

# Programación II — Trabajo Práctico: UML

**Alumno:** Fermín Alliot

**Carrera:** Tecnicatura Universitaria en Programación

**Materia:** Programación II

**Comisión:** “3”

**GitHub:** <https://github.com/Falliot00/UTN-TUPaD-P2/tree/main/5.%20UML>

---

## Criterios y convenciones que utilizo

- **UML (texto):**
    - Asociación unidireccional: ClaseA → ClaseB (1..1)
    - Asociación bidireccional: ClaseA —— ClaseB con multiplicidades (1..1)
    - Agregación: ClaseA ⇝ ClaseB (1..1) (rombo blanco en A)
    - Composición: ClaseA ◆— ClaseB (1..1) (rombo negro en A)
    - Dependencia: ClaseA - - -> ClaseB <<call|create>>
  - **Java:**
    - **Asociación unidireccional:** atributo + setter/constructor, puede ser null.
    - **Agregación:** el contenedor **recibe** la instancia creada afuera.
    - **Composición:** el contenedor **crea** la parte (ciclo de vida ligado).
    - **Bidireccional:** setters **sincronizados** con chequeo para evitar recursión.
- 

## PARTE A — Relaciones 1:1

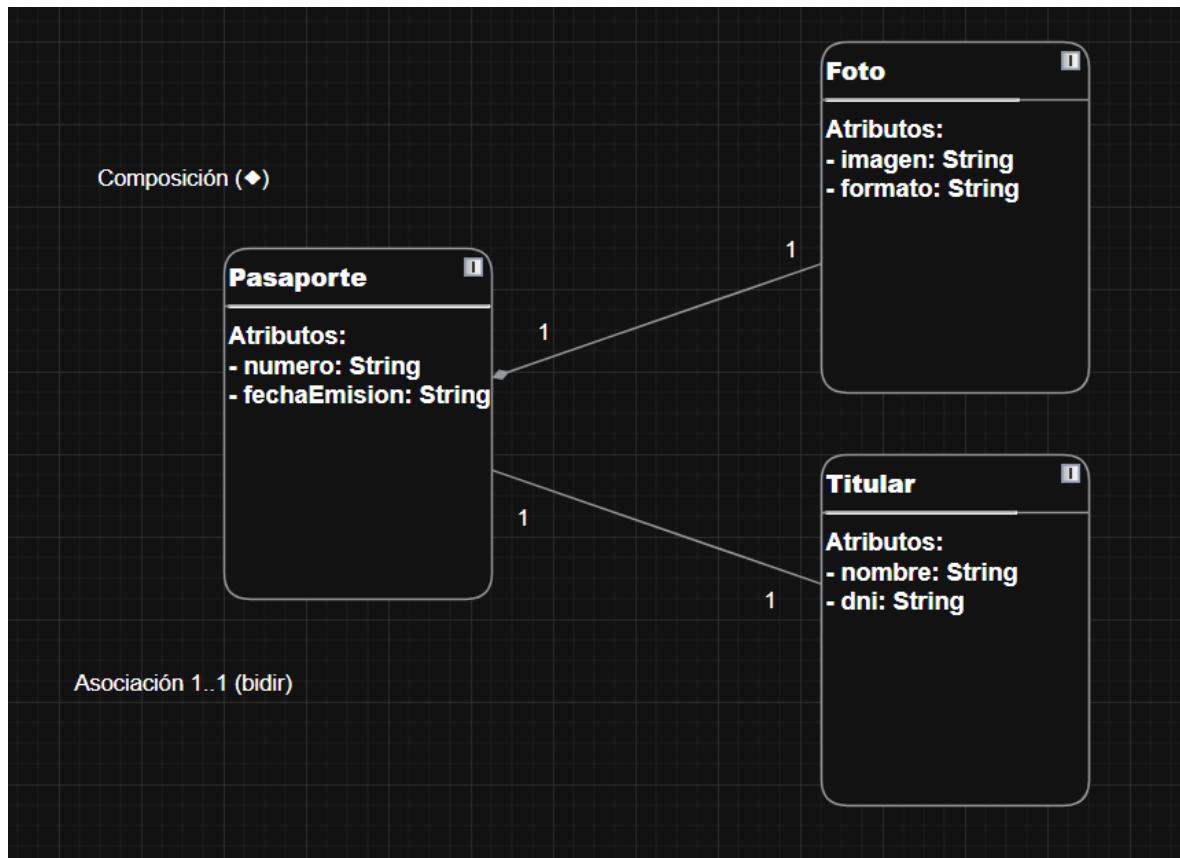
### 1) Pasaporte — Foto — Titular

a) **Composición:** Pasaporte ◆—> Foto (1..1)

b) **Asociación bidireccional:** Pasaporte —— Titular (1..1)

