

Equipo docente: Mg. María Alejandra Vranic  
Lic. Romina Mansilla  
Lic. Gustavo Siciliano  
Lic. Ezequiel Scordamaglia

## Trabajo Práctico Integrador: Sistema para un comercio “Almacen Granate”

IDE: Eclipse IDE for Enterprise Java Developers <https://www.eclipse.org/>

UML: dia 0.97+git [live.gnome.org/Dia](http://live.gnome.org/Dia)

TP Grupal: 4 integrantes

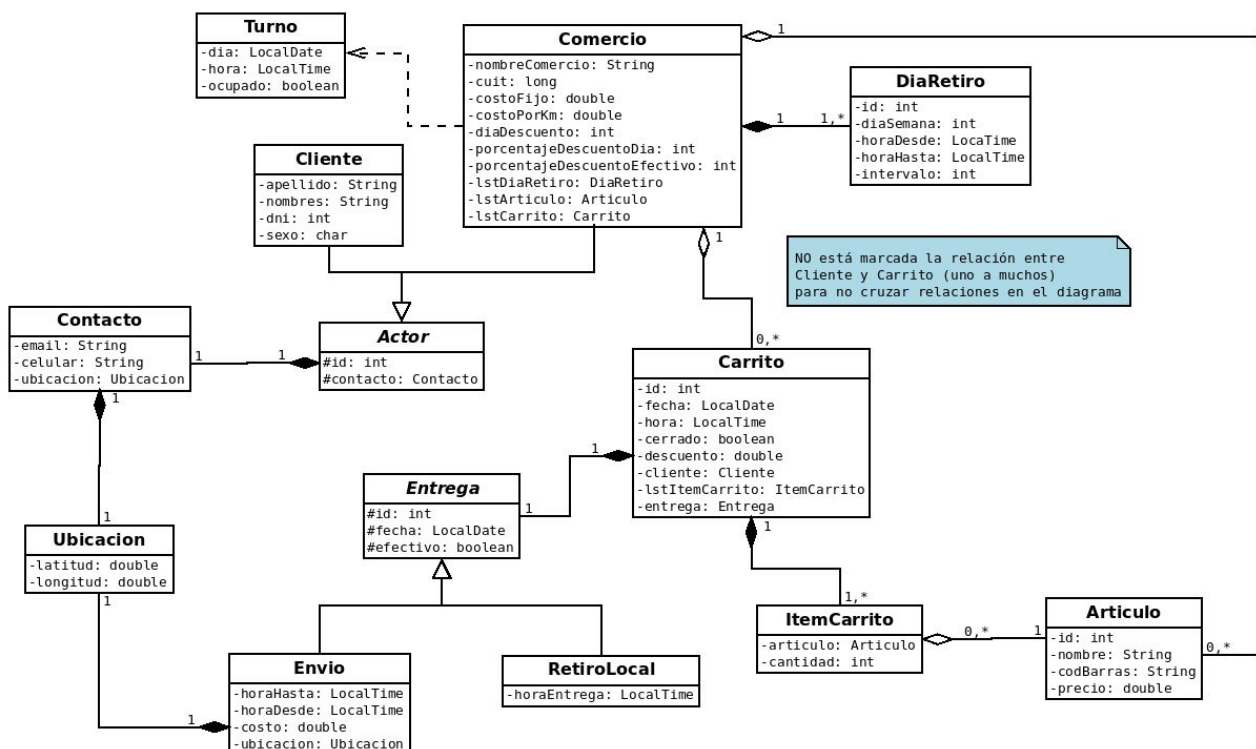
Evaluación: La evaluación de TP es personal se consideran los aportes de cada integrante al trabajo según cada informe y las defensas de cada uno de los integrantes en clase.

Defensas: En clase por grupos se exponen los avances del proyecto.

El sistema de laboratorio para un Comercio: “Almacen Granate”, debe permitir resolver la siguiente funcionalidad:

- Comprar digitalmente un carrito
- Gestionar turnos de retiro de la compra
- Gestionar entrega a domicilio
- Gestionar ofertas

La capa de modelo se ilustra en el siguiente Diagrama de clases:



El sistema debe implementar la siguiente funcionalidad:

### **Validar identificador:**

DNI: el ente nacional para validación de DNI es el ReNaPer para este TP de laboratorio solo validar que el ingreso sea numérico.

CUIT: según el Algoritmo: verifica un CUIT de la Guía 6 (Métodos Estáticos y Excepciones)

### **Envío a domicilio**

El costo de envío es el resultado de un costo fijo en función de la distancia del local al domicilio.

Para calcular la distancia se propone el siguiente método:

#### **Cálculo de distancia:**

```
public double distanciaCoord(double lat1, double lng1, double lat2, double lng2) {
    double radioTierra = 6371; //en kilómetros
    double dLat = Math.toRadians(lat2 - lat1);
    double dLng = Math.toRadians(lng2 - lng1);
    double sindLat = Math.sin(dLat / 2);
    double sindLng = Math.sin(dLng / 2);
    double va1 = Math.pow(sindLat, 2) + Math.pow(sindLng, 2) * Math.cos(Math.toRadians(lat1)) *
        Math.cos(Math.toRadians(lat2));
    double va2 = 2 * Math.atan2(Math.sqrt(va1), Math.sqrt(1 - va1));
    return radioTierra * va2;
}
```

### **Retirar en el negocio**

Para retirar en el negocio el comercio determina para cada día de la semana de entrega, la franja horaria y el intervalo de tiempo.

### **Descuento**

- 1) El comercio ofrece un día a la semana con la compra de 2 artículos iguales aplica un porcentaje de descuento al precio de la 2° unidad. Por ejemplo si compra 7 artículos iguales tendrá un descuento =  $3 * \text{precioArticulo} * \text{porcentajeDeDescuento} / 100$
- 2) El comercio sobre la compra aplica un porcentaje de descuento sobre el costo del carrito algunos días o si el cliente realiza el pago en efectivo o débito.

El CU que determina el descuento sobre el carrito aplicará el mayor de los dos de los descuentos sobre la compra.

### **Agregar al carrito**

Cuando el cliente agrega al carrito un artículo y cantidad se debe verificar si existe un objeto ItemCarrito para ese artículo, en caso que existe se incrementa la cantidad de lo contrario se crea y se agrega a la lista.

### **Sacar del carrito**

Cuando el cliente saca del carrito una cantidad de un artículo, si la cantidad es menor la resta y si es igual elimina el objeto ItemCarrito.

Algunos de los CU que debe implementar el sistema:

- 1) # validarIdentificadorUnico():boolean //valida DNI o CUIT según la sub-clase
- 2) + traerHoraRetiro (LocalDate fecha): LocalTime
- 3) + generarTurnosLibres (LocalDate fecha) : Turno //retorna una lista de objetos Turno libres

- 4) + traerTurnosOcupados(LocalDate fecha):Turno //retorna una lista de objetos Turno dados
- 5) + generarAgenda (LocalDate fecha) : Turno //retorna una lista de objetos Turno indicando si está ocupado o libre.
- 6) + agregarDiaRetiro(int diaSemana, LocalTime horaDesde, LocalTime horaHasta, int intervalo):boolean
- 7) + validarCodBarras():boolean //Se valida por el sistema EAN -13  
[http://www.alimentosargentinos.gob.ar/contenido/revista/pdfs/07/07\\_03\\_codigo.htm](http://www.alimentosargentinos.gob.ar/contenido/revista/pdfs/07/07_03_codigo.htm)
- 8) + agregar(Articulo articulo, int cantidad):boolean
- 9) + calcularSubTotalItem():double
- 10) + calcularTotalCarrito() : doble
- 11) + calcularDescuentoDia (int diaDescuento, double porcentajeDescuento):double
- 12) + calcularDescuentoEfectivo (double porcentajeDescuentoEfectivo): double
- 13) + calcularDescuentoCarrito (int diaDescuento, double porcentajeDescuento,double porcentajeDescuentoEfectivo):double //determina cual es el descuento mayor
- 14) + setDescuento():
- 15) + totalAPagarCarrito() : double
- 16) + setCosto(Ubicacion ubicacion, double costoFijo, double costoPorKm):
- 17) + traerUbicacion() : Ubicacion

### **Capa Test**

La capa de testeo debe crear los objetos necesarios en una clase ejecutable para los posibles escenarios según el caso de uso a testear.