



PROGRAMACIÓN II

Trabajo Práctico 5: Relaciones UML 1 a 1

Alumno: Victor Barroeta

DNI: 95805903

Comisión: M2025 – 2

Matricula: 100393

Link de Github:

https://github.com/victormbar/UTN_Prog2_VictorBarroeta/tree/main/javaUTN_TPs/src/TPs_JAVA_TP5

OBJETIVO GENERAL

Modelar clases con relaciones 1 a 1 utilizando diagramas UML. Identificar correctamente el tipo de relación (asociación, agregación, composición, dependencia) y su dirección, y llevarlas a implementación en Java.

MARCO TEÓRICO

Concepto	Aplicación en el proyecto
Asociación	Relación entre clases con referencia mutua o directa, puede ser uni o bidireccional
Agregación	Relación de "tiene un" donde los objetos pueden vivir independientemente
Composición	Relación fuerte de contención, el ciclo de vida del objeto contenido depende del otro
Dependencia de uso	Una clase usa otra como parámetro en un método, sin almacenarla como atributo
Dependencia de creación	Una clase crea otra en tiempo de ejecución, sin mantenerla como atributo
Asociación	Relación entre clases con referencia mutua o directa, puede ser uni o bidireccional
Agregación	Relación de "tiene un" donde los objetos pueden vivir independientemente



Caso Práctico

Desarrollar los siguientes ejercicios en Java. Cada uno deberá incluir:

- Diagrama UML
- Tipo de relación (asociación, agregación, composición, dependencia)
- Dirección (unidireccional o bidireccional)
- Implementación de las clases con atributos y relaciones definidas

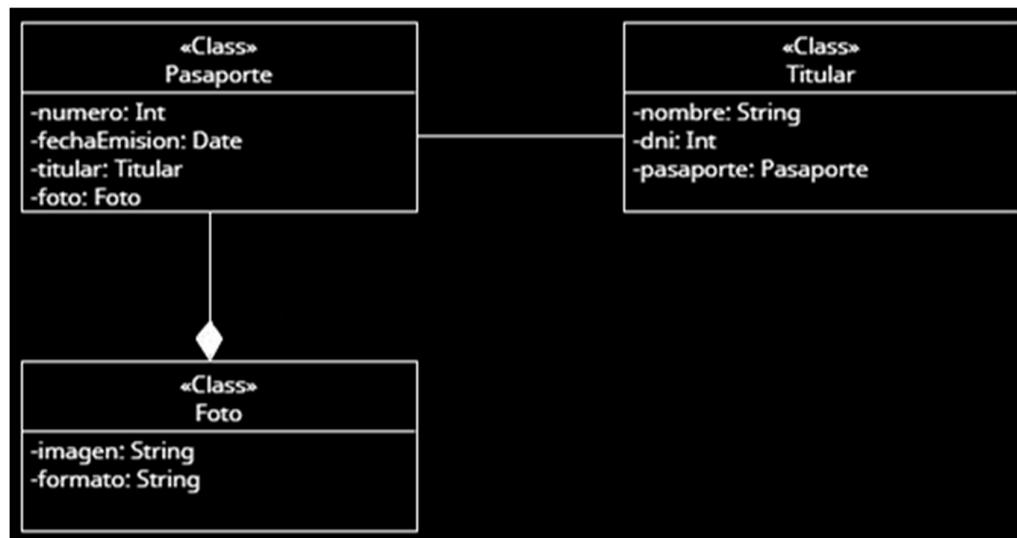
Ejercicios de Relaciones 1 a 1

1. Pasaporte - Foto - Titular
 - a. Composición: **Pasaporte → Foto**
 - b. Asociación bidireccional: **Pasaporte ↔ Titular**

Clases y atributos:

- i. Pasaporte: numero, fechaEmision
- ii. Foto: imagen, formato
- iii. Titular: nombre, dni

Diagrama UML



Código:



```

import java.util.Date;

public class ejercicio1 {

    public class Pasaporte{
        private int numero;
        private Date fechaEmision;
        private Titular titular;
        private Foto foto;

        public Pasaporte(int numero, Date fechaEmision, Titular titular, Foto foto) {
            this.numero = numero;
            this.fechaEmision = fechaEmision;
            this.titular = titular;
            this.foto = foto;
        }
    };

    public class Foto{
        private String imagen;
        private String formato;

        public Foto(String imagen, String formato) {
            this.imagen = imagen;
            this.formato = formato;
        }
    }

    public class Titular{
        private String nombre;
        private int dni;
        private Pasaporte pasaporte;

        public Titular(String nombre, int dni, Pasaporte pasaporte) {
            this.nombre = nombre;
            this.dni = dni;
            this.pasaporte = pasaporte;
        }
    }
}

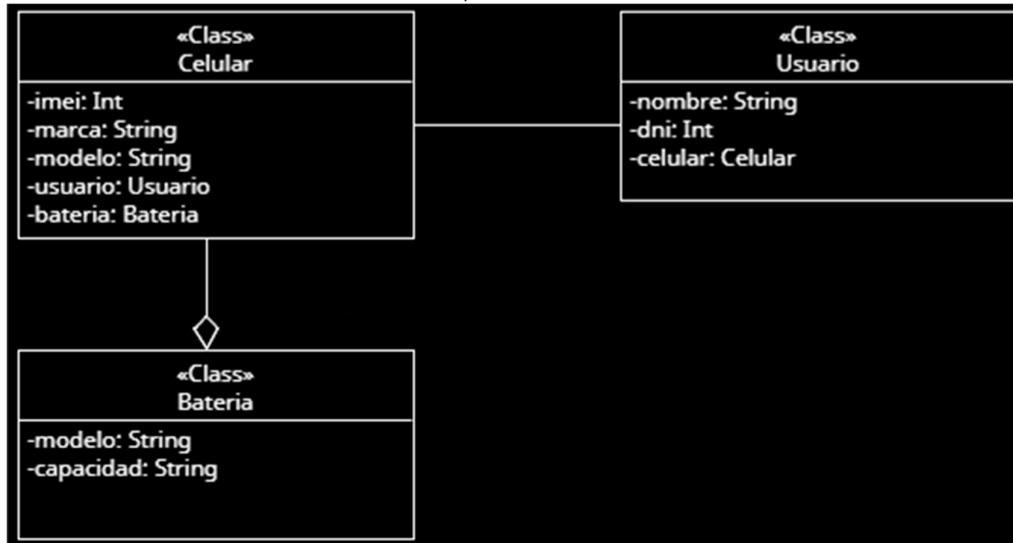
```

