Universidad Nacional de Río Cuarto
Facultad de Ciencias Exactas, Físico-Químicas y Naturales
Departamento de Computación
Asignatura: **ANÁLISIS Y DISEÑO DE SISTEMAS (3303)**
Año 2017

PRÁCTICO Nº 5

TEMA: Diagramas de Secuencia

Objetivos

Con este práctico se espera que el estudiante pueda:

- Modelar la interacción entre los componentes principales del sistema.
- Visualizar mediante un diagrama de secuencia la comunicación y colaboración a través del tiempo de los objetos participantes.
- Describir funcionalidades identificadas en la etapa de análisis y detallarlas en la etapa de diseño.

1. Video Club

El diagrama de clases que se encuentra a continuación provee una vista estática del sistema de administración de un video club. Utilizando los Diagramas de Secuencia de UML describa la secuencia de mensajes necesarios para cumplir con cada uno de los escenarios presentados.

En todos los casos asuma que la interacción inicia desde el método *main()* de la clase principal.

a) El dueño del video club quiere registrar en el sistema el género “Comedia Romántica”. Note que la clase **VideoAdm** cuenta con el método *crearNuevoGenero*, el mismo deberá ser utilizado y descripto en la secuencia de mensajes.

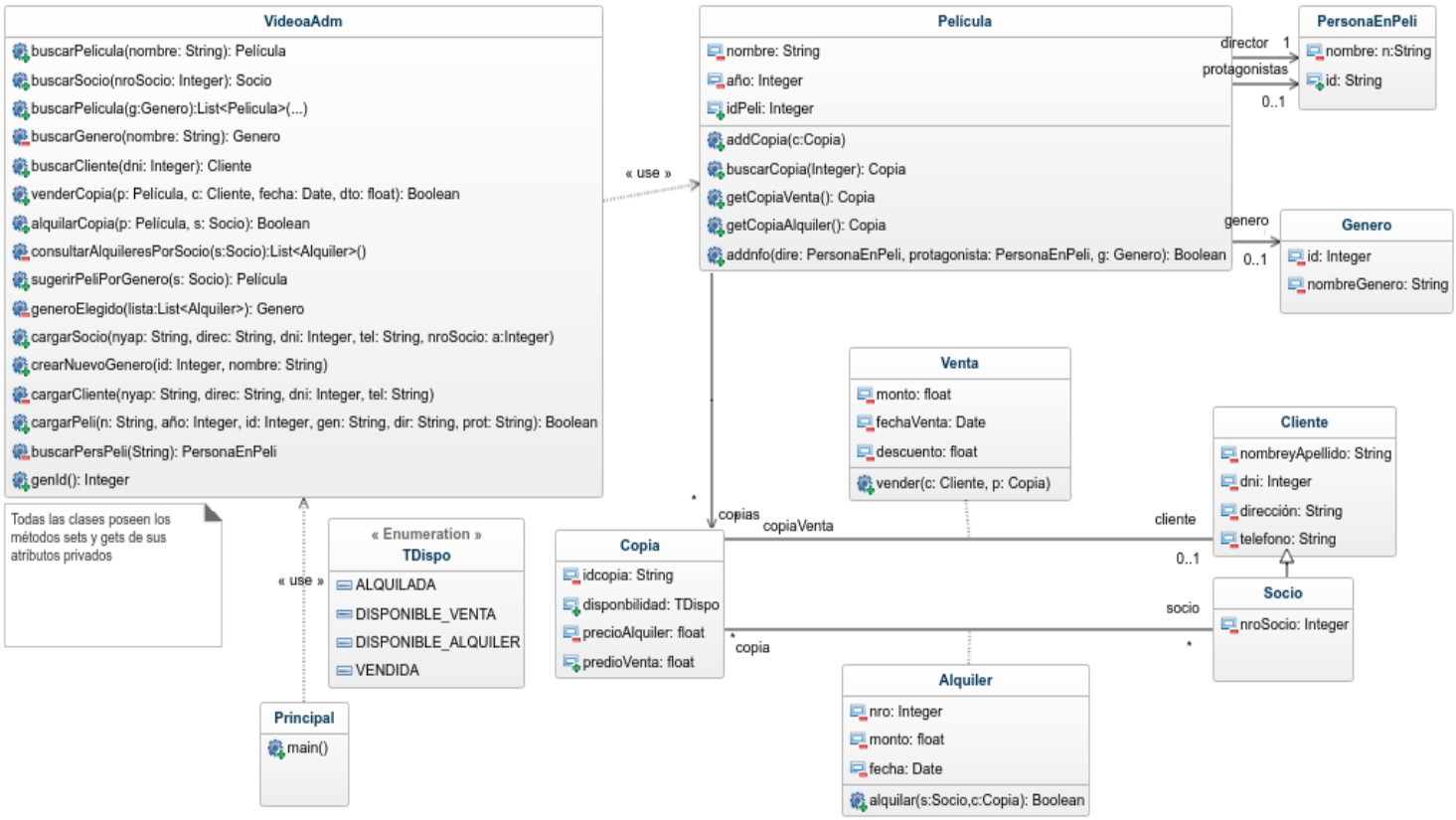
b) Un empleado del video club desea obtener la dirección del socio “Jimmy Page” nro. socio 1734. La secuencia de mensajes deberá acceder a través de la clase *VideoAdm*.

