Proyecto Convivium

Documento de Diseño

EC-DD. Versión 3.0
20 de abril de 2022
Estatus: Restringido

Autores:
Carlos Anívarro Batiste
Daniel Barahona Martín
Daniel Cerrato Sánchez
David T. Garitagoitia Romero
Luis Lepore Pérez de Obanos

Empresa: LCD^3

Tabla de control de versiones

ID	Versión	Fecha	Descripción del Elemento / Versión	Realización	Validación	Revisión formal
EC-DC	1.0	12/03/2022	Diagrama de Clases del Sistema inicial a nivel de la Capa de Modelo.	Equipo de Desarollo	Equipo de Desarollo	
EC-DS04	1.0	12/03/2022	Diagrama de Secuencia correspondiente al Caso de Uso 04 - Registrar Restaurante.	Equipo de Desarollo	Equipo de Desarollo	
EC-DS05	1.0	12/03/2022	Diagrama de Secuencia correspondiente al Caso de Uso 05 - Registrar Producto Local/Externo.	Equipo de Desarollo	Equipo de Desarollo	
EC-DS06	1.0	12/03/2022	Diagrama de Secuencia correspondiente al Caso de Uso 06 - Registrar Menú.	Equipo de Desarollo	Equipo de Desarollo	
EC-DS10	1.0	12/03/2022	Diagrama de Secuencia correspondiente al Caso de Uso 10 - Realizar Menú.	Equipo de Desarollo	Equipo de Desarollo	
EC-DS12	1.0	12/03/2022	Diagrama de Secuencia correspondiente al Caso de Uso 12 - Aceptar pedido.	Equipo de Desarollo	Equipo de Desarollo	
EC-DS13	1.0	12/03/2022	Diagrama de Secuencia correspondiente al Caso de Uso 13 - Pagar pedido.	Equipo de Desarollo	Equipo de Desarollo	
EC-DS14	1.0	12/03/2022	Diagrama de Secuencia correspondiente al Caso de Uso 14 - Liquidar Restaurante.	Equipo de Desarollo	Equipo de Desarollo	
EC-DS16	1.0	12/03/2022	Diagrama de Secuencia correspondiente al Caso de Uso 16 - Introducir Datos del Cliente.	Equipo de Desarollo	Equipo de Desarollo	
EC-DS20	1.0	12/03/2022	Diagrama de Secuencia correspondiente al Caso de Uso 20 - Cerrar pedido.	Equipo de Desarollo	Equipo de Desarollo	
EC-DD Línea base	1.0	12/03/2022	Documento de Diseño inicial. Estructura básica con puntos principales.	Equipo de Desarollo	Equipo de Desarollo	David T. Garitagoitia Romero
EC-DD Línea base	2.0	04/04/2022	Documento de Diseño intermedio. Cambios correctivos sobre la entrega inicial del documento.	Equipo de Mantenimiento	Daniel Cerrato Sánchez	David T. Garitagoitia Romero
EC-DS20	2.1	07/04/2022	Diagrama de Secuencia correspondiente al Caso de Uso 20 - Cerrar pedido con los cambios para la encuesta de satisfacción.	Equipo de Mantenimiento	Carlos Anívarro Batiste	
EC-DC	2.1	07/04/2022	Diagrama de Clases del Sistema con	Equipo de	Carlos Anívarro	

			los cambios para la encuesta de satisfacción.	Mantenimiento	Batiste	
EC-DD	2.1	07/04/2022	Documento de Diseño intermedio. Introducción del primer cambio pedido: Encuestas de satisfacción.	Equipo de Mantenimiento	Daniel Cerrato Sánchez	
EC-DC	2.2	18/04/2022	Diagrama de Clases del Sistema con los cambios para evitar errores de registro con el carácter “ñ” o con diéresis.	Equipo de Mantenimiento	Carlos Anívarro Batiste	
EC-DS01	2.2	18/04/2022	Diagrama de Secuencia correspondiente al Caso de Uso 01 - Registro de usuario.	Equipo de Mantenimiento	Carlos Anívarro Batiste	
EC-DD	2.2	18/04/2022	Documento de Diseño intermedio del tercer cambio (modificación en CU01 de tipo correctivo para evitar errores de registro con el carácter “ñ” o con diéresis).	Equipo de Mantenimiento	Daniel Cerrato Sánchez	
EC-DC	2.3	20/04/2022	Diagrama de Clases del Sistema con los cambios para el resumen de pedidos y ofertas.	Equipo de Mantenimiento	Daniel Cerrato Sánchez	
EC-DS22	2.3	20/04/2022	Diagrama de Secuencia correspondiente al Caso de Uso 22 - Dar de alta cupón.	Equipo de Mantenimiento	Daniel Cerrato Sánchez	
EC-DS21	2.3	20/04/2022	Diagrama de Secuencia correspondiente al Caso de Uso 21 - Enviar email con resumen y promoción mensual.	Equipo de Mantenimiento	Daniel Cerrato Sánchez	
EC-DD	2.3	20/04/2022	Documento de Diseño intermedio con funcionalidad del segundo cambio pedido: Resumen de pedidos y ofertas, todo de carácter mensual.	Equipo de Mantenimiento	Carlos Anívarro Batiste	
EC-DD Línea base	3.0	26/04/2022	Documento de Diseño final.	Equipo de Mantenimiento	Carlos Anívarro Batiste	David T. Garitagoitia Romero

Resumen

Este documento contiene una especificación formal de la arquitectura del sistema *Convivium*, la cual se ha determinado en base al análisis y educción de casos de uso que se presentaban en el Documento de Análisis [EC-DA Versión 3.0]. El sistema *Convivium* se concibe para facilitar la distribución de comida a domicilio, en un entorno en el que clientes, repartidores y propietarios de restaurantes pueden realizar diversas funciones con eficiencia.

Fundamentalmente, este documento tiene por objetivo dar una explicación introductoria a la arquitectura de clases, siguiendo el patrón Modelo-Vista-Controlador, junto con un diagrama explicativo en UML. En dicha arquitectura se describe un diseño tentativo para las jerarquías de usuarios, productos y los diversos sistemas externos con los que se interactúa (sistemas de pago, bases de datos, geolocalización…).

Después, se incluirán los diagramas de secuencia asociados a los casos de uso detallados en el Documento de Análisis [EC-DA Versión 3.0], tales como registrar un restaurante, hacer un pedido, etc. Para cada caso de uso se dará su descripción más básica junto con el pertinente diagrama de secuencia realizado en UML.

Índice de Contenidos

1. Descripción de la Arquitectura del Sistema **6**

2. Diagrama de Clases **8**

3. Diagramas de Secuencia **10**

3.1. Diagrama de Secuencia Caso 1 10
3.2. Diagrama de Secuencia Caso 4 10
3.3. Diagrama de Secuencia Caso 5 12
3.4. Diagrama de Secuencia Caso 6 13
3.5. Diagrama de Secuencia Caso 10 14
3.6. Diagrama de Secuencia Caso 12 15
3.7. Diagrama de Secuencia Caso 13 16
3.8. Diagrama de Secuencia Caso 14 17
3.9. Diagrama de Secuencia Caso 16 18
3.10. Diagrama de Secuencia Caso 20 19
3.11. Diagrama de Secuencia Caso 21 19
3.12. Diagrama de Secuencia Caso 22 20

4. Glosario **22**

1. Descripción de la Arquitectura del Sistema

El objetivo de Convivium es crear una aplicación que cubra un nicho de mercado centrado en la gestión y reparto de comida a domicilio. En el sistema habrá cuatro perfiles distintos de usuario que se explicarán a fondo más adelante. De esta forma se conseguirá un ambiente en el que convivan dueños de restaurantes, repartidores, clientes y un administrador, y que todos ellos disfruten de los beneficios que aporta la aplicación.

La arquitectura de este sistema informático se basa en un patrón MVC (Modelo Vista Controlador).

El patrón MVC, el cual parte de las iniciales de *Model-View-Controller*, es un patrón arquitectural basado en capas, las cuales buscan lograr su principal objetivo que consiste en la separación de los componentes de la aplicación, diferenciando los datos, lógica de negocio y su representación para cubrir la necesidad de la facilidad de reutilización eficiente de código.

La capa de modelo será la encargada de trabajar y operar con los datos con mecanismos tanto para consultar los mismos, como para actualizaciones de dichos datos.

La capa de vista es aquella que contiene la parte del código para presentar la información del modelo, produciendo la interfaz que muestre de forma adecuada dichos datos.

La capa de controlador es aquella encargada de responder a los eventos y acciones de los usuarios, invocando como respuesta de estas acciones al modelo y enviando respuestas a la vista, actuando como intermediario entre usuario y sistema, así como coordinador general del sistema final.

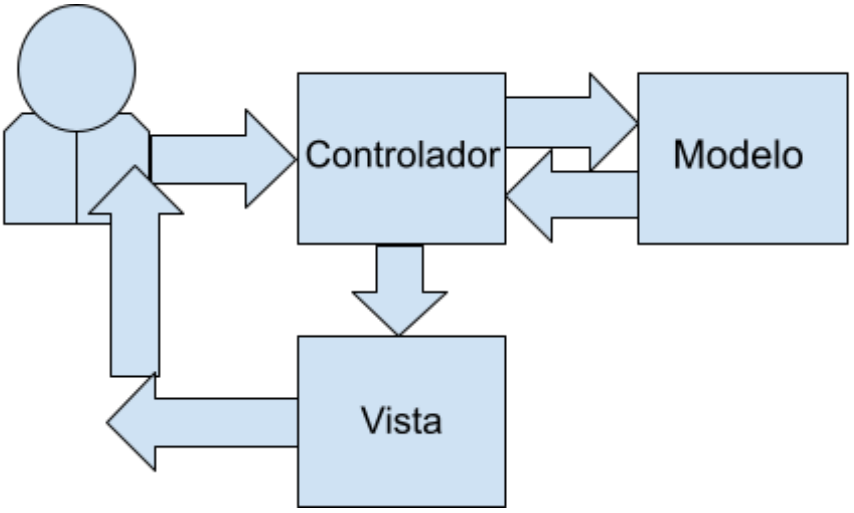


Figura 1. Patrón MVC

Para cumplir con este diseño, el diagrama de clases refleja el equivalente. En otras palabras, el controlador es la clase *Sistema*, que maneja todo el flujo de datos para actualizar el modelo *BaseDeDatos* de forma que la vista *GUI* muestre la información correcta al usuario final. El resto de clases (Usuario, Pedido, Restaurante…) serían el equivalente al modelo *BaseDeDatos* pero en memoria volátil, o sea en memoria RAM. ¿Qué significa esto último? Dichas clases permitirán reflejar la información obtenida de la base de datos en formato objeto, es decir, otorga un formato que la clase *Sistema* podrá manejar. ¿Su propósito? Permitir la comunicación entre la aplicación y el usuario final, aportando un formato consistente de almacenamiento temporal que luego pueda ser utilizado para guardar o cargar datos en la base de datos.
Ahora que entendemos el propósito del diseño, sólo queda aclarar para qué sirve cada clase (aunque su propio nombre ya lo indica):

- Usuario. Clase abstracta que define los atributos y métodos comunes para los diferentes tipos de usuario de la aplicación.
- Administrador. Usuario capaz de editar, añadir o eliminar funcionalidad de la aplicación (por ejemplo, eliminar la posibilidad de hacer menús con productos externos).
- Propietario. Usuario que puede registrar sus restaurantes, así como los productos que ofrecen éstos.
- Repartidor. Usuario que recibirá notificaciones y avisos cuando haya un pedido realizado por su zona de tránsito.
- Cliente. Usuario capaz de realizar pedidos, aceptarlos o cancelarlos y pagarlos.
- Restaurante. Clase que almacenará los datos de un restaurante. Depende fuertemente de su Propietario.
- Item. Clase abstracta que define los atributos y métodos comunes para los diferentes tipos de productos de los restaurantes.
- ProductoLocal. Tipo de producto, de carácter local, que registra el Propietario en uno o varios de sus restaurantes. Almacena los datos de dicho producto local.
- ProductoExterno. Tipo de producto, de carácter ajeno, que registra el Propietario en uno o varios de sus restaurantes. Almacena los datos de dicho producto externo.
- Menú. Tipo de producto que consiste en la agrupación de uno o varios Item. Sigue el patrón Composite.
- Pedido. Clase que almacenará los datos de un pedido realizado por un Cliente. Depende fuertemente del Cliente.
- EstadoPedido. Enumeración que determina el estado de un pedido en un momento dado.
- Domicilio. Clase fuertemente dependiente del Cliente que indica todo lo necesario para poder entregar el Pedido de un Cliente en el lugar adecuado.
- Sistema. Clase *maestra* que controla todo el flujo de datos de la aplicación. Se encarga de comunicarse con la base de datos para la persistencia en guardado y cargado de datos, así como con la interfaz GUI para que muestre datos actualizados.
- GUI. Clase *representativa* que se identifica con la interfaz de la aplicación. No contiene métodos concretos pues poseerá todo el código necesario para

implementar la interfaz gráfica (es la Vista del patrón MVC). Será el controlador quien esté "pendiente" de la GUI para realizar la lógica pertinente.

- BaseDeDatos. Clases *representativa* vacía que indica la existencia de una base de datos para la persistencia de datos, pero no se implementará una clase como tal. La lógica de las consultas a base de datos las hará el propio Sistema.
- SistemaGeolocalización. Sigue la misma línea que BaseDeDatos. Es un objeto externo que ayudará a distribuir la carga de pedidos entre los repartidores.
- PasarelaPago. Clase que hará las comprobaciones de tarjeta y las liquidaciones gracias a la comunicación con un objeto externo dado por un servicio de terceros.

2. Diagrama de Clases

El presente diagrama de clases muestra una aplicación donde gran parte de la funcionalidad reside entre la clase Sistema y la Base de Datos (a partir de ahora BBDD). ¿Esto qué refleja?

Existen dos tipos de aplicaciones: por un lado aquellas que guardan todo en memoria volátil, es decir, en la aplicación, y luego, antes de terminar la ejecución, vuelcan toda la información en la BBDD. Luego, están aquellas aplicaciones que guardan cada cambio que se hace en la BBDD, en "tiempo real".

En este proyecto se ha optado por seguir el diseño del segundo caso expuesto, el cual nos permite proporcionar un servicio de bajo consumo capaz de funcionar en sistemas informáticos con hardware no necesariamente puntero. También es posible recurrir a este diseño ya que se cuenta con una conexión a BBDD estable, de lo contrario no sería la solución más óptima en cuanto a diseño.

A continuación, en la Figura 2, se muestra el diagrama de Clases del Sistema. que ha sido etiquetado con la denominación de control de versiones: **Elemento de Configuración EC-DC, versión 2.3.**

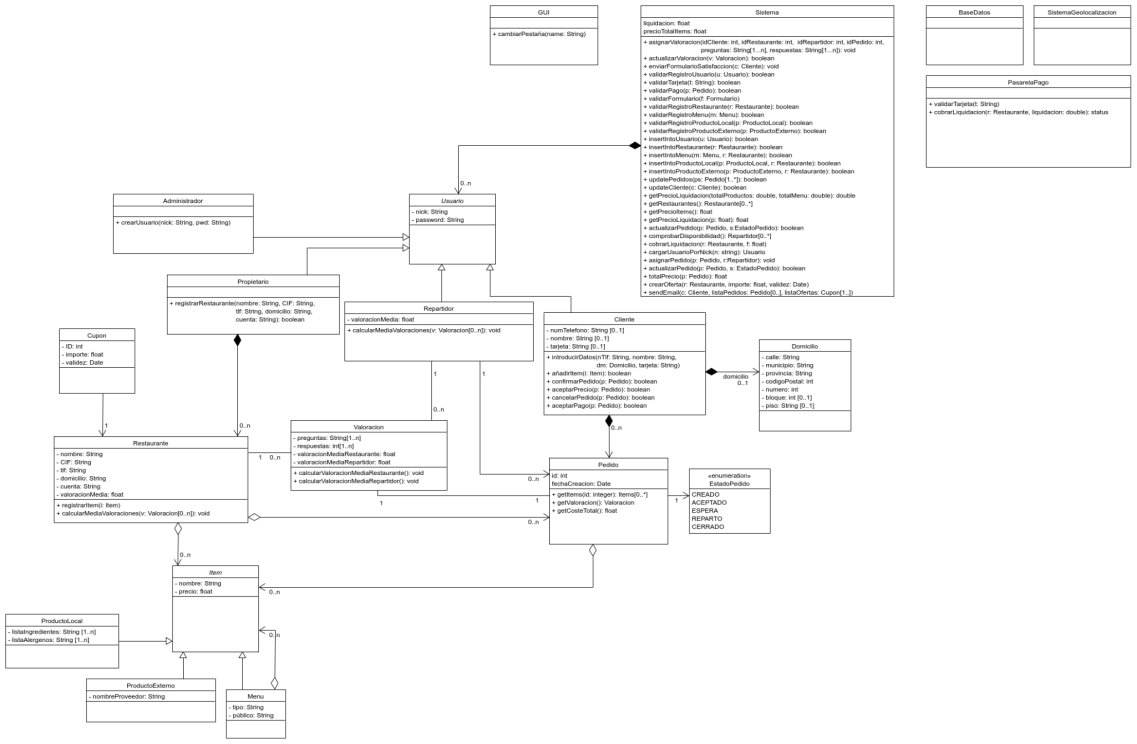


Figura 2. Diagrama de Clases del Sistema.

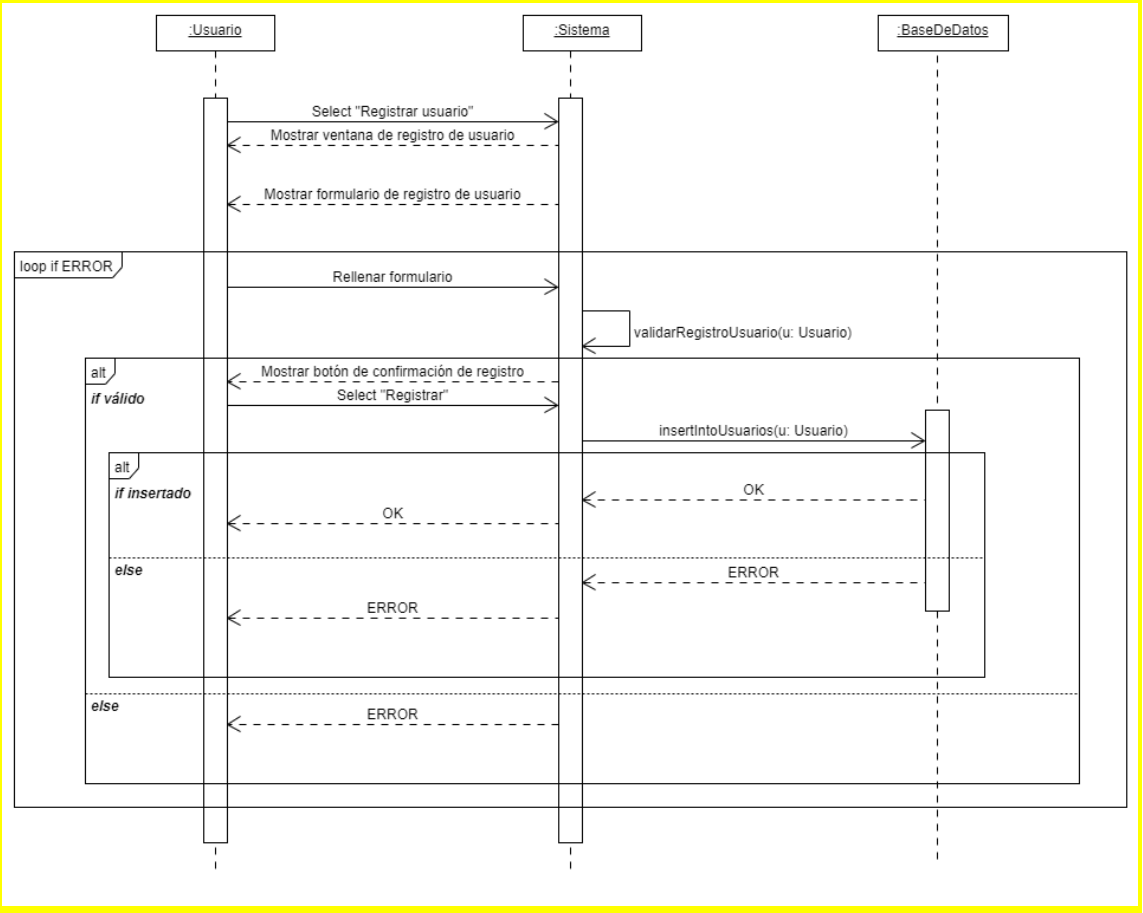
Los getters incluidos en el diagrama no son simples, ya que acceden a la base de datos para realizar consultas.

3. Diagramas de Secuencia

En este apartado se muestra, de forma detallada, el flujo de acciones e interacciones entre clases que tienen lugar en el desarrollo de diversos casos de uso. Esto se hace mediante diagramas de secuencia, cada uno asociado a uno de los casos de uso anteriormente expuestos en el Documento de Análisis.

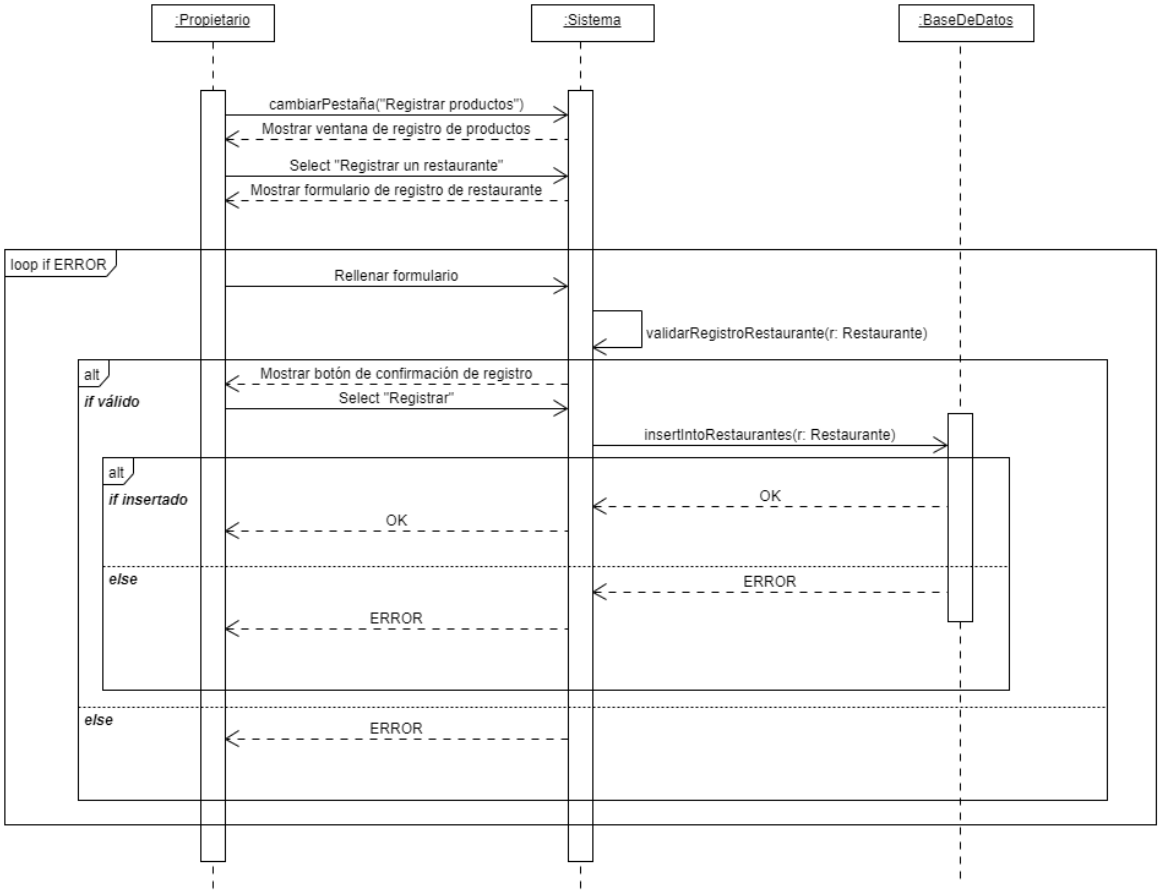
3.1. Diagrama de Secuencia Caso 1

Este caso de uso modela el registro de un usuario por parte de un usuario del sistema. Etiquetado con la denominación de control de versiones: **Elemento de Configuración EC-DS01, versión 2.3.**



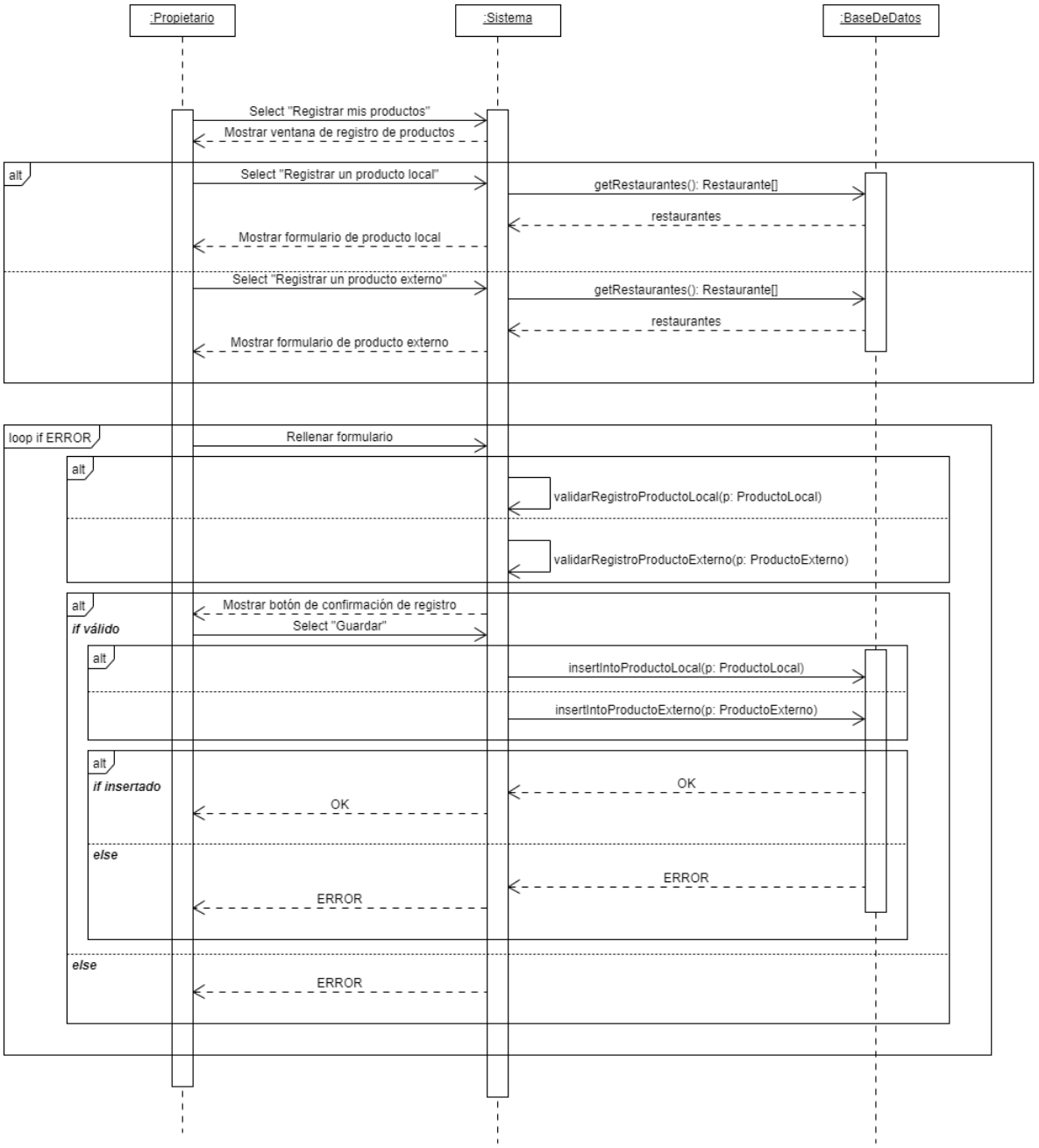
3.2. Diagrama de Secuencia Caso 4

Este caso de uso modela el registro de un restaurante por parte de un usuario con rol "Propietario de Restaurantes". Etiquetado con la denominación de control de versiones: **Elemento de Configuración EC-DS04, versión 1.0.**


:Propietario
:Sistema
:BaseDeDatos
cambiarPestaña("Registrar productos")
Mostrar ventana de registro de productos
Select "Registrar un restaurante"
Mostrar formulario de registro de restaurante
loop if ERROR
Rellenar formulario
validarRegistroRestaurante(r: Restaurante)
alt
if válido
Mostrar botón de confirmación de registro
Select "Registrar"
insertIntoRestaurantes(r: Restaurante)
alt
if insertado
OK
OK
else
ERROR
ERROR
else
ERROR

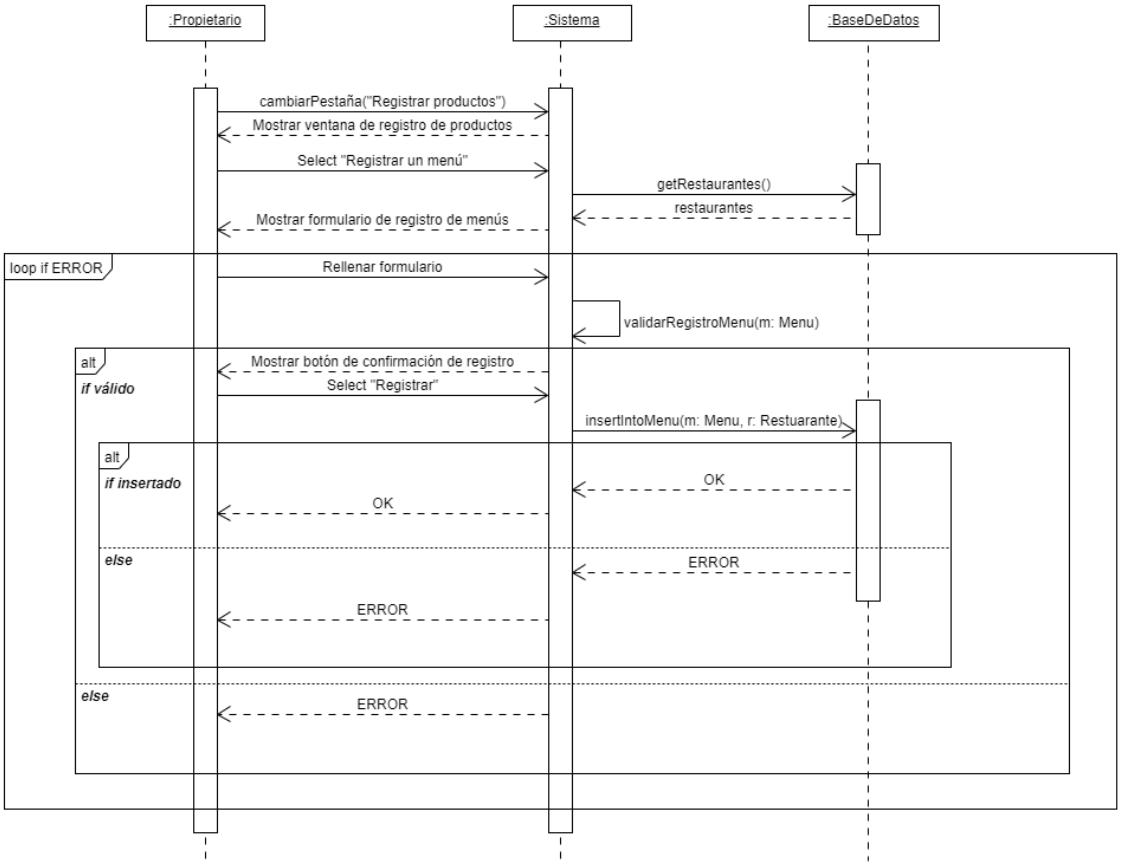
3.3. Diagrama de Secuencia Caso 5

Este caso de uso modela el registro de un producto (local o externo) por parte de un usuario con rol "Propietario de Restaurantes". Etiquetado con la denominación de control de versiones: **Elemento de Configuración EC-DS05, versión 1.0.**



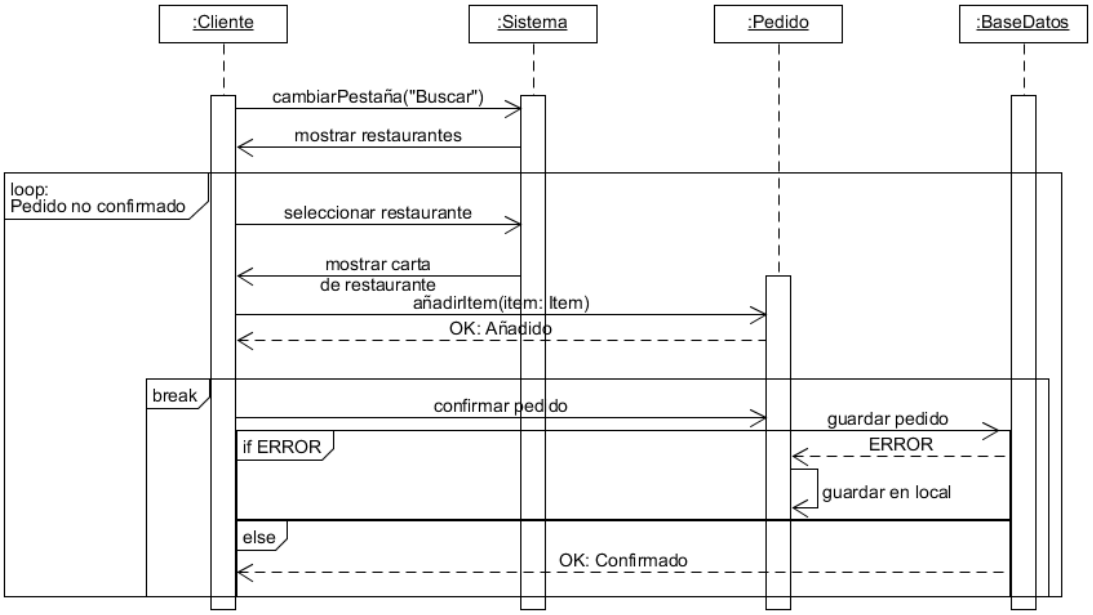
3.4. Diagrama de Secuencia Caso 6

Este caso de uso modela el registro de un menú por parte de un usuario con rol "Propietario de Restaurantes". Etiquetado con la denominación de control de versiones: **Elemento de Configuración EC-DS06, versión 1.0.**



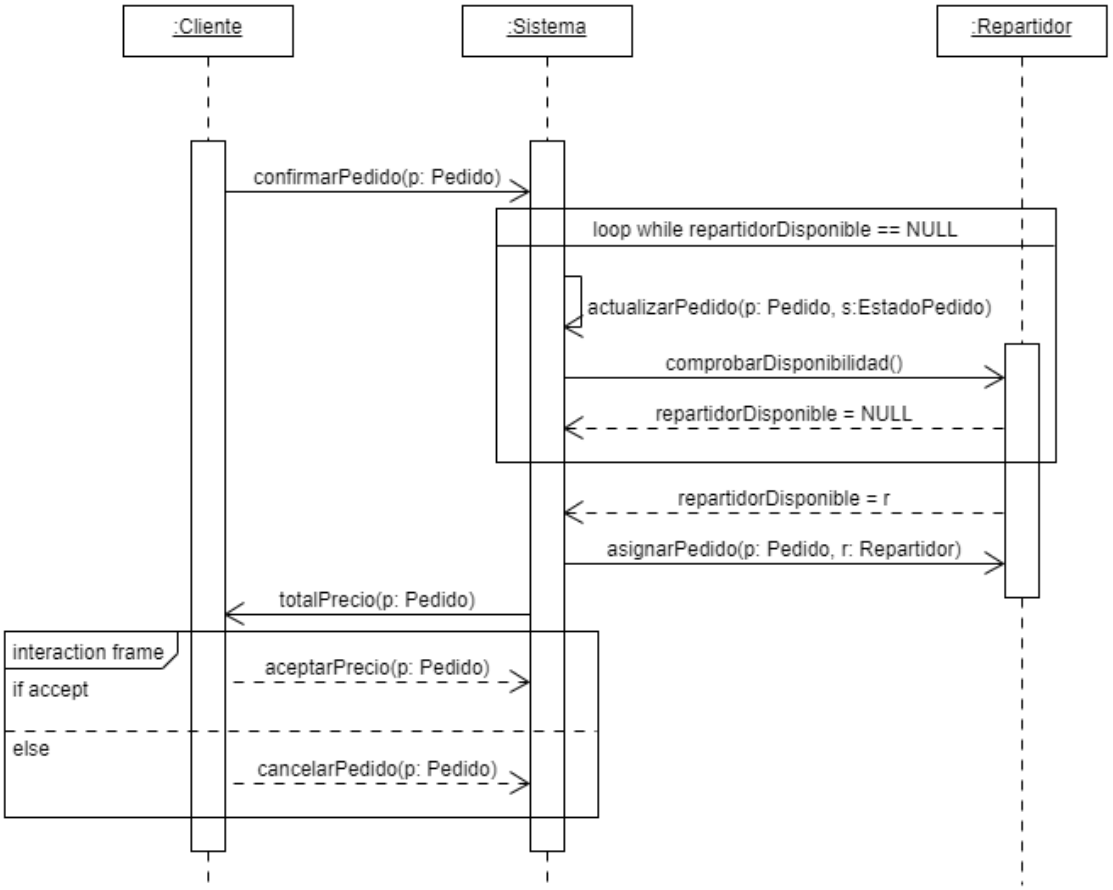
3.5. Diagrama de Secuencia Caso 10

Este caso de uso modela la realización de un pedido por parte de un usuario con rol "Cliente del Servicio". Etiquetado con la denominación de control de versiones: **Elemento de Configuración EC-DS10, versión 1.0.**



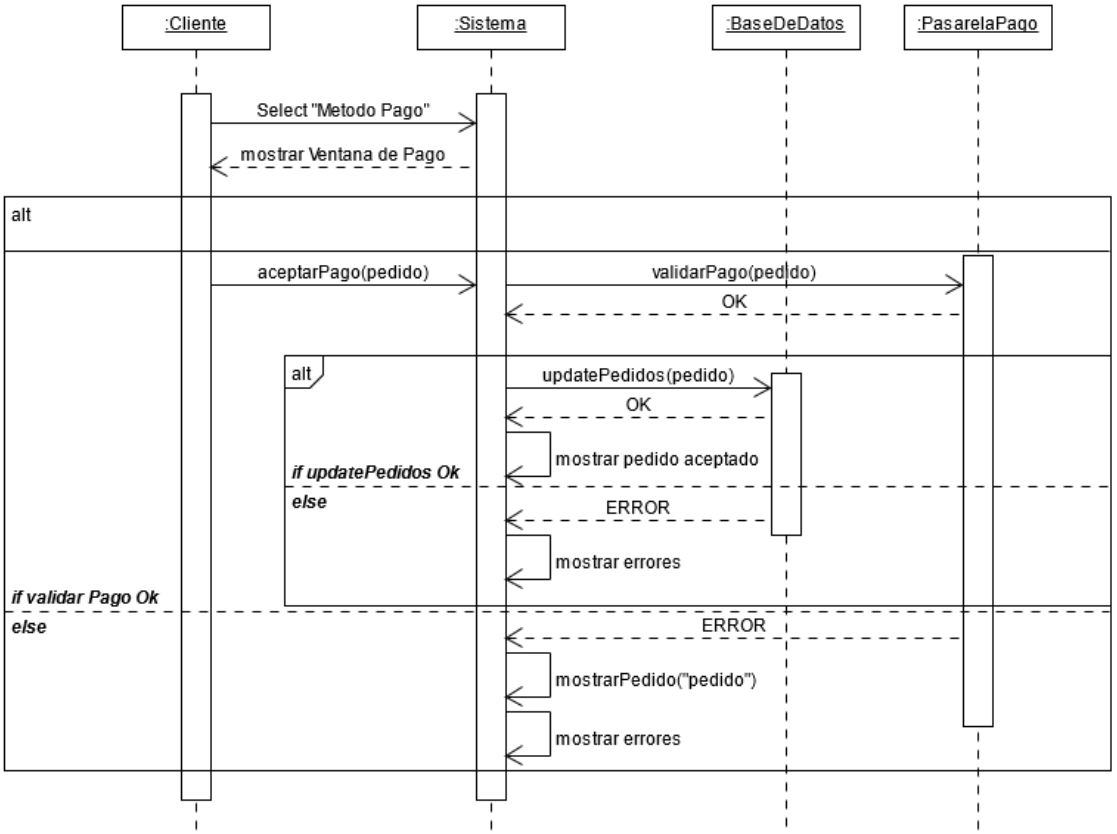
3.6. Diagrama de Secuencia Caso 12

Este caso de uso modela la aceptación de un pedido por parte de un usuario con rol "Cliente del Servicio". Etiquetado con la denominación de control de versiones: **Elemento de Configuración EC-DS12, versión 1.0.**



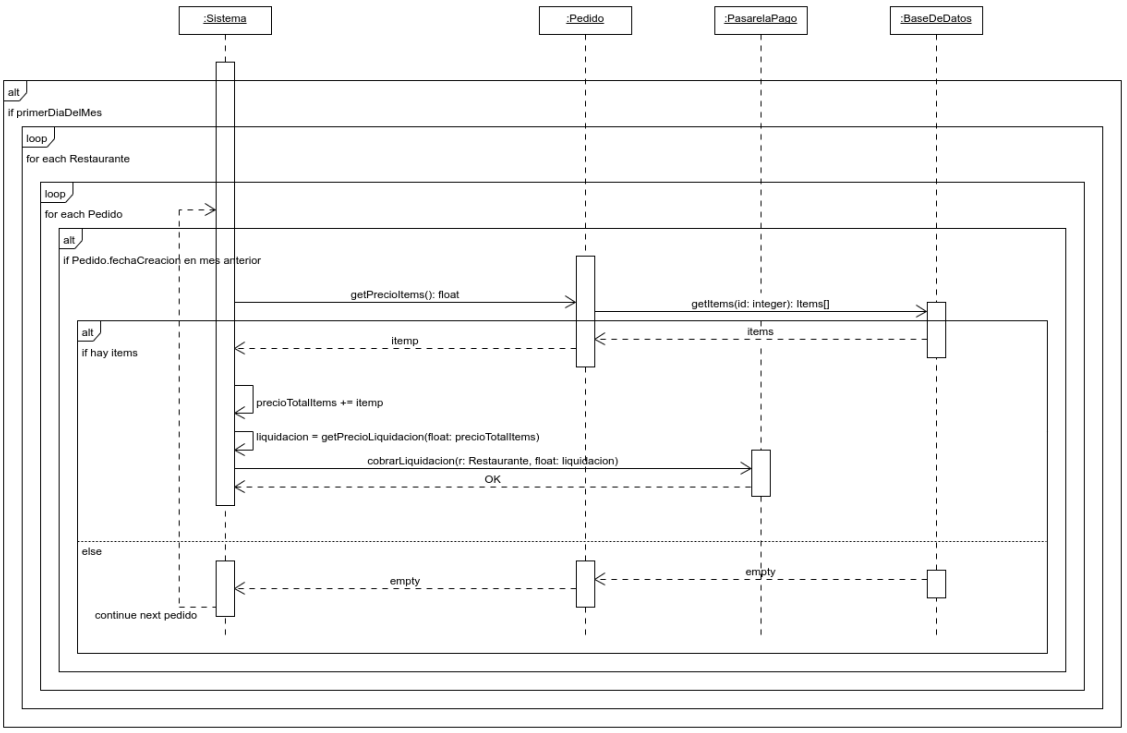
3.7. Diagrama de Secuencia Caso 13

Este caso de uso modela el pago de un pedido por parte de un usuario con rol "Cliente del Servicio". Etiquetado con la denominación de control de versiones: **Elemento de Configuración EC-DS13, versión 1.0.**



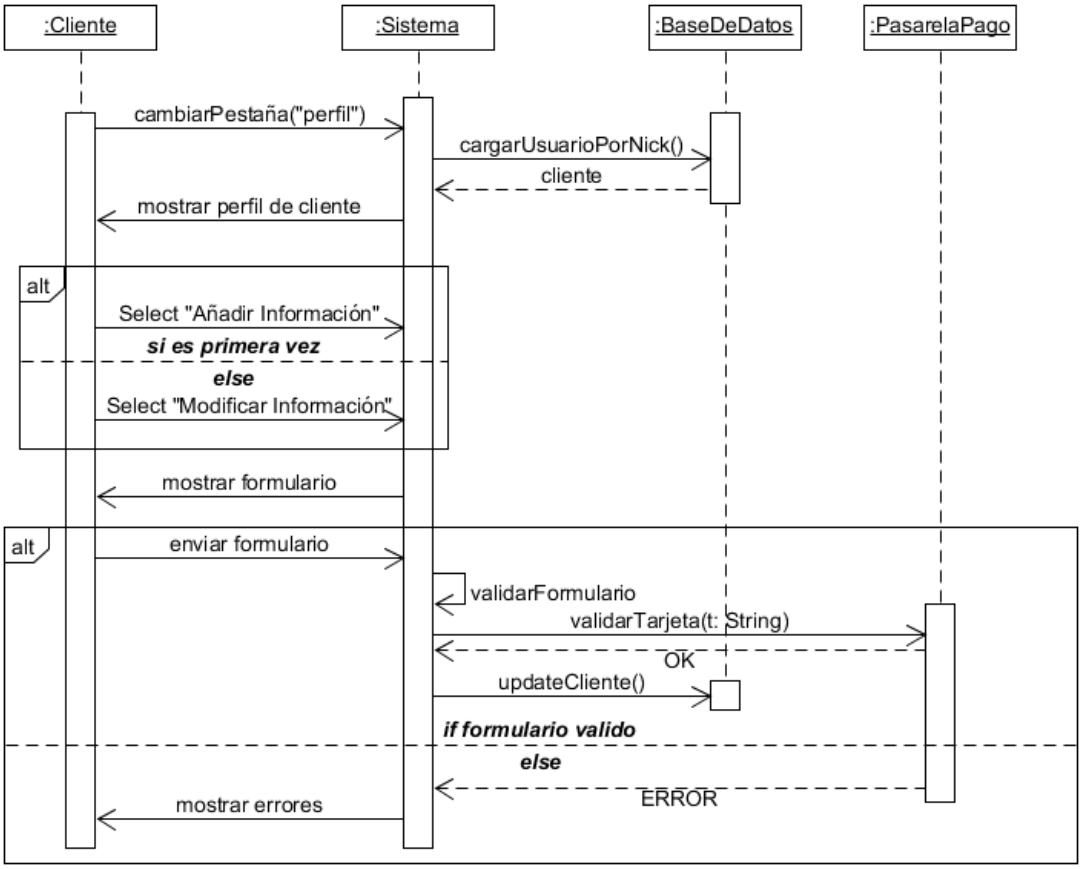
3.8. Diagrama de Secuencia Caso 14

Este caso de uso modela la liquidación de un restaurante por parte del sistema. Etiquetado con la denominación de control de versiones: **Elemento de Configuración EC-DS14, versión 1.0.**



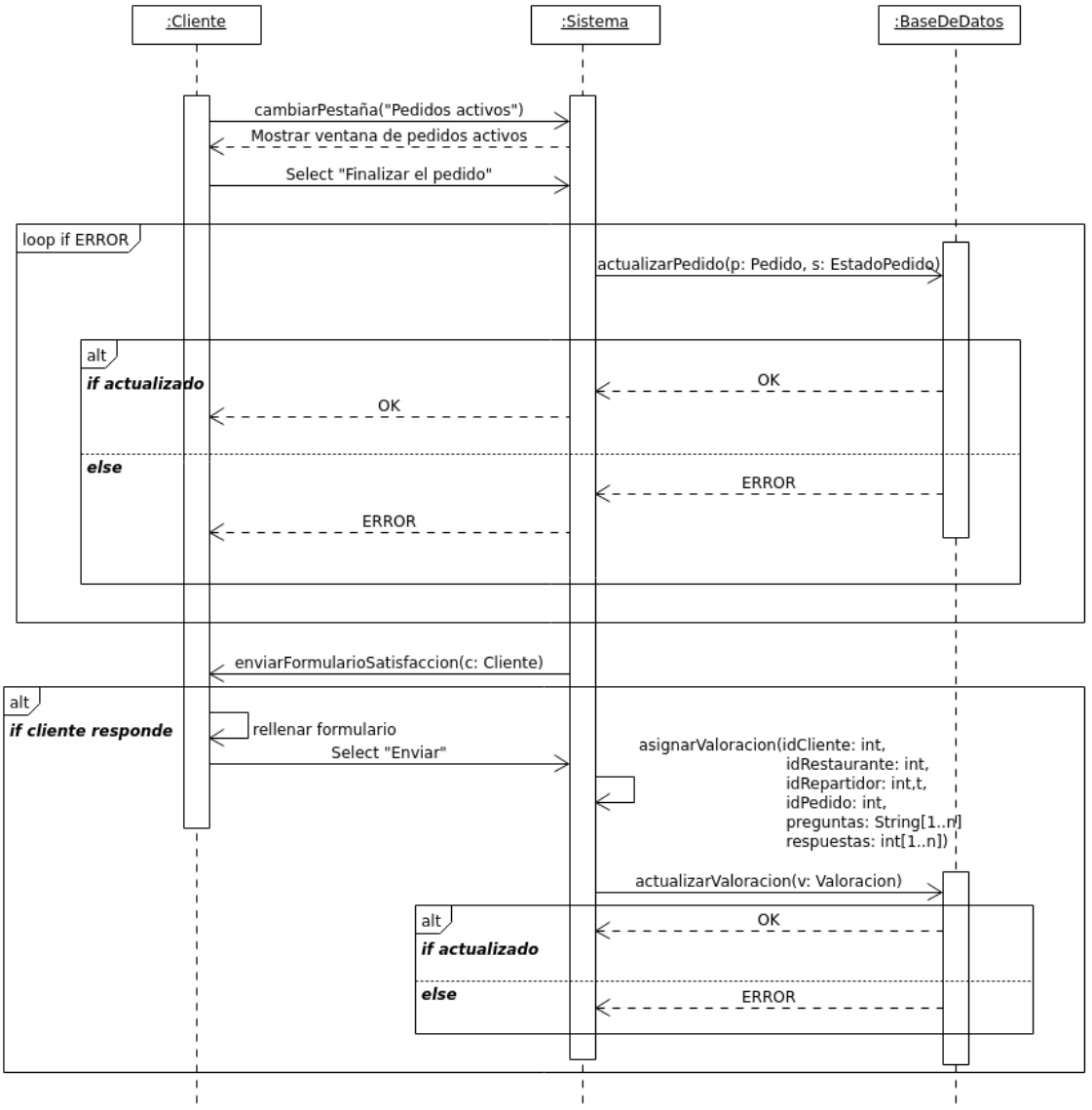
3.9. Diagrama de Secuencia Caso 16

Este caso de uso modela la introducción de los datos del cliente por parte de un usuario con rol "Cliente del Servicio". Etiquetado con la denominación de control de versiones: **Elemento de Configuración EC-DS16, versión 1.0.**



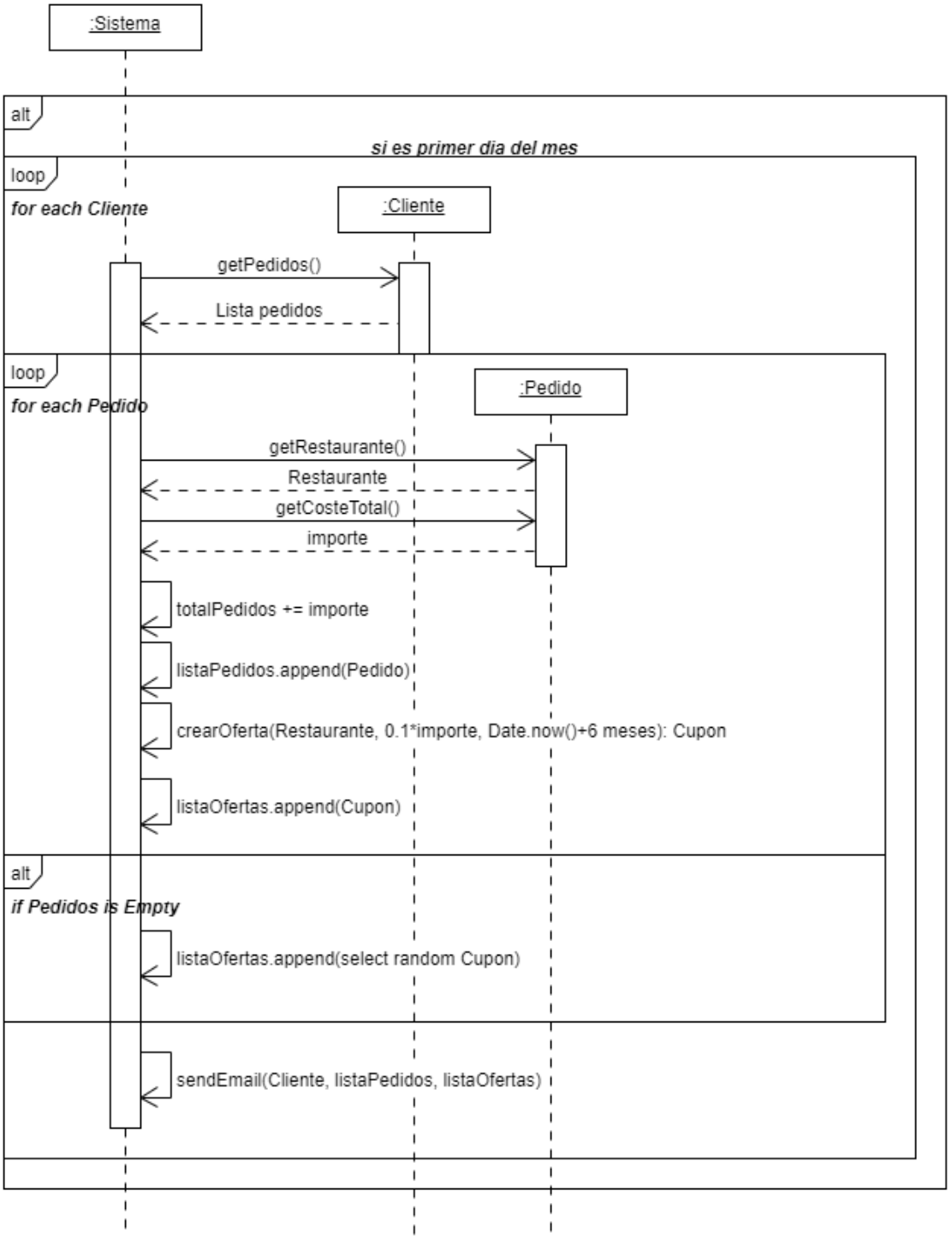
3.10. Diagrama de Secuencia Caso 20

Este caso de uso modela el cierre de un pedido y el envío de la encuesta de satisfacción por parte de un usuario con rol "Cliente del Servicio". Etiquetado con la denominación de control de versiones: **Elemento de Configuración EC-DS20, versión 2.2.**



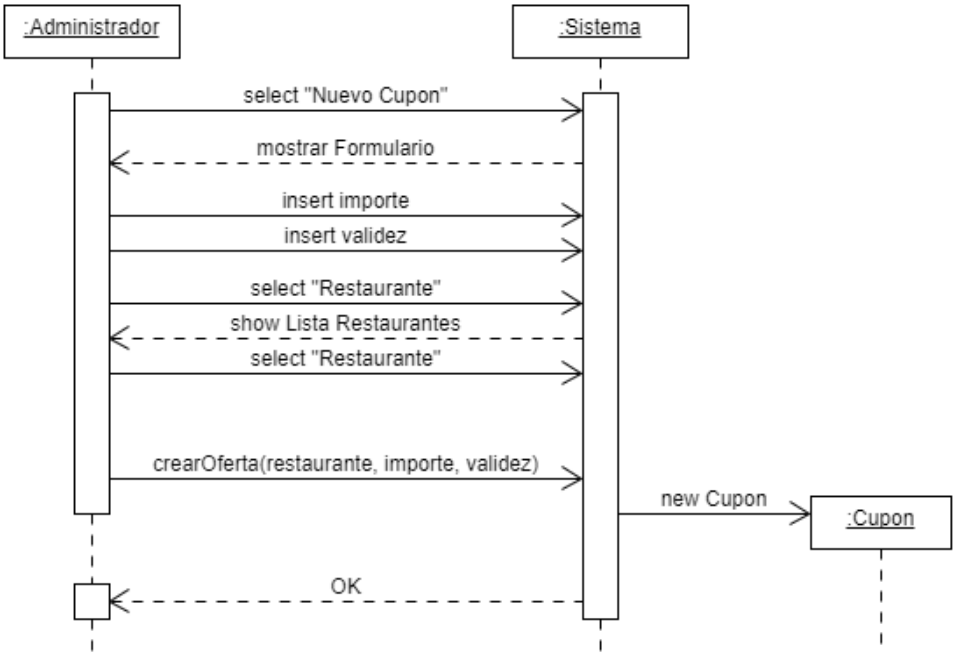
3.11. Diagrama de Secuencia Caso 21

Este caso de uso modela el envío del email mensual a todos los clientes registrados, por parte del Sistema. Etiquetado con la denominación de control de versiones: **Elemento de Configuración EC-DS21, versión 2.3.**



3.12. Diagrama de Secuencia Caso 22

Este caso de uso modela la alta de una oferta en forma de cupón en el sistema, por parte del Administrador. Etiquetado con la denominación de control de versiones: **Elemento de Configuración EC-DS22, versión 2.3.**


:Administrador
:Sistema
select "Nuevo Cupon"
mostrar Formulario
insert importe
insert validez
select "Restaurante"
show Lista Restaurantes
select "Restaurante"
crearOferta(restaurante, importe, validez)
new Cupon
:Cupon
OK

4. Glosario

TÉRMINO	DESCRIPCIÓN
BBDD	Base de datos.
Clase maestra	Clase, del diagrama de clases, que maneja la mayoría de la funcionalidad de la aplicación. Se da, sobre todo, cuando se desarrolla el patrón de diseño Modelo-Vista-Controlador, siendo ésta el controlador.
Clase representativa	Clase, del diagrama de clases, que no requiere de una implementación como tal a la hora de entrar en la fase de programación. Es una clase de carácter informativo.
Diagrama de secuencia	Diagrama en el que se representa la secuencia de acciones que se realizarán para completar un requisito.
Educción	Deducción.
Formato consistente	Se dice del formato de un objeto cuando es común para dos o más comunicadores. Es decir, si dos entornos diferentes son capaces de comunicarse mediante un objeto X, se dice que ese objeto X tiene un formato consistente (p.e. la base de datos maneja datos simples y la aplicación maneja datos complejos: se necesita una clase que transforme la información simple de la base de datos a un objeto complejo que entienda la aplicación).
Getter	Función que trata de devolver un atributo de una clase o de obtener algún dato de algún objeto.
Memoria volátil o memoria RAM	Memoria que no almacena su estado al sufrir una desconexión. Esto es, si una aplicación se cierra por completo o el ordenador que la ejecuta se apaga, la información recabada por dicha aplicación se pierde.