# Proyecto CometelooToo

## Documento de Diseño

9 de marzo de 2020

Informe realizado para


Carlos Molinero Alvarado
Rafael Sánchez Sánchez
Alejandro Santorum Varela
Manuel Soto Jiménez

S3M enterprise

c/ de Silva, 9
28004 Madrid
Tlf.: 34-91-524 02 00
Fax: 34-91-524 02 73
contacto@s3menterprise.es
www.s3menterprise.es

# Resumen del documento

En el siguiente documento se expondrá el diseño del **proyecto Cometelotoo**.

En primer lugar, se **describirá el diseño de la arquitectura**, buscando una implementación acorde a los estándares actuales, distinguiendo servidor, y dos tipos de cliente.

A continuación se aporta el **diagrama de clases** de la aplicación, donde se intenta guiar su implementación en un lenguaje orientado a objetos.

Finalmente, se aportan los **diagramas de secuencia** de las cuatro historias de usuario implementadas en el primer sprint de SCRUM, que se corresponden con las que el equipo dio más prioridad. Con ello, se definen las interacciones entre participantes y agentes en la implementación de la funcionalidad de la historia.

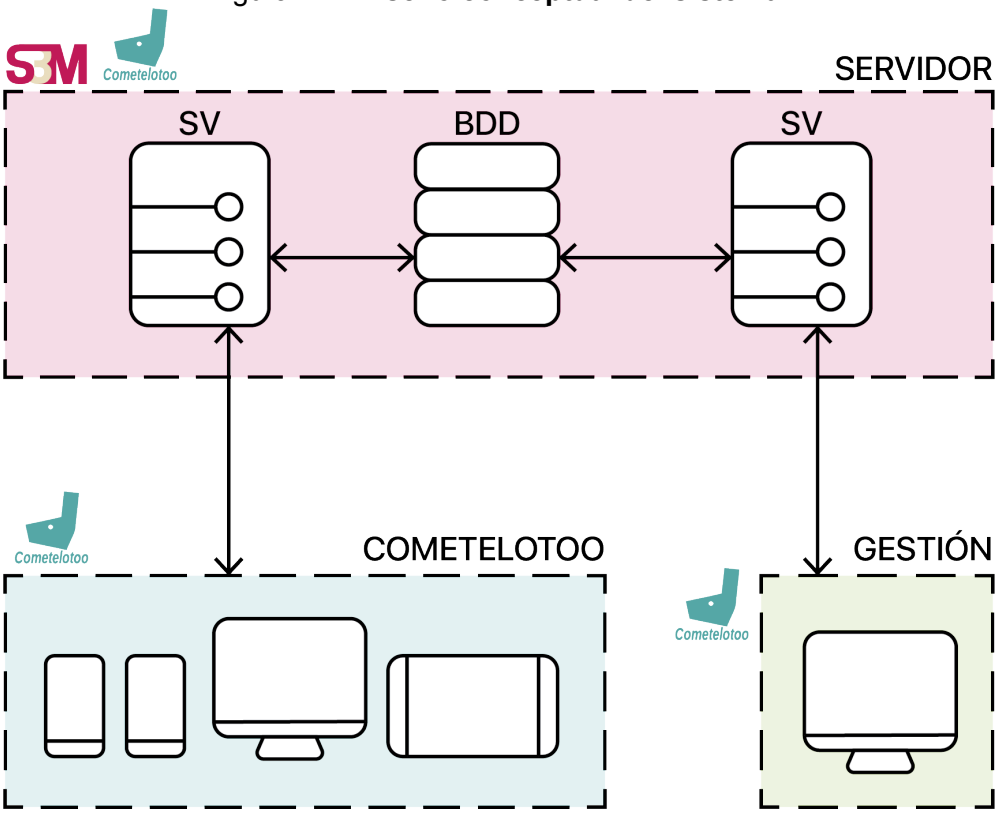
# Índice

# 1. Descripción de la Arquitectura del Sistema

Planteamos un sistema **cliente - servidor** para el despliegue de la aplicación. Los clientes (el gestor y los usuarios de la aplicación) se conectarán con nuestro servidor de aplicaciones mediante **HTTPS** a través de una *REST API*.

Con este nivel de abstracción conseguimos la uniformidad en la implementación de los servicios ofrecidos, que será común para los diferentes clientes de la aplicación (web, tablet y móvil). A su vez, el servidor de aplicaciones se comunicará con la base de datos, que no se ubicará en el mismo sistema físico necesariamente.

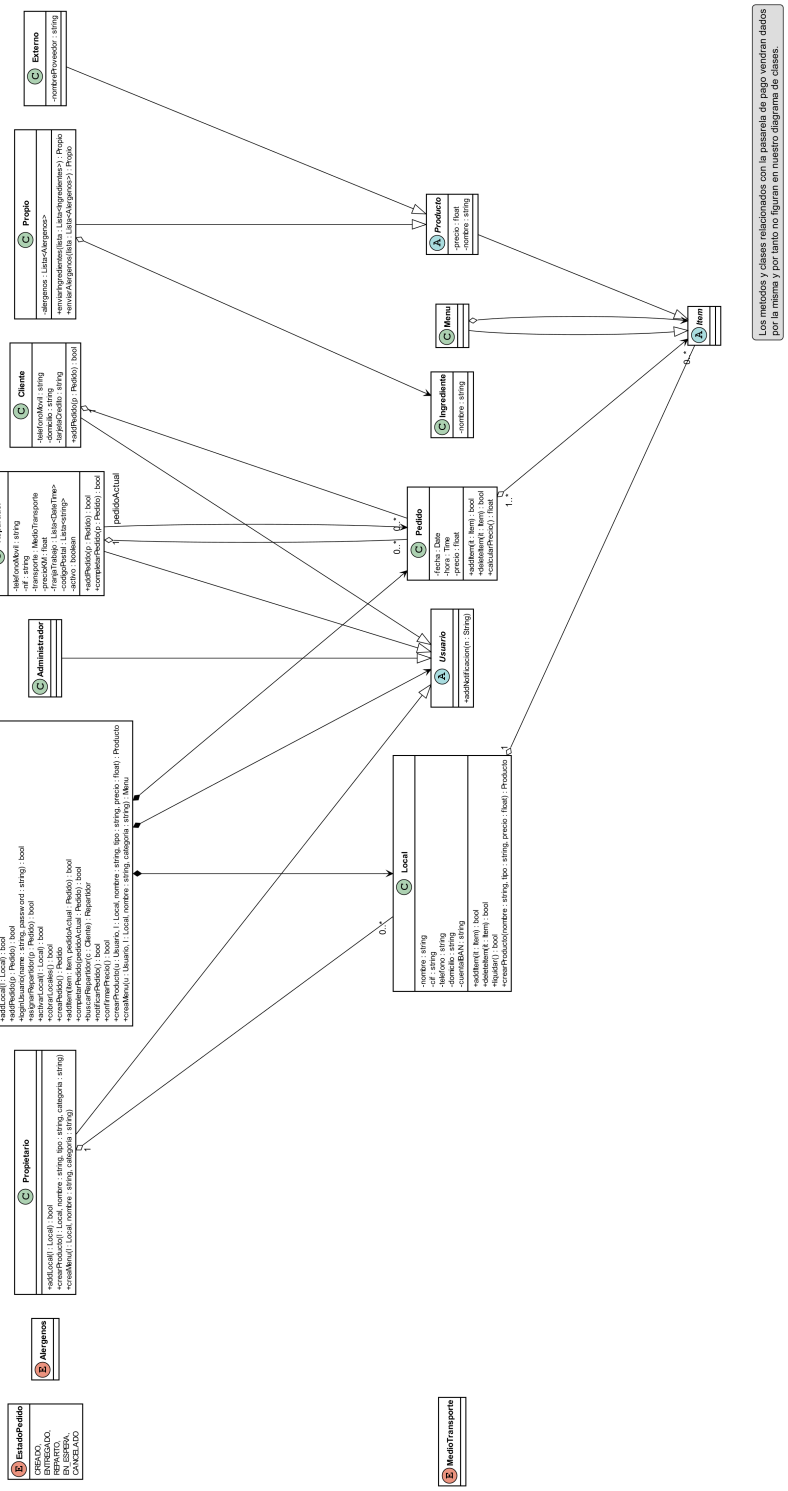
Figura 1.1: **Diseño conceptual del sistema**



# 2. Diagrama de Clases

Hemos seguido un diseño orientado a objetos. Cabe destacar que la clase `App` es un *Singleton*, con lo que contará con una instancia única en el servidor. Además, hemos usado el patrón *Composite* en el diseño de los Menús, que heredan de la clase abstracta `Item` y tienen a su vez una lista de `Items` como atributo.

Figura 2.1: **Diagrama de clases para la lógica y modelo de datos de Cometelotoo**



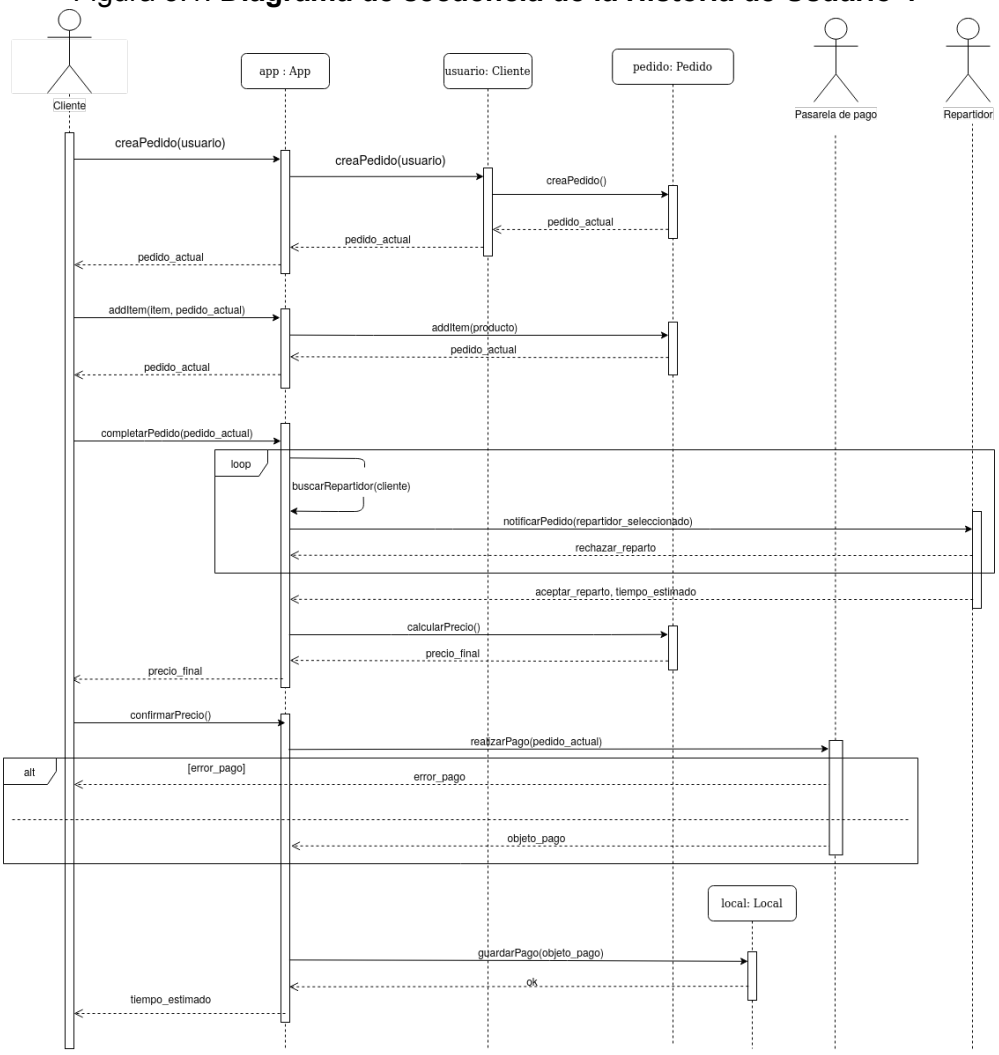
# 3. Diagramas de secuencia

Si hay algo a destacar entre todos los diagramas de secuencia es en el hecho de permitir al usuario únicamente acceder al objeto *App*. Por lo demás, como es de esperar, variará en función de la lógica a implementar en la historia de usuario.

## 3.1. Diagrama de secuencia Caso 4

Para la realización de un pedido se destaca la aparición tanto de la pasarela de pago (para poder efectuar el pago del pedido) así como del repartidor (componente clave en la asignación del mismo para un pedido).

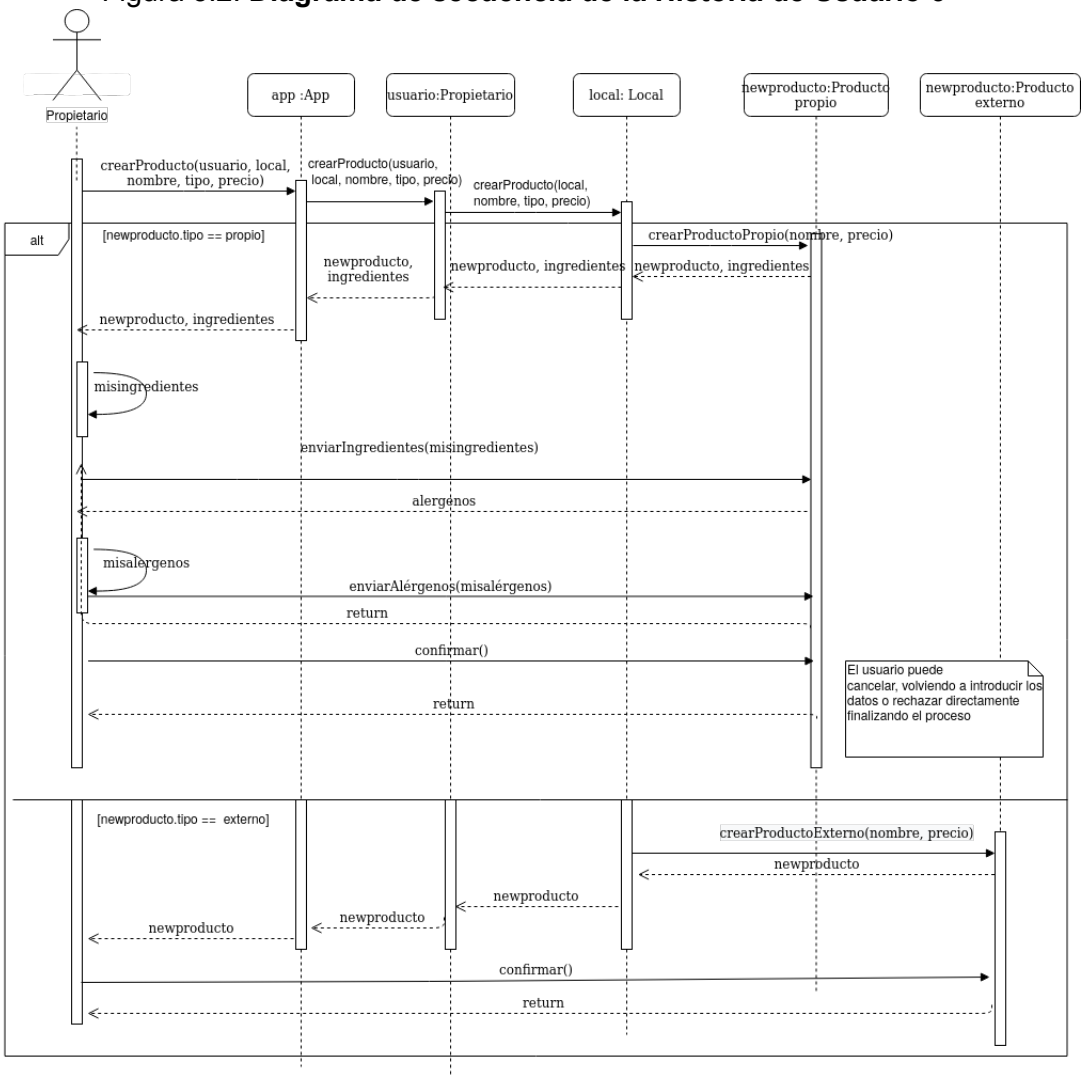
Figura 3.1: **Diagrama de secuencia de la Historia de Usuario 4**



## 3.2. Diagrama de secuencia Caso 6

Salta a la vista en la adición de un producto el hecho de haber dos caminos separados casi por completo, que dependen del tipo de producto a añadir, puesto que se espera del propietario que introduzca atributos distintos.

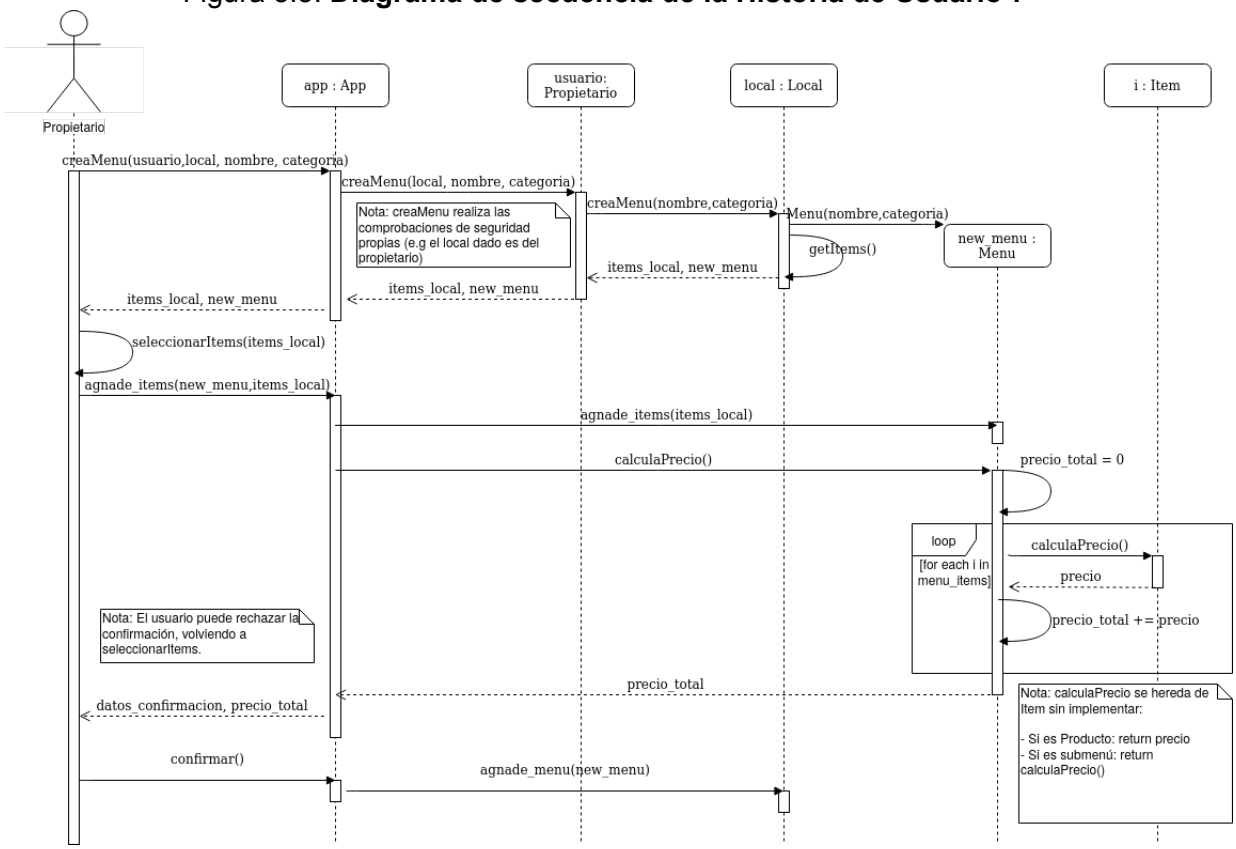
Figura 3.2: **Diagrama de secuencia de la Historia de Usuario 6**



## 3.3. Diagrama de secuencia Caso 7

De la historia de usuario 7, el registro de un menú, se destaca en especial la redefinición de *calculaPrecio()* en función del tipo de *Item* que sea a partir de herencias. De esta forma, conseguimos en el *Composite* la recursión en el cálculo de precio de un menú simplificando el futuro código evitando distinciones entre submenús y productos.

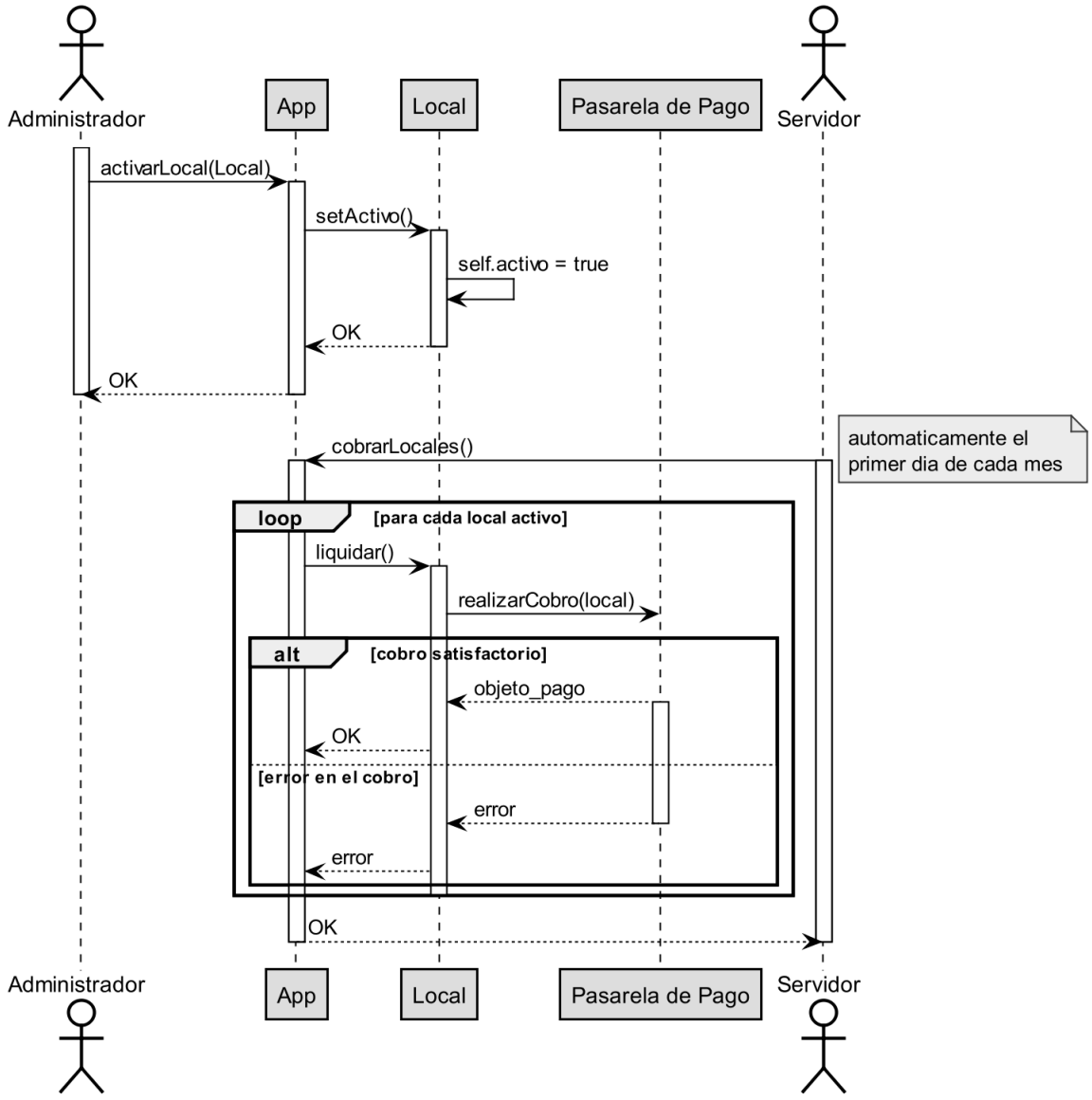
Figura 3.3: **Diagrama de secuencia de la Historia de Usuario 7**