#### UML

- Pasaporte ◆— Foto
- Pasaporte —— Titular (1..1 ↔ 1..1)



## Java

```

// a) Composición: Pasaporte crea y posee Foto
class Foto {
    private String imagen;
    private String formato;
    public Foto(String imagen, String formato) {
        this.imagen = imagen; this.formato = formato;
    }
    public String getImagen() { return imagen; }
    public String getFormato() { return formato; }
}

class Pasaporte {
    private String numero;
    private String fechaEmision;
    private Foto foto; // composición
    private Titular titular; // asociación bidireccional

    public Pasaporte(String numero, String fechaEmision, String img,
String formato) {
        this.numero = numero; this.fechaEmision = fechaEmision;
        this.foto = new Foto(img, formato); // composición: creación
        interna
    }
}

```

```

public void setTitular(Titular t) {
    this.titular = t;
    if (t != null && t.getPasaporte() != this) t.setPasaporte(this);
}
public Titular getTitular() { return titular; }
}

class Titular {
    private String nombre;
    private String dni;
    private Pasaporte pasaporte; // asociación bidireccional

    public Titular(String nombre, String dni) { this.nombre = nombre;
this.dni = dni; }
    public void setPasaporte(Pasaporte p) {
        this.pasaporte = p;
        if (p != null && p.getTitular() != this) p.setTitular(this);
    }
    public Pasaporte getPasaporte() { return pasaporte; }
}