2. Celular - Batería - Usuario

- a. Agregación: **Celular → Batería**
- b. Asociación bidireccional: **Celular ↔ Usuario**

Clases y atributos:

- i. Celular: imei, marca, modelo
- ii. Batería: modelo, capacidad
- iii. Usuario: nombre, dni





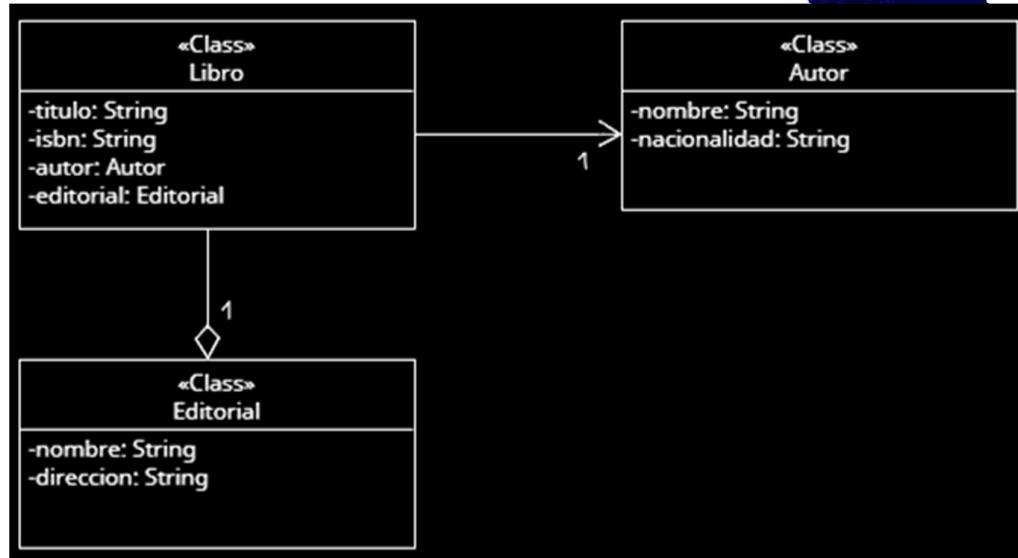
```
public class ejercicio2 {  
  
    public class Celular{  
        private int imei;  
        private String marca;  
        private String modelo;  
        private Usuario usuario;  
        private Bateria bateria;  
  
        public Celular(int imei, String marca, String modelo, Usuario usuario,  
                      Bateria bateria) {  
            this.imei = imei;  
            this.marca = marca;  
            this.modelo = modelo;  
            this.usuario = usuario;  
            this.bateria = bateria;  
        }  
    };  
  
    public class Bateria{  
        private String modelo;  
        private String capacidad;  
  
        public Bateria(String modelo, String capacidad) {  
            this.modelo = modelo;  
            this.capacidad = capacidad;  
        }  
    }  
  
    public class Usuario{  
        private String nombre;  
        private int dni;  
        private Celular celular;  
  
        public Usuario(String nombre, int dni, Celular celular) {  
            this.nombre = nombre;  
            this.dni = dni;  
            this.celular = celular;  
        }  
    };  
}
```

3. Libro - Autor - Editorial

- Asociación unidireccional: **Libro → Autor**
- Agregación: **Libro → Editorial**

Clases y atributos:

- Libro: titulo, isbn
- Autor: nombre, nacionalidad
- Editorial: nombre, dirección



```

/*
 * @author vcmam
 */
public class ejercicio3 {

    public class Libro{
        private String titulo, isbn;
        private Autor autor;
        private Editorial editorial;

        public Libro(String titulo, String isbn, Autor autor, Editorial editorial) {
            this.titulo = titulo;
            this.isbn = isbn;
            this.autor = autor;
            this.editorial = editorial;
        }
    }

    public class Autor{
        private String nombre, nacionalidad;

        public Autor(String nombre, String nacionalidad) {
            this.nombre = nombre;
            this.nacionalidad = nacionalidad;
        }
    }

    public class Editorial{
        private String nombre, direccion;

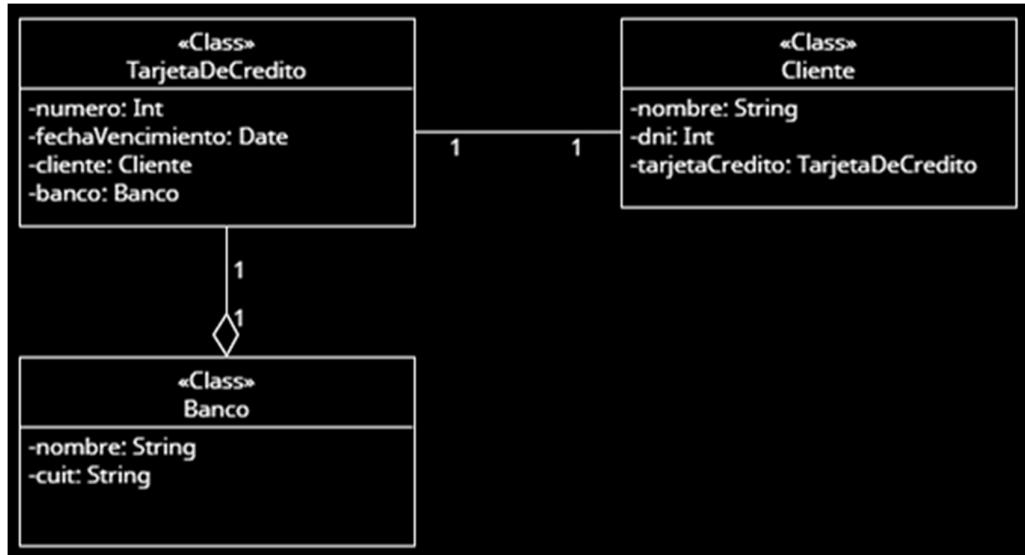
        public Editorial(String nombre, String direccion) {
            this.nombre = nombre;
            this.direccion = direccion;
        }
    }
}
  
