Patrones de Diseño: Ejercicios.

Patrones de Creación

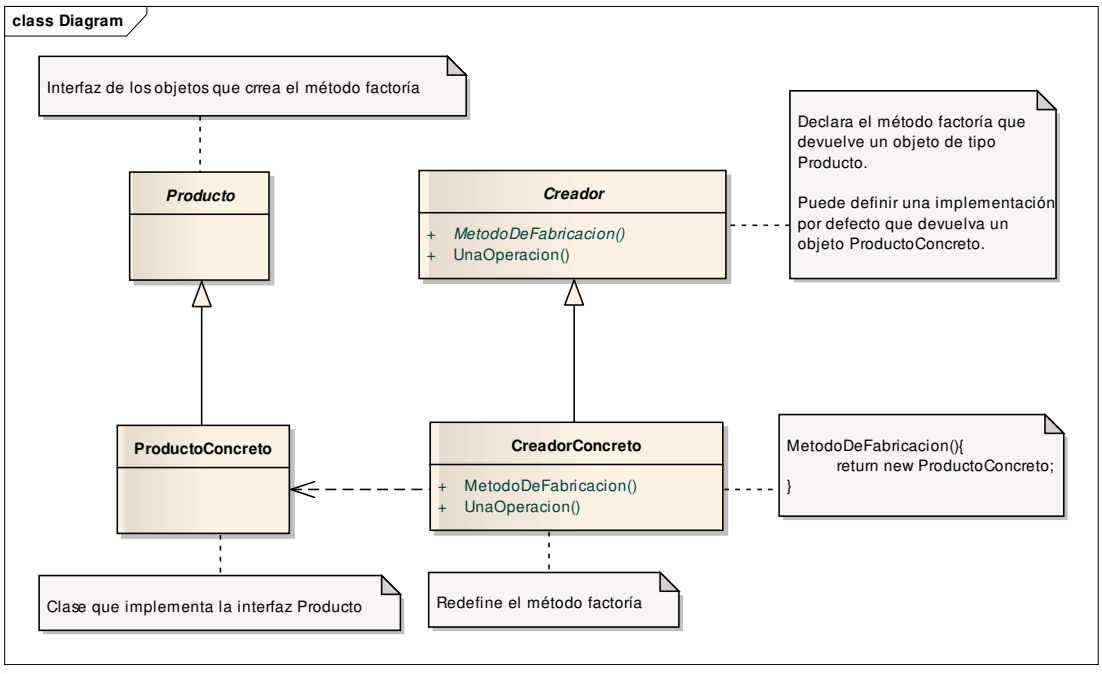
Ejercicio 1

- En una empresa de juguetes se están planteando construir tres clases de juguetes: coches, aviones y naves espaciales con acciones de animación asociadas a su encendido y apagado. Estas acciones serán diferentes para cada juguete. Todos los juguetes tendrán un nombre. ¿Qué patrón aplicarías para facilitar la creación de los distintos tipos de juguetes?

Ejercicio 1. Solución

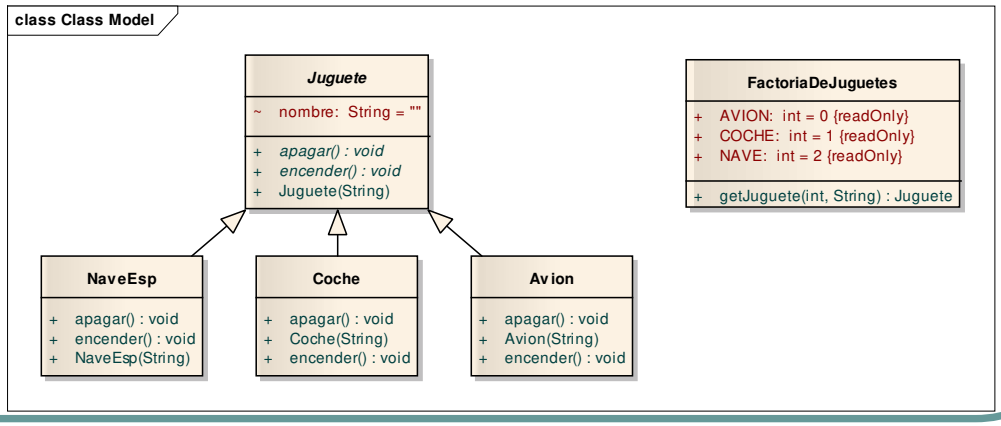
- El patrón que mejor se adapta a este ejercicio es el Factory Method.
  - *Define una interfaz para crear un objeto pero deja que sean las subclases quienes decidan que clase instanciar. Permite que una clase delegue en sus subclases la creación del objeto.*
  - *Nota: También se podría aplicar el patrón Builder.*

