Documento de diseño: Manejo de un Supermercado
Proyecto 02

Camilo Morillo Cervantes y Juan Sebastián Ortega Romero

Departamento de Ingeniería de Sistemas y Computación, Universidad de los Andes

16 de noviembre de 2021

1. Actualización de Objetos/Roles

i. Diagrama de clases de alto nivel: Actualización de la Interfaz

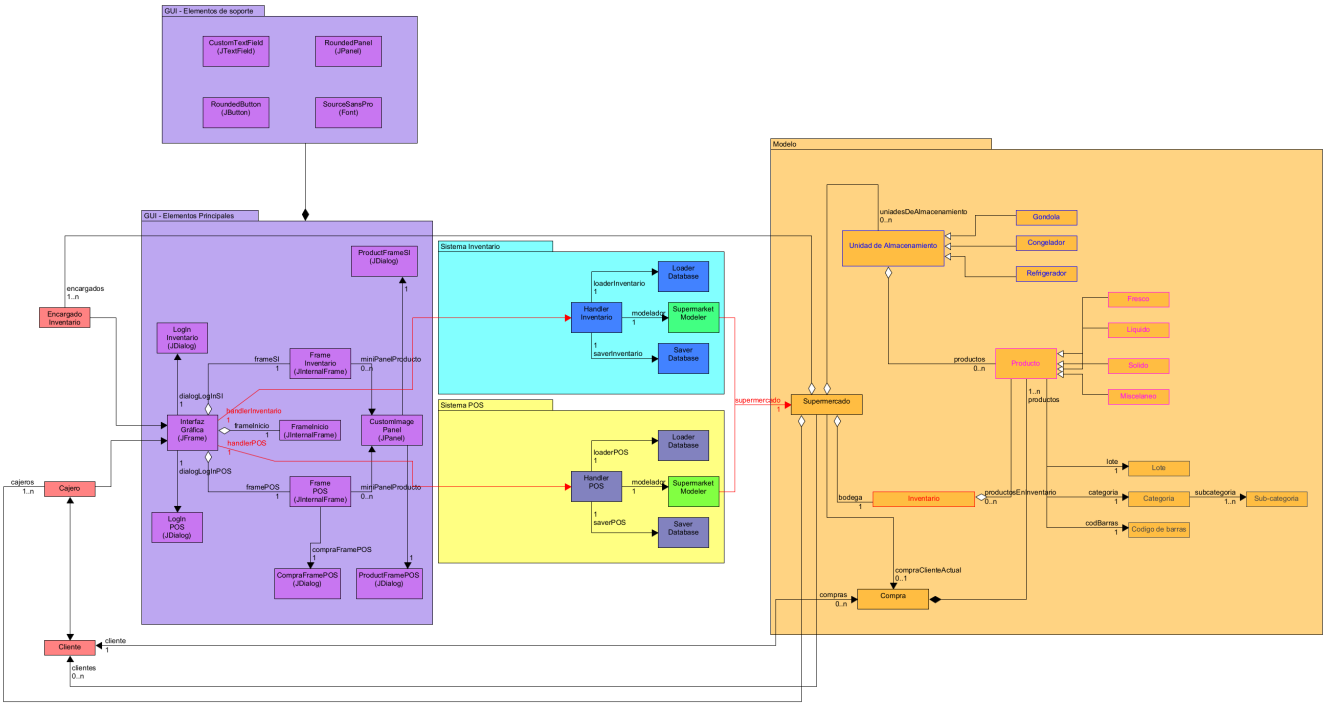

Figura 1: Diagrama de clases de alto nivel

Decisiones importantes: Con el objetivo de que el acceso y la manipulación del sistema fuera más intuitivo para ambos actores involucrados (cajeros y encargados del inventario), se optó por centralizar la interfaz en una ventana principal (JFrame) la cual incluye el único método *main* de todo el aplicativo, siendo este el punto de partida de ambos sistemas. No obstante, para conservar la unicidad de cada sistema, se desarrollaron métodos de acceso (Log-In) particulares tanto para POS como para Inventario, los cuales determinan que JFrame interno se abre dentro de la ventana principal. De esta forma, cada sistema cuenta con un JInternalFrame propio que contiene todo lo necesario para resolver los requerimientos específicos.