```

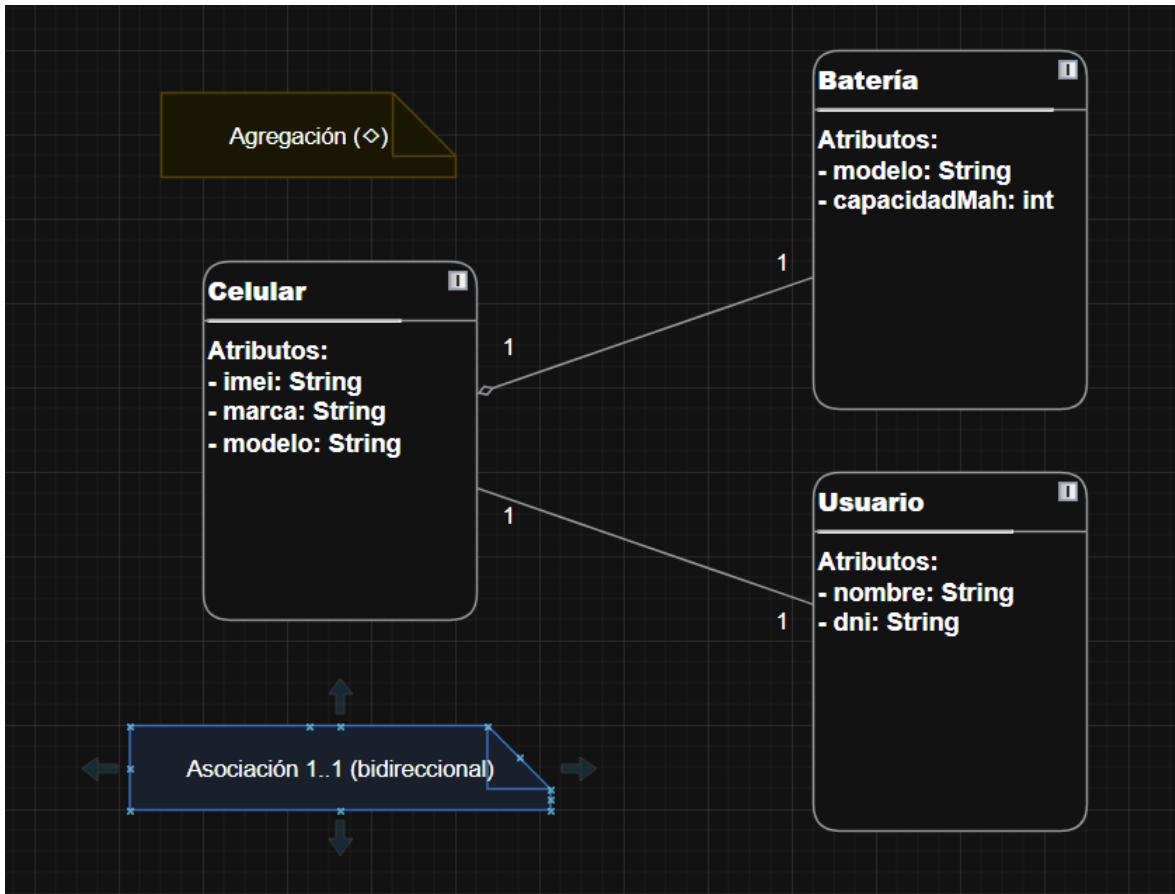
---

## 2) Celular — Batería — Usuario

- a) Agregación: Celular  $\diamond\rightarrow$  Batería (1..1)
- b) Asociación bidireccional: Celular — Usuario (1..1)

### UML

- Celular  $\diamond\leftarrow$  Batería
- Celular — Usuario



## Java

```

class Bateria {
    private String modelo; private int capacidadMah;
    public Bateria(String modelo, int capacidadMah) { this.modelo = modelo; this.capacidadMah = capacidadMah; }
}
class Usuario {
    private String nombre; private String dni; private Celular celular;
    public Usuario(String nombre, String dni) { this.nombre = nombre; this.dni = dni; }
    public void setCelular(Celular c) { this.celular = c; if (c != null && c.getUsuario() != this) c.setUsuario(this); }
    public Celular getCelular() { return celular; }
}
class Celular {
    private String imei, marca, modelo; private Bateria bateria; private Usuario usuario;
    public Celular(String imei, String marca, String modelo, Bateria bateria) {
        this.imei = imei; this.marca = marca; this.modelo = modelo;
        this.bateria = bateria; // agregación
    }
    public void setUsuario(Usuario u) { this.usuario = u; if (u != null)
  
```

```

    && u.getCelular() != this) u.setCelular(this); }
    public Usuario getUsuario() { return usuario; }
}