c) El dueño desea actualizar la dirección del cliente “Brian May” DNI 14.253.899 quién cambió su domicilio a “Queen 7240”.

d) El dueño del video club compró una copia más de la película “Loco por Mary”. La película ya se encuentra registrada en el sistema. La copia nueva posee un precio de Alquiler de $30 y un precio de venta de $500. Por defecto las copias que se cargan en el sistema se setean como DISPONIBLE_ALQUILER.

e) El empleado del video club desea registrar el alquiler de una copia de la película “T.N.T.” por parte del socio “Angus Young” nro. socio 666.

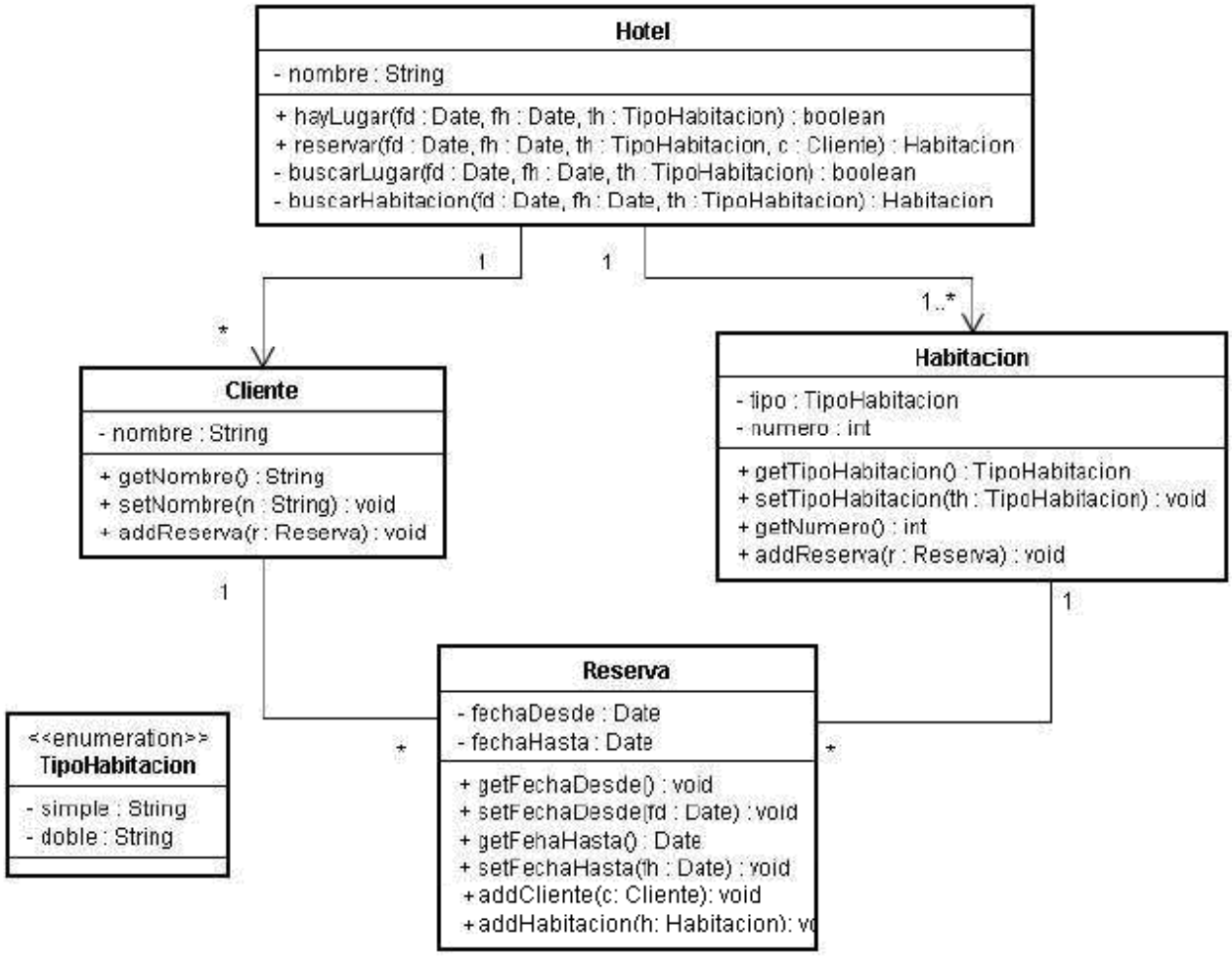
f) El señor “Jimi Hendrix” DNI 12.999.858 domiciliado en “Sarmiento 1078”, tel. 4645443 desea alquilar una copia de la película “Red House” que se encuentra en el video club. Jimi no es socio, por lo que antes de poder registrar el alquiler va a ser necesario cargarlo al sistema como socio. Dentro de la secuencia de mensajes propuesta deberá utilizar el método cargarSocio de la clase VideoAdmin.



