

# SISTEMA DE GESTIÓN UNIVERSITARIA DE CAFETERÍA Y BIBLIOTECA

David Menoyo Ros

Álvaro Zamorano Ortega

MEMORIA FINAL

## Enunciado y Requisitos

El objetivo del programa es el desarrollo de un sistema que gestione las operaciones de las cafeterías y bibliotecas de una universidad. Las operaciones se dirigirán al personal que componga la universidad, permitiendo también que el personal de cafetería y biblioteca puedan realizar las mismas operaciones que el resto de personas. Debido a esto, se desea implementar una aplicación que permita:

- **Gestión de biblioteca:** Gestionar las diferentes publicaciones y las diferentes salas que posee la biblioteca. Asimismo, se requerirá que el bibliotecario valide las diferentes concesiones.
- **Gestión de cafetería:** Gestionar los diferentes productos de la cafetería, así como realizar el proceso de compra y simulación del proceso de elaboración de comidas.

La aplicación podrá ser ejecutada desde cualquier dispositivo, siendo una aplicación de escritorio en el que cada usuario tendrá una cuenta propia de acceso (dependiendo del usuario la aplicación ofrecerá unas funcionalidades u otras).

Los elementos a tener en cuenta por la aplicación son los siguientes:

### Usuarios

En la aplicación habrá 4 tipos de usuarios:

- **Estudiante:** Utilizarán las acciones de compra y préstamo de la universidad, así como su consulta.
- **Profesor:** Mismas acciones que los estudiantes.
- **Bibliotecario:** Además de las funciones de los estudiantes y profesores, realizarán las acciones de control y gestión de la biblioteca.
- **Camarero:** Además de las funciones de los estudiantes y profesores, realizarán las acciones de control y gestión de la cafetería.

Los atributos comunes a camareros, profesores, estudiantes y bibliotecarios serán:

- Apellidos
- Nombre
- Dirección
- Cuenta bancaria
- Dni
- Email
- Password
- Teléfono.

### Productos de Biblioteca

La biblioteca ofrecerá diferentes productos, desde publicaciones a salas de estudio.

### Catálogo de publicaciones

La Biblioteca Universitaria posee los siguientes tipos de publicaciones:

- Libros.
- Revistas.
- Proyectos fin de grado.

Los datos a considerar para cada Publicación dependen del tipo del mismo. Son comunes a todas ellas:

- ISBN: Es la identificación única para una publicación.
- Título: Título de la publicación.
- Autores: Nombre y apellidos de los autores.
- Fecha de publicación: La fecha en que fue publicada.

- **Materia:** Indica la materia de la publicación.

## **Libros**

Para todos los libros, se deben añadir:

- **Editorial:** identifica la editorial que ha publicado el libro
- **Localidad:** indica la localidad de la publicación.
- **Edición:** en este apartado se reflejará la edición de la obra.

## **Revistas**

Para todas las revistas, se deben añadir:

- **Periodicidad:** se indica si es trimestral, semestral o anual.
- **Volumen:** número del volumen.
- **Número:** indica el número de la revista.

## **Proyectos**

Para todos los proyectos, se deben añadir:

- **Tribunal:** se indica el tribunal del proyecto.
- **Titulación:** indica la titulación a la que pertenece el proyecto.
- **Calificación:** indica la nota sacada.

## [Catálogo de otros productos de la biblioteca](#)

## **Salas**

Para todas las salas, se deben añadir:

- **Aforo:** aforo permitido.
- **Capacidad:** capacidad permitida.

## [Productos de Cafetería](#)

En la cafetería se ofrecerá comidas, bebidas y menús, todos con los atributos:

- **Identificador**
- **Nombre**
- **Precio**

Y el menú se compondrá de comidas y bebidas, concretamente:

- **PrimerPlato:** Comida.
- **SegundoPlato:** Comida.
- **Postre:** Comida.
- **Bebida:** Bebida.

En la aplicación se permitirá una opción en la que el usuario pueda elegir cualquier tipo de comida y bebida en los apartados correspondientes del menú.

## [Módulo de Gestión de Biblioteca](#)

En este apartado definimos las operaciones y actividades que se podrán realizar en el ámbito de los diferentes usuarios que interaccionen con la biblioteca.

Operaciones permitidas a los **bibliotecarios**:

- Alta/Baja/Modificación de publicaciones y concesiones de salas.
- Simular la entrega de la publicación o sala.
- Consulta del estado los préstamos de los usuarios

Operaciones permitidas a los diferentes **usuarios** de la aplicación(profesores, alumnos, camareros y los propios bibliotecarios):

- Búsqueda de publicaciones, permitiendo filtrar por nombre de autor, título, materia y fecha de publicación. También se podrán buscar las salas por su capacidad.
- Ordenación de publicaciones de forma ascendente o descendente, permitiendo ordenar por nombre de autor, título, materia o fecha de publicación.
- Solicitar préstamos:
  - Según el cliente que haya entrado en la aplicación, el número de los días de préstamo será distinto para cada cliente:
    - Profesor:
      - Libros: 10 días
      - Proyectos: 12 días
      - Revistas: 6 días
      - Salas: 2 horas
    - Estudiante:
      - Libros: 6 días
      - Proyectos: 10 días
      - Revistas:3 días
      - Salas: 2 horas
    - Camareros y Bibliotecarios:
      - Libros: 5 días
      - Proyectos: No pueden
      - Revistas: No pueden
      - Salas: 2 horas.
  - Se generará un fichero de texto con los datos de las publicaciones y del usuario que pide el préstamo.
- Realizar devoluciones de préstamos o salas:
  - Si se excede el límite de tiempo, se notificará de una infracción.
  - Se generará un fichero de texto con la información de la devolución.
- Visualización del histórico de los préstamos concedidos.

### Módulo de Gestión de Cafetería

En este apartado definimos las operaciones y actividades que se podrán realizar en el ámbito de los diferentes usuarios que interaccionen con la cafetería.

Operaciones permitidas a los **camareros**:

- Alta/Baja/Modificación del catálogo de productos de cafetería.
- Simular la creación de los productos de la cafetería, es decir, simular que los distintos productos se cocinan.
- Consulta de las comandas pendientes.

Operaciones permitidas a los diferentes usuarios de la aplicación (profesores, alumnos, bibliotecarios y camareros):

- Compra de productos de cafetería:
  - Se generará un archivo de texto con los datos del usuario y de los productos.

## Instrucciones de ejecución y manual de usuario

La aplicación ha sido desarrollada en lenguaje Java y realizada en el entorno de desarrollo NetBeans en Swing. Para instalar la aplicación es necesario tener la máquina virtual de Java y el entorno de desarrollo NetBeans. El ejecutable con extensión .jar desde el que se puede abrir la aplicación se encuentra en la carpeta del proyecto.

A continuación se detallarán cada una de las secciones de la aplicación para un correcto uso:

### Login

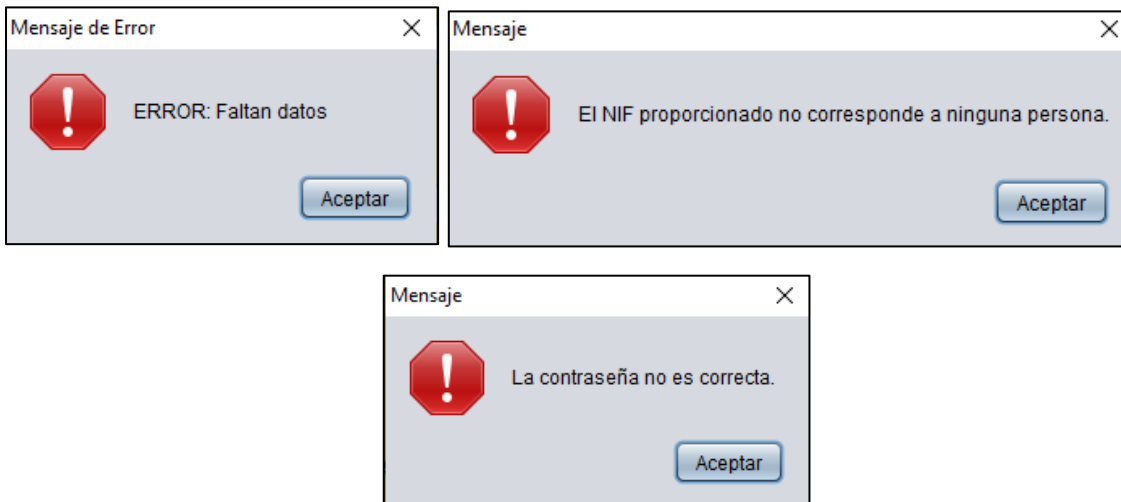
La primera ventana que el usuario encontrará será la ventana de login en que el usuario introducirá su DNI y Contraseña.



**Usuarios disponibles:** Camarero{DNI=1,Contraseña=1234}, Bibliotecario{2,1234} y Estudiante{3,1234}.

(En caso de no disponer de esos usuarios iniciales, simplemente habría que ejecutar el script **PruebaSingletonProxyServer.java** del paquete **SingletonProxyServidor**. De esta manera se generarían los datos iniciales necesarios para la simulación de la aplicación)

En caso de introducir nada o introducir un usuario o contraseña incorrecta la aplicación nos mostrará los siguientes mensajes:



### Ventana Principal

Una vez introducido las credenciales correctamente, se mostrará una ventana en donde se podrá elegir la opción de entrar al módulo de gestión de la biblioteca o cafetería, así como entrar como camarero o bibliotecario a la gestión de la biblioteca o cafetería.

SISTEMA CAFETERIA Y BIBLIOTECA

Nombre: Alvaro  
Apellidos: Zamorano

**Biblioteca:**

**Cafetería:**

**Bibliotecario:**

**Camarero:**

Si se quiere acceder al módulo de gestión para camareros y bibliotecarios, pero no se tienen permisos, se mostrará el siguiente mensaje:



### Ventana de gestión de biblioteca como usuario normal

Si se ha seleccionado la opción de acceder al módulo de gestión de biblioteca como usuario normal en la ventana principal, en esta ventana se mostrarán las diferentes opciones permitidas para todos los usuarios de la aplicación.