```

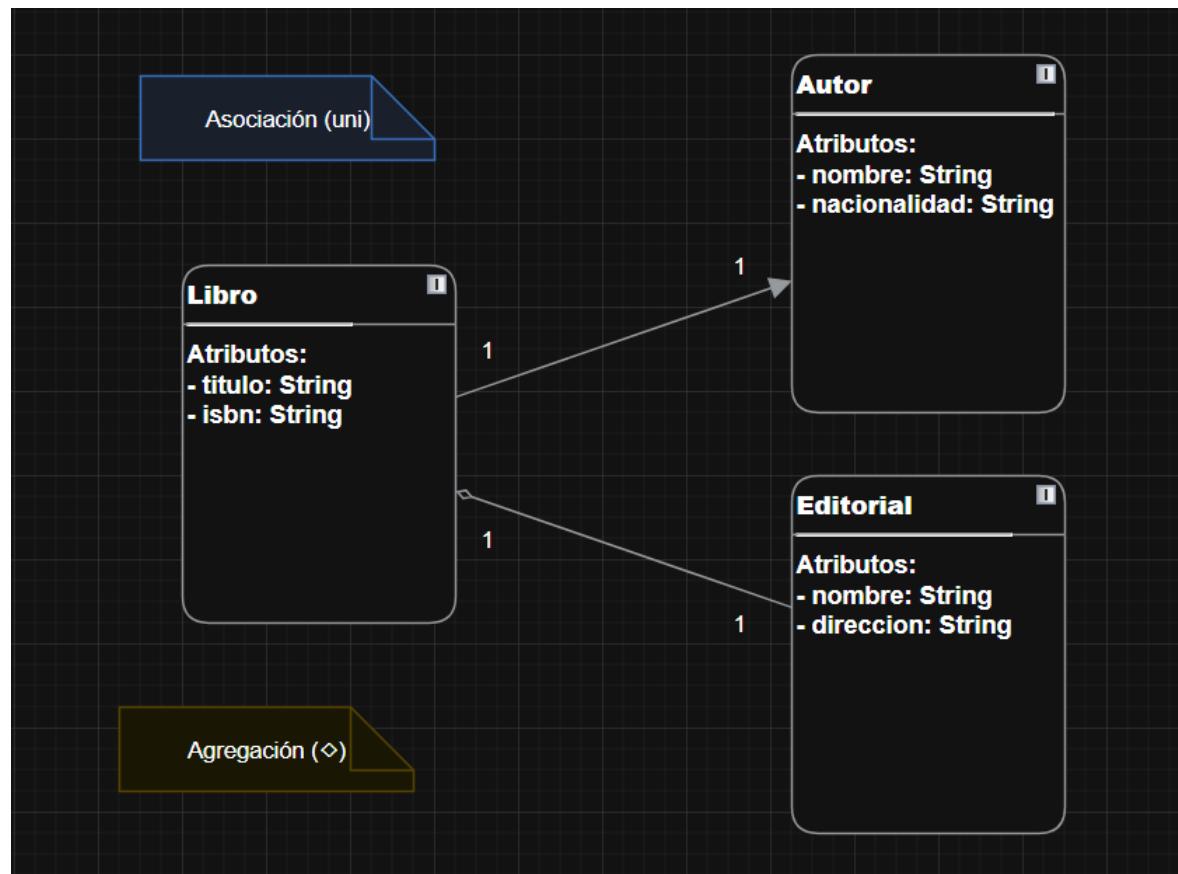
---

### 3) Libro — Autor — Editorial

- a) Asociación unidireccional: Libro → Autor (1..1)
- b) Agregación: Libro ◊→ Editorial (1..1)

#### UML

- Libro → Autor
- Libro ◊→ Editorial



#### Java

```

class Autor { private String nombre, nacionalidad; public Autor(String n,
String nac){ this.nombre=n; this.nacionalidad=nac; } }
class Editorial { private String nombre, direccion; public
Editorial(String n, String d){ this.nombre=n; this.direccion=d; } }
class Libro {
    private String titulo, isbn; private Autor autor; private Editorial
editorial; // asoc+agreg

```

```

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

```

---

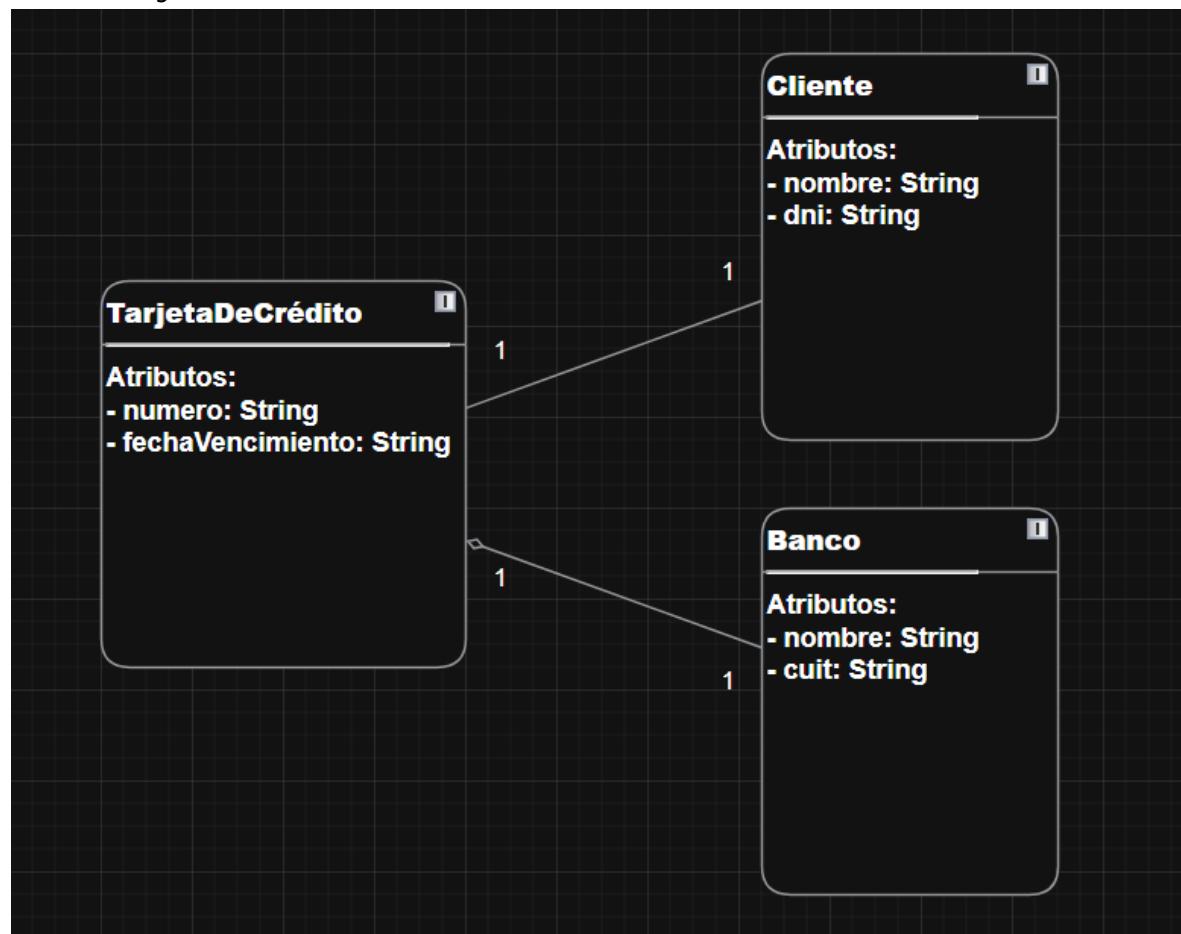
## 4) TarjetaDeCrédito — Cliente — Banco

a) Asociación bidireccional: TarjetaDeCrédito — Cliente

b) Agregación: TarjetaDeCrédito  $\rightarrowtail$  Banco

### UML

- Tarjeta — Cliente
- Tarjeta  $\rightarrowtail$  Banco



### Java

```

class Banco { private String nombre; private String cuit; public
Banco(String n, String c){ this.nombre=n; this.cuit=c; } }

```

```

class Cliente {
    private String nombre, dni; private TarjetaDeCredito tarjeta;
    public Cliente(String nombre, String dni) { this.nombre=nombre;
this.dni=dni; }
    public void setTarjeta(TarjetaDeCredito t){ this.tarjeta=t;
if(t!=null && t.getCliente()!=this) t.setCliente(this);}
    public TarjetaDeCredito getTarjeta(){ return tarjeta; }
}
class TarjetaDeCredito {
    private String numero, fechaVencimiento; private Cliente cliente;
private Banco banco;
    public TarjetaDeCredito(String numero, String fechaVencimiento, Banco
banco){
        this.numero=numero; this.fechaVencimiento=fechaVencimiento;
this.banco=banco; // agregación
    }
    public void setCliente(Cliente c){ this.cliente=c; if(c!=null &&
c.getTarjeta()!=this) c.setTarjeta(this);}
    public Cliente getCliente(){ return cliente; }
}

```

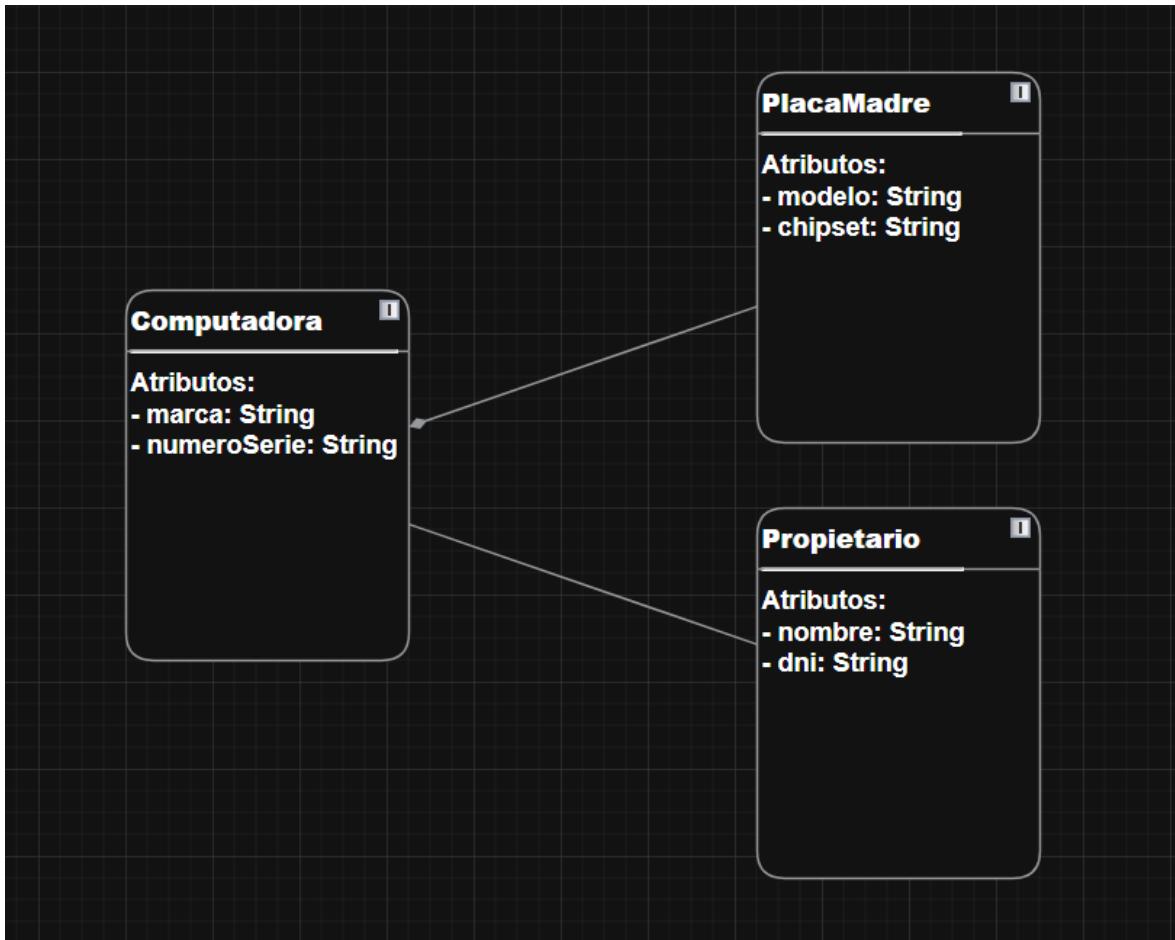
---

## 5) Computadora — PlacaMadre — Propietario

- a) **Composición:** Computadora ◆→ PlacaMadre
- b) **Asociación bidireccional:** Computadora — Propietario

### UML

- Computadora ◆→ PlacaMadre
- Computadora — Propietario



## Java

```

class PlacaMadre { private String modelo, chipset; public
PlacaMadre(String m, String c){ this.modelo=m; this(chipset=c); }
class Propietario {
    private String nombre, dni; private Computadora pc;
    public Propietario(String n, String d){ this.nombre=n; this.dni=d; }
    public void setPc(Computadora pc){ this.pc=pc; if(pc!=null &&
pc.getPropietario()!=this) pc.setPropietario(this);}
    public Computadora getPropietarioPc(){ return pc; }
}
class Computadora {
    private String marca, numeroSerie; private PlacaMadre placa; private
Propietario propietario;
    public Computadora(String marca, String numeroSerie, String
modeloPlaca, String chipset){
        this.marca=marca; this.numeroSerie=numeroSerie; this.placa=new
PlacaMadre(modeloPlaca, chipset); // composición
    }
    public void setPropietario(Propietario p){ this.propietario=p;
if(p!=null && p.getPropietarioPc()!=this) p.setPc(this);}
    public Propietario getPropietario(){ return propietario; }
}