Otra observación importante es el hecho de que se buscó mantener la jerarquía de elementos y conexiones que ya se venía manejando en el desarrollo directamente anterior. En consecuencia, únicamente los elementos de mayor jerarquía en cada grupo de clases (modelo, procesamiento e interfaz) tienen acceso

directo a algún elemento de otro grupo. En el diagrama de alto nivel, estas conexiones especiales son representadas por flechas de color rojo que conectan las siguientes clases de alta jerarquía: **InterfazGráfica - HandlerInventario/POS - Supermercado**.

Finalmente, cabe resaltar y aclarar la relación entre los elementos del grupo *GUI - Elementos de soporte* y *GUI - Elementos principales*. Los elementos del primer grupo son componentes típicos de la librería Swing modificados estéticamente mediante el uso de Java2D para lograr una apariencia similar a la del diseño conceptual. Por otro lado, los elementos del segundo grupo son mucho más complejos y de mayor jerarquía, conteniendo las clases que estructuran a grandes rasgos la interfaz gráfica. De esta forma, los elementos del primer grupo componen y están presentes en todas las clases del segundo grupo.

ii. Diagrama de clases detallado: Descomposición actualizada

a) Modelo

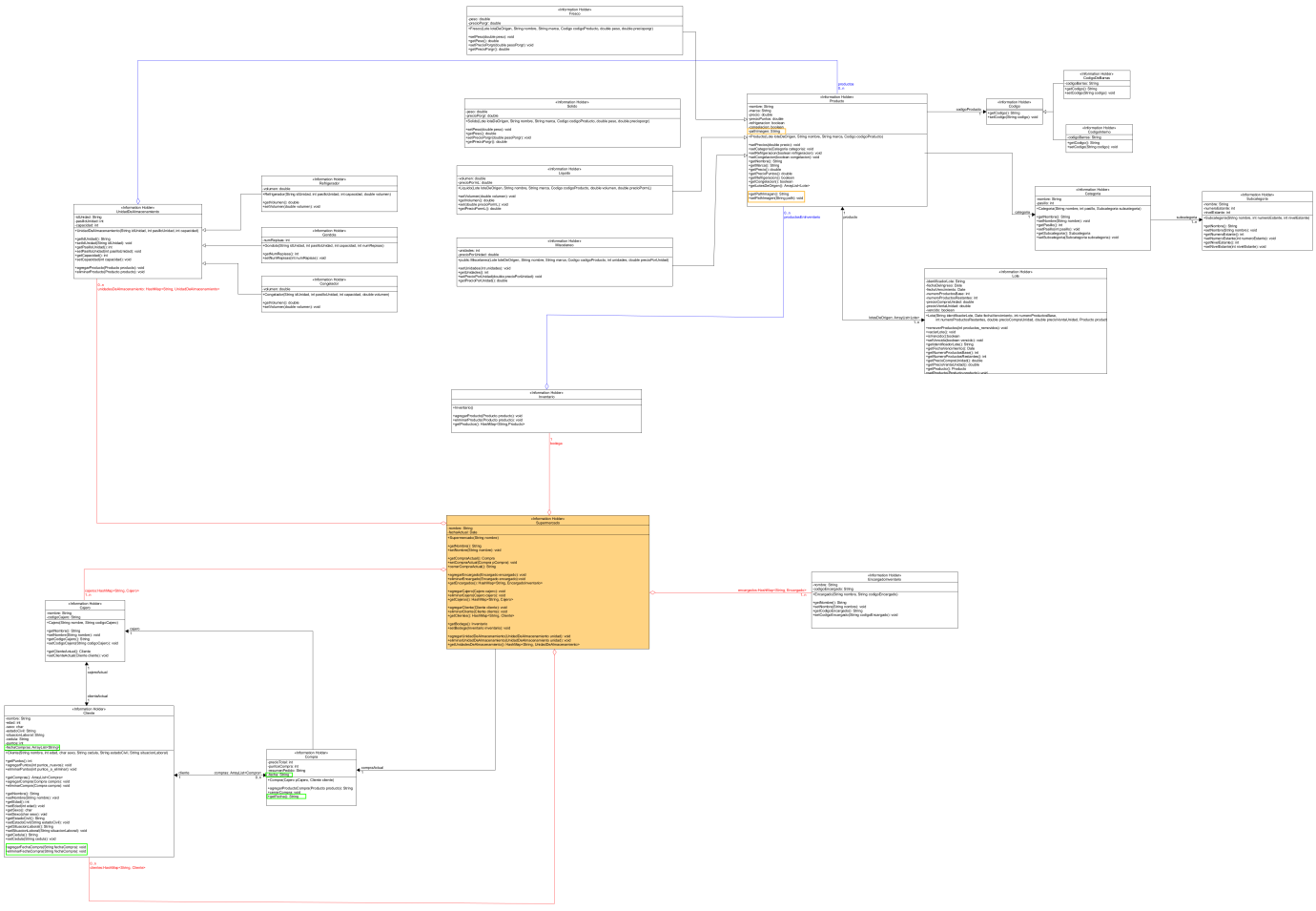

Figura 2: Diagrama de clases detallado: Modelo

Decisiones importantes: Como se puede observar en el diagrama de clases anterior, no se realizaron grandes modificaciones al modelo compartido de la aplicación, por lo que las clases y sus relaciones son exactamente iguales a las vistas en el desarrollo anterior. Sin embargo, debido a la necesidad de incluir un diagrama que representara las compras de un cliente en la interfaz, así como la inclusión de la posibilidad

de vincular una imagen con cada producto, se optó por realizar algunos cambios mínimos en el modelo que permitieran almacenar la información de las fechas y las imágenes, para que posteriormente la interfaz pudiera interactuar y representar dicha información. Rodeado de un rectángulo verde es posible apreciar la creación del atributo **fecha** en el objeto de tipo *compra*, el propósito de este atributo y sus métodos derivados es asignarle la fecha actual a una compra en el momento que esta es finalizada. De esta forma, las compras relacionadas a un cliente siempre tendrán una fecha real asociada. En el caso de las imágenes, es posible observar dentro del rectángulo naranja presente en la clase Producto, el atributo **pathImagen**, este atributo almacena el path de una imagen dentro de la carpeta *imagenesProductos*. Mediante estos atributos, la interfaz será capaz de interpretar las fechas relacionadas a las compras de un cliente específico y así mismo podrá encontrar y renderizar las imágenes relacionadas a cada uno de los productos.

b) Procesamiento POS

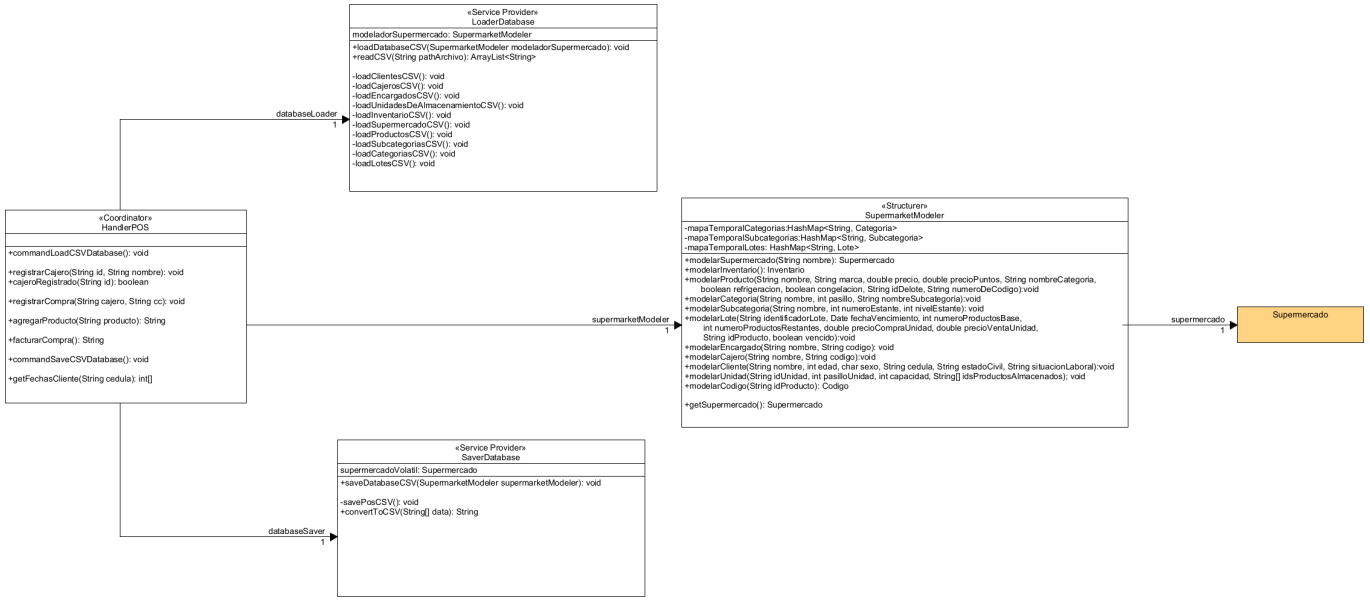

Figura 3: Diagrama de clases detallado: Sistema POS

Decisiones importantes: Puesto que el sistema ya resolvía la totalidad de los requerimientos funcionales, únicamente fue necesario agregar un método que se encargara de obtener las fechas en las que un cliente tiene compras registradas (getFechasCliente en HandlerPOS), esto con el objetivo de generar el diagrama de compras en la interfaz.

c) Procesamiento Sistema Inventario

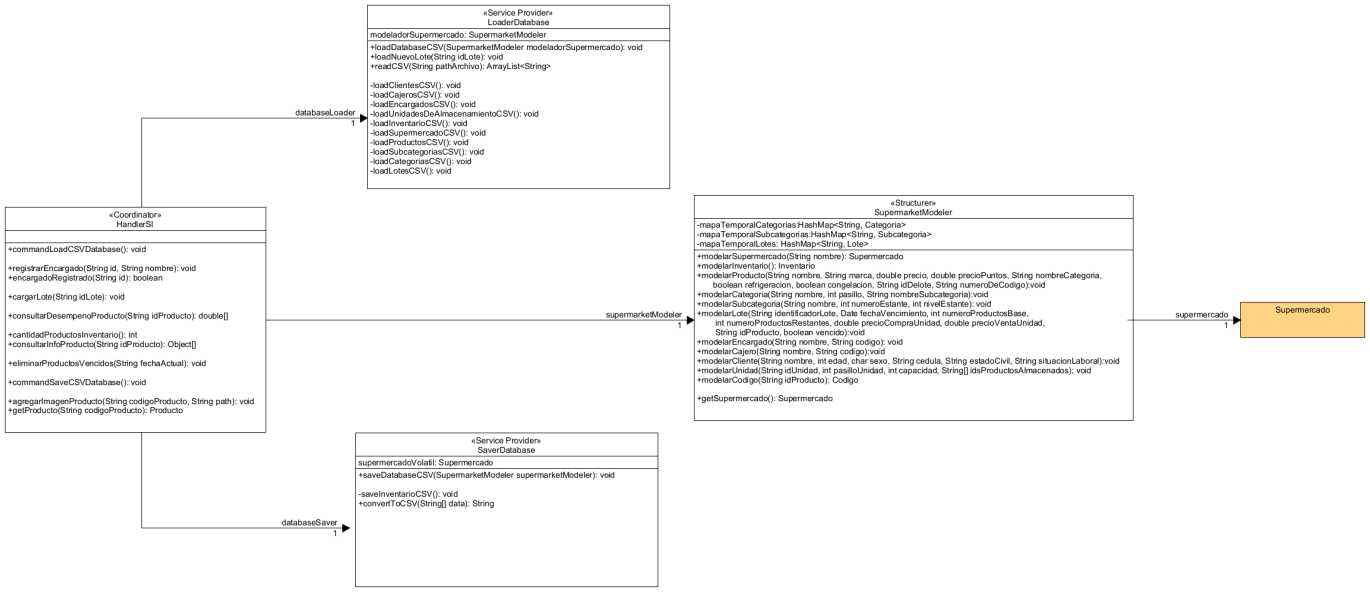

Figura 4: Diagrama de clases detallado: Sistema Inventario

Decisiones importantes: Puesto que el sistema ya resolvía la totalidad de los requerimientos funcionales, únicamente fue necesario agregar dos métodos (agregarImagenProducto y getProducto) a la clase HandlerSI) para que lograr completar el proceso de la relación entre productos e imágenes.

d) Diagrama de alto nivel: Interfaz (Sistema Inventario y POS)

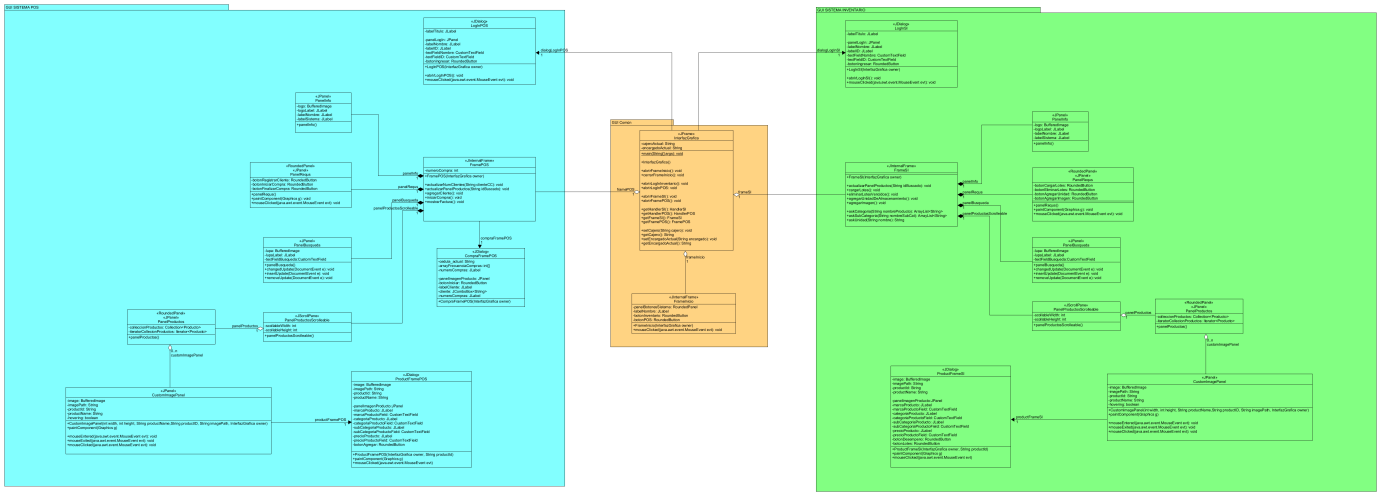

Figura 5: Diagrama de clases detallado: Interfaz

Decisiones importantes: Note la presencia de los elementos de soporte al interior de los múltiples paneles: RoundedPanel, RoundedButton y CustoTextField. Los tres elementos mencionados no son más

que clases estilizadas que heredan de JPanel, JButton y JTextField respectivamente, por lo que su comportamiento es exactamente igual al esperado. Ya que los cambios realizados al interior de estas clases fueron netamente estilísticos, se consideró innecesario representarlas por aparte ya que virtualmente se comportan igual que la superclase de Swing de la cual hereda cada una.

2. Funcionalidades Críticas

Dentro del listado de los requerimientos funcionales se seleccionó una muestra de funcionalidades **críticas** para el desempeño final de la aplicación. Adicionalmente, la solución a dichos requerimientos ilustra el flujo de solución de los demás:

Sistema Inventario

1. Un **encargado del inventario** debe poder realizar múltiples *consultas* ya sea sobre la totalidad del **inventario** o sobre un **producto** particular perteneciente a este.
2. Un **encargado del inventario** debe poder *consultar* las *ganancias o pérdidas* de los **productos** registrados en el **inventario**.
3. Un **encargado del inventario** ha de poder eliminar los **productos** que ya han superado su fecha de vencimiento.
4. Un **encargado del inventario** debe poder *cargar* la información de los **lotes** entrantes al **inventario** a través de un único archivo.
5. Un **encargado del inventario** debe poder cargar una imagen y asociarla directamente con un producto específico. Esta relación debe ser visible, por lo que la imagen ha de poder ser mostrada por el programa.

Sistema POS

1. Un **cajero** ha de poder *registrar* la información de un **cliente** en el sistema de puntos propio del **supermercado**.
2. Un **cajero** ha de poder registrar las **compras** del cliente en el sistema (ya sea a través de un código de barras o el peso y código del producto) y descontar los productos automáticamente del inventario.
3. Un **cajero** debe poder conocer en que casos no existe la disponibilidad de un producto y en dado caso, no realizar la venta.
4. Un **cajero** ha de poder generar un recibo referente a la **compra** de un **cliente**.
5. Un **cajero** ha de poder visualizar de forma gráfica la frecuencia con la que un cliente ha realizado compras en el último año en el momento de iniciar una nueva compra.

*Es importante recordar que la información total del sistema Supermercado debe ser persistente, es decir que debe poderse cargar y archivar en una base de datos externa. Esto debe poderse realizar en ambos sistemas.

i. Carga de datos desde la Data Base (Sistema Inventario):

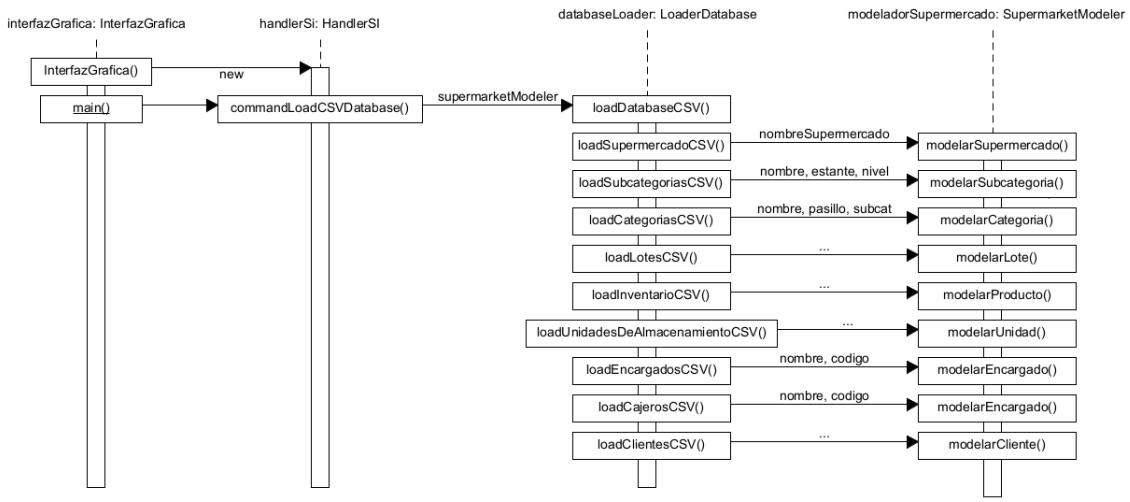

Figura 6: Diagrama de Flujo de la carga de datos.

ii. Guardado de datos a la Data Base (Sistema Inventario):

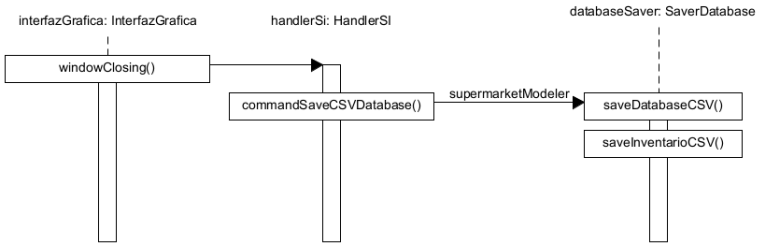

Figura 7: Diagrama de Flujo del guardado de datos.

iii. Registrar Compra (Sistema POS):

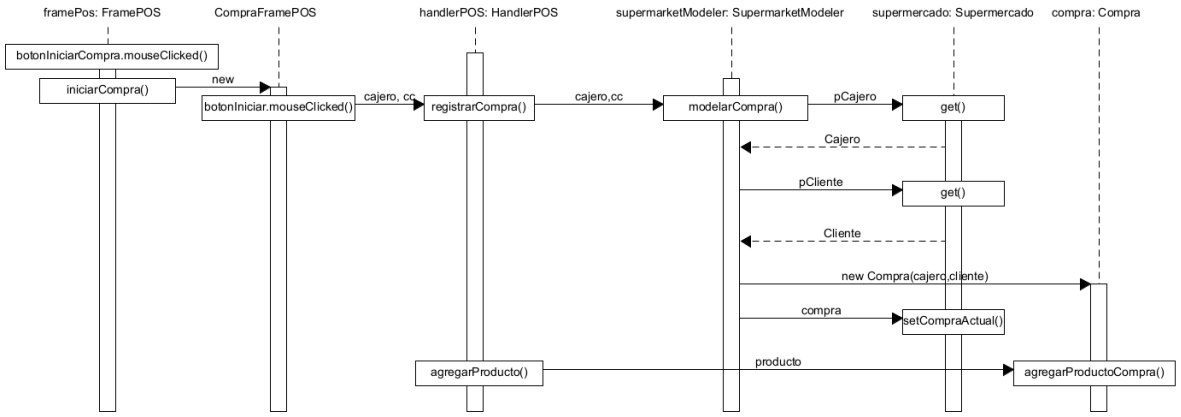

Figura 8: Diagrama de Flujo del registro de una compra mediante el sistema POS.

iv. Facturar Compra (Sistema POS):

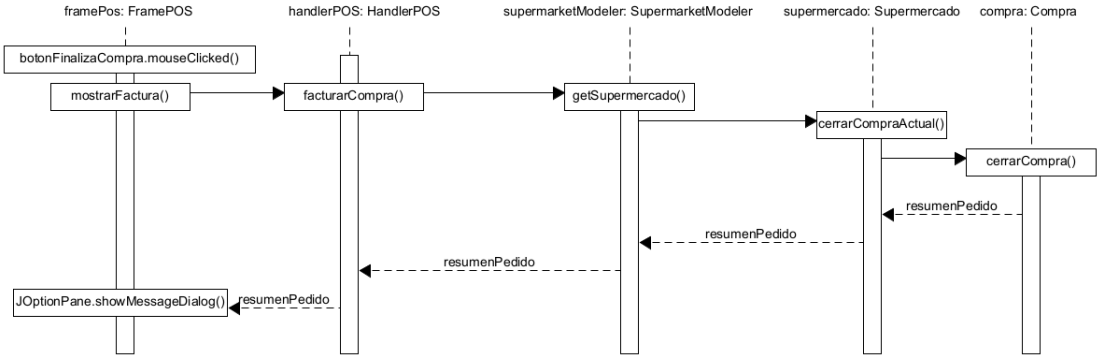

Figura 9: Diagrama de Flujo de la facturación de una compra mediante el sistema POS.

v. Cargar información de nuevo lote (Sistema Inventario):

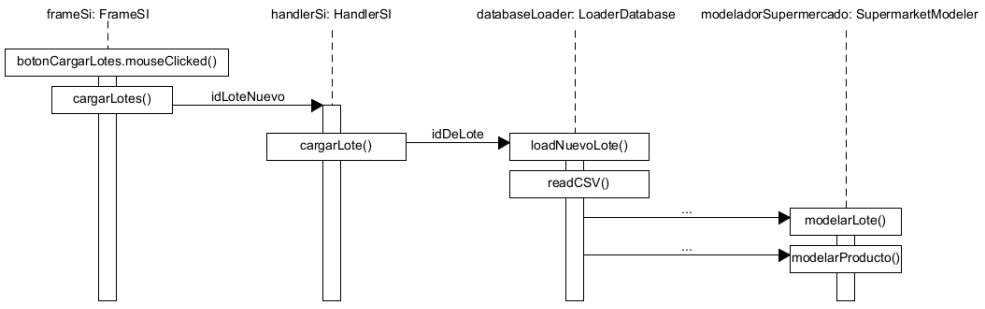

Figura 10: Diagrama de Flujo al cargar un nuevo lote mediante el sistema Inventario.