2. Hotel California

Dado el siguiente Diagrama de Clases, modelar mediante un diagrama de secuencia el siguiente escenario: “El señor Roberto Rodriguez consulta por disponibilidad de una habitación simple por tres días en el hotel California a partir del 20/05/17, el encargado del hotel le informa que hay disponibilidad por lo que el cliente decide realizar la reservación”.

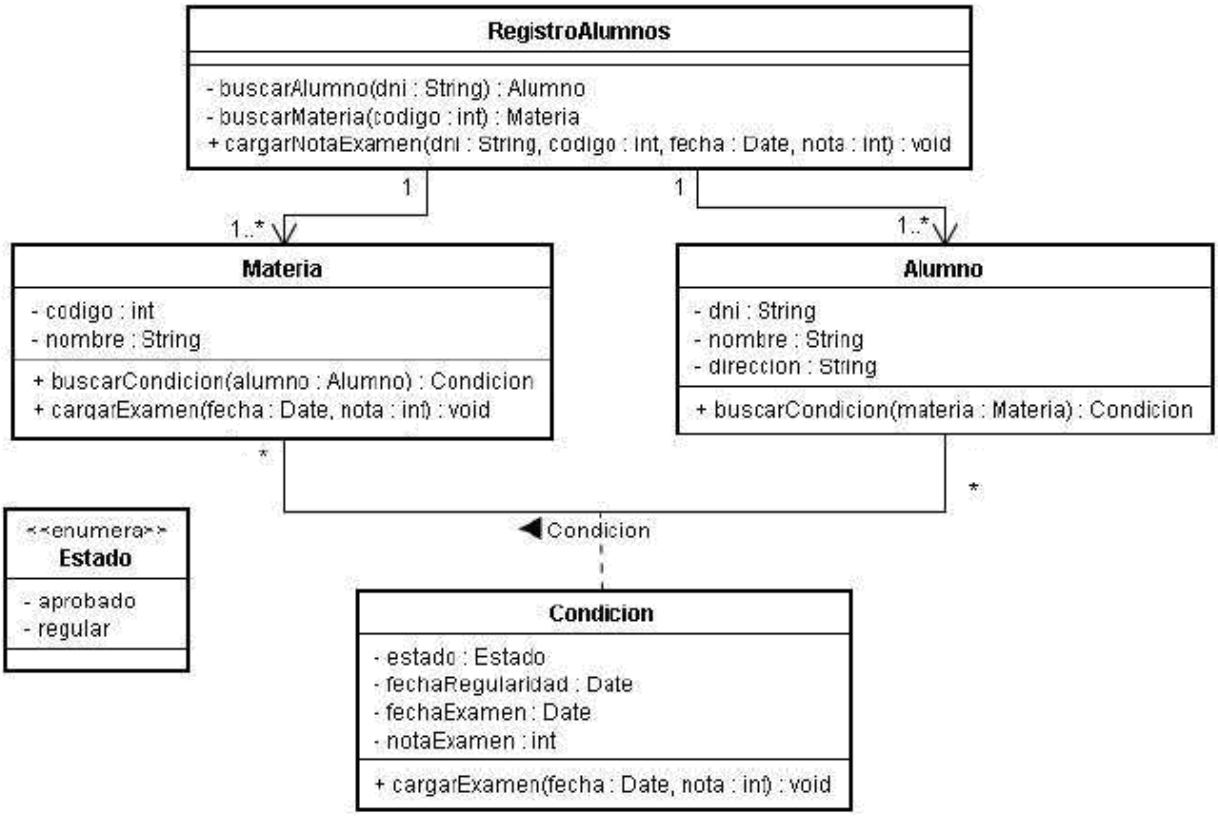
Nota: La secuencia de mensajes debe iniciar a través de un método *main*() de una clase *Principal.* La secuencia deberá describir el comportamiento del método *reservar*.



3. Sistema Integral de Alumnos (SIAL)

Dado el siguiente Diagrama de Clases, modelar mediante un diagrama de secuencia el siguiente escenario: “El Registro de Alumnos debe cargar la nota obtenida por el alumno Clark Kent con DNI 30.255.310 en la materia Análisis y Diseño de Sistemas, con código 3303. El alumno en esta materia está en condición de regular. La nota a registrar es un 10 (diez), y la fecha es 21/03/2017”.

Nota: La secuencia de mensajes debe iniciar a través de un método *main*() de una clase *Principal.* La secuencia deberá describir el comportamiento del método *cargarNotaExamen*.



4. Ejecución de programa

Dado el siguiente código escrito en el lenguaje JAVA, realizar un diagrama de secuencia de UML que modele el escenario o secuencia de interacciones que se desencadenan si se ejecuta ***menu.ejecutar(2)*** de la clase ***Principal***.

```
public class Menu {
    protected Vector comandos = new Vector();

    public void agregarComando(Comando c) {……..}
    public void suprimirComando(Comando c) {.....}
    public Comando getComando(int i) {.....}
    public void ejecutar(int i) {
            Comando c = this.getComando(i);
            if (c != null) {
                    c.ejecutar();
            }
        }
}
```

```
public class Dato {
        protected String valor;

        public Dato() {}
        public void m1() {…....}
        public void m2() {…....}
        public void m3() {…....}
}
```

```
public abstract class Comando {
        protected Dato dato;
        public void setDato(Dato d) {...}
        public Dato getDato() {...}
        abstract public void ejecutar();
}
public class Comando1 extends Comando{
        public void ejecutar() {
                dato.m1();
                dato.m2();
        }
}
public class Comando2 extends Comando{
        public void ejecutar() {
                dato.m2();
                dato.m3();
        }
}
public class Comando3 extends Comando{
        public void ejecutar() {
                dato.m3();
                dato.m1();
        }
}
```

```
public class Principal {
        public static void main(String[ ] args) {
        Menu menu = new Menu();
        Dato dato = new Dato();

        Comando comando1 = new Comando2();
        comando1.setDato(dato);
        menu.agregarComando(comando1);

        Comando comando2 = new Comando2();
        comando2.setDato(dato);
        menu.agregarComando(comando2);

        Comando comando3 = new Comando2();
        comando3.setDato(dato);
        menu.agregarComando(comando3);

        menu.ejecutar(2);
        }
}
```

5. Restaurante “A mi Gusto”

Dado el modelo de clases del restaurante “A mi Gusto”, modele el siguiente escenario con un Diagrama de Secuencia UML. “El cliente Aníbal Guevara, DNI 16.4587.96, desea abonar el monto total de su cena en efectivo, realizada el día 12/03/2008 a las 12 hs. y fue atendido por el mesero “Morales”. El cliente ha consumido un plato especial del día y una gaseosa. La propina del mozo corresponde al 10% del monto total y el impuesto al 3 % del mismo”.

calcularMontoTotal: calcula el monto total de la consumición, sumando el monto de la comida, más el adicional por impuestos y la propina del mozo.

buscarCliente: retorna un cliente específico.

buscarOrden*: retorna una orden específica.*

calcularOrden*: calcula el monto correspondiente a la comida, sumando el monto de la bebida más el monto del plato.*

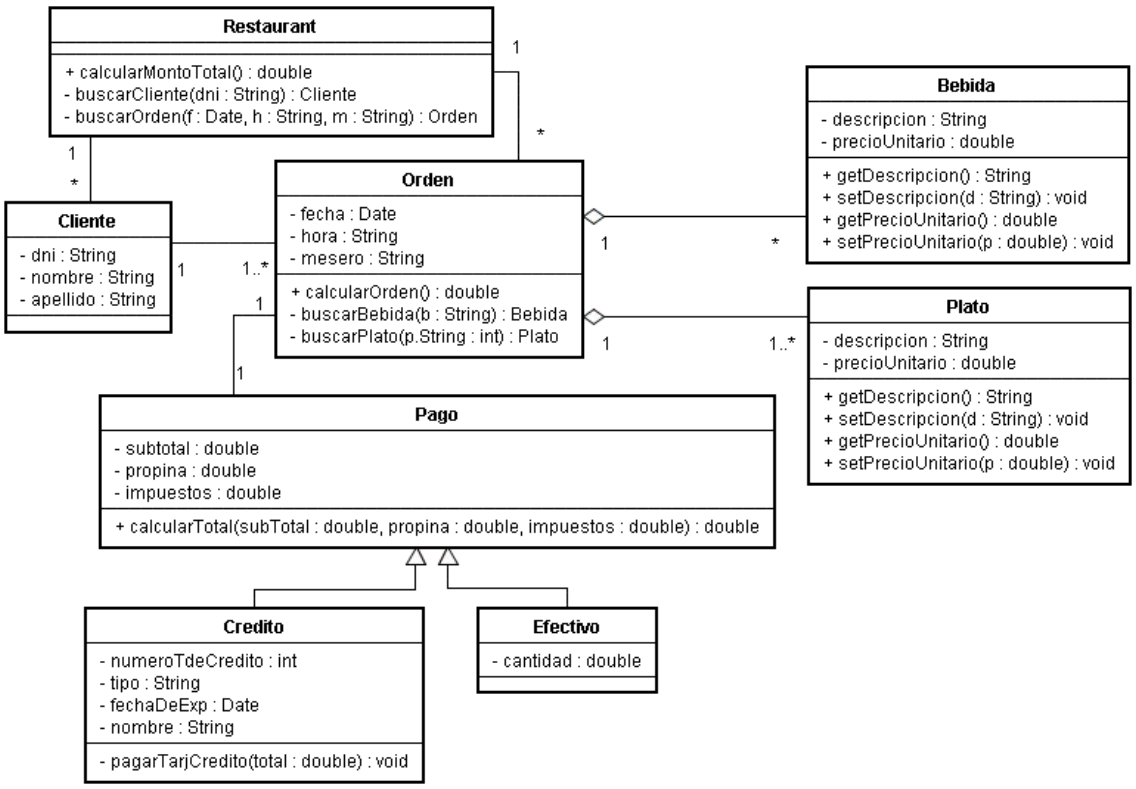
buscarBebida*: retorna una bebida específica.*

buscarPlato*: retorna un plato específico.*

calcularTotal*: calcula el monto total de la consumición, dados el subtotal (bebida +*

plato), el valor de la propina y el de los impuestos.

