Sistema de Préstamos Bibliotecarios – Taller POO en C#

Santiago Andrés Echeverri Montoya

Programación de Software

7/09/2025

**1. Definir el caso de negocio**

* Problema: En la biblioteca no existe un control digital sencillo para registrar préstamos y devoluciones de libros, lo que ocasiona pérdidas, retrasos en devoluciones y dificultad para saber qué libro está disponible.
* Usuarios:
  + Bibliotecario (administra libros, socios y préstamos).
  + Socio (estudiante o docente, solicita los libros).
* Valor esperado: Un sistema de consola que permita gestionar los libros, registrar préstamos/devoluciones y aplicar reglas de negocio (15 días para estudiantes, 30 para docentes).

**2. Historias de usuario**

HU1 – Registrar libro  
Como bibliotecario, quiero registrar un libro físico o digital, para que quede disponible en el inventario.

* Given un libro con Id único, When lo registro, Then queda disponible para préstamo.

HU2 – Registrar socio  
Como bibliotecario, quiero registrar un socio indicando si es estudiante o docente, para que el sistema aplique las reglas correctas de préstamo.

* Given un socio con Id único, When lo registro, Then queda disponible en la lista de socios.

HU3 – Prestar libro  
Como bibliotecario, quiero registrar un préstamo a un socio, para que el libro quede marcado como no disponible.

* Given un libro disponible, When lo presto a un socio válido, Then el sistema calcula la fecha de devolución según su tipo.

HU4 – Devolver libro  
Como bibliotecario, quiero registrar la devolución de un libro, para que quede nuevamente disponible en el inventario.

* Given un préstamo activo, When el socio devuelve el libro, Then el libro vuelve a estar disponible.

HU5 – Listar libros  
Como bibliotecario, quiero listar los libros de la biblioteca, para que pueda ver cuáles están disponibles o en préstamo.

* Given libros registrados, When consulto la lista, Then se muestran con su estado actual (disponible/no disponible).

HU6 – Controlar política de préstamo  
Como sistema, quiero aplicar diferentes políticas de préstamo (15 días estudiantes, 30 días docentes), para que los usuarios tengan claridad de cuándo devolver los libros.

* Given un préstamo, When el socio es estudiante, Then se calcula fecha devolución en 15 días.
* Given un préstamo, When el socio es docente, Then se calcula fecha devolución en 30 días.

**3. Requerimientos**

De negocio (RN):

* RN-01: Todo libro registrado debe tener un Id único.
* RN-02: Un libro no puede prestarse si ya está en préstamo.
* RN-03: El préstamo a estudiantes dura 15 días.
* RN-04: El préstamo a docentes dura 30 días.
* RN-05: Solo los socios registrados pueden realizar préstamos.
* RN-06: Un socio no puede tener más de **2 libros prestados simultáneamente.**

Funcionales (RF):

* RF-01: CRUD de libros (crear, listar, consultar).
* RF-02: CRUD de socios.
* RF-03: Registrar préstamo de libro a socio.
* RF-04: Registrar devolución de libro.
* RF-05: Validar disponibilidad de libro antes de prestar.
* RF-06: Aplicar reglas de tiempo según tipo de socio.
* RF-07: Mostrar listado de libros con su estado.
* RF-08: Menú de consola interactivo.

No funcionales (RNF):

* RNF-01: El sistema debe responder en < 1 segundo con hasta 100 registros.
* RNF-02: Manejo de errores con mensajes claros (ej. “No existe socio”).
* RNF-03: Código limpio y mantenible (nombres claros en español).
* RNF-04: Portabilidad en Windows/Linux con .NET 8.
* RNF-05: Testabilidad mediante clases separadas (Domain, Application, Infrastructure).
* RNF-06: Persistencia opcional en memoria o archivo .json.

**3. Diseño de POO aplicado**

🔹 1. Encapsulamiento

* Los atributos internos de las clases están protegidos y solo se accede a ellos mediante métodos controlados:
  + En Libro, la propiedad Disponible no se puede modificar directamente, solo cambia mediante los métodos Prestar() y Devolver().
  + En Socio, los datos (Id, Nombre, Tipo) se asignan en el constructor y no se pueden alterar desde fuera.
* Esto garantiza que el estado de los objetos sea consistente y evita manipulaciones indebidas.

🔹 2. Abstracción

* Se modelan solo los elementos esenciales del dominio:
  + Libro es una clase abstracta que define lo básico de cualquier libro (Id, Título, Disponible), pero delega los detalles a sus clases hijas.
  + La interfaz IPoliticaPrestamo abstrae la regla de negocio de cálculo de fecha de devolución, sin importar si es para estudiante o docente.
* Gracias a la abstracción, el sistema se centra en lo importante para la biblioteca sin exponer la complejidad al usuario.

🔹 3. Herencia

* Se aprovecha la reutilización de código mediante jerarquías:
  + LibroFisico y LibroDigital heredan de Libro, compartiendo atributos comunes y agregando sus propios detalles (Ubicacion o Url).
  + Las clases PoliticaPrestamoNormal y PoliticaPrestamoDocente implementan la misma interfaz (IPoliticaPrestamo), cada una con reglas diferentes.
* La herencia permite que el sistema crezca en el futuro sin duplicar lógica.

🔹 4. Polimorfismo

* Se usa el mismo contrato para comportamientos distintos:
  + En Prestamo, se recibe un objeto de tipo IPoliticaPrestamo, lo que permite que dependiendo del socio (estudiante o docente) se aplique una política diferente sin cambiar el código del préstamo.
  + Al listar los libros, tanto LibroFisico como LibroDigital se tratan como un Libro, mostrando polimorfismo de clases.
* Esto hace que el sistema sea flexible y extensible, ya que se pueden añadir nuevas políticas o tipos de libros sin alterar el resto del código.

**4.Modelo Conceptual**

Entidades y relaciones

1. Libro

* Representa el material disponible en la biblioteca.
* Tiene atributos como Id, Título y Disponible.
* Puede ser de dos tipos:
  + LibroFísico (con atributo Ubicación).
  + LibroDigital (con atributo Url).
* Relación: un Libro puede estar asociado a cero o un Préstamo (si está prestado o no).

2. Socio

* Representa a la persona que accede a los servicios de la biblioteca.
* Tiene atributos como Id, Nombre y Tipo (Estudiante o Docente).
* Relación: un Socio puede tener muchos Préstamos a lo largo del tiempo.

3. Préstamo

* Representa la acción de prestar un libro a un socio.
* Guarda FechaPréstamo y FechaDevolución.
* Relación:
  + Un Préstamo está asociado a un Libro.
  + Un Préstamo está asociado a un Socio.
* Además, utiliza una Política de Préstamo (normal o docente) para calcular la fecha de devolución.

Resumen de relaciones:

* Un Socio puede tener muchos Préstamos.
* Un Libro puede estar en cero o un Préstamo a la vez.
* Cada Préstamo une a un Socio con un Libro bajo una política de préstamo.

**5.Plan de Pruebas - Sistema de Préstamos Bibliotecarios**

**Casos de Prueba Manuales**

* + CP-01: Registro de Libro Físico Exitoso

Given: Sistema iniciado When: Registro libro con ID="L001", Título="El Principito", Ubicación="Estante A-3" Then: El libro queda registrado y disponible

Datos de entrada: ID=L001, Título=El Principito, Ubicación=Estante A-3

* + CP-02: Préstamo a Estudiante

Given: Libro L001 disponible y socio estudiante S001 registrado When: Realizo préstamo del libro L001 al socio S001 Then: Libro marcado como no disponible

Datos de entrada: fecha devolución = fecha actual + 15 días Datos de entrada: LibroId=L001, SocioId=S001

* + CP-03: Validación de Libro Ya Prestado

Given: Libro L001 ya prestado a socio S001 When: Intento prestar el mismo libro L001 a socio S002 Then: Sistema muestra error "El libro no está disponible"

Datos de entrada: LibroId=L001, SocioId=S002

* + CP-04: Validación Máximo 2 Libros por Socio

Given: Socio S001 ya tiene 2 libros prestados When: Intento prestar un tercer libro L003 al mismo socio Then: Sistema muestra error "El socio ya tiene el máximo de libros prestados (2)"

Datos de entrada: LibroId=L003, SocioId=S001

* + CP-05: Devolución Exitosa

Given: Libro L001 prestado al socio S001 When: Proceso devolución del libro L001 Then: Libro marcado como disponible, préstamo marcado como inactivo

Datos de entrada: LibroId=L001

* + CP-06: Política de Préstamo Docente

Given: Libro L002 disponible y socio docente D001 registrado When: Realizo préstamo del libro L002 al socio D001 Then: Fecha devolución = fecha actual + 30 días.

Datos de entrada: LibroId=L002, SocioId=D001

**Casos de Prueba de Error**

* + CPE-01: Registro con ID Duplicado

Given: Ya existe libro con ID="L001" When: Intento registrar otro libro con ID="L001" Then: Error "Ya existe un libro con ese Id"

* + CPE-02: Préstamo a Socio Inexistente

Given: No existe socio con ID="S999" When: Intento prestar libro L001 al socio S999 Then: Error "No existe el socio"

* + CPE-03: Devolución de Libro No Prestado

Given: Libro L001 está disponible (no prestado) When: Intento devolver el libro L001 Then: Error "El libro no está prestado"