SISTEMA CAFETERIA Y BIBLIOTECA

Nombre: Alvaro  
Apellidos: Zamorano

**Consultar Préstamos -**

**Realizar Préstamo-** ID:

**Devolver Préstamo-** ID:

**Búsqueda Publicaciones:**

### Búsqueda de publicaciones

En la ventana aparecerá un componente de tipo JComboBox que permitirá seleccionar el tipo de filtro (autor, fecha, materia, título y capacidad de salas) para buscar en los diferentes productos de la biblioteca. De este modo, se escribirá el valor que queramos buscar y seleccionamos el botón buscar:

Autor ▼ 
 Buscar Ordenar Asc Ordenar Des

1 - Libro: Id= L1, Editorial= libro1, LocalidadPublicacion= ed, Edicion= 1, ISBN= ISBN, Titulo= titu

Si no se ha introducido un valor, se mostrará el siguiente mensaje de error:



#### Ordenación de publicaciones

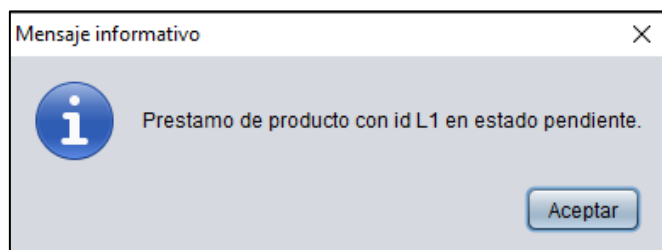
Al igual que en la búsqueda, seleccionamos el tipo de filtro que queremos para ordenar las publicaciones y damos click al botón ordenar asc (ascendentemente) u ordenar des (descendentemente) según queramos.

Autor ▼ 
 Buscar Ordenar Asc Ordenar Des

1 - Libro: Id= L1, Editorial= libro1, LocalidadPublicacion= ed, Edicion= 1, ISBN= ISBN, Titulo= titu  
 2 - Revista: Id= R1, Periodicidad= revista1, Volumen= 2, Numero= 3, ISBN= ISBN, Titulo= titulo, A  
 3 - Proyecto: Id= P1, Tribunal= [autor1], Titulacion= proyecto1, Calificacion= 3.5, ISBN= ISBN, Titu

#### Solicitar préstamo

Para solicitar un préstamo se deberá introducir el identificador del producto de la biblioteca que quiera tomar prestado y dar click al botón prestar. Si se ha realizado de forma correcta, se mostrará el siguiente mensaje:

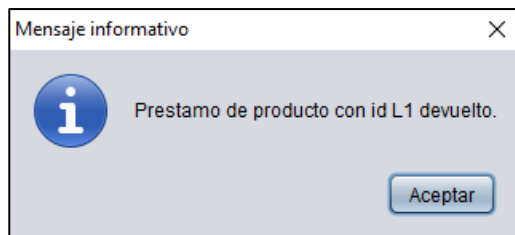


Al contrario, si no se puede tomar prestado el producto se mostrará los siguientes mensajes de error dependiendo del por qué:



### Devolver préstamo

Para devolver un producto prestado se deberá introducir el identificador del producto prestado y dar click al botón devolver. Si se ha realizado de forma correcta se mostrará el siguiente mensaje informativo:



Al contrario, si no se puede devolver el producto o se ha introducido mal el identificador se mostrará los siguientes mensajes de error:



### Consultar préstamos

Si se da click al botón acceder de consulta de préstamos, aparecerá una nueva ventana en la que se mostrarán los préstamos que ha realizado el usuario. Con los botones Ant y Sig se mostrarán los demás préstamos realizados.



**SISTEMA CAFETERIA Y BIBLIOTECA**

**CONSULTA PRESTAMOS**

**Ant** **Sig**

Producto  
Id Producto: L1, DNI Persona: 1

Fecha Prestamo  
Wed Jan 15 12:37:01 CET 2020

Fecha Devolucion Determinada  
Mon Jan 20 12:37:01 CET 2020

Fecha devolución Usuario  
Wed Jan 15 12:37:08 CET 2020

Días/Horas permitidos  
5 dias

Estado  
Devuelto

### Ventana de gestión de cafetería como usuario normal

En la ventana principal, si se ha seleccionado entrar a la cafetería, se mostrará una nueva ventana en la que aparecerán los diferentes productos de la cafetería a comprar. De este modo, se podrán seleccionar cualquiera de ellos, y una vez seleccionados, se deberá dar click al botón entregar comanda.

**SISTEMA CAFETERIA Y BIBLIOTECA**

Nombre: Alvaro  
Apellidos: Zamorano

**Comidas**  
macarrones1  
lentejas1  
cocido1  
ternera1  
ensalada1

**Postres**  
tarta de chocolate  
helado1  
café1

**Bebidas**  
cocacola1  
fanta1  
trina1

**Menús**  
menu1  
menuBarato1  
menuCaro1

**Entregar Comanda**

Si se ha realizado de forma correcta, se mostrará el siguiente mensaje:

**Mensaje informativo**

**i** Comanda pendiente

**Aceptar**

### Ventana de gestión de biblioteca para usuarios bibliotecarios

Si en la ventana principal se ha elegido la opción de entrar al módulo de gestión de bibliotecarios, se mostrará (si se accede a la aplicación como usuario con permisos de bibliotecario) una nueva ventana en la que se visualizarán las diferentes operaciones permitidas para los bibliotecarios.

SISTEMA CAFETERIA Y BIBLIOTECA

Nombre: Alvaro  
Apellidos: Zamorano

**Alta Publicaciones-** **Acceder**

**Alta Infraestructuras-** **Acceder**

**Baja Productos-** **Acceder**

**Consulta Prestamos-** **Acceder**

**Conceder Prestamos-** **Acceder**

### Alta de publicaciones

Si se accede al alta de nuevas publicaciones, se mostrará una nueva ventana en la que se visualizarán diferentes campos a rellenar para dar de alta una nueva publicación en la aplicación. Se deberá seleccionar el tipo de publicación en el componente JComboBox debido a que cada tipo (Libro, Revista y Proyecto) tienen diferentes atributos.

SISTEMA CAFETERIA Y BIBLIOTECA

Identificador:  Seleccione Tipo: **Libro**

ISBN:  **Añadir**

Título:

Autor:

Fecha de publicación:

Materia:

Editorial:

Localidad de publicación:

Edición:

Si se ha añadido de forma correcta, aplicación mostrará el siguiente mensaje:

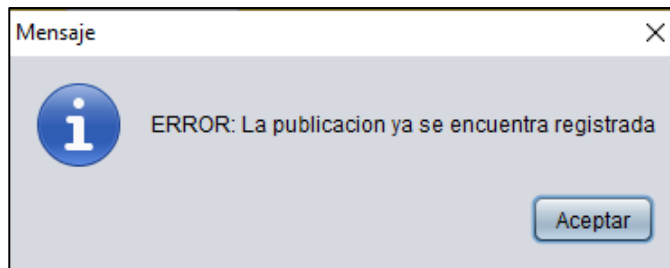
Mensaje

**i** Publicacion Libro1 registrada correctamente

**Aceptar**

Para el atributo fecha de deberá rellenar de la forma “dd/mm/yy hh:mm:ss” y para el tribunal de los proyectos los nombres separados por comas.

Si se añade una publicación con un identificador ya registrado, aparecerá el siguiente error:



#### Alta de infraestructuras

Si se accede al alta de infraestructuras, se mostrará una nueva ventana en la que se mostrarán campos a rellenar para dar de alta una nueva infraestructura.

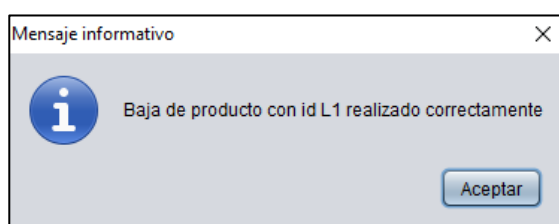
A screenshot of a web application window titled "SISTEMA CAFETERIA Y BIBLIOTECA". The background is yellow. In the top right corner, there is a dropdown menu labeled "Sala". Below this, there are three input fields, each preceded by a label: "Capacidad:", "Aforo:", and "Identificador:". To the right of the "Capacidad:" field is a dark button labeled "Añadir".

#### Baja de productos

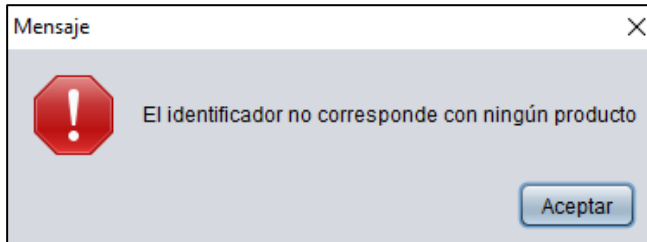
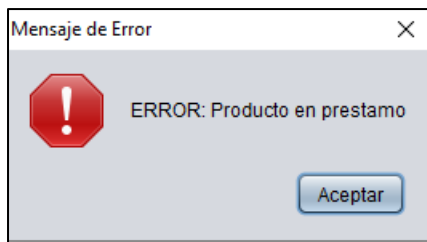
Si se accede a baja de productos, se mostrará una nueva ventana en la que se deberá indicar el identificador del producto a borrar. Si se selecciona consultar, se mostrarán los datos del producto con identificador indicado, y si selecciona eliminar, se eliminará el producto de la aplicación.

A screenshot of a web application window titled "SISTEMA CAFETERIA Y BIBLIOTECA". The background is yellow. At the top, there is a label "Introduce id de Producto a eliminar:" followed by a text input field. Below the input field are two dark buttons: "Consultar" on the left and "Eliminar" on the right. At the bottom of the window is a large, empty white rectangular area.