```



4. TarjetaDeCrédito - Cliente - Banco
- Asociación bidireccional: **TarjetaDeCrédito ↔ Cliente**
 - Agregación: **TarjetaDeCrédito → Banco**

Clases y atributos:

- TarjetaDeCrédito: numero, fechaVencimiento
- Cliente: nombre, dni
- Banco: nombre, cuit





```

import java.util.Date;

public class ejercicio4 {

    public class TarjetaDeCredito{
        private int numero;
        private Date fechaVencimiento;
        private Cliente cliente;
        private Banco banco;

        public TarjetaDeCredito(int numero, Date fechaVencimiento, Cliente cliente, Banco banco) {
            this.numero = numero;
            this.fechaVencimiento = fechaVencimiento;
            this.cliente = cliente;
            this.banco = banco;
        }
    }

    public class Cliente{
        private String nombre;
        private int dni;
        private TarjetaDeCredito tarjetaDeCredito;

        public Cliente(String nombre, int dni, TarjetaDeCredito tarjetaDeCredito) {
            this.nombre = nombre;
            this.dni = dni;
            this.tarjetaDeCredito = tarjetaDeCredito;
        }
    }

    public class Banco{
        private String nombre, cuit;

        public Banco(String nombre, String cuit) {
            this.nombre = nombre;
            this.cuit = cuit;
        }
    }
}

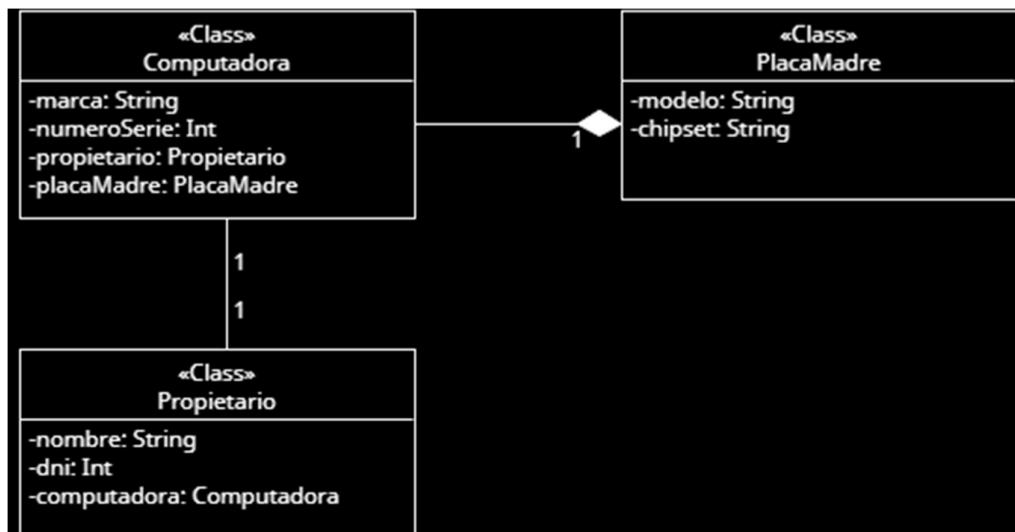
```

5. Computadora - PlacaMadre - Propietario

- a. Composición: **Computadora → PlacaMadre**
- b. Asociación bidireccional: **Computadora ↔ Propietario**

Clases y atributos:

- i. Computadora: marca, numeroSerie
- ii. PlacaMadre: modelo, chipset
- iii. Propietario: nombre, dni





```

/*
 * @author vcmn
 */
public class ejercicio5 {

    public class Computadora{
        private String marca;
        private int numSerie;
        private Propietario propietario;
        private PlacaMadre placaMadre;

        public Computadora(String marca, int numSerie, Propietario propietario, PlacaMadre placaMadre) {
            this.marca = marca;
            this.numSerie = numSerie;
            this.propietario = propietario;
            this.placaMadre = placaMadre;
        }
    }

    public class Propietario{
        private String nombre;
        private int dni;
        private Computadora computadora;

        public Propietario(String nombre, int dni, Computadora computadora) {
            this.nombre = nombre;
            this.dni = dni;
            this.computadora = computadora;
        }
    }

    public class PlacaMadre{
        private String modelo, chipset;

        public PlacaMadre(String modelo, String chipset) {
            this.modelo = modelo;
            this.chipset = chipset;
        }
    }
}

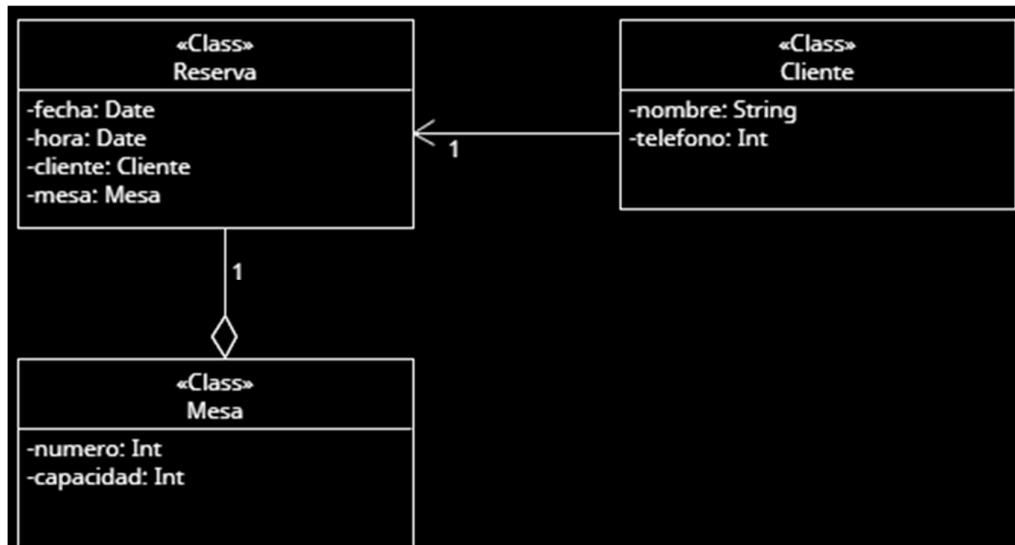
```

6. Reserva - Cliente - Mesa

- a. Asociación unidireccional: **Reserva → Cliente**
- b. Agregación: **Reserva → Mesa**

Clases y atributos:

- i. Reserva: fecha, hora
- ii. Cliente: nombre, telefono
- iii. Mesa: numero, capacidad





```
import java.util.Date;

public class ejercicio6 {

    public class Reserva{
        private Date fecha, hora;
        private Cliente cliente;
        private Mesa mesa;

        public Reserva(Date fecha, Date hora, Cliente cliente, Mesa mesa) {
            this.fecha = fecha;
            this.hora = hora;
            this.cliente = cliente;
            this.mesa = mesa;
        }
    }

    public class Cliente{
        private String nombre;
        private int dni;

        public Cliente(String nombre, int dni) {
            this.nombre = nombre;
            this.dni = dni;
        }
    }

    public class Mesa{
        private int numero, capacidad;

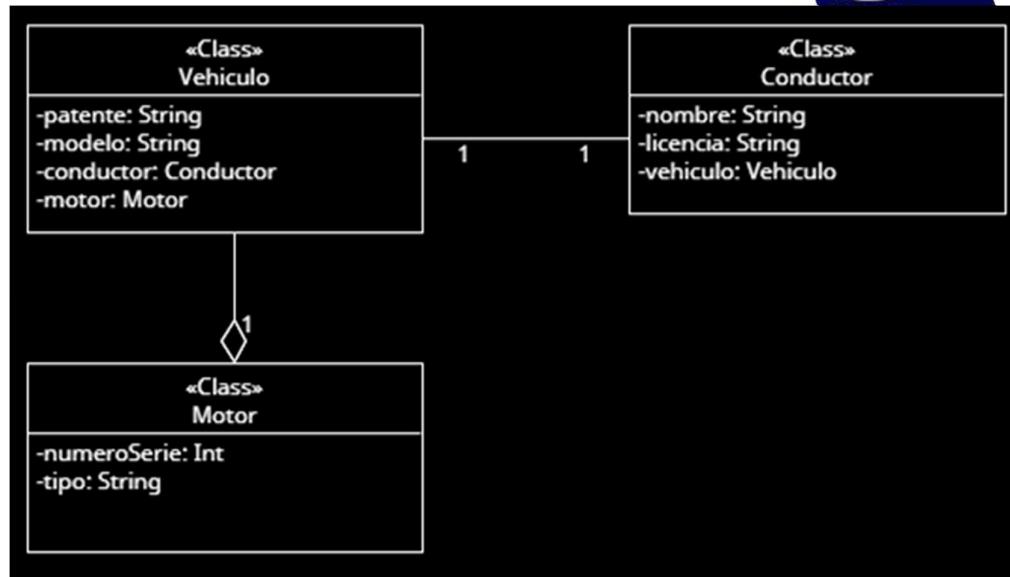
        public Mesa(int numero, int capacidad) {
            this.numero = numero;
            this.capacidad = capacidad;
        }
    }
}
```

7. Vehículo - Motor - Conductor

- Agregación: **Vehículo → Motor**
- Asociación bidireccional: **Vehículo ↔ Conductor**

Clases y atributos:

- Vehículo: patente, modelo
- Motor: tipo, numeroSerie
- Conductor: nombre, licencia



```

/*
 * @author vcmam
 */
public class ejercicio7 {

    public class Vehiculo{
        private String patente, modelo;
        private Conductor conductor;
        private Motor motor;

        public Vehiculo(String patente, String modelo, Conductor conductor, Motor motor) {
            this.patente = patente;
            this.modelo = modelo;
            this.conductor = conductor;
            this.motor = motor;
        }
    }

    public class Conductor{
        private String nombre, licencia;
        private Vehiculo vehiculo;

        public Conductor(String nombre, String licencia, Vehiculo vehiculo) {
            this.nombre = nombre;
            this.licencia = licencia;
            this.vehiculo = vehiculo;
        }
    }

    public class Motor{
        private int numeroSerie;
        private String tipo;

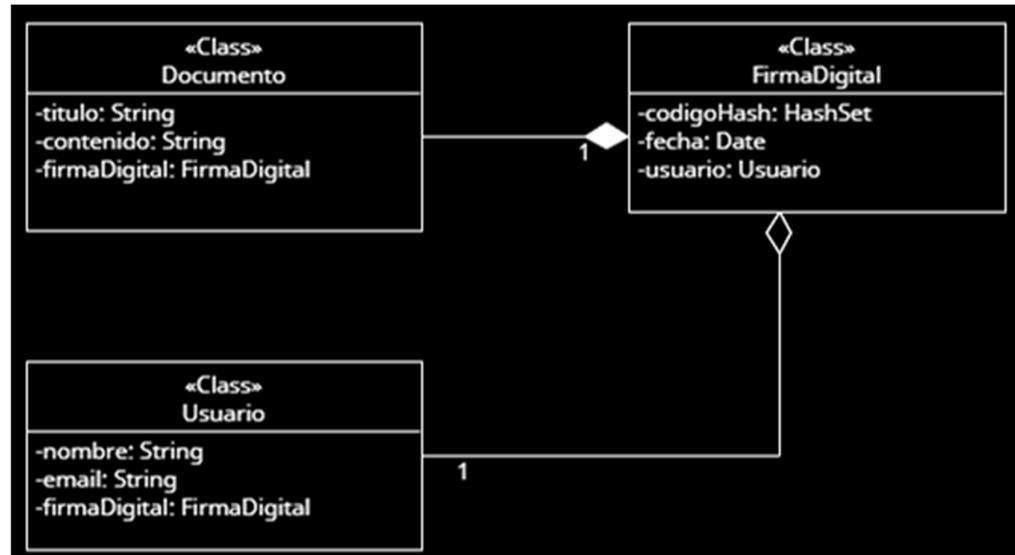
        public Motor(int numeroSerie, String tipo) {
            this.numeroSerie = numeroSerie;
            this.tipo = tipo;
        }
    }
}
  
```

8. Documento - FirmaDigital - Usuario
 - a. Composición: **Documento → FirmaDigital**
 - b. Agregación: **FirmaDigital → Usuario**

Clases y atributos:



- i. Documento: titulo, contenido
- ii. FirmaDigital: codigoHash, fecha
- iii. Usuario: nombre, email



```

import java.util.Date;
import java.util.HashSet;

public class ejercicio8 {

    public class Documento{
        private String titulo, contenido;
        private FirmaDigital firmaDigital;

        public Documento(String titulo, String contenido, FirmaDigital firmaDigital) {
            this.titulo = titulo;
            this.contenido = contenido;
            this.firmaDigital = firmaDigital;
        }
    }

    public class FirmaDigital{
        private HashSet codigoHash;
        private Date fecha;
        private Usuario usuario;

        public FirmaDigital(HashSet codigoHash, Date fecha, Usuario usuario) {
            this.codigoHash = codigoHash;
            this.fecha = fecha;
            this.usuario = usuario;
        }
    }

    public class Usuario{
        private String nombre, email;
        private FirmaDigital firmaDigital;

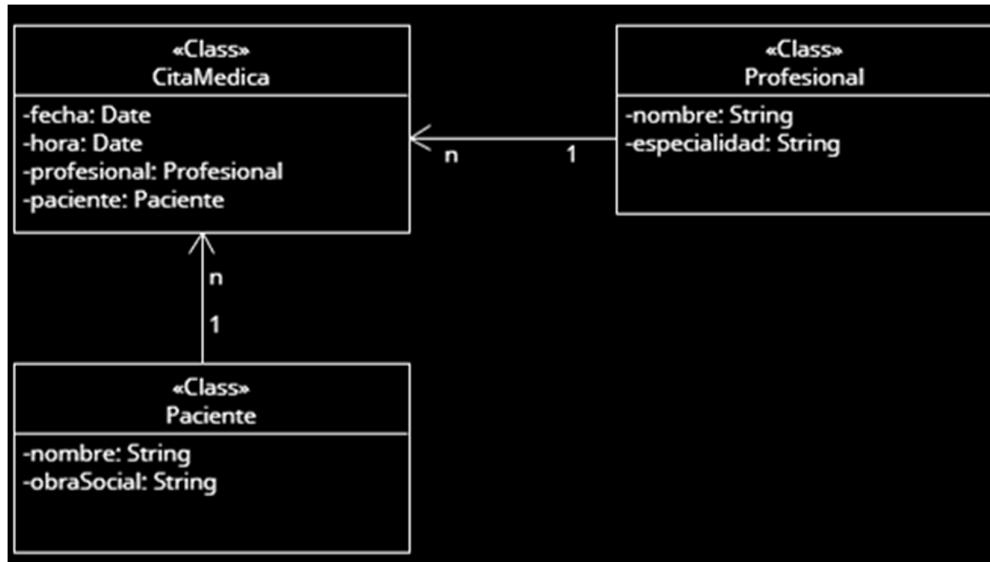
        public Usuario(String nombre, String email, FirmaDigital firmaDigital) {
            this.nombre = nombre;
            this.email = email;
            this.firmaDigital = firmaDigital;
        }
    }
}
  
```

9. CitaMédica - Paciente - Profesional
- Asociación unidireccional: **CitaMédica → Paciente**,
 - Asociación unidireccional: **CitaMédica → Profesional**



Clases y atributos:

- i. CitaMédica: fecha, hora
- ii. Paciente: nombre, obraSocial
- iii. Profesional: nombre, especialidad



```

/*
 * @author vcmann
 */
import java.util.Date;

public class ejercicio9 {

    public class CitaMedica{
        private Date fecha, hora;
        private Profesional profesional;
        private Paciente paciente;

        public CitaMedica(Date fecha, Date hora, Profesional profesional, Paciente paciente) {
            this.fecha = fecha;
            this.hora = hora;
            this.profesional = profesional;
            this.paciente = paciente;
        }
    }

    public class Profesional{
        private String nombre, especialidad;

        public Profesional(String nombre, String especialidad) {
            this.nombre = nombre;
            this.especialidad = especialidad;
        }
    }

    public class Paciente{
        private String nombre, obraSocial;

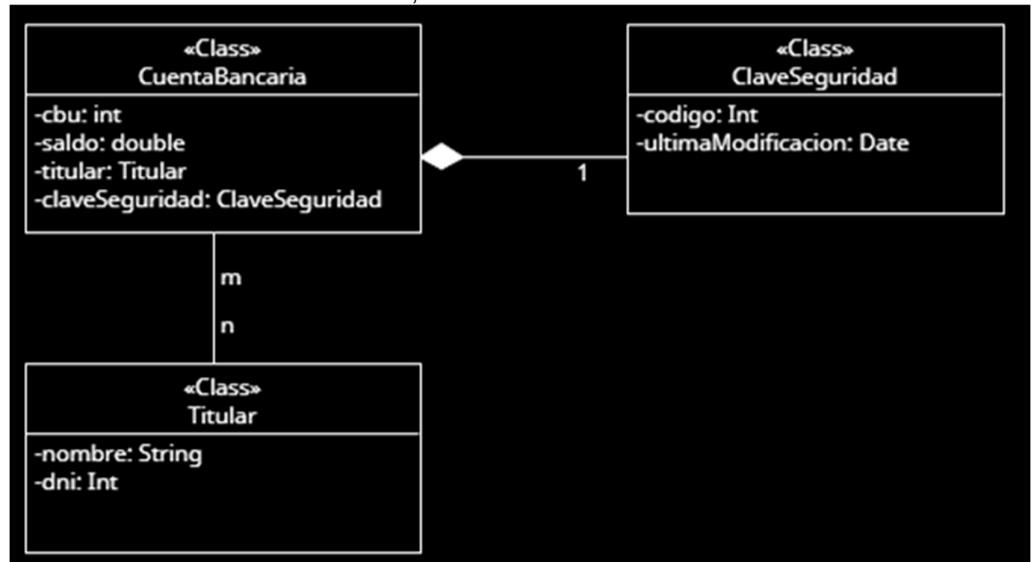
        public Paciente(String nombre, String obraSocial) {
            this.nombre = nombre;
            this.obraSocial = obraSocial;
        }
    }
}
  
```



10. CuentaBancaria - ClaveSeguridad - Titular
- Composición: **CuentaBancaria → ClaveSeguridad**
 - Asociación bidireccional: **CuentaBancaria ↔ Titular**

Clases y atributos:

- CuentaBancaria: cbu, saldo
- ClaveSeguridad: codigo, ultimaModificacion
- Titular: nombre, dni.



```

import java.util.Date;

public class ejercicio0 {

    public class CuentaBancaria{
        private int cbu;
        private double saldo;
        private Titular titular;
        private ClaveSeguridad claveSeguridad;

        public CuentaBancaria(int cbu, double saldo, Titular titular, ClaveSeguridad claveSeguridad) {
            this.cbu = cbu;
            this.saldo = saldo;
            this.titular = titular;
            this.claveSeguridad = claveSeguridad;
        }
    }

    public class Titular{
        private String nombre;
        private int dni;

        public Titular(String nombre, int dni) {
            this.nombre = nombre;
            this.dni = dni;
        }
    }

    public class ClaveSeguridad{
        private int codigo;
        private Date ultimaModificacion;

        public ClaveSeguridad(int codigo, Date ultimaModificacion) {
            this.codigo = codigo;
            this.ultimaModificacion = ultimaModificacion;
        }
    }
}
  
```



DEPENDENCIA DE USO

La clase usa otra como **parámetro de un método**, pero **no la guarda como atributo**.

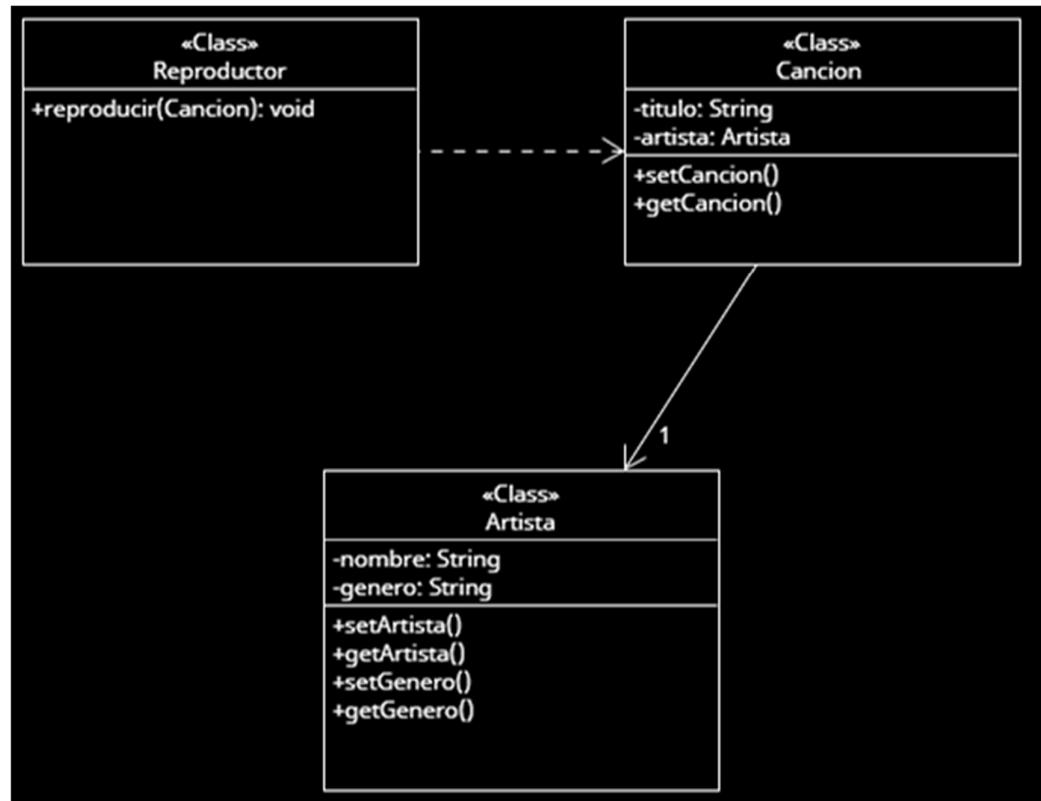
Ejercicios de Dependencia de Uso

11. Reproductor - Canción - Artista

- a . Asociación unidireccional: **Canción → Artista**
- b . Dependencia de uso: **Reproductor.reproducir(Cancion)**

Clases y atributos:

- i. Canción: título.
- ii. Artista: nombre, género.
- iii. Reproductor->método: void reproducir(Cancion cancion)





```
public class ejercicio11 {

    public class Artista{
        private String nombre, genero;

        public Artista(String nombre, String genero) {
            this.nombre = nombre;
            this.genero = genero;
        }

        public String getNombre() {
            return nombre;
        }

        public void setNombre(String nombre) {
            this.nombre = nombre;
        }

        public String getGenero() {
            return genero;
        }

        public void setGenero(String genero) {
            this.genero = genero;
        }
    }

    public class Cancion{
        private String titulo;
        private Artista artista;

        public Cancion(String titulo, Artista artista) {
            this.titulo = titulo;
            this.artista = artista;
        }

        public String getTitulo() {
            return titulo;
        }

        public Artista getArtista() {
            return artista;
        }
    }
}
```



```
private String titulo;
private Artista artista;

public Cancion(String titulo, Artista artista) {
    this.titulo = titulo;
    this.artista = artista;
}

public String getTitulo() {
    return titulo;
}

public void setTitulo(String titulo) {
    this.titulo = titulo;
}

public Artista getArtista() {
    return artista;
}

public void setArtista(Artista artista) {
    this.artista = artista;
}

}

public class Reproductor{

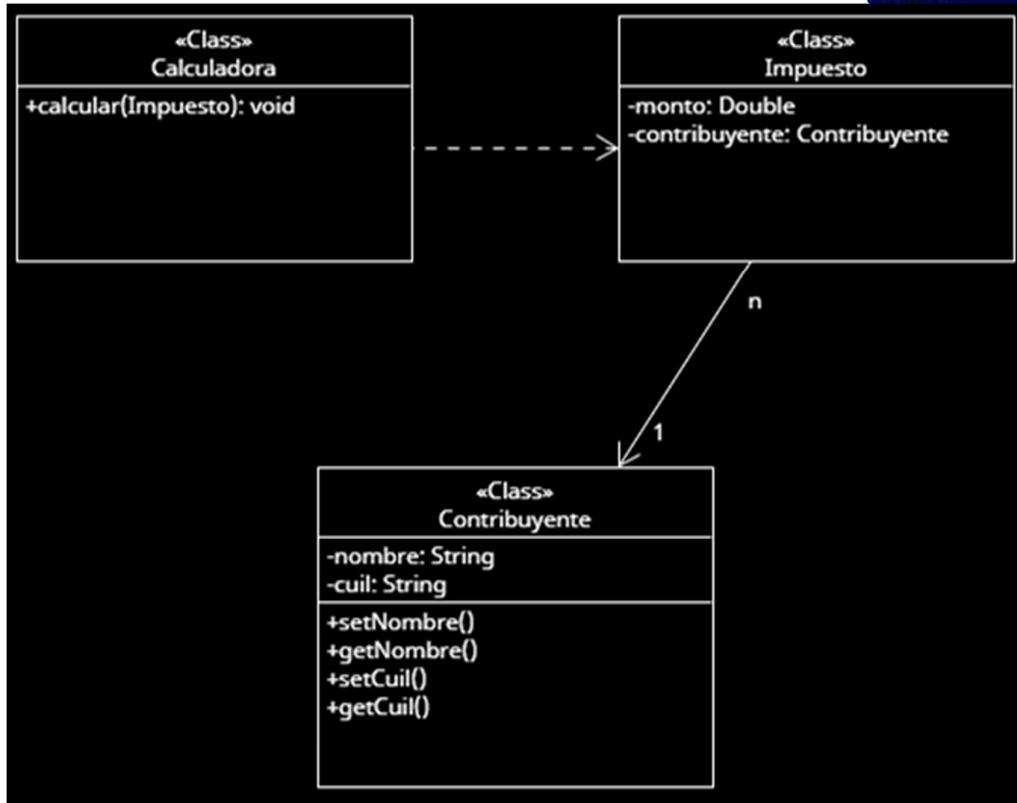
    public void reproducir(Cancion cancion){
        cancion.getTitulo();
        System.out.println("Reproduciendo el titulo: "+ cancion.getTitulo());
    }
}
}
```

12. Impuesto - Contribuyente - Calculadora

- Asociación unidireccional: **Impuesto → Contribuyente**
- Dependencia de uso: **Calculadora.calcular(Impuesto)**

Clases y atributos:

- Impuesto: monto.
- Contribuyente: nombre, cuil.
- Calculadora->método: void calcular(Impuesto impuesto)





```
public class ejercicio12 {

    public class Contribuyente{
        private String nombre, cuil;

        public Contribuyente(String nombre, String cuil) {
            this.nombre = nombre;
            this.cuil = cuil;
        }

        public String getNombre() {
            return nombre;
        }

        public void setNombre(String nombre) {
            this.nombre = nombre;
        }

        public String getCuil() {
            return cuil;
        }

        public void setCuil(String cuil) {
            this.cuil = cuil;
        }
    }

    public class Impuesto{
        private double monto;
        private Contribuyente contribuyente;

        public Impuesto(double monto, Contribuyente contribuyente) {
            this.monto = monto;
            this.contribuyente = contribuyente;
        }

        public double getMonto() {
            return monto;
        }
    }
}
```



```
public double getMonto() {
    return monto;
}

public void setMonto(double monto) {
    this.monto = monto;
}

public Contribuyente getContribuyente() {
    return contribuyente;
}

public void setContribuyente(Contribuyente contribuyente) {
    this.contribuyente = contribuyente;
}

public class Calculadora{

    public void calcular(Impuesto imp){
        imp.setMonto(imp.getMonto() * 0.2);
        //Seteo el nuevo monto con un 20% de aumento
    }
}
}
```

DEPENDENCIA DE CREACIÓN

La clase crea otra dentro de un método, pero no la conserva como atributo..

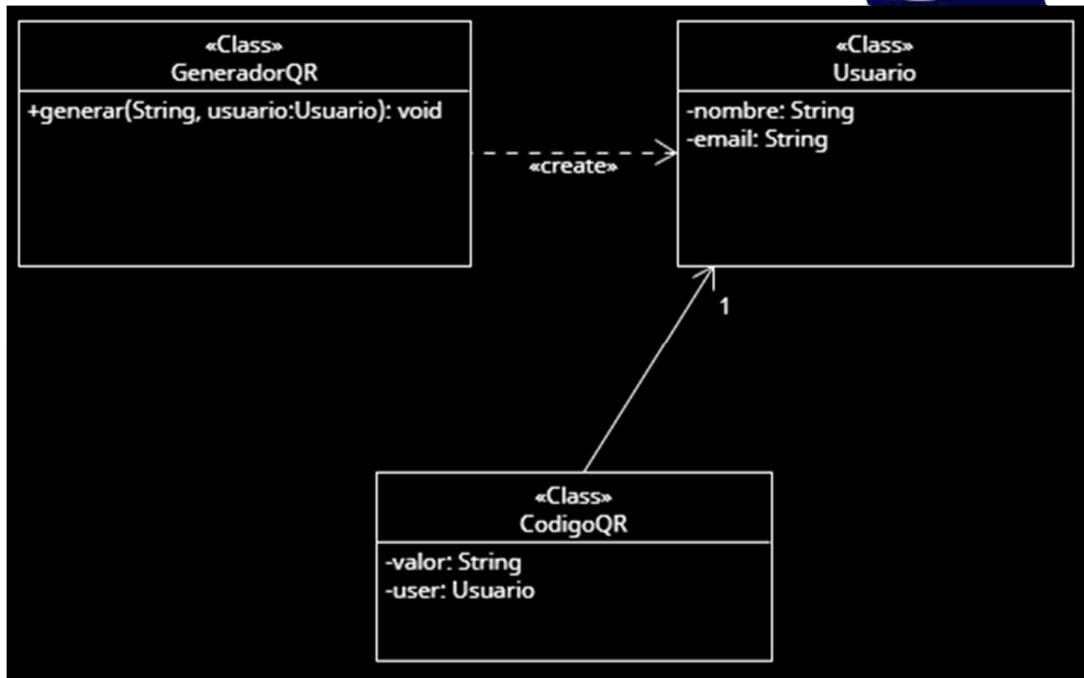
Ejercicios de Dependencia de Creación

13. GeneradorQR - Usuario - CódigoQR

- a. Asociación unidireccional: **CódigoQR → Usuario**
- b. Dependencia de creación: **GeneradorQR.generar(String, Usuario)**

Clases y atributos:

- i. CódigoQR: valor.
- ii. Usuario: nombre, email.
- iii. GeneradorQR->método: void generar(String valor, Usuario usuario)





```
public class ejercicio13 {

    public class Usuario{
        private String nombre, email;

        public Usuario(String nombre, String email) {
            this.nombre = nombre;
            this.email = email;
        }

        public String getNombre() {
            return nombre;
        }

        public void setNombre(String nombre) {
            this.nombre = nombre;
        }

        public String getEmail() {
            return email;
        }

        public void setEmail(String email) {
            this.email = email;
        }
    }

    public class CódigoQR{
        private String valor;
        private Usuario user;

        public CódigoQR(String valor, Usuario user) {
            this.valor = valor;
            this.user = user;
        }

        public String getValor() {
            return valor;
        }
    }
}
```



```
public class CódigoQR{
    private String valor;
    private Usuario user;

    public CódigoQR(String valor, Usuario user) {
        this.valor = valor;
        this.user = user;
    }

    public String getValor() {
        return valor;
    }

    public void setValor(String valor) {
        this.valor = valor;
    }

    public Usuario getUser() {
        return user;
    }

    public void setUser(Usuario user) {
        this.user = user;
    }
}

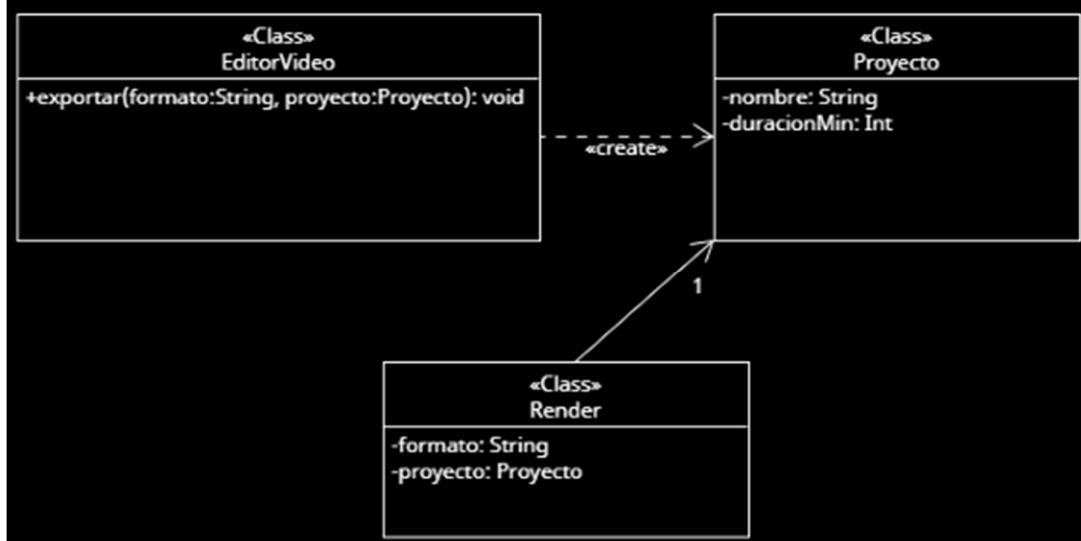
public class GeneradorQR{

    public void generar(String valor, Usuario user){
        CódigoQR new_qr = new CódigoQR(valor, user);
        System.out.println("Usuario: "+ new_qr.getUser().getNombre()+"\nCódigo QR: " +
                           new_qr.getValor());
    }
}
```



14. EditorVideo - Proyecto - Render

- a. Asociación unidireccional: Render → Proyecto
- b. Dependencia de creación: **EditorVideo.exportar(String, Proyecto)**
- c. Clases y atributos:
 - i. Render: formato.
 - ii. Proyecto: nombre, duracionMin.
 - iii. EditorVideo->método: void exportar(String formato, Proyecto proyecto)





```
* @author vcman
*/
public class ejercicio14 {

    public class Proyecto {
        private String nombre;
        private int duracionMin;

        public Proyecto(String nombre, int duracionMin) {
            this.nombre = nombre;
            this.duracionMin = duracionMin;
        }

        public String getNombre() {
            return nombre;
        }

        public void setNombre(String nombre) {
            this.nombre = nombre;
        }

        public int getDuracionMin() {
            return duracionMin;
        }

        public void setDuracionMin(int duracionMin) {
            this.duracionMin = duracionMin;
        }
    }

    public class Render {
        private String formato;
        private Proyecto proyecto;

        public Render(String formato, Proyecto proyecto) {
            this.formato = formato;
            this.proyecto = proyecto;
        }
    }
}
```



```
}

public class Render {
    private String formato;
    private Proyecto proyecto;

    public Render(String formato, Proyecto proyecto) {
        this.formato = formato;
        this.proyecto = proyecto;
    }

    public String getFormato() {
        return formato;
    }

    public void setFormato(String formato) {
        this.formato = formato;
    }

    public Proyecto getProyecto() {
        return proyecto;
    }

    public void setProyecto(Proyecto proyecto) {
        this.proyecto = proyecto;
    }
}

private class EditorVideo{
    public void exportar(String formato, Proyecto proyecto){
        Render new_render = new Render(formato, proyecto);
        System.out.println("Proyecto renderizando: "+ new_render.getProyecto()
            .getNombre() +"."+ new_render.getFormato());
    }
}
}
```

CONCLUSIONES ESPERADAS

- Diferenciar claramente los tipos de relaciones entre clases (asociación, agregación, composición).
- Representar las relaciones con la dirección adecuada en diagramas UML.
- Comprender e implementar dependencias de uso y de creación.
- Aplicar relaciones 1 a 1 en el diseño e implementación de clases en Java.
- Reforzar el análisis de modelos orientados a objetos y la capacidad de abstracción.