Ejercicio 1. Solución



Ejercicio 1. Solución

- Identificamos a continuación los elementos del patrón:
  - Producto: Juguete.
  - ProductoConcreto: NaveEsp, Coche, Avion.
  - Creador: FactoriaDeJuguetes.



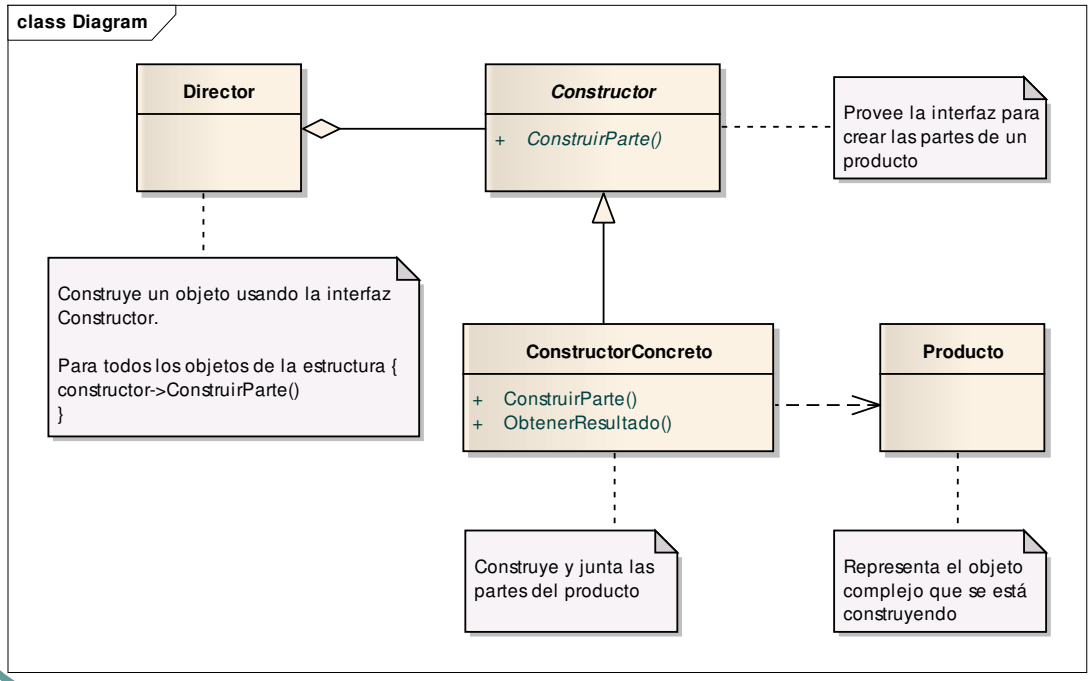
Ejercicio 2

- Un estudio de arquitectura pretende realizar una aplicación que le ayude en el diseño de sus inmuebles. En principio el estudio diseña dos tipos de inmuebles: apartamentos y chalets, pero la aplicación debe poder incorporar fácilmente el diseño de otros inmuebles. Cada inmueble se diseña de forma similar, diseñando las habitaciones, el salón, la cocina, los cuartos de baño y la terraza. Todo inmueble tiene una superficie y las habitaciones y los cuartos de baño un nombre. ¿Qué patrón aplicarías para facilitar el diseño de los distintos tipos de inmuebles?

Ejercicio 2. Solución

- El patrón que mejor se adapta a este ejercicio es el Builder.
  - *Separa la construcción de un objeto complejo de su representación, de forma que el mismo proceso de construcción pueda crear diferentes representaciones.*

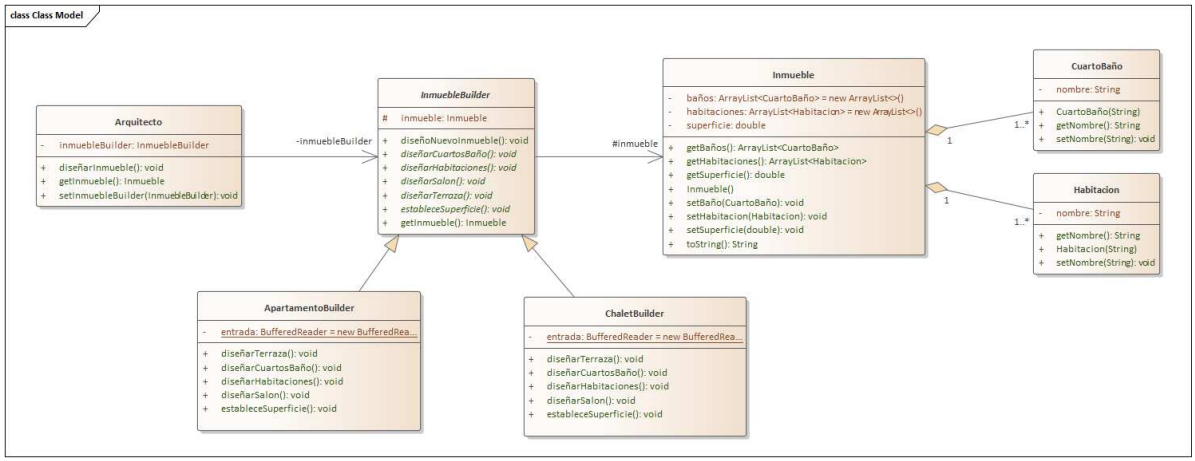
Ejercicio 2. Solución



Ejercicio 2. Solución

- Identificamos a continuación los elementos del patrón Builder:
  - Constructor: InmuebleBuilder.
  - Constructor concreto: ChaletBuilder, ApartamentoBuilder.
  - Producto: Inmueble.
  - Director: Arquitecto.

Ejercicio 2. Solución



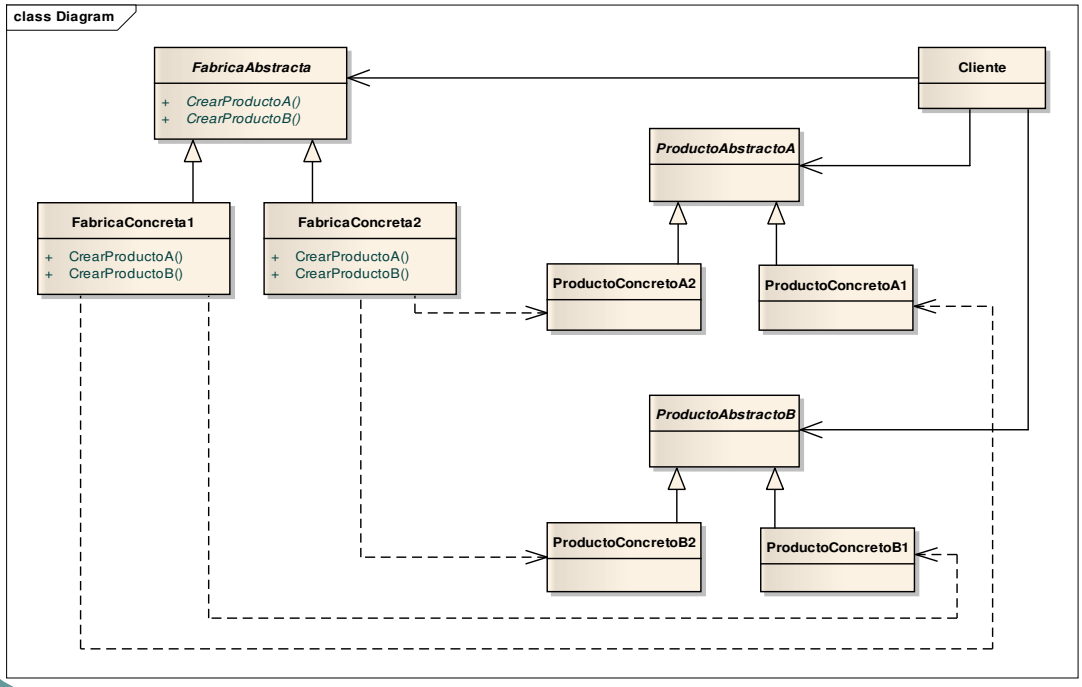
Ejercicio 3

- Se desea realizar una aplicación que simule un restaurante donde se sirven distintos tipos de menús. Los menús están compuestos por primer plato, segundo y postre y dentro de cada uno podremos elegir entre distintos tipos de alimentos. ¿Cómo representarías la estructura de clases para poder crear dos tipos de menús y que patrón utilizarías?

Ejercicio 3. Solución

- El patrón que mejor se adapta a este ejercicio es el Abstract Factory.
  - Proporciona una interfaz para crear familias de objetos relacionados o que dependen entre sí, sin especificar sus clases concretas.

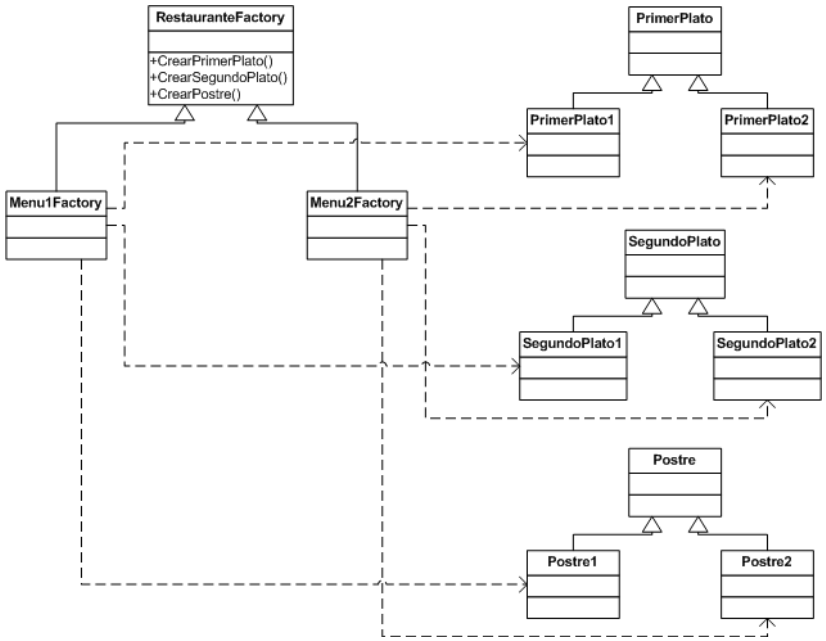
Ejercicio 3. Solución



Ejercicio 2. Solución

- Identificamos a continuación los elementos del patrón Abstract Factory:
  - FabricaAbstracta: RestauranteFactory.
  - FabricaConcreta: Menu1Factory, Menu2Factory.
  - ProductoAbstracto: PrimerPlato, SegundoPlato, Postre.
  - ProductoConcreto: PrimerPlato1, PrimerPlato2, SegundoPlato1, SegundoPlato2, Postre1, Postre2.

Ejercicio 3. Solución



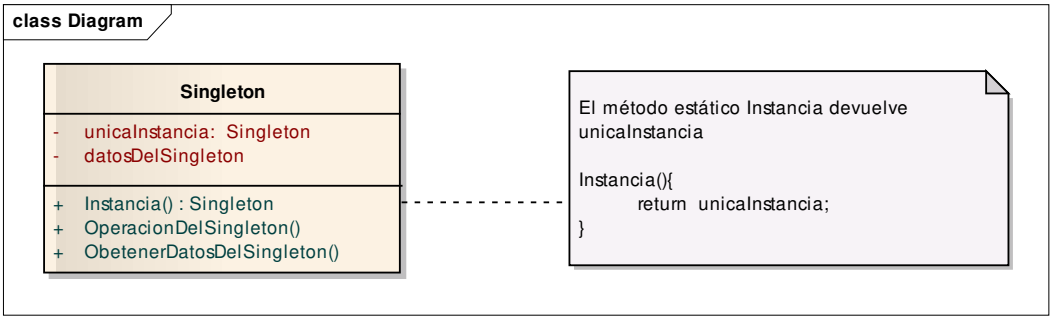
Universidad de Alcalá

Ejercicio 4

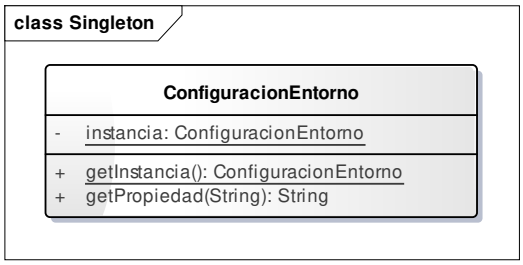
- Tenemos un fichero de configuración cuyos valores pueden ser usados desde cualquier parte de la aplicación. ¿Qué patrón aplicarías para facilitar el acceso a estos valores?