Si se ha eliminado de forma correcta, se mostrará el siguiente mensaje informativo:



Al contrario, si hay algún error se mostrarán los siguientes mensajes



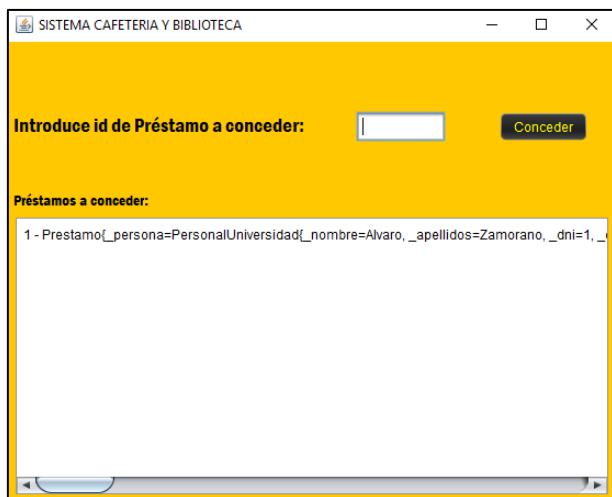
### Consulta de préstamos

Si se da acceso a la consulta de préstamos, aparecerá una nueva ventana en la que se mostrarán los préstamos realizados por los usuarios en la biblioteca del bibliotecario. Con los botones Ant y Sig se mostrarán los demás préstamos realizados.

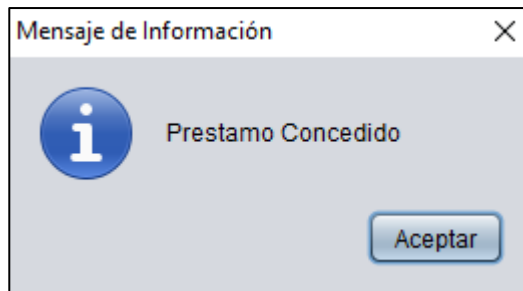


### Conceder préstamos

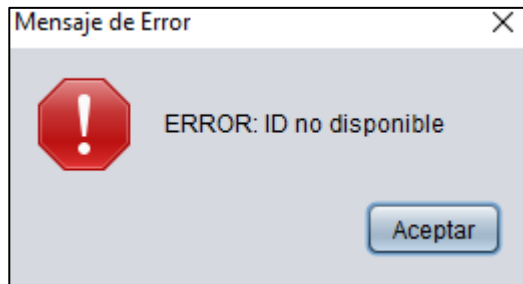
Si se selecciona al botón de conceder préstamos, se mostrará una nueva ventana en la que aparecerán aquellos préstamos en estado pendiente. Para concederlos, se deberá introducir el identificador del producto a prestar y dar click al botón conceder.



Si se ha realizado de forma satisfactoria, se mostrará el siguiente mensaje informativo:



Al contrario, se mostrarán el siguiente mensaje de error:



### Ventana de gestión de cafetería para usuarios camareros

Si en la ventana principal se ha elegido la opción de entrar al módulo de gestión de bibliotecarios, se mostrará (si se accede a la aplicación como usuario con permisos de camarero) una nueva ventana en la que se visualizarán las diferentes operaciones permitidas para los camareros.



### Alta de productos de cafetería

Si se selecciona el botón de alta de productos de cafetería, se abrirá una nueva ventana en la que se podrán dar de alta nuevos productos de cafetería en aplicación. Si se selecciona que quiere añadir un nuevo menú, se deberán seleccionar a su vez el primer plato, segundo plato, postre y bebida, además de rellenar los campos de nombre, precio e identificador.

Si se han rellenado correctamente todos los campos, se mostrará el siguiente mensaje informativo:

### Baja de productos de cafetería

Si se selecciona el botón de baja de productos, se mostrará una nueva ventana en la que se podrán eliminar cualquier producto de cafetería de la aplicación. Para ello se deberá introducir el identificador del producto y dar al botón de eliminar.

Si se ha eliminado de forma correcta, la aplicación mostrará el siguiente mensaje informativo:

### Conceder comanda

Si se selecciona el botón de cocinar comanda, se mostrará una nueva ventana en la que se visualizarán las comandas pendientes. Para cocinarlas, se deberá introducir el número de comanda que se muestre en el cuadro.

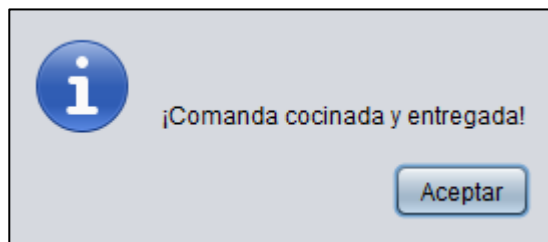
SISTEMA CAFETERIA Y BIBLIOTECA

Introduce id de Comanda a conceder:

Comandas a cocinar:

```
1 - Comanda{persona=PersonalUniversidad[_nombre=Alvaro, _apellidos=Zamorano, _dni=1, _e
2 - Comanda{persona=PersonalUniversidad[_nombre=Alvaro, _apellidos=Zamorano, _dni=1, _e
```

Si se ha cocinado correctamente, aparecerá, tras 5 segundos, el siguiente mensaje:



## PATRONES UTILIZADOS

Se han utilizado un total de 12 patrones:

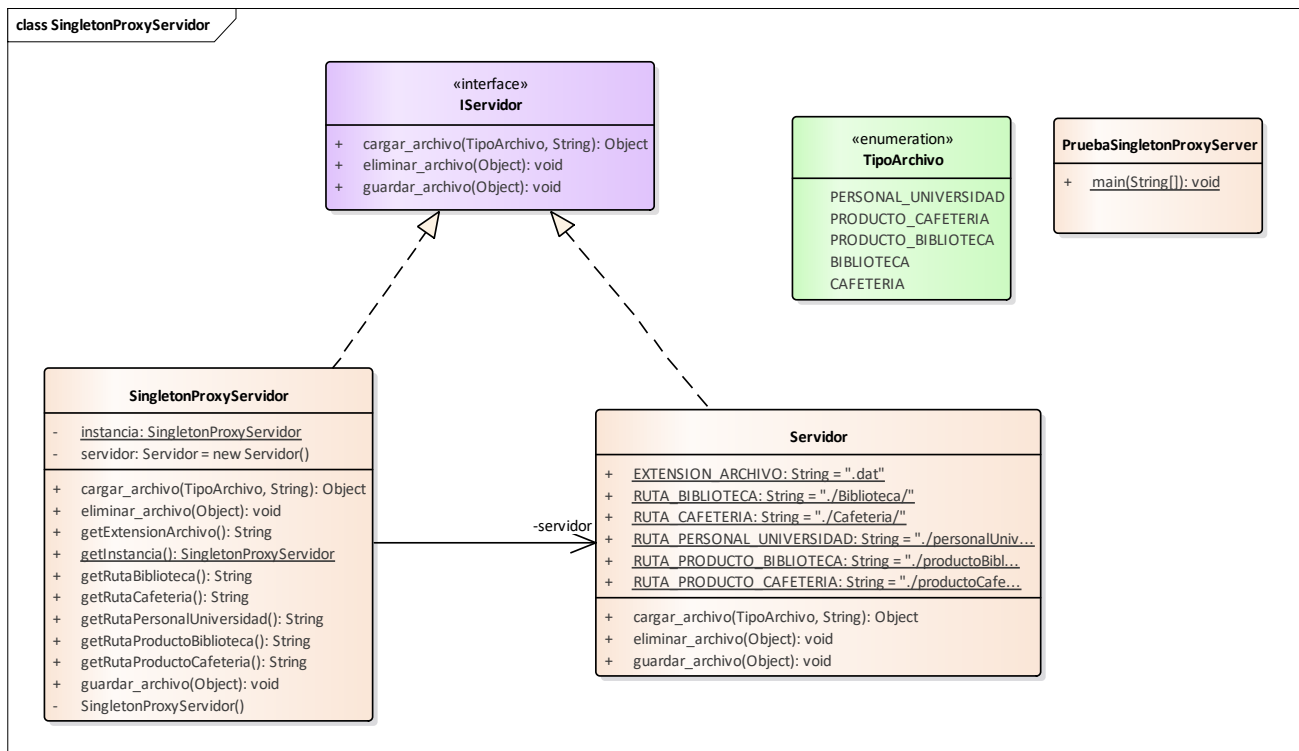
- Patrones de creación:
  - Factory Method
  - Builder
  - Prototype
  - Singleton
- Patrones estructurales
  - Bridge
  - Proxy
  - Adapter
- Patrones de comportamiento
  - State
  - Strategy
  - Observer
  - Iterator
  - Template Method

Veamos el diagrama y la explicación de cada patrón. Explicaremos los patrones que se utilicen de manera relacionada de manera conjunta.

### Patrón Proxy y Singleton para el servidor

Empezaremos por una de las partes más importantes del sistema, esta es la del **servidor** que se encarga de almacenar y obtener los diferentes objetos de la aplicación que se serializan.

Para confeccionar la clase Servidor, se empezó por confeccionar un **Proxy** que creara una indirección entre el cliente y el propio Servidor, pero más adelante, debido a la gran importancia del Servidor, se combinó el patrón Proxy con el Singleton, de manera que la instanciación y acceso a dicho componente fuera global y única. Veámoslo en el diagrama:

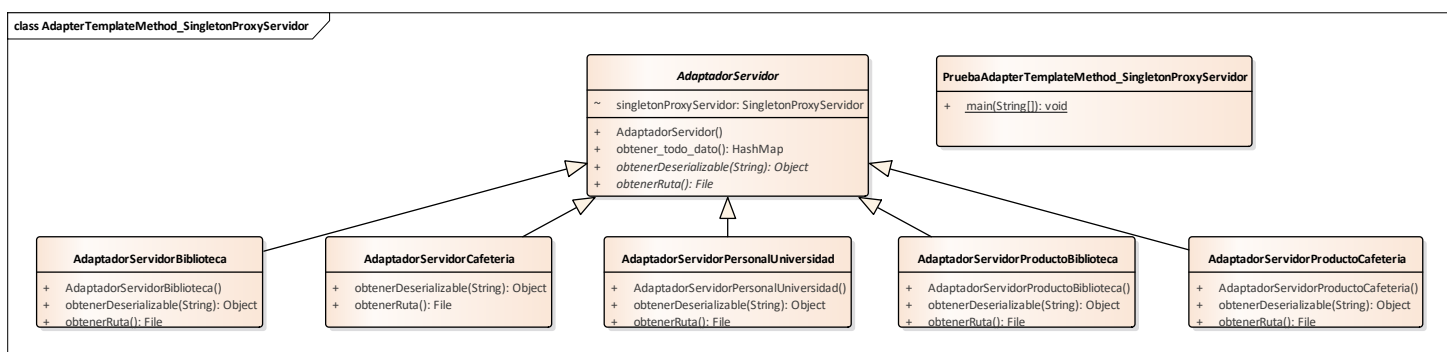




## Patrón Adapter y Template Method para extender funcionalidad del servidor

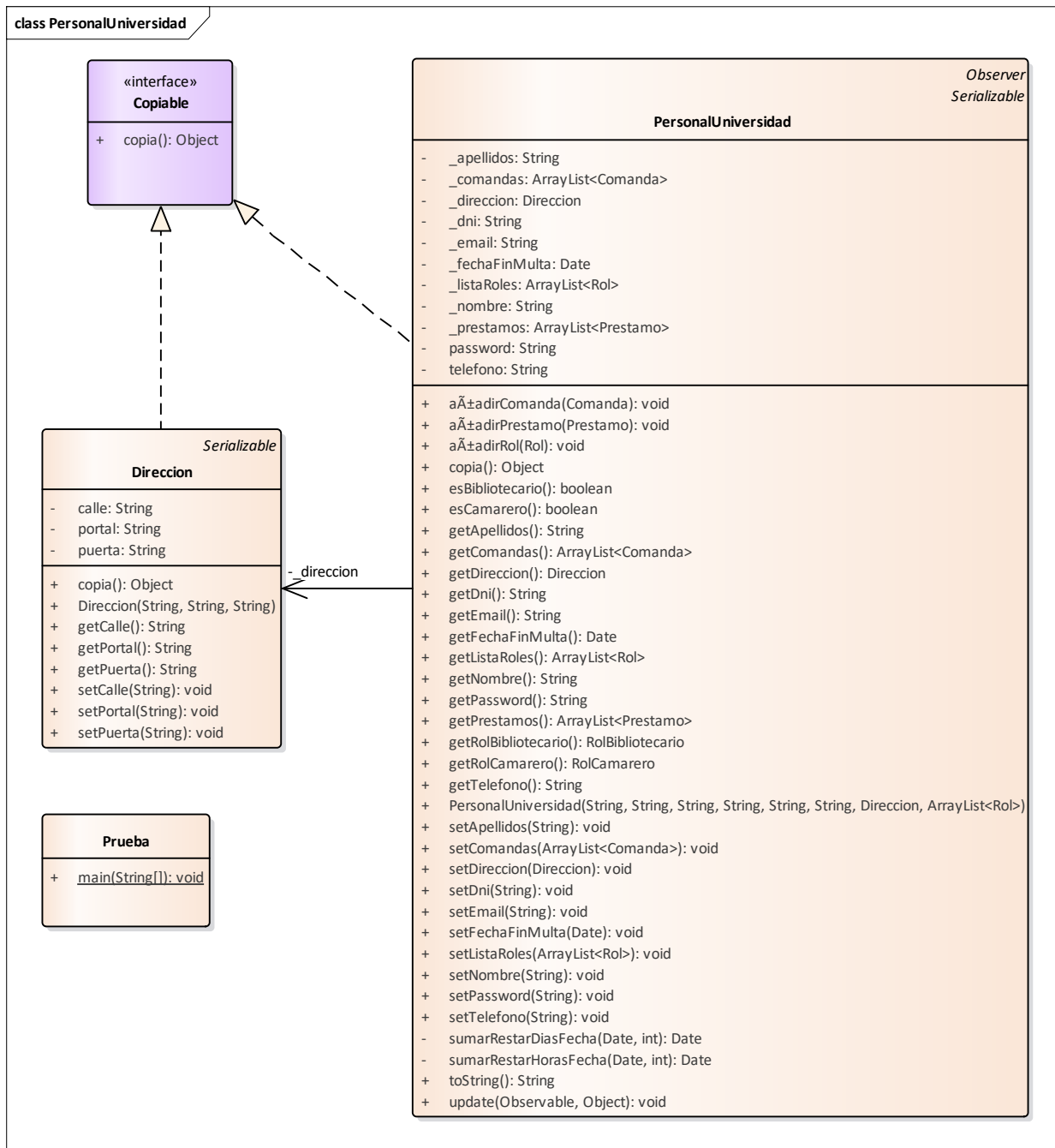
No obstante, a medida que empezamos a desarrollar la aplicación, nos encontramos con la necesidad de obtener no solo un solo objeto almacenado, sino obtener todos los objetos de un tipo concreto, por lo que necesitaríamos añadir métodos al sistema Singleton y Proxy del Servidor, pero aprovechamos esta oportunidad para utilizar el patrón **Adapter**, ya que dicha patrón nos conseguiría extender la funcionalidad del sistema previamente confeccionado, sin realizar en este ninguna modificación. En definitiva, estamos extendiendo su funcionalidad con simplemente añadir un Adaptador, el cual simplemente utilizará los métodos ya existentes del servidor, pero tantas veces como archivos de un cierto tipo haya en la base de datos.

Además, a la hora de confeccionarlo, nos percatamos de que debido a nuestra forma de almacenar los datos, estábamos creando varios condicionales, así como códigos repedidos excepto por dos partes concretas del mismo, por lo que aprovechas esta oportunidad perfecta para implementar el patrón **Template Method**, de manera que finalmente obtenemos un adaptador abstracto con el método extendido, el cual utiliza dos métodos abstractos que se define según el adaptador concreto implementado. Vemos esta parte en el siguiente diagrama, e n donde el método del adaptador es obtener\_todo\_dato(), y los métodos abstractos que definen las concreciones son obtenerDeserializable() y obtenerRuta():



## Patrón **Prototype** para crear personal universitario

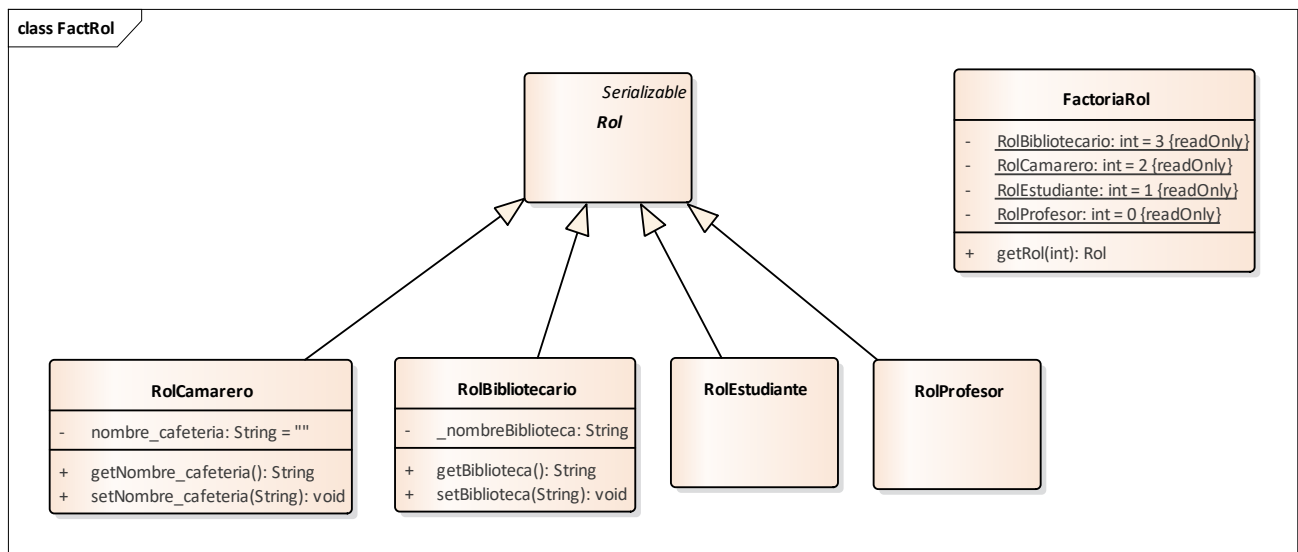
Para la creación de la clase que representa a los usuarios de la aplicación se utilizó el patrón **Prototype**. La elección de este patrón se debe a que queríamos rapidez en la creación del mismo, ya que probablemente se trata de la clase con más atributos de todo el sistema, y así nos limitamos a instanciar copias (en concreto “copia profunda”) de un prototipo de personal universitario previo.



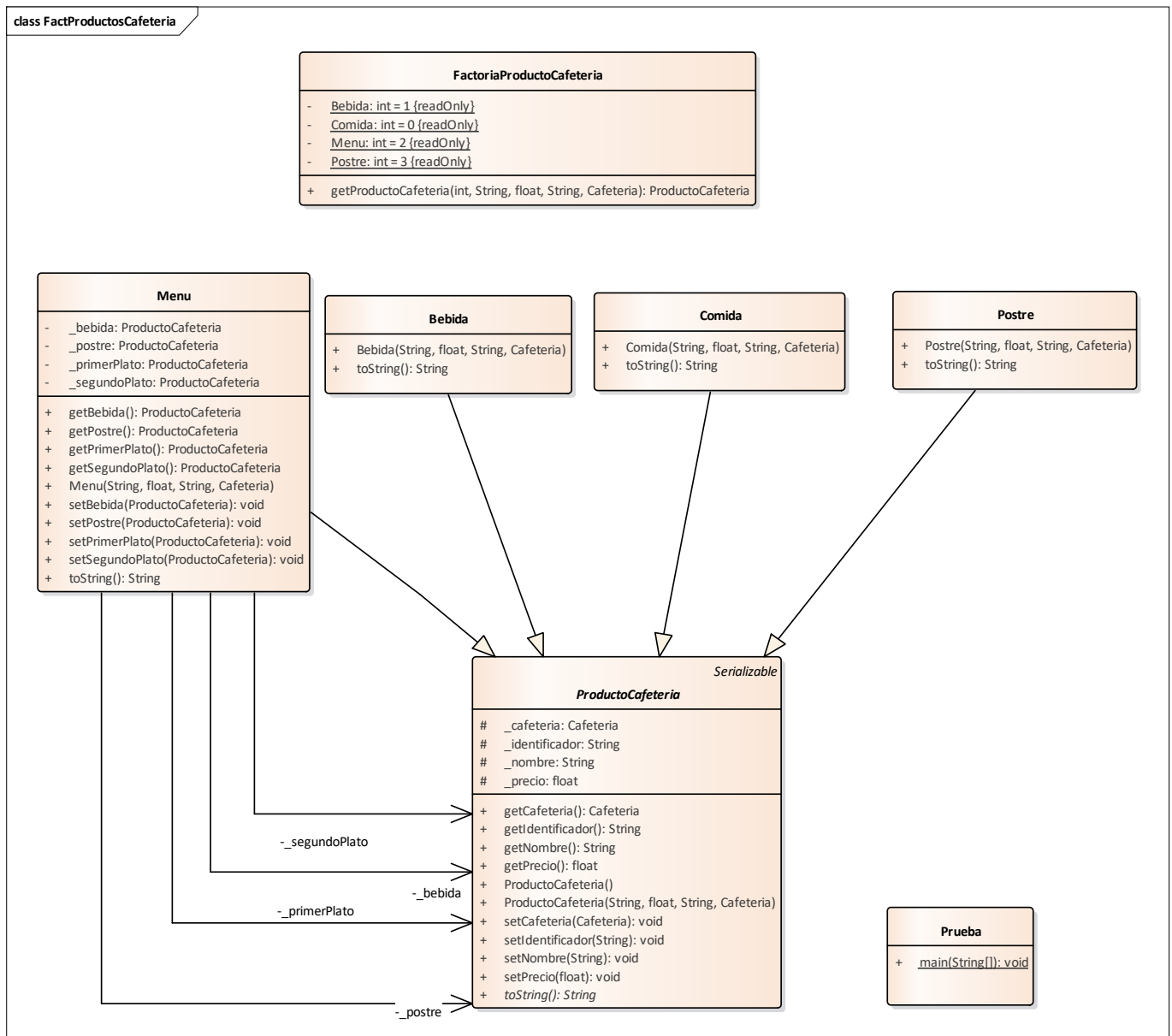
Además, aunque lo comentaremos en el patrón **Factory Method**, la clase **PersonalUniversidad** de apoya en el patrón fundamental **Delegation** y sobre todo **Marker Interface** que le confiere el atributo **\_listaRoles**, de esta manera conseguimos composición y no herencia.

Patrón **Factory Method** para crear Roles, Productos de biblioteca y de cafetería( con el patrón **Builder** también).

Para la creación de los productos de cafetería y biblioteca, así como los distintos roles del PersonalUniversidad, se utilizó el patrón **Factory Method** debido a su extrema sencillez, ya que además en nuestro caso no necesitamos un creador abstracto, sino que directamente la clase factoría está totalmente definida. Veamos los diferentes diagramas:

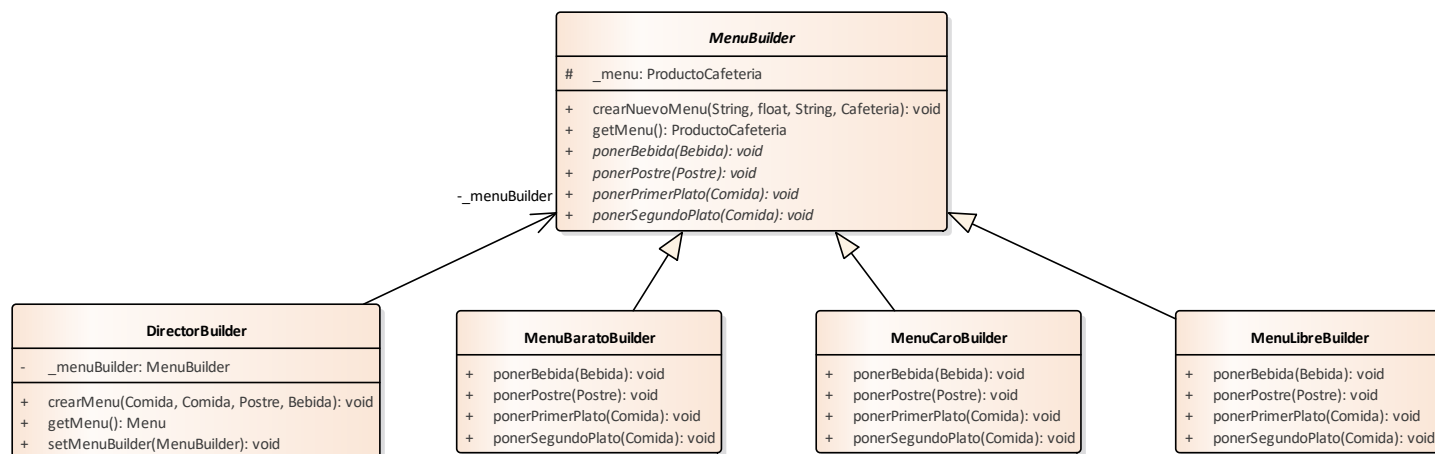


Para la factoría de productos de la cafetería ocurre algo extraño que finalmente hemos dejado codificado ya que se puede utilizar sin problemas, veamos primero el diagrama de clases:



Como vemos, podemos crear los 4 tipos de productos de cafetería sin problemas, no obstante, realmente la creación del Menú se realiza con el patrón **Builder**, el cual es el que más flexibilidad nos dá para la composición de los diferentes atributos. Probablemente lo más sensato es quitar dicha clase de la FactoriaProductoCafeteria, pero actualmente es el **Builder** quién utiliza la FactoriaProductoCafeteria para crear el Menú, y no deberíamos utilizar sin el Builder. Veamos el diagrama del patrón Builder para el Menú.

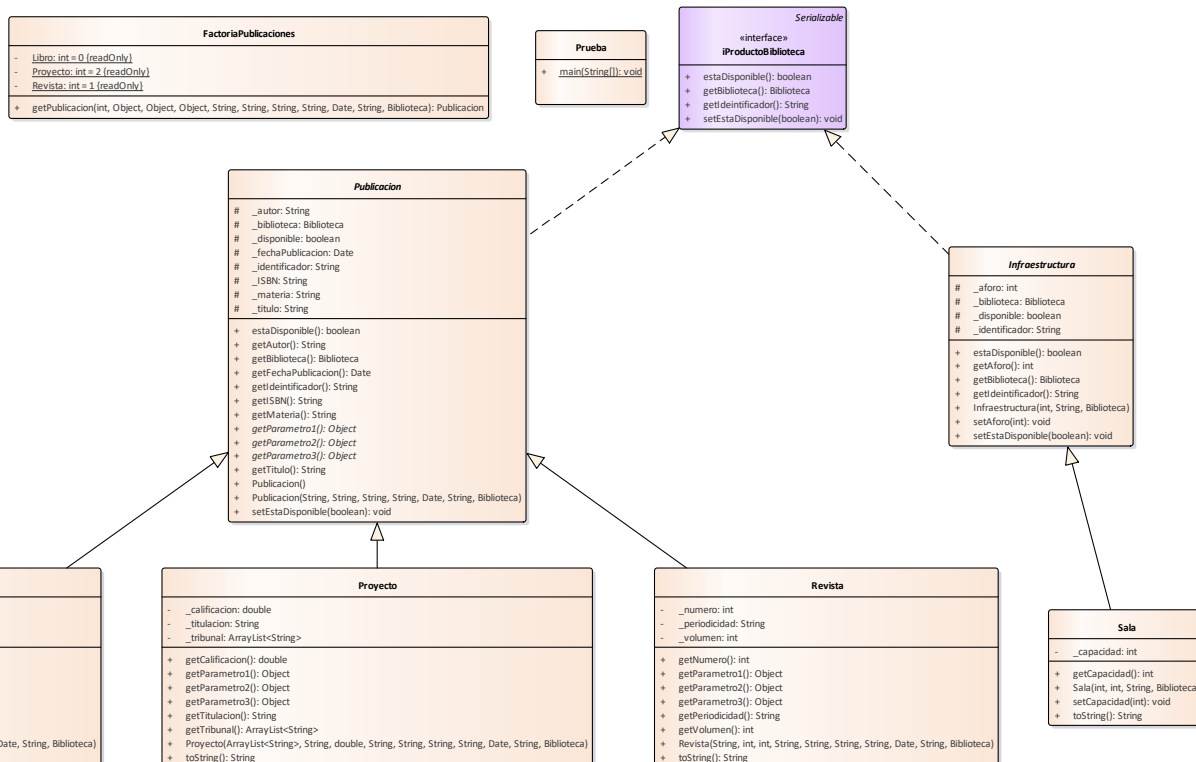
class BuilderMenu



En el método crearNuevoMenu(String,float,String,Cafeteria) es donde se utiliza la factoria de productos de cafetería para instanciar el Menú con los atributos comunes pasados por parámetro, y más adelante con la utilización del director y las concreciones se definirá y retornará el Menu correctamente creado.

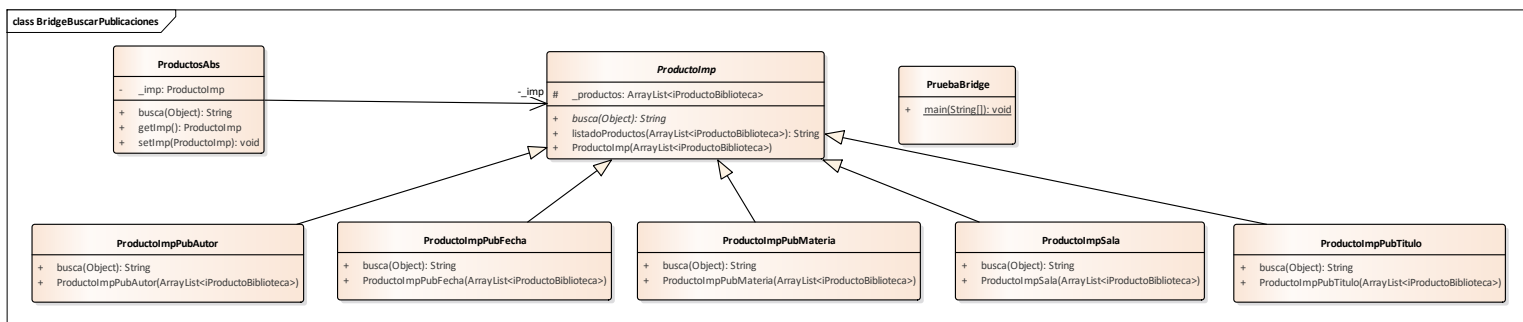
Y por último, para la creación de los productos de la cafetería, aparte de utilizar la factoría, se hace uso del patrón **Interface y Abstract Class** para poder diferenciar entre Publicaciones e Infraestructuras. Veamoslo en el diagrama:

class FactPublicaciones



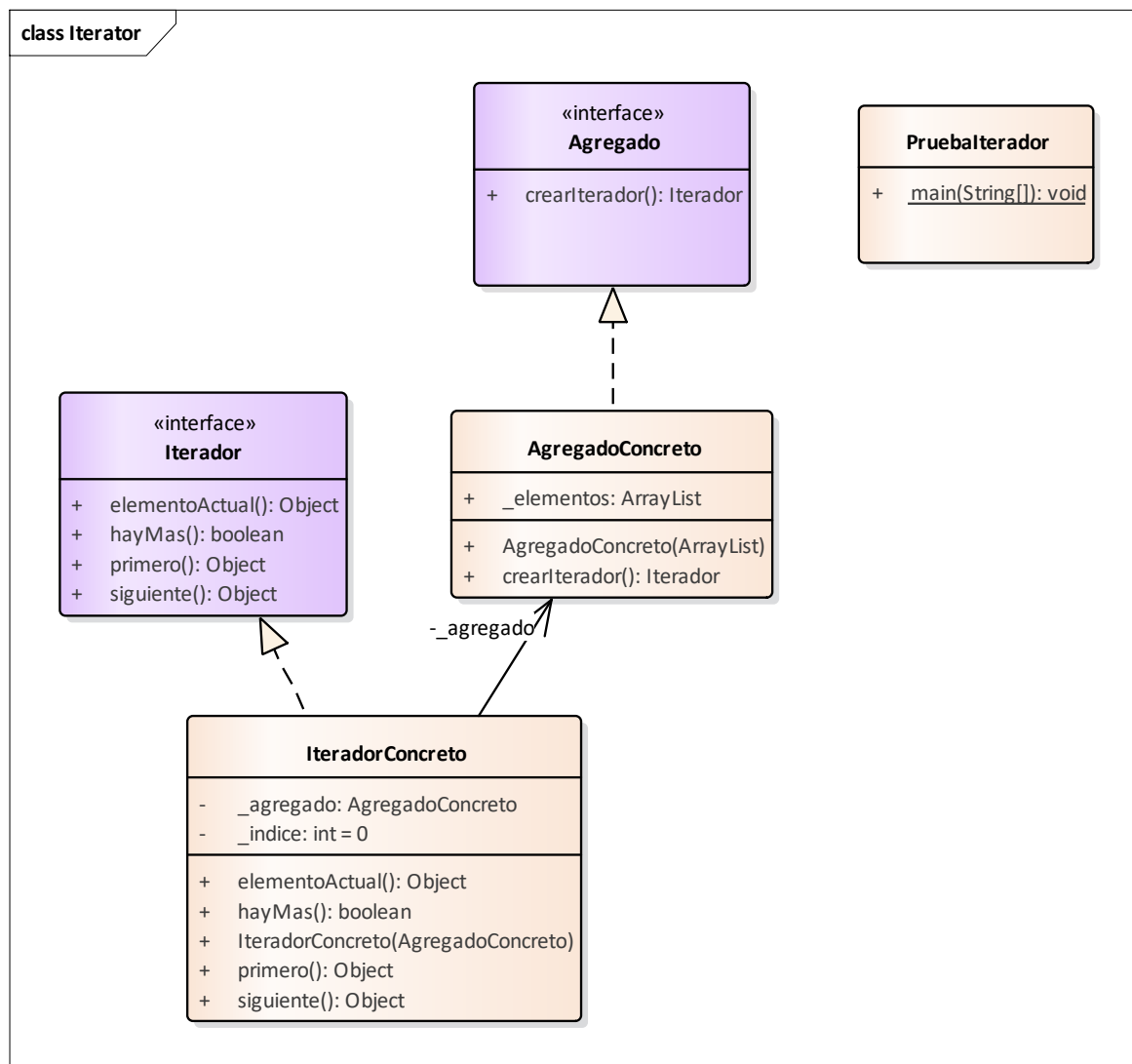
## Patrón Bridge para la búsqueda de productos de biblioteca según diferentes criterios

Se utilizó el patrón **Bridge** para poder realizar la búsqueda de los productos de la biblioteca según diversos criterios, de manera que ambas partes pudieran variar independientemente, por lo que conseguimos desacoplar la implementación interna de su abstracción funcional.



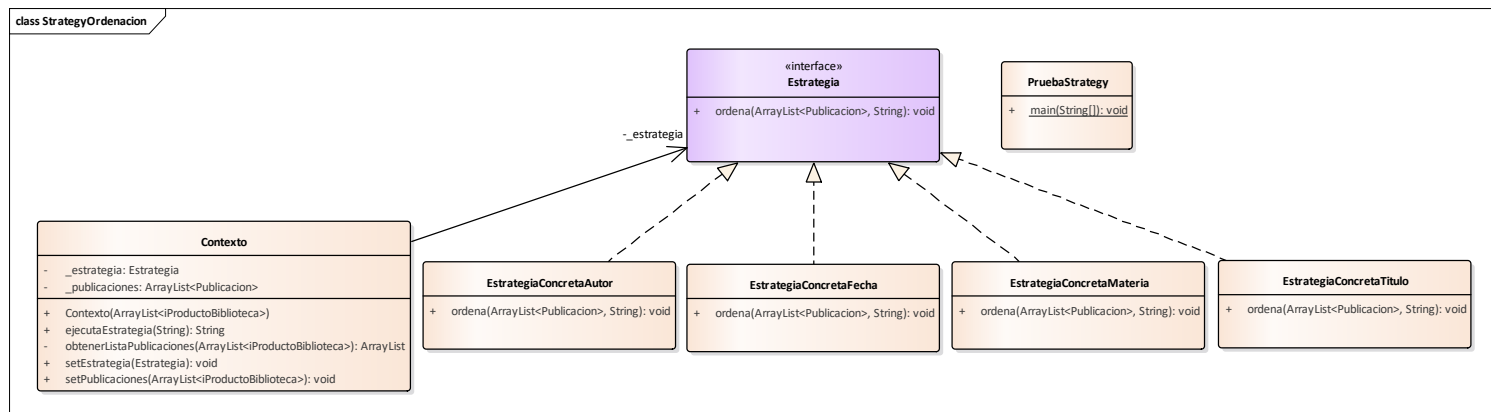
## Patrón Iterator para recorrer listas de objetos

Creación de este patrón para recorrer lista de objetos, en nuestro caso para los arraylist típicos de Publicaciones y Productos de cafetería. Este patrón no tiene más misterios, simplemente cumple las especificaciones del mismo. Mostramos a continuación su diagrama:



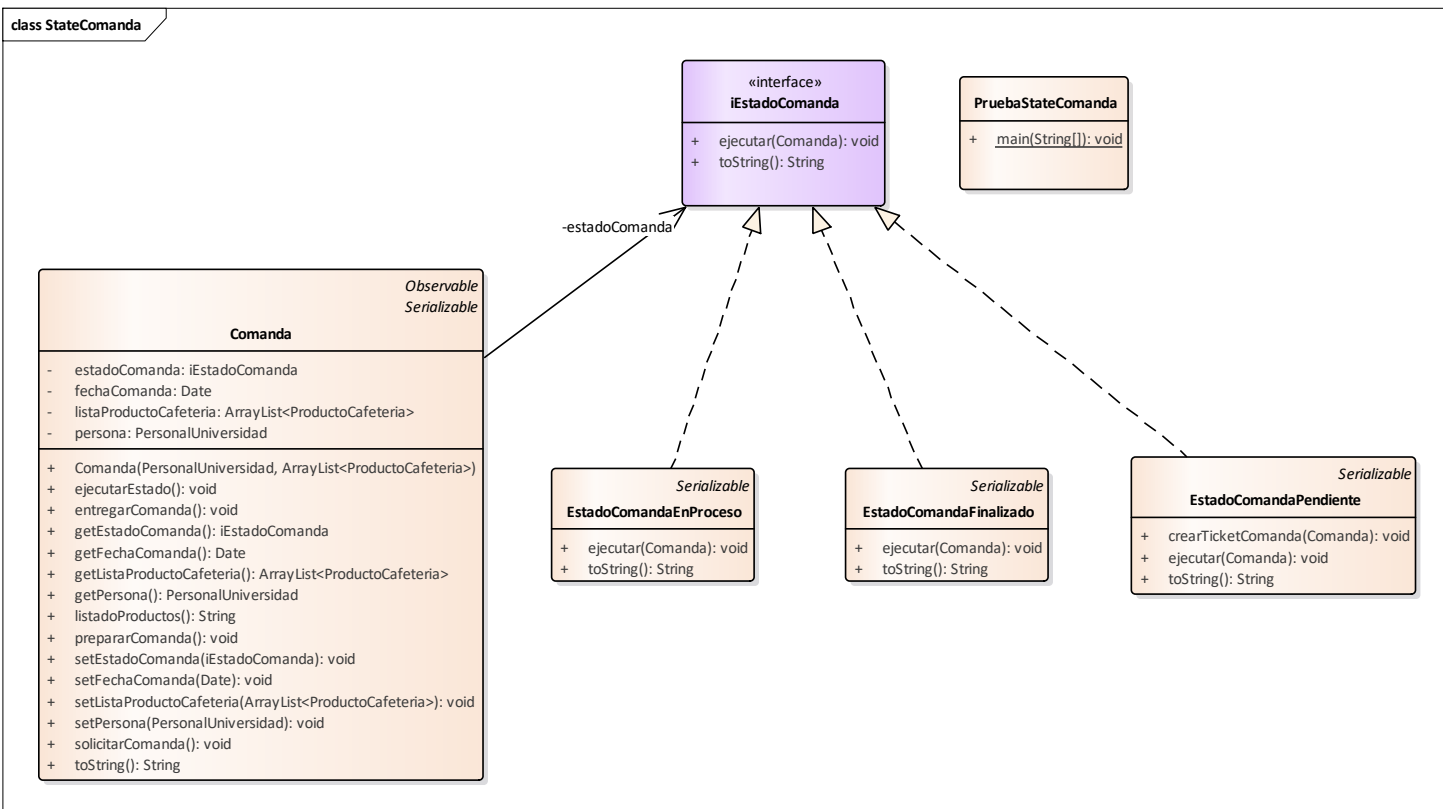
## Patrón Strategy para las estrategias de ordenación

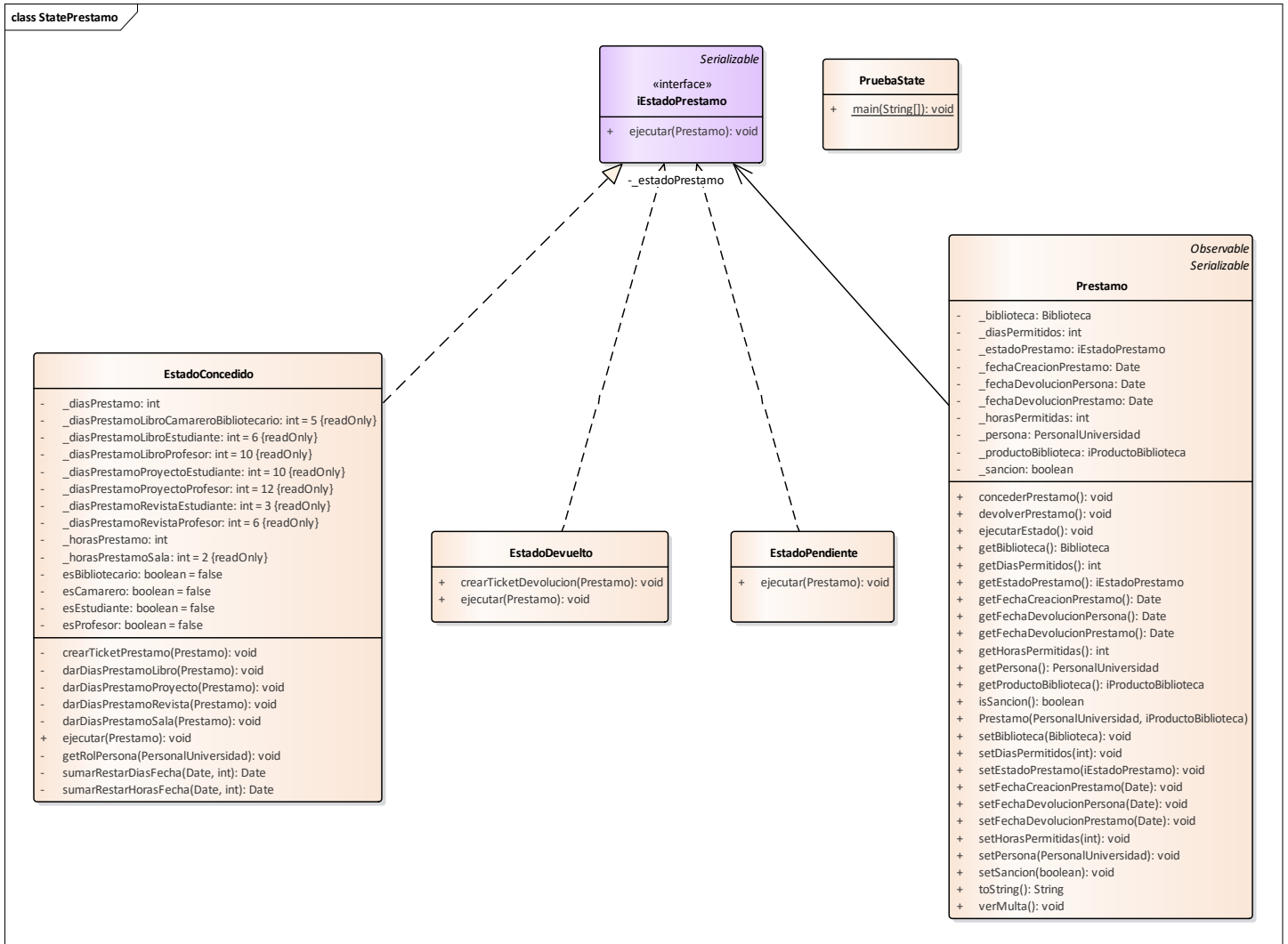
Para poder variar el algoritmo de ordenación se utilizó el patrón **Strategy**. Si diagrama es el que se muestra a continuación:



## Patrón State para los estados de las comandas de la cafetería y los préstamos de la biblioteca

Con el patrón State conseguimos que el objeto Comanda y Préstamo se comporten de manera distinta dependiendo de su estado distinto, por ejemplo si la comanda está en estado pendiente, la información que se obtendrá de ella será la representación de dicho estado, así como que podrá ser utilizado por otras clases al poseer ese estado. En cambio, cuando pase a estado en proceso, se simulará el proceso de cocinar la comida que durará 6 segundos, y finalmente, cuando esté en estado finalizado, se imprimirá la representación de dicho estado, así como que se avisará a la persona que solicitó la comanda para que se entere de que ya puede empezar a comer (de hecho, esta última parte se consigue a su vez con el patrón **Observer** que explicaremos más adelante). Veamos finalmente el diagrama del patrón State:





Patrón **Observer** para la finalización de las comandas y para la concesión de los préstamos que notifiquen al PersonalUniversidad concreto que lo pidió

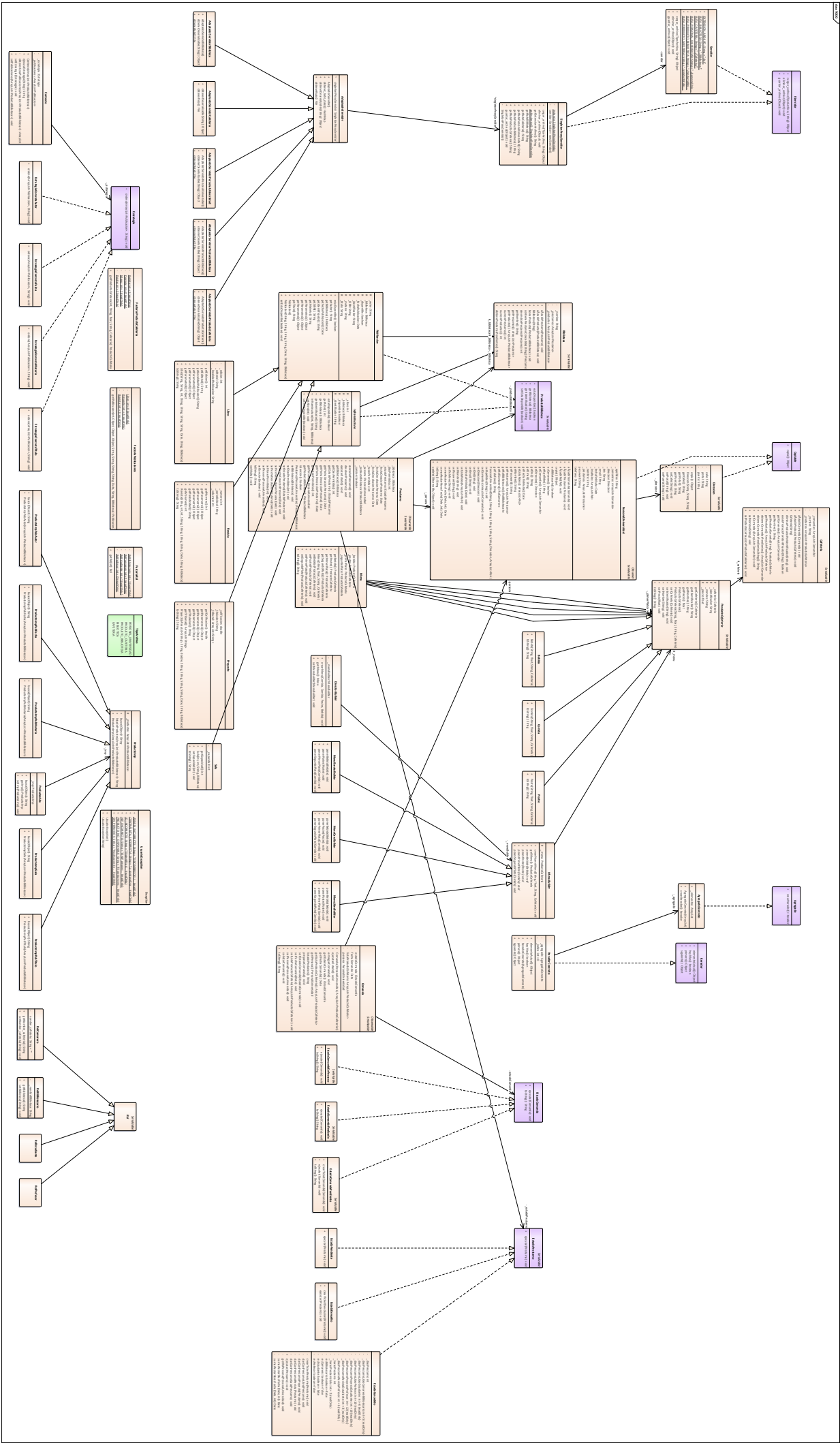
Utilizamos las utilidades que java nos ofrece para implementar el patrón observer, en concreto, implementamos la interfaz Observer en la clase PersonalUniversidad, y extendemos de Observable en la clase Comanda y Prestamos puesto que serán los encargados de realizar la notificación que lance la ejecución del método update de los observers, que en nuestro caso es el PersonalUniversidad concreto que realizara la Comanda o Préstamo.

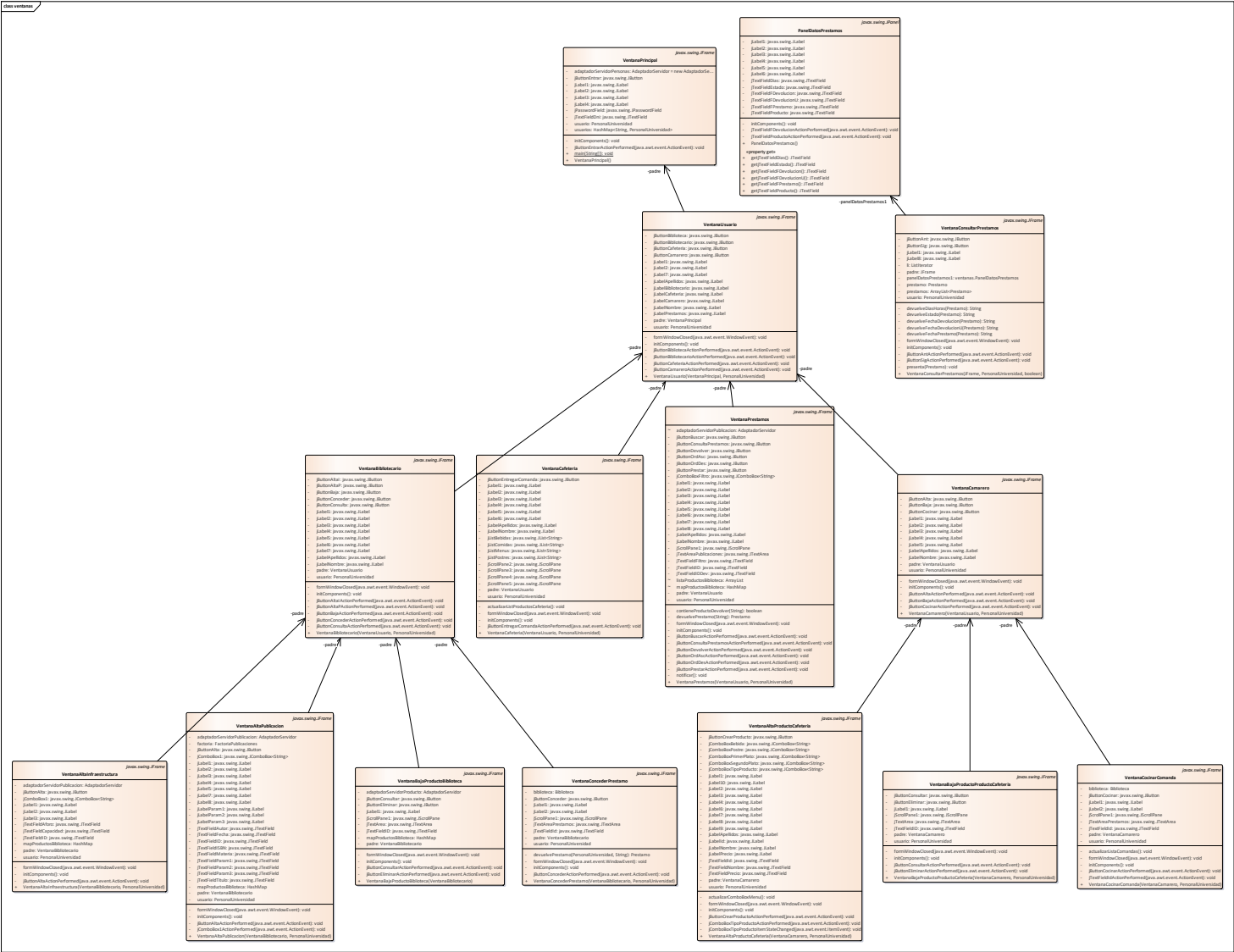
### Diagrama Final de la lógica de de la aplicación y de interfaz gráfica

Incorporamos en la siguiente página el diagrama generado por Enterprise Architect para la lógica de nuestra aplicación, lo cual son simplemente los 12 patrones anteriormente explicados.

Además, también mostramos el diagrama final que generan las interfaces gráficas creadas, de manera que se pueda ver fácilmente sus relaciones, las cual explican casi por sí solas la interacción que realizará el usuario con la aplicación.

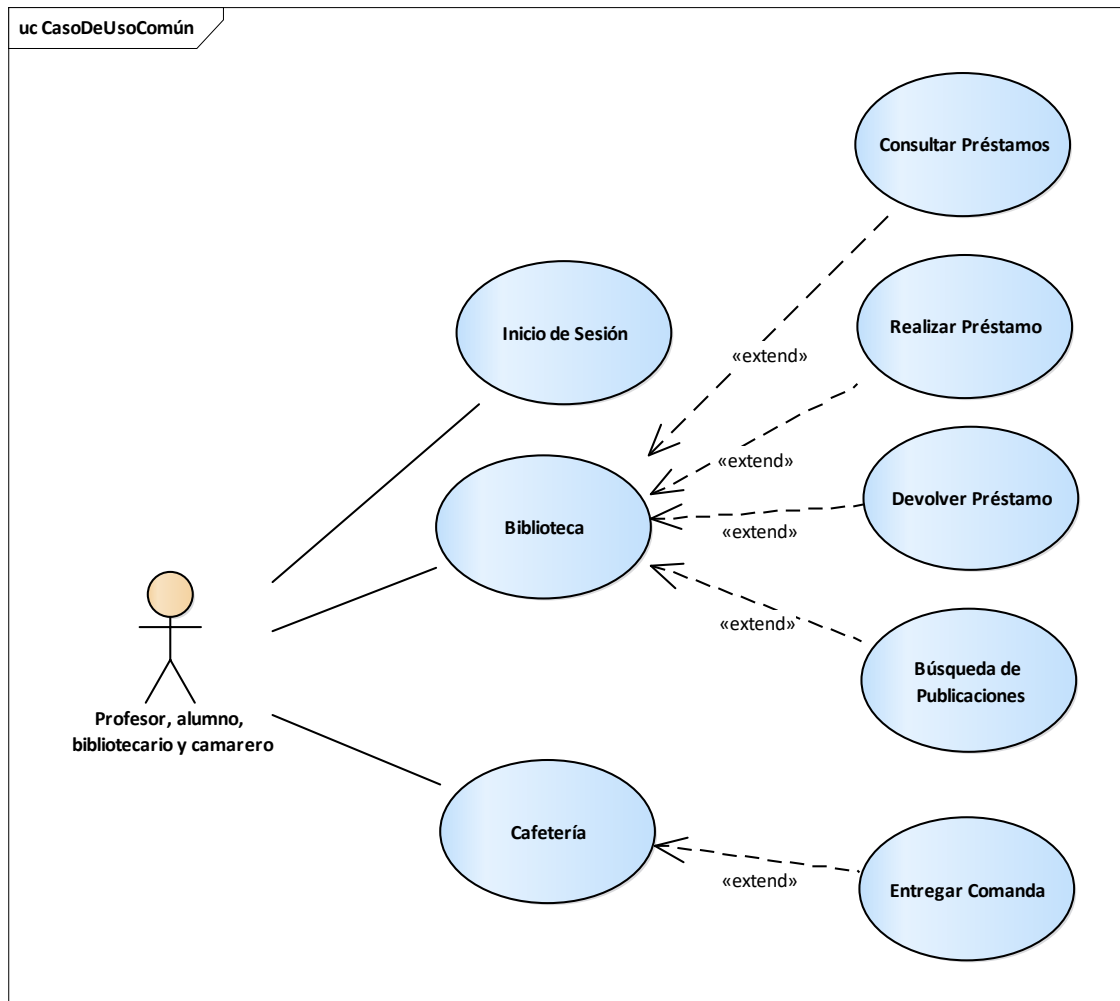






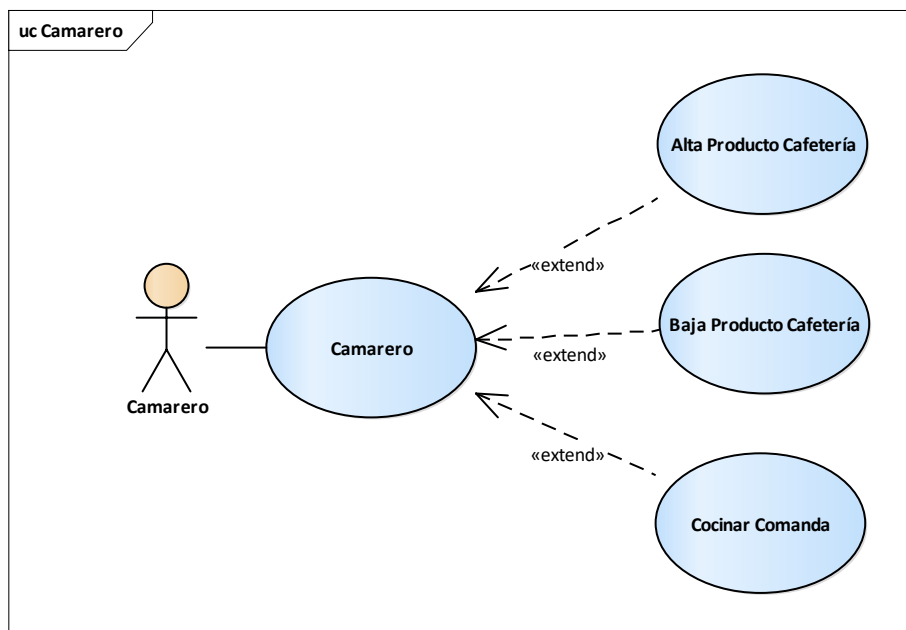
## CASOS DE USO

Empezamos por el caso de uso que todo usuario del sistema puede realizar:



Como vemos, los cuatro tipos de usuarios pueden realizar inicio de sesión, así como las actividades propias de una biblioteca y una cafetería.

Ahora bien, el usuario de tipo **Camarero** posee las siguientes peculiaridades:



Y en cuanto al **Bibliotecario** algo muy similar también:

