**Enunciado:**

Un supermercado internacional con cientos de centros repartidos por 23 países desea una gestión de tickets.

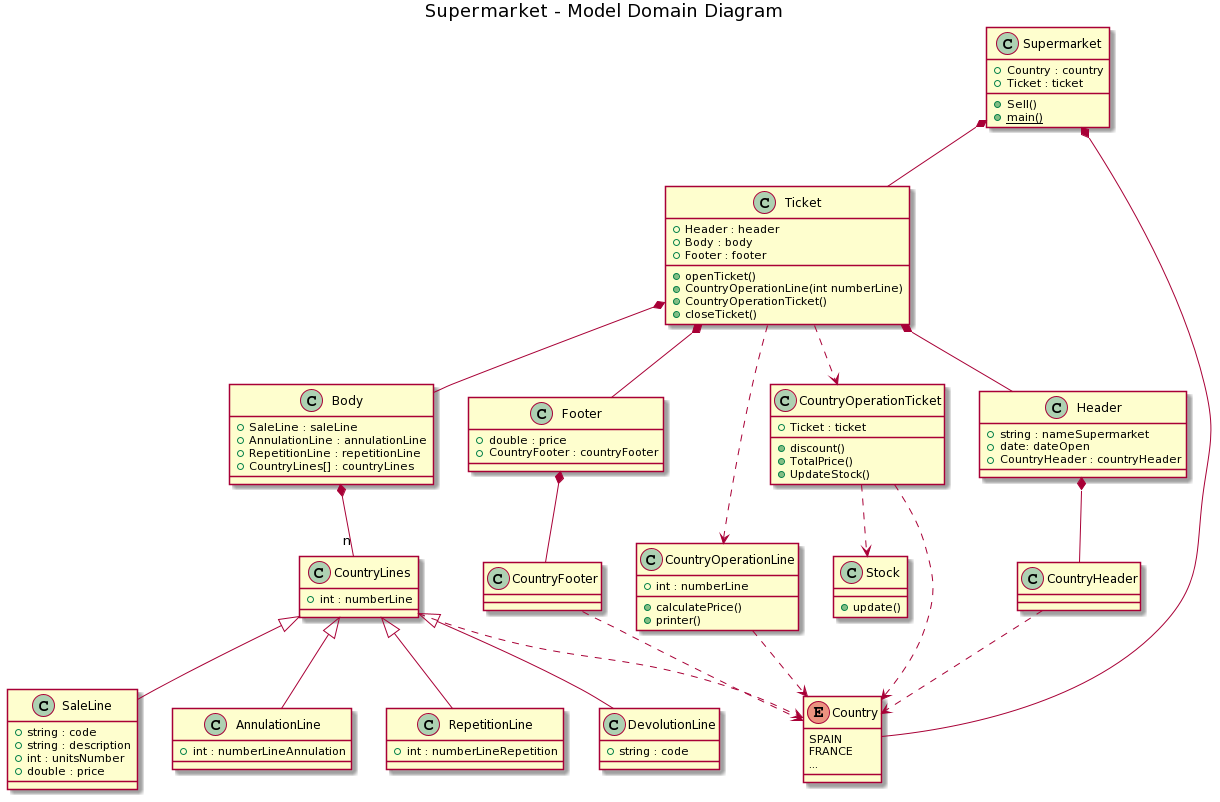
Un ticket consta de:

* **Cabecera**:
  + Distintos campos dependiendo del país.
  + Siempre el nombre el nombre de la cadena y la fecha y hora de comienzo del ticket de compra.
* **Cuerpo**:
  + Una secuencia de líneas. Cada una de las líneas puede ser:
    - **De venta**, que refleja el código, descripción, unidades y precio.
    - **De repetición**, que refleja otra línea exactamente igual a la dada por el número de la secuencia de líneas.
    - **De anulación**, que refleja el número de línea anterior anulada con un saldo negativo igual a la línea que anula.
    - **De devolución**, que refleja el código
    - ….
  + **Dependiendo del país** existirán unos tipos de líneas u otros, aunque venta, repetición y anulación estás presentes en todos. Y, por ejemplo, en España existen devoluciones, pero en Francia no.
* **Pie**:
  + Distintos campos dependiendo del país.
  + Siempre tiene el precio final con impuestos del ticket de compra.

Existen dos tipos de operaciones a realizar: **Considerando el país, se realizarán unas operaciones u otras.**

* **Operación a nivel de línea:** Que se realiza con cada parte del ticket según se introduce en el sistema. Por ejemplo:
  + Calcular precio.
  + Visualizar.
  + Imprimir.
  + Descuentos por línea.
* **Operación a nivel de ticket:** Que se realiza con todas las partes del ticket según se cierra. Por ejemplo:
  + Descuentos globales.
  + Precio global.
  + Modificación del almacén.

**Diagrama del modelo del dominio resultante:**



**Patrones aplicados:**

1. Un **Abstract Factory**, para que en función del país devuelva una familia (Menú, Ticket, OperationLine, OperationTicket) concreta de clases para cada país (Spain, France):

|  |  |  |
| --- | --- | --- |
| Client: Supermarket | AbstractFactory: CountryFactory | AbstracProductA: Menu |
|  | ConcreteFactoryA: SpainFactory | AbstracProductB: Ticket |
|  | ConcreteFactoryB: FranceFactory | AbstracProductC: OperationLine |
|  |  | AbstracProductD: OperationTicket |
|  |  | ConcreteProductAA: SpainMenu |
|  |  | ConcreteProductAB: SpainTicket |
|  |  | ConcreteProductAC: SpainOperationLine |
|  |  | ConcreteProductAD: SpainOperationTicket |
|  |  | ConcreteProductBA: FranceMenu |
|  |  | ConcreteProductBB: FranceTicket |
|  |  | ConcreteProductBC: FranceOperationLine |
|  |  | ConcreteProductBD: FranceOperationTicket |

1. Un **Command** para conseguir un menú las opciones dinámicas. Convierte cada opción de menú en un objeto.

* Client: Supermarket
* Invoker: Menu
* Command: CommandMenu
* ConcreteCommandA: MenuLineSale
* ConcreteCommandB: MenuLineAnnulation
* ConcreteCommandC: MenuLineDevolution
* ConcreteCommandD: MenuLineRepetition
* Receiver: Ticket

1. Un **TemplateMethod** para la construcción de las opciones del menú de cada país.

* Client: Supermarket
* AbstractClass: Menu
* ConcreteClassX: SpainMenu
* ConcreteClassY: FranceMenu

1. Un **Command** para conseguir las operaciones de línea de cada país. Convierte cada operación de línea en un objeto.

* Client: Supermarket
* Invoker: Operation
* Command: CommandOperation
* ConcreteCommandA: CommandOperationLinePrice
* ConcreteCommandB: CommandOperationLineDisplay
* ConcreteCommandC: CommandOperationLinePrinter
* ConcreteCommandD: CommandOperationLineDiscount
* Receiver: Ticket

1. Un **TemplateMethod** para la construcción de las opciones de línea de cada país.

* Client: Supermarket
* AbstractClass: Operation
* ConcreteClassX: SpainOperationLine
* ConcreteClassY: FranceOperationLine

1. Un **Composite** para las operaciones de línea de los tickets se manejen como si fueran una sola.

* Client: Ticket
* Component: CommandOperation
* Leaf: CommandOperationLinePrice, CommandOperationLineDisplay, CommandOperationLinePrinter, CommandOperationLineDiscount
* Composite: CommandOperationLineComposite

1. Un **Command** para conseguir las operaciones globales de ticket de cada país. Convierte cada operación global de ticket en un objeto.

* Client: Supermarket
* Invoker: Operation
* Command: CommandOperation
* ConcreteCommandA: CommandOperationTicketDiscount
* ConcreteCommandB: CommandOperationTicketPrice
* ConcreteCommandC: CommandOperationTicketStock
* Receiver: Ticket

1. Un **TemplateMethod** para la construcción de las opciones globales de ticket de cada país.

* Client: Supermarket
* AbstractClass: Operation
* ConcreteClassX: SpainOperationTicket
* ConcreteClassY: FranceOperationTicket

1. Un **Composite** para las operaciones globales de los tickets se manejen como si fueran una sola.

* Client: Ticket
* Component: CommandOperation
* Leaf: CommandOperationTicketDiscount, CommandOperationTicketPrice, CommandOperationTicketStock
* Composite: CommandOperationGlobalComposite

1. Un **Strategy** para la inyección de las operaciones concretas en el ticket.

* Client: supermarket
* Context: Ticket
* Strategy: CommandOperation
* ConcreteStrategyA: CommandOperationLine
* ConcreteStrategyB: CommandOperationTicket

1. Un **Builder** para crear el ticket dependiendo del país.

* Director: CountryFactory
* Builder: TicketBuilder
* ConcreteBuilderA: SpanishTicketBuilder
* ConcreteBuilderB: FrenchTicketBuilder

**Diagrama de clases resultante:** Debido a la cantidad de clases se ha partido el diagrama en 2 partes (Ticket+menú y Operaciones)