Ejercicio 4. Solución

- El patrón que mejor se adapta a este ejercicio es el Singleton.
  - *Garantiza que una clase solo tenga una instancia, y proporciona un punto de acceso global a ella. Todos los objetos que utilizan una instancia de esa clase usan la misma instancia.*



Ejercicio 4. Solución



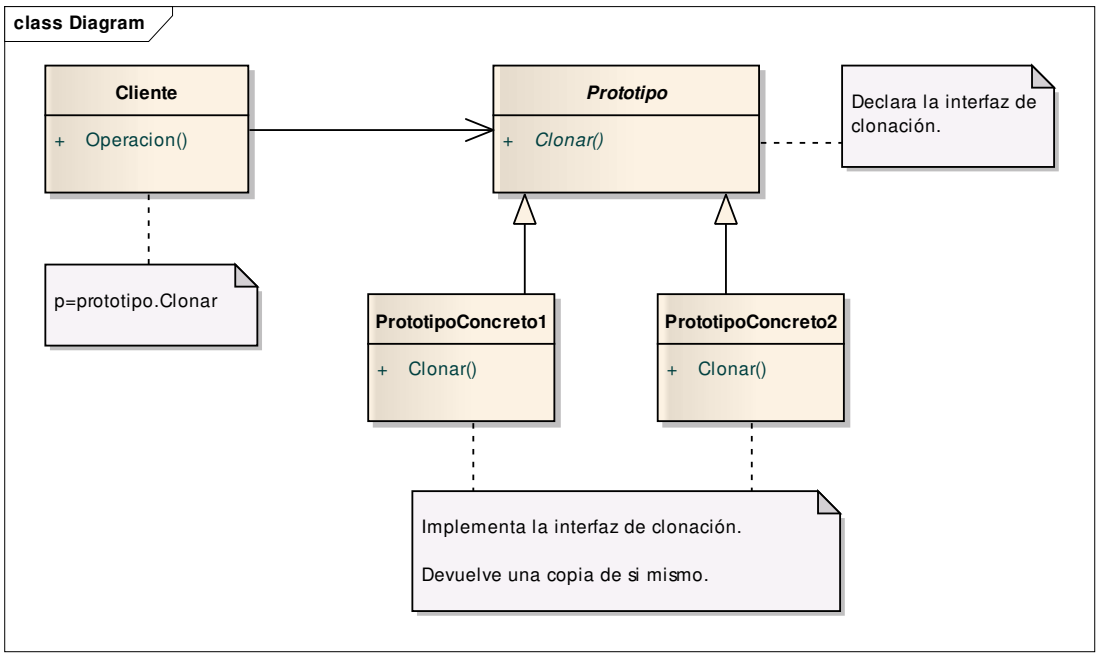
Ejercicio 5

- Hemos de crear una aplicación para la consulta de datos almacenados en una base de datos de histórico. Sabemos que el volumen de datos que se devuelve en cada consulta y el coste del acceso a base de datos son altos. ¿Qué patrón aplicarías para gestionar los distintos tipos de ordenación de los datos devueltos por la consulta?

Ejercicio 5. Solución

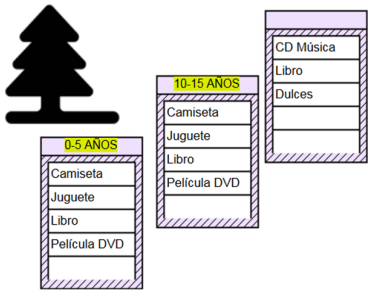
- El patrón que mejor se adapta a este ejercicio es Propotype.
  - *Se realiza una primera lectura de base de datos. En caso de necesitarse una ordenación diferente a la inicial, se clona el resultado de la primera lectura y se ordena. De esta forma evitamos un segundo acceso a base de datos.*

Ejercicio 5. Solución



Ejercicio 6

- Imagina tu sorpresa esta Navidad cuando, en lugar de regalos tradicionales debajo del árbol, encuentra cajoneras con etiquetas y tiradores. Cuando sacas un cajón, obtienes un regalo. El regalo es único, pero coincide con la etiqueta (y categoría) del cajón. Algunas cajoneras están etiquetadas con rangos de edad. Los cajones de estas cajoneras producirán diferentes tipos de regalos, pero coincidirán con el rango de edad.
- ¿Qué patrón o patrones utilizarías?

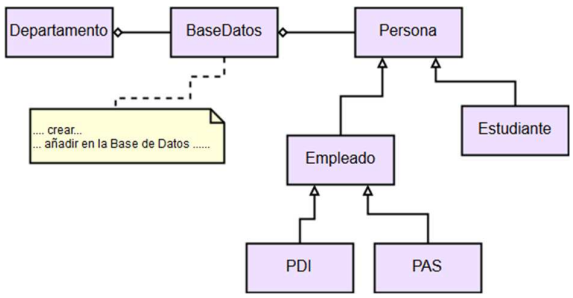


Ejercicio 6. Solución

- Esta situación se representa con dos patrones de diseño: Factory Method y Abstract Factory. Las cajoneras con rangos de edad representan el patrón Abstract Factory porque cada producto está relacionado con una familia de productos. La cajonera con una sola etiqueta representa el patrón de diseño Factory Method.

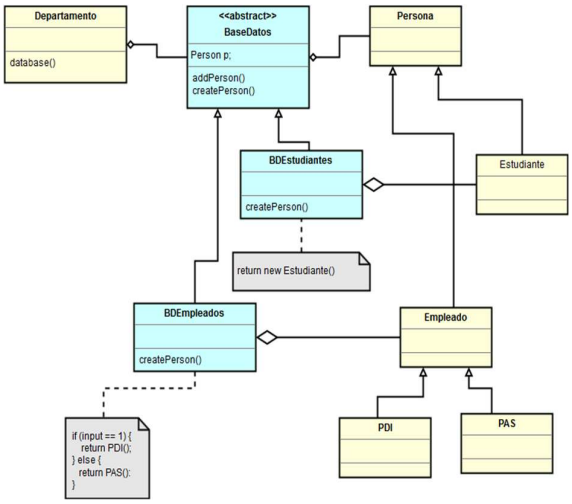
Ejercicio 7

- Un departamento de la Universidad tiene, en su Base de Datos interna, el directorio de los estudiantes y empleados (que pueden ser PDI o PAS) que depende del departamento. En la siguiente imagen se puede observar la estructura de clases que maneja el sistema. Tal y como está ahora, el personal administrativo añade manualmente objetos de tipo Estudiante, PDI y PAS. Modificar la estructura para añadir la creación de estos objetos mediante un patrón de creación.



Ejercicio 7. Solución

- Existen 2 posibles soluciones: con Factory Method (si lo incluimos como parte de la clase “BaseDatos”) o Prototype (si lo incluimos como parte de la clase “Persona”, aunque también utiliza el Factory Method para crear el objeto correspondiente).

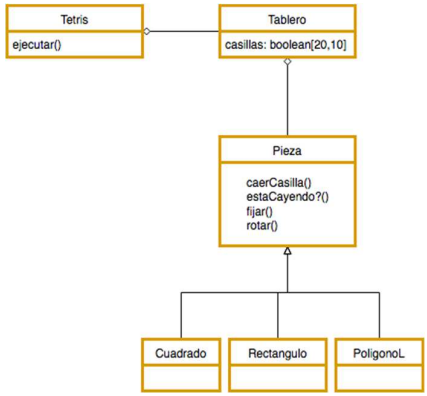


Ejercicio 8

- Imagina que tienes que diseñar el juego del Tetris. Este juego utiliza un tablero de 20x10 casillas (en forma vertical). De la parte superior caen piezas. Las piezas tienen diferentes formas (rectángulo, cuadrado y polígono en forma de L). Las piezas pueden tener varias implementaciones visuales (para este caso, colores negro y azul). El jugador puede rotar las piezas según caen. Las piezas, una vez que tocan el fondo (u otra pieza) se quedan fijas. Solo cae una pieza cada vez (cuando la pieza se queda fija, cae la siguiente).
  1. Diseñar el modelo de clases de este juego.
  2. Aplicar, al modelo anterior, un patrón de creación que permita gestionar la creación de nuevas piezas.

Ejercicio 8. Solución

Diagrama de clases:



Abstract Factory: