Orientación a Objetos 2 - 1er Parcial - 2012

1. Patrones

En una cafetería se desea implementar el pago de los pedidos procesados mediante diferentes medios de pagos (también conocidos como gateways). Por el momento se requiere la posibilidad de realizar el pago por medio de Débito automático, y DineroMail, sin embargo no se descarta la posibilidad de utilizar otros medios en el futuro.
Se distingue del modelo del sistema (Figura 1) la clase **Cliente** con variables de instancia para el nombre, apellido, mail y CBU (Clave Bancaria Uniforme) y sus correspondientes accesors. Y ,por otro lado, la clase **Pedido** que conoce el cliente que lo solicitó mediante la variable de instancia *cliente*, un mensaje **#monto** que calcula el costo del pedido y un mensaje **#pagarCon:unMedio.** Este último retorna true o false dependiendo si pudo o no realizar el pago.

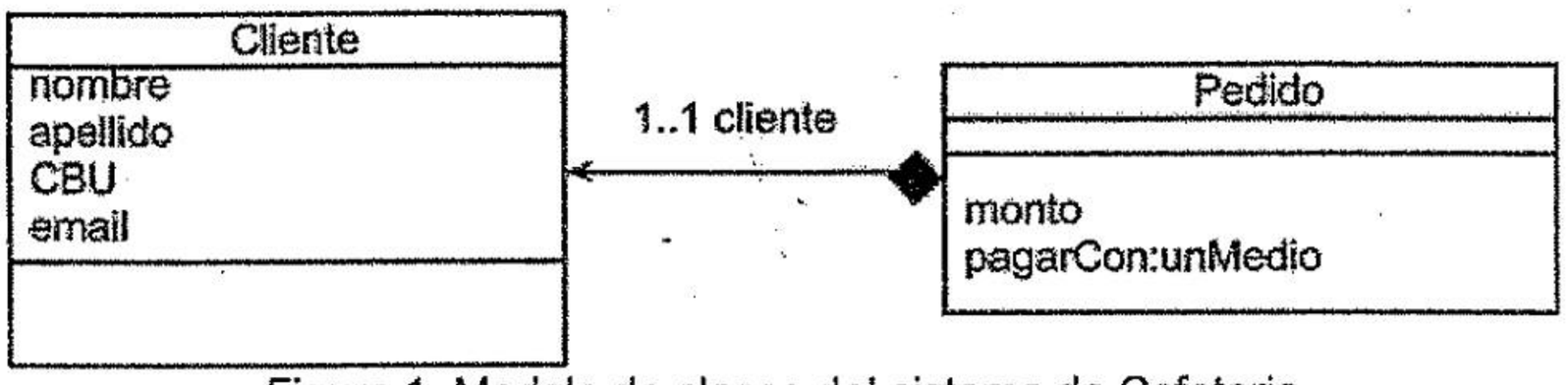

Figura 1. Modelo de clases del sistema de Cafeteria

Cada uno de los medios de pago dispone de una API con su correspondiente procedimiento de uso:

- Débito automático: El pago requiere una autorización previa al pago donde se valida si el cliente dispone de fondos. Para realizar el pago usted dispone de una clase llamada **DebitoGateway** con dos mensajes de clase: **#autorizar:unMonto conCBU:unCBU** y **#pagar:unMonto conCBU:unCBU**. El primero permite autorizar el monto de pago por un CBU retornando una colección de strings que representan los errores de autorización. El segundo método, permite realizar el pago si y sólo si la colección de errores es vacía.
- DineroMail: El pago se realiza en un solo paso. Para realizar el pago usted dispone de una clase llamada **DineroMailGateway** con un mensaje de clase **#realizarPago:unCorreo** que recibe como parámetro la cuenta de correo de cliente.

Tenga en cuenta que las API de los medio de pago no pueden ser modificados ya que es utilizado por varios sistemas.

Tareas

a. Diseñe una solución que permita soportar los diferentes medio de pagos, y definá e implemente el comportamiento del mensaje que permite realizar el pago de un pedido (**#pagarCon:unMedio**). Realice el diagrama de clases UML correspondiente.
b. Implemente en Smalltalk su diseño.
c. Muestre en un workspace cómo se instancia un pago por débito automático.

2. Refactoring

Se dispone de la clase *SupervisorWindow* que permite recolectar información asociado a un Supervisor tal como su nombre y apellido. En la Figura 2, se puede ver el modelo de clases donde clase *SupervisorAplicationModel* es una subclase de ApplicationModel, posee dos variables de instancias con sus accesors para las propiedades nombre y apellido y dos aspectos para las dos variables de instancia.
En la Figura 3, se puede apreciar la interfaz de usuario donde tras ingresar los datos del nombre y apellido se permite "grabar" los datos ingresados.
Los métodos de la clase *SupervisorApplicationModel* son:

```
SupervisorApplicationModel >>grabar
```