## 3.4. Diagrama de secuencia Caso 12

La historia de usuario 12 se corresponde con la adición al sistema de un local al que realizar cobros y el cobro en sí. Vemos dos agentes claros, el administrador al añadir el local a la lista y el servidor que realizará el cobro de forma periódica. Además, figura una pasarela de pago externa con la que nos comunicaremos para efectuar las transacciones necesarias.

Figura 3.4: **Diagrama de secuencia de la Historia de Usuario 12**



# 4. Glosario

| Término | Descripción |
|---|---|
| Cliente-Servidor | Modelo de diseño de software en el que las tareas se reparten entre los proveedores de recursos o servicios, llamados servidores, y los demandantes, llamados clientes |
| Composite | Patrón de diseño utilizado para construir objetos complejos a partir de otros más simples y similares entre sí, gracias a la composición recursiva y a una estructura en forma de árbol. |
| Diseño | Etapa de la Ingeniería del Software enfocada a toda la actividad implicada en conceptualizar, enmarcar, implementar, poner en funcionamiento y, finalmente, modificar sistemas complejos; que sigue a la etapa de Análisis. |
| Herencia | Concepto de programación orientado a la reutilización y extensibilidad. Una clase hija ”hereda”de su clase padre obteniendo su comportamiento (métodos) y atributos. |
| HTTPS | Protocolo seguro de transferencia de hipertexto (en inglés, *Hypertext Transfer Protocol Secure*). Protocolo de aplicación basado en el protocolo HTTP, destinado a la transferencia segura de datos de hipertexto, es decir, es la versión segura de HTTP |
| REST API | Transferencia de estado representacional (en inglés *REpresentational State Transfer*). Estilo de arquitectura software para sistemas hipermedia distribuidos como la *World Wide Web*. |
| Servidor de Aplicaciones | Componente de un Sistema Informático encargada de tener implementada toda la lógica de negocio. |
| Singleton | Patrón de diseño que permite restringir la creación de objetos pertenecientes a una clase o el valor de un tipo a un único objeto. |