributos.
  - Comportamiento: expresa cómo se producen los cambios de estado.

Características OO

- Encapsulación
- Herencia
- Paso de Mensajes
- Polimorfismo

Encapsulación

- También llamado abstracción
  - Agrupar bajo una misma entidad los datos y las funciones (métodos) que trabajan con esos datos.
- Efecto
  - Independiza la implementación interna del interface del objeto

Herencia

- Sirve para construir clases a partir de clases ya existentes.
- Las nuevas clases incorporan estructura y comportamiento de la clase de la que heredan.
- Superclase/Subclase, Clase Padre/Clase Hija, Clase Base/Clase Derivada.
- La definición de una subclase siempre debe incluir al menos un detalle no derivado de ninguna de sus superclases

Paso de Mensajes

- El único mecanismo para modificar el estado actual de un objeto son sus METODOS o SERVICIOS.
- Los objetos se comunican entre sí mediante el paso de mensajes.
- Consiste en pedir a un objeto que ejecute un servicio.
  - ObjetoDestino.servicio(parámetros).

Paso de mensajes

- Expresa la acción que un objeto realiza sobre otro
- El Sistema se ve como una simulación del mundo real: objetos colaboran enviándose mensajes para realizar una tarea colectiva

Métodos

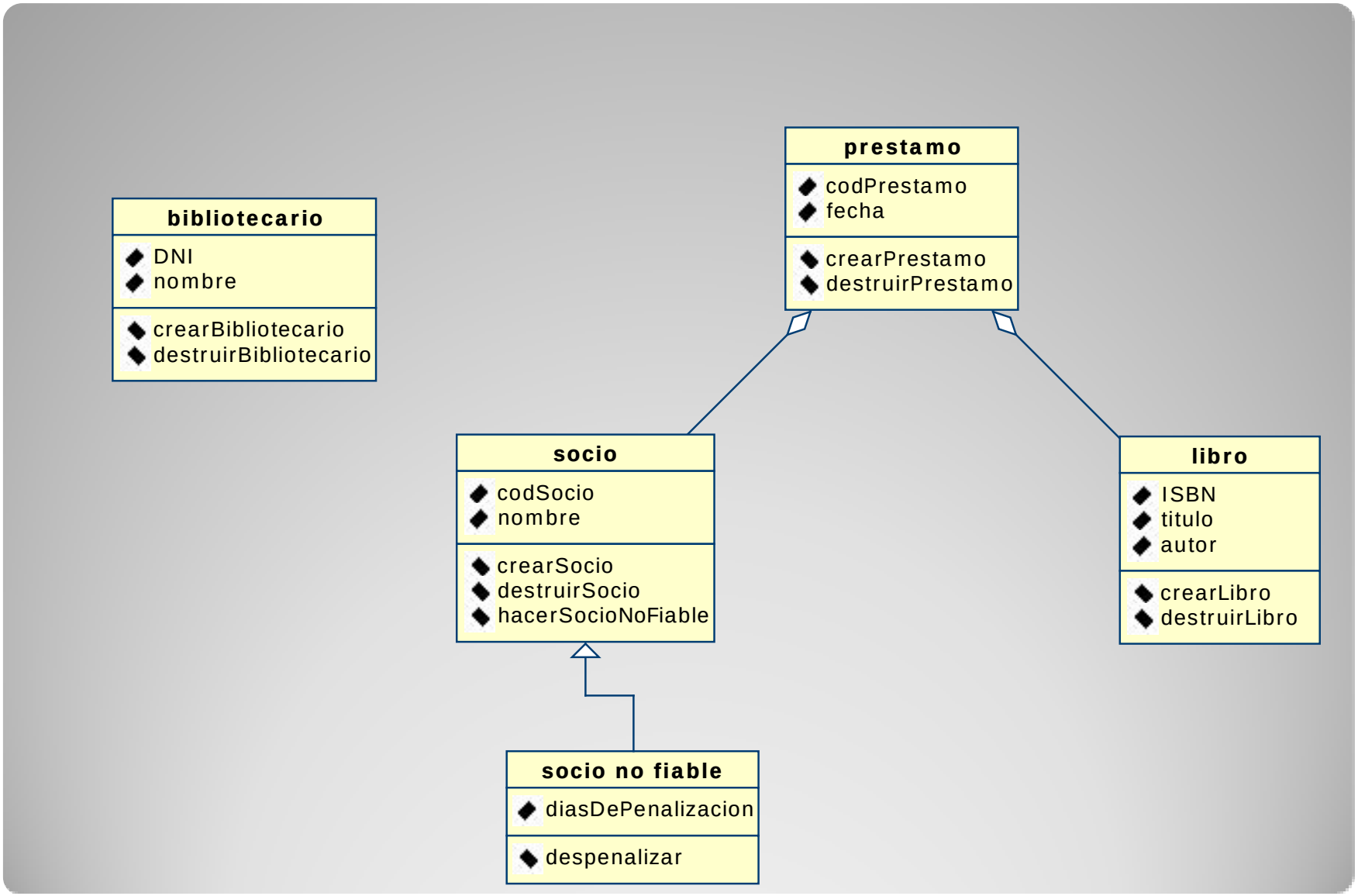
- Operaciones sobre los atributos de un objeto
- Un mensaje es el vehículo mediante el cual un objeto1 solicita a un objeto2 que ejecute uno de sus métodos.
- Tipos de operaciones:
  - Modificadoras.
  - Selectoras.
  - Iteradoras.
  - Constructoras y destructoras.

Polimorfismo

- Mecanismo que permite definir el mismo interface para un conj. de objetos con comportamiento totalmente distinto
- Posibilidad de que distintos tipos de objetos respondan de distinta manera al mismo mensaje
  - Es el objeto destino el encargado de saber cómo responder de la manera apropiada
  - figura.area()
  - figura = cuadrado, rectángulo, triángulo, círculo.

Ejercicio “La Biblioteca”

“Una biblioteca tiene libros que presta a sus socios. Los socios se caracterizan por un código de socio y su nombre. Los libros, por su ISBN, título y autor. Un préstamo relaciona la acción de prestar un libro a un socio en una fecha y tiene un código de préstamo. Cuando un socio devuelve un libro el bibliotecario disminuye su cantidad de libros prestados. Un socio con tres o mas libros prestados es un socio no fiable…”


bibliotecario
DNI
nombre
crearBibliotecario
destruirBibliotecario
prestamo
codPrestamo
fecha
crearPrestamo
destruirPrestamo
socio
codSocio
nombre
crearSocio
destruirSocio
hacerSocioNoFiable
libro
ISBN
titulo
autor
crearLibro
destruirLibro
socio no fiable
diasDePenalizacion
despenalizar

Ejemplo Código C++

```
#include <iostream.h>
#include <stdlib.h>

class CDummy {
  public:
    CDummy () {n++;};
    ~CDummy () {n--;};
     static int getN() {return n;};
  private:
     static int n;
};
int CDummy::n=0;
int main () {
  CDummy a;
  CDummy j=new CDummy();
  CDummy *cplus=new CDummy; //tb puede ir con paréntesis
  cout << a.getN() << endl;
  }
```

¿Es correcto este programa?

Constructor: Ejercicio