Laboratorio de Computación II

TP 4

Temática: Sistema de fábrica de productos recreativos para gatos

Alumno: Marcos Seghesio

División: 2 D

Resumen del proyecto

ACLARACION IMPORTANTE: EL SCRIPT DE LA BASE DE DATOS SE ENCUENTRA EN LA CARPETA DEL PROYECTO JUNTO A ESTE ARCHIVO PDF

El proyecto está pensado para ser un sistema donde pueda ingresarse los distintos insumos necesarios para la fabricación de productos a un inventario de insumos. Posteriormente puede darse de alta los productos requeridos, utilizándose los insumos necesarios del stock. En caso de no tener insumos suficientes, el mismo sistema indicará que insumos faltan y dará la posibilidad de realizar un pedido. Una vez ingresados productos exitosamente estos estarán disponibles en la línea de producción para que se ejecuten los procesos específicos de fabricación. Cuando los productos se encuentren terminados, se los puede separar a un almacén de productos completos.

Acontecimientos generales del sistema

1. Usuario ingresa un insumo al stock de Fábrica.
2. Usuario ingresa un producto a línea de producción.
3. Usuario ejecuta un proceso de línea de producción.
4. Usuario despacha un producto a stock de productos terminados.
5. Usuario crea un reporte de fábrica

1 - Usuario ingresa un insumo al stock de Fábrica

El usuario mediante un formulario en WinForm ingresa todos los campos necesarios para ingresar un insumo válido. Una vez cargado correctamente el insumo con todos los campos correspondientes se agrega al stock de insumos un nuevo elemento. En el caso que un insumo sea de las mismas características que uno ya existente, se suman sus cantidades colocándose la fecha de ingreso del insumo más reciente.

2 - Usuario ingresa un producto a línea de producción

El usuario ingresa mediante el formulario un producto, teniendo todas las posibilidades para elegir las características del producto. Una vez accionado el botón de Ingresar Producto el sistema evaluará contar con todos los insumos en cuanto a tipo y cantidad requerida.

- En el caso de no haber stock, el mismo algoritmo proveerá una lista con el tipo y cantidad exacta de insumos faltantes, brindándose la posibilidad de realizar un pedido de insumos, repitiéndose el acontecimiento del caso 1 para todo el listado de insumos faltantes.

- En el caso de tener stock se descontarán la cantidad de insumos que demanda el producto y se agregará el mismo a la línea de producción de productos, con el estado de “Planificado”.

3 - Usuario ejecute un proceso de línea de producción

Una vez que exista al menos un producto en línea de producción, desde la pestaña de línea de producción del formulario de Fábrica, aparecerán lo/los mismos con su estado actual. El proceso de producción para los dos tipos de productos existentes es el siguiente:

Torre: Planificado -> MaderasLijadas -> Alfombrado -> AdicionalesAgregados -> Completo

Estante: Planificado -> MaderasLijadas -> Barnizado -> Alfombrado -> Completo

Procesos de fabricación

1. Lijar Maderas: todo producto que esté en estado de Planificado pasará a tener el estado de MaderasLijadas.
2. Barnizar: proceso solo aplicado a productos del tipo Estante que esté con estado de MaderasLijadas, los cuales modificarán su estado a Barnizado
3. Alfombrar: proceso aplicado a estantes con estado Barnizado y Torres con estado MaderasLijadas, los cuales modificarán s u estado a Alfombrado
4. Agregar Yute: este proceso solo se aplica a productos del tipo Torre en estado Alfombrado que además tengan en el campo metrosYute un valor mayor a cero. Este proceso modifica el campo de yuteInstalado a true y cambia el estado del producto a AdicionalesInstalados
5. Ensamblar: este proceso se aplica a los productos en tres situaciones distintas:
    a. El Producto es Estante y se encuentra en estado Alfombrado.
    b. El producto es Torre y se encuentra en estado AdicionalesAgregados.
    c. El producto es torre y se encuentra en estado Alfombrado y el campo metrosYute tiene valor cero.

En todas las circunstancias mencionadas, el producto modificará su estado a Completo.

4 - Usuario despacha un producto a stock de productos terminados

Desde la pestaña de línea de producción del formulario de Fabrica el usuario puede seleccionar la opción de despachar los productos completos al stock de productos terminados. Al accionar el botón se eliminarán de la línea de producción

todos los productos que tengan el estado de “Completo” y estos podrán encontrarse en el formulario de productos terminados. En el caso de no haber productos para despachar, la aplicación informará que no se han podido despachar productos.

5 - Usuario crea un reporte para fábrica

Desde el botón de Crear Reporte del menú lateral el usuario puede obtener un informe en pdf con el estado de todos los campos de Insumos y Productos del objeto Fabrica. Este documento se abrirá de forma emergente una vez creado el informe.

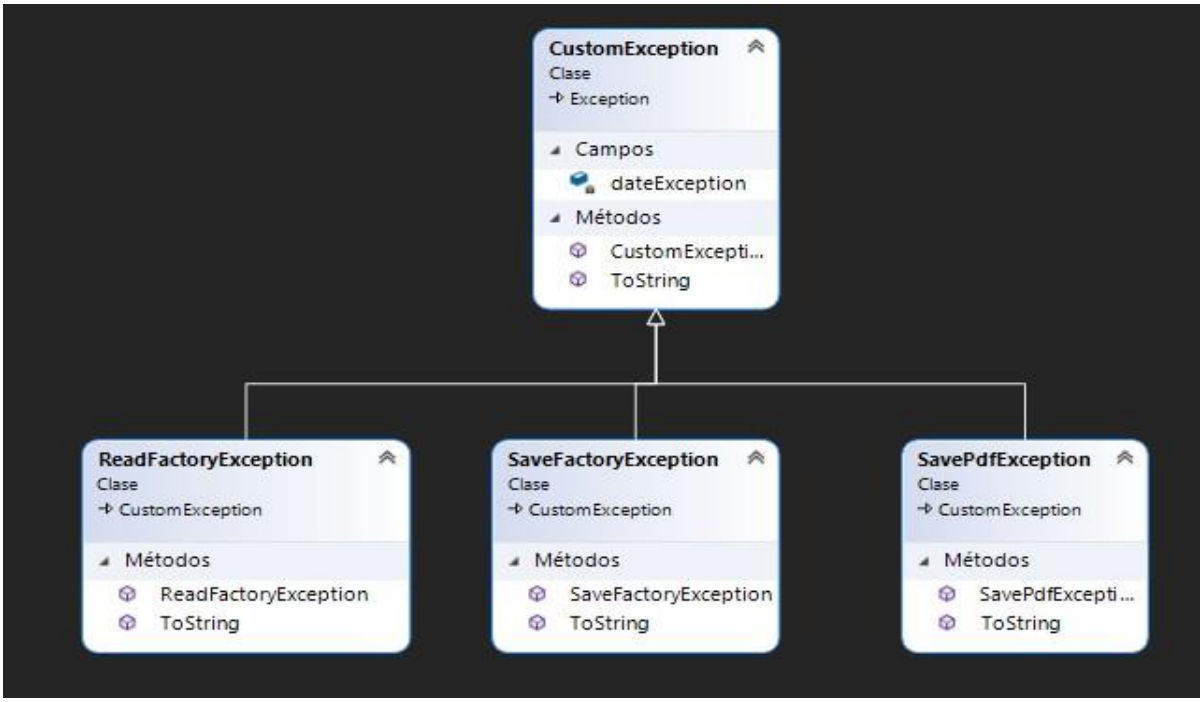
Temas propios de las clases 15-19

Clase 15 Excepciones (Entidades.Exceptions)

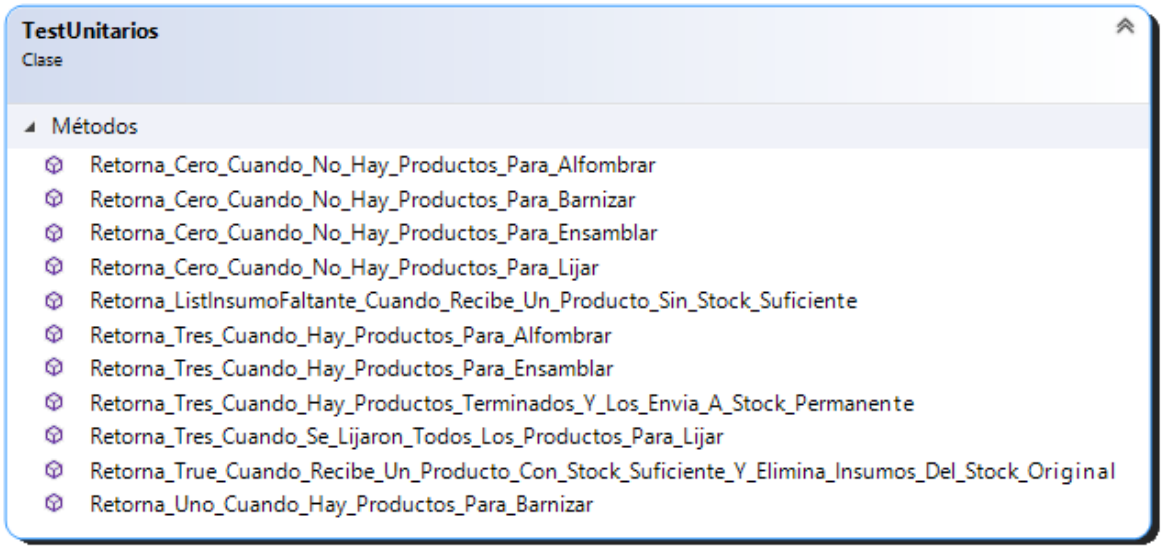
La solución cuenta en su proyecto de Entidades una carpeta con una carpeta llamada Exceptions la cual contiene una clase base denominada CustomException. Esta clase como único campo tiene dateException del tipo DateTime, lo cual está pensada para poder guardar el horario en el que se produjo una excepción en la aplicación. Las clases derivadas de esta son las únicas que pueden instanciarse y vas a lanzarse en los siguientes casos:

- ReadFactoryException: problemas en la ruta de los archivos XML de donde se extraen los datos de los atributos de la fábrica, o si los archivos están corruptos.
- SaveFactoryException: problemas en la ruta donde se intentarán almacenar los archivos XML de los atributos de la fábrica.

- SavePdfException: problemas al crear reportes en pdf de la fábrica. Suele lanzarse esta excepción especialmente cuando se intenta guardar un nuevo pdf con el mismo nombre y ruta teniendo el archivo abierto.



Clase 16 Test Unitarios (proyecto TestUnitarios)



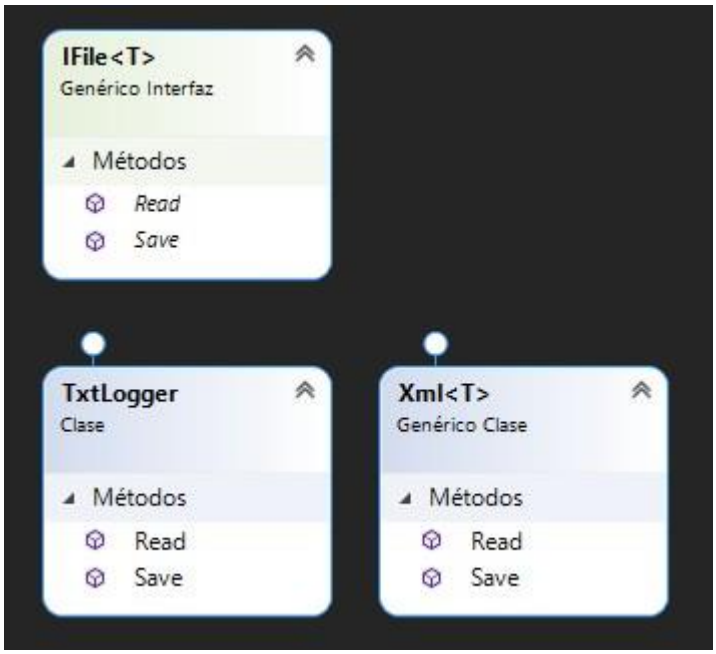
Los Test Unitarios están enfocados en evaluar las distintas fases del proceso de dar de alta Insumos y Productos, y que los procesos de Fabricación se ejecuten en los productos aptos.

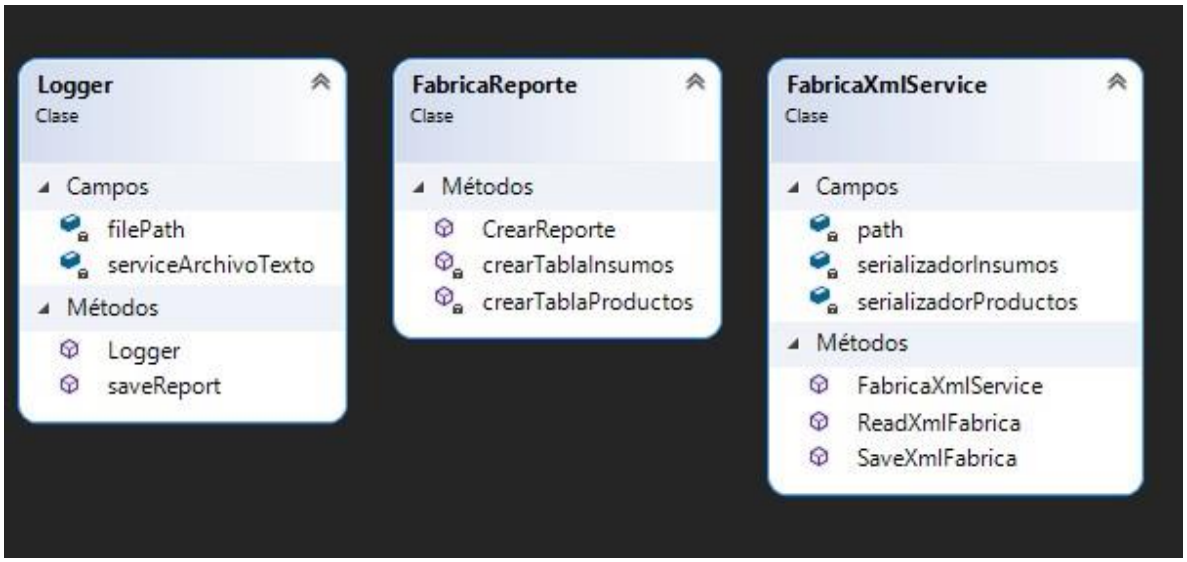
- Evaluar que haya suficiente stock de Insumos antes de dar por agregado un producto a línea de producción. Y que en caso de que intencionalmente no se haya dado el stock necesario, que retorne los insumos correctos como faltantes.
- Evaluar cada fase del proceso de fabricación y que, en caso de no haber productos aptos para ejecutar el proceso, que retorne 0 la cantidad de productos modificados.

Clase 17 Tipos genéricos y Clase 18 Interfaces

Los Tipos genéricos están presentes en el proyecto File en las Interfaz IFile<T> la cual serán implementada tanto para manipular archivos de texto como archivos XML. Sus correspondientes implementaciones que serán desarrollados en los apartados de las clases 18 y 19.

Clase 19 Archivos (Proyecto Files, Entidades.Logger, Entidades.Reportes y Entidades.Services)





En este proyecto se manipulan archivos en tres circunstancias:

- Creación del reporte de fábrica en PDF: con la clase FabricaReporte se crea el contenido del reporte para posteriormente crear el PDF. Con la clase FileStream se crea el archivo y con las clases derivadas del paquete iTextSharp se agregan los elementos del reporte.
- Cargar o guardar datos de la fábrica: los atributos de stockInsumos, lineaProduccion y stockProductosTerminados pueden ser cargados de un archivo XML desde el formulario. Para lograr esto la clase FabricaXmlService tiene atributos como serializadorInsumos y serializadorProductos los cuales implementan la interfaz genérica IFile<T> pudiendo des serializar y serializar objetos del tipo List<Producto> y List<Insumo>.
- Uso de la clase Logger: esta clase se encargará de guardar fecha, hora y descripción general de las excepciones surgidas al leer y modificar archivos en la aplicación. Este archivo puede consultarse desde el botón de bitácora de errores del menú principal, el cual creará un formulario con la información del documento. La lectura y escritura del logger se realiza mediante la clase TxtLogger la cual implemente la interfaz IFile<T>, adoptando el comodín T el tipo de dato string.

Correcciones realizadas al TP3 y algunas aclaraciones

- Mejorada la interfaz gráfica en la vista de la línea de producción haciéndola más intuitiva al momento de llevar adelante un proceso de producción.

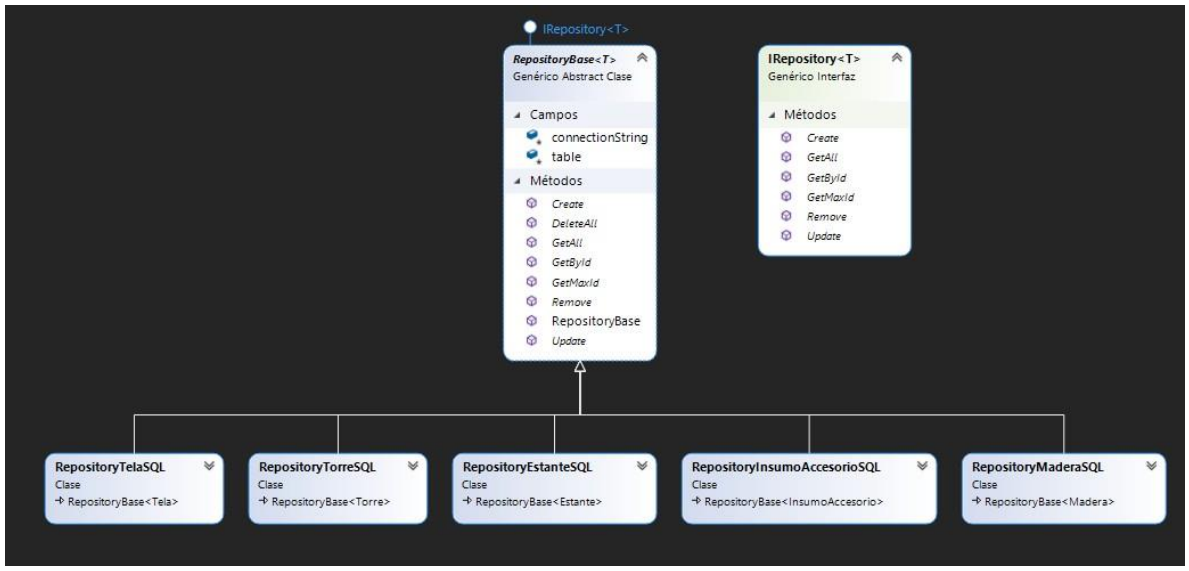
- Corregida excepción en el test de Consola.
- Creación de métodos adicionales en los test unitarios, para evitar la redundancia en el código que había en los mismos.
- Si bien se incorporó base de datos, sigue existiendo la posibilidad de serializar y desserializar en Xml, teniendo en cuenta que al serializar se toma la información de la base de datos y se persiste en el archivo, y en el proceso inverso se carga la información del Xml en la base de datos.

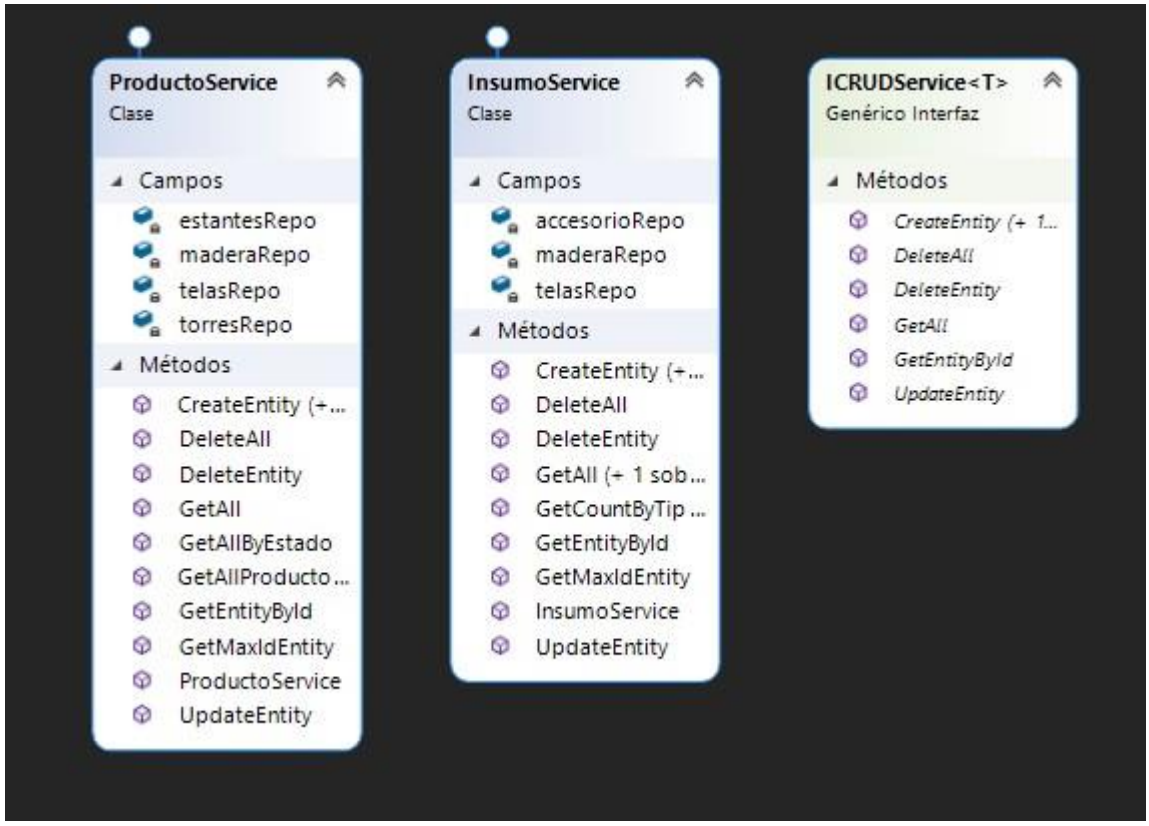
TP4

Clase 21-22 SQL y Base de Datos (Entidades.Repositories y Entidades.Services)

En la entrega del TP3 se utilizó una Colección del tipo List para almacenar los datos de Insumo y Producto en la clase Fabrica, dejando la persistencia de la información en archivos Xml. Para esta entrega se creo una base de datos en SQLServer para almacenar la información en distintas tablas que representan los objetos Producto e Insumo. Se creo la interfaz Genérica IRepository que establece los métodos básicos para la conexión a base de datos. La clase abstracta RepositoryBase es quien implementa esta interfaz y le ofrece la posibilidad a las distintas clases derivadas de heredar de ella (RepositoryEstanteSQL, RepositoryMaderaSQl, entre ellas).

Debido a que tanto Producto como Insumo son clases abstractas que se instancian solo por medio de sus derivadas, para simplificar el enlace entre los métodos de la clase de negocio (Fabrica) y los repositorios, se crearon clases de servicio que permiten recibir objetos del tipo Insumo o Producto y comunicarse eficazmente con la base de datos, no teniendo que desarrollar estas tareas desde la clase Fabrica. Por ejemplo, si tenemos un listado de Insumos que puede contener objetos del tipo Madera o Tela, al pasarle este listado a la clase servicio, esta clase es la que determinara a que repositorio debe llamar según corresponda.





Clase 23 y 24 Hilos, Eventos y Delegados (Entidades.Fabrica,Entidades.Reportes, VistaProyectos.FormDetalles)

La clase fábrica cuenta con un evento del tipo Delegado ModificacionFabrica que se encarga de emitir un evento cuando hay las clases de servicio ejecutan una tarea de creación, modificación y/o en la base de datos. El evento CambioRealizado se encuentra dentro del método EmitirEvento, el cual se emite solo si la propiedad LanzarEventos es true.

Se hicieron modificaciones en la clase FabricaReporte para que la creación del pdf se realice en un hilo secundario, el cual es instanciado en la clase de FormPrincipal. Artificialmente se hizo demorar la creación del informe para que un objeto del tipo ProgressBar se vaya actualizando mediante un método manejador que está suscripto al evento ActualizacionInforme del tipo InformeRealizado. Como la creación de un archivo PDF está sujeta a posibles excepciones, se maneja la excepción almacenando el error en el log además de usar el evento EnviarErrorInforme para indicar al formulario que hubo errores en el pdf. Esta excepción se da en este caso al intentar crear un nuevo reporte en pdf teniendo el archivo abierto.

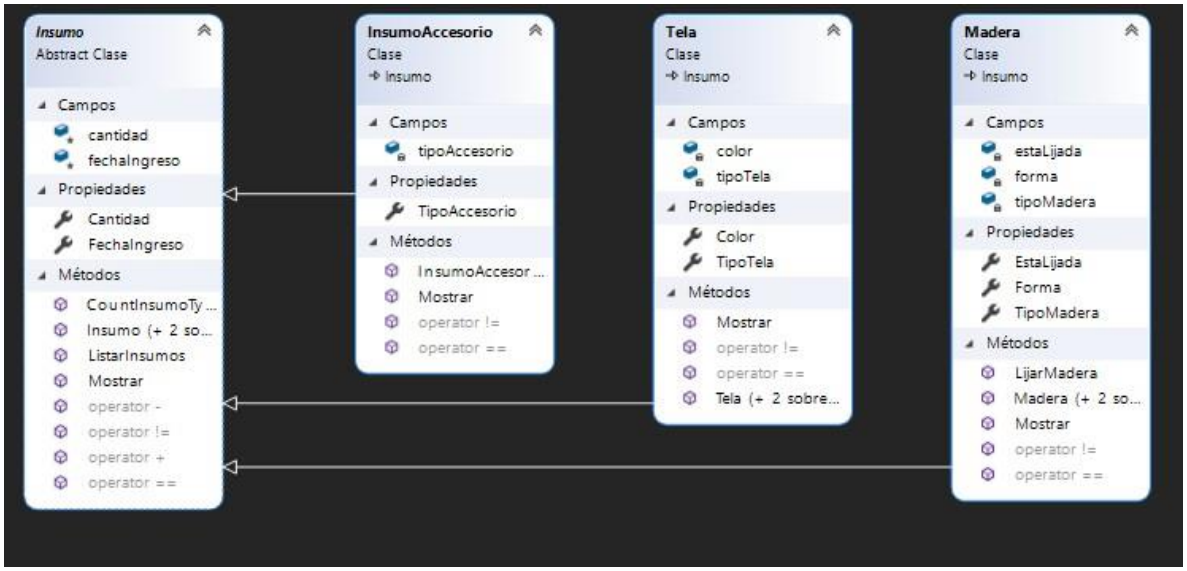
Se creó otro evento en la clase FormDetalles que se invoca en el caso de que se abandona el formulario o se ejecuta un proceso mediante el botón correspondiente.

Clase 25 Métodos de Extensión (Entidades.MetodosExepción)

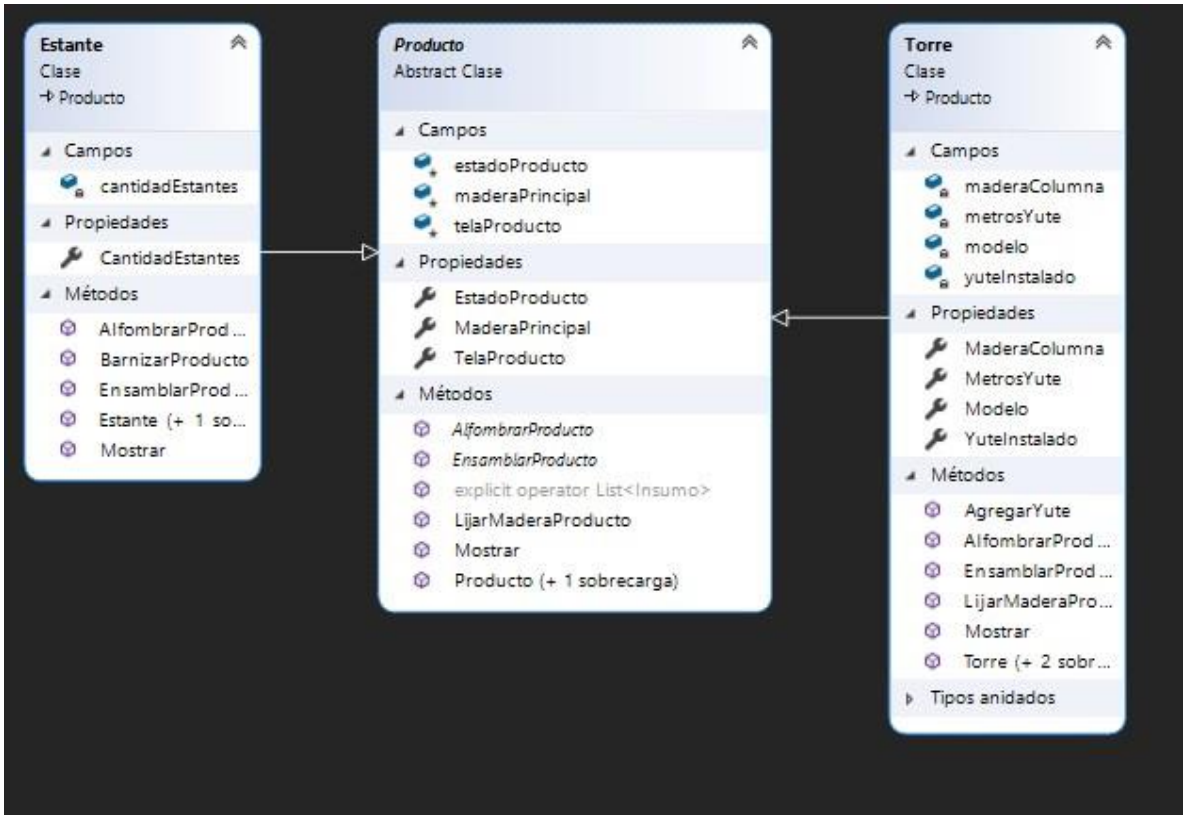
Para esta consigna, se procedió a crear un método que se encargue de hacer una clonación de un objeto del tipo Producto, es decir crear un objeto con los mismos valores en los atributos a otro, pero con referencias en memoria distinta.

Anexo

Clase Base Insumo y derivadas



Clase base Producto y derivadas



Clase Fabrica