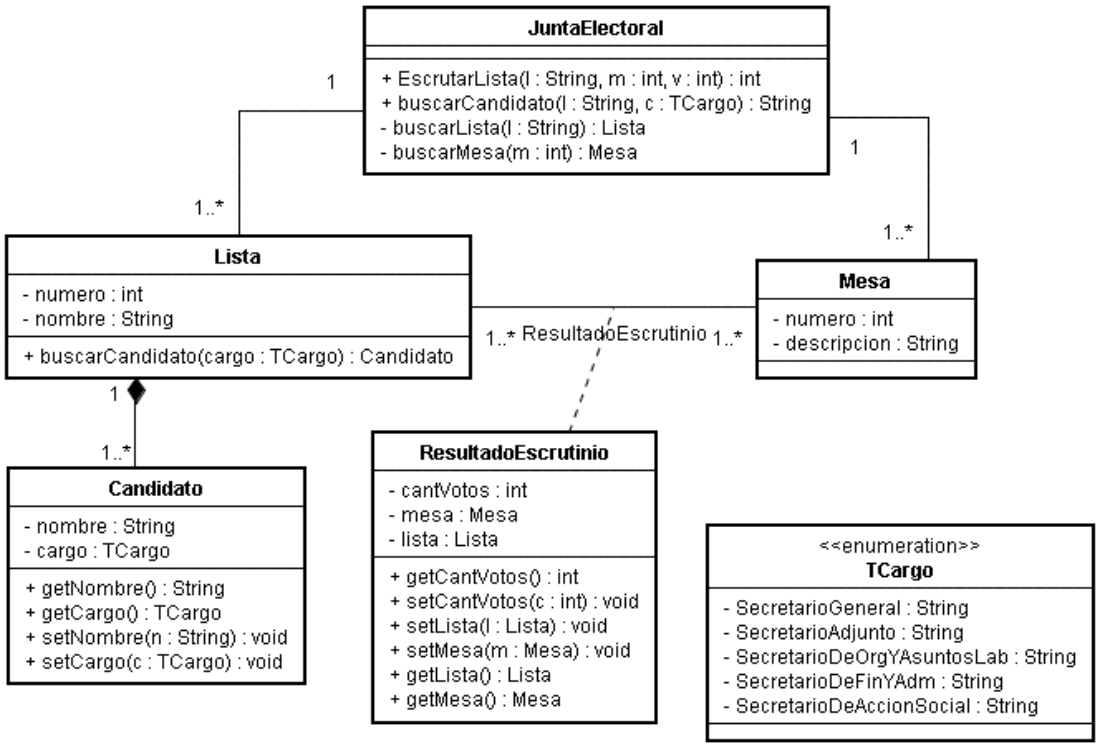
Nota: La secuencia de mensajes debe iniciar a través de un método *main*() de una clase *Principal.* La secuencia deberá describir el comportamiento del método *calcularMontoTotal()*.



6. Sistema Electoral

Dado el siguiente diagrama de clases, modelar mediante un Diagrama de Secuencia de UML el siguiente escenario: “Se desea registrar la cantidad de votos obtenidos por la lista “Siempre Unidos” en la mesa 145 en la última elección de la Comisión Directiva de una organización social. Esta cantidad es de 128 votos. Además, se quiere obtener el nombre del candidato titular al cargo de Secretario General.

Nota: La secuencia de mensajes debe iniciar a través de un método *main*() de una clase *Principal.* La secuencia deberá describir el comportamiento del método *EscrutarLista*.


JuntaElectoral
+ EscrutarLista(l : String, m : int, v : int) : int
+ buscarCandidato(l : String, c : TCargo) : String
- buscarLista(l : String) : Lista
- buscarMesa(m : int) : Mesa
1
1
1..*
1..*
Lista
- numero : int
- nombre : String
+ buscarCandidato(cargo : TCargo) : Candidato
1..* ResultadoEscrutinio 1..*
Mesa
- numero : int
- descripcion : String
1
1..*
Candidato
- nombre : String
- cargo : TCargo
+ getNombre() : String
+ getCargo() : TCargo
+ setNombre(n : String) : void
+ setCargo(c : TCargo) : void
ResultadoEscrutinio
- cantVotos : int
- mesa : Mesa
- lista : Lista
+ getCantVotos() : int
+ setCantVotos(c : int) : void
+ setLista(l : Lista) : void
+ setMesa(m : Mesa) : void
+ getLista() : Lista
+ getMesa() : Mesa
<<enumeration>>
TCargo
- SecretarioGeneral : String
- SecretarioAdjunto : String
- SecretarioDeOrgYAsuntosLab : String
- SecretarioDeFinYAdm : String
- SecretarioDeAccionSocial : String