```

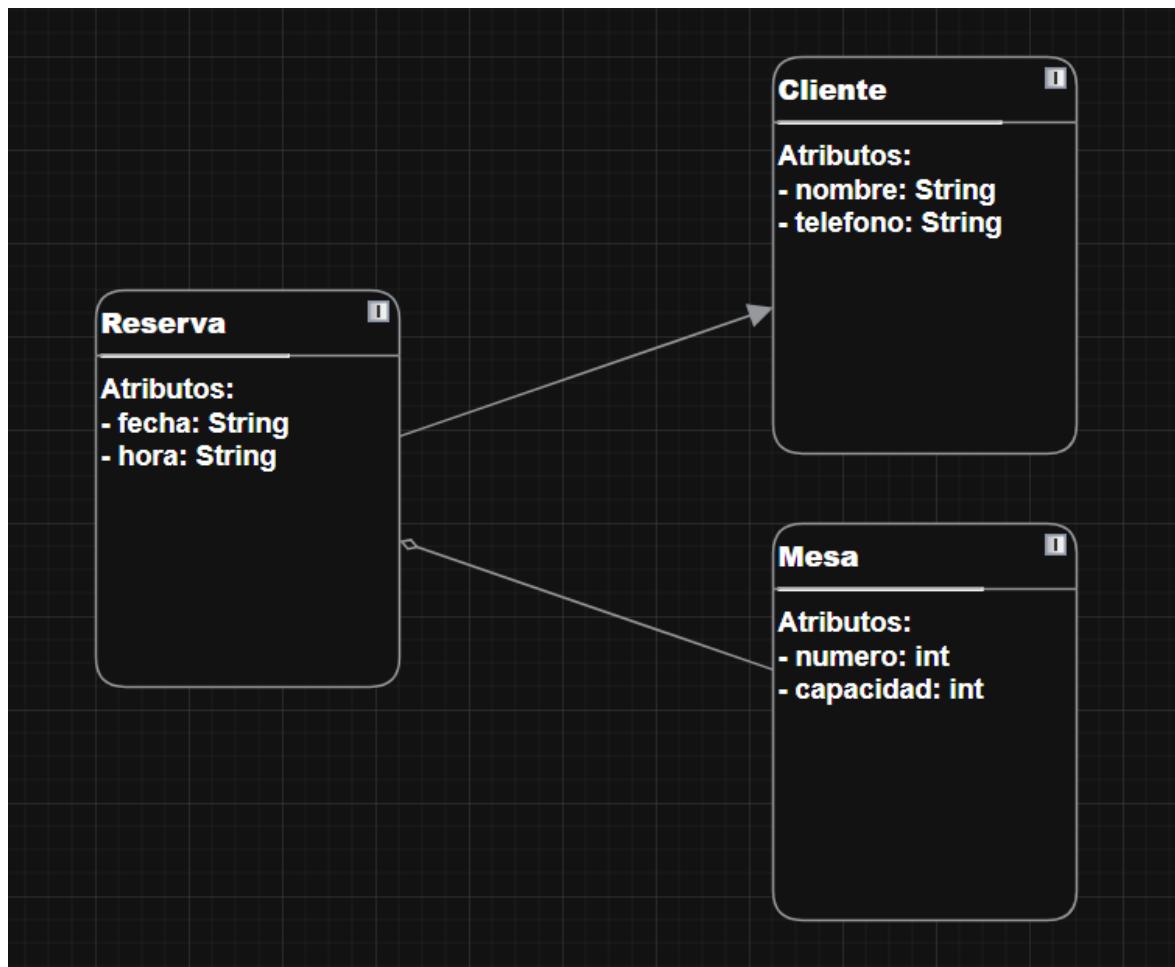
---

## 6) Reserva — Cliente — Mesa

- a) Asociación unidireccional: Reserva → Cliente
- b) Agregación: Reserva ⌂→ Mesa

### UML

- Reserva → Cliente
- Reserva ⌂→ Mesa



### Java

```
class ClienteR { private String nombre, telefono; public ClienteR(String n, String t){ this.nombre=n; this.telefono=t; } }
class Mesa { private int numero, capacidad; public Mesa(int n,int c){ this.numero=n; this.capacidad=c; } }
class Reserva {
    private String fecha, hora; private ClienteR cliente; private Mesa mesa;
    public Reserva(String fecha, String hora, ClienteR cliente, Mesa
```

```

mesa){
    this.fecha=fecha; this.hora=hora; this.cliente=cliente;
this.mesa=mesa;
}
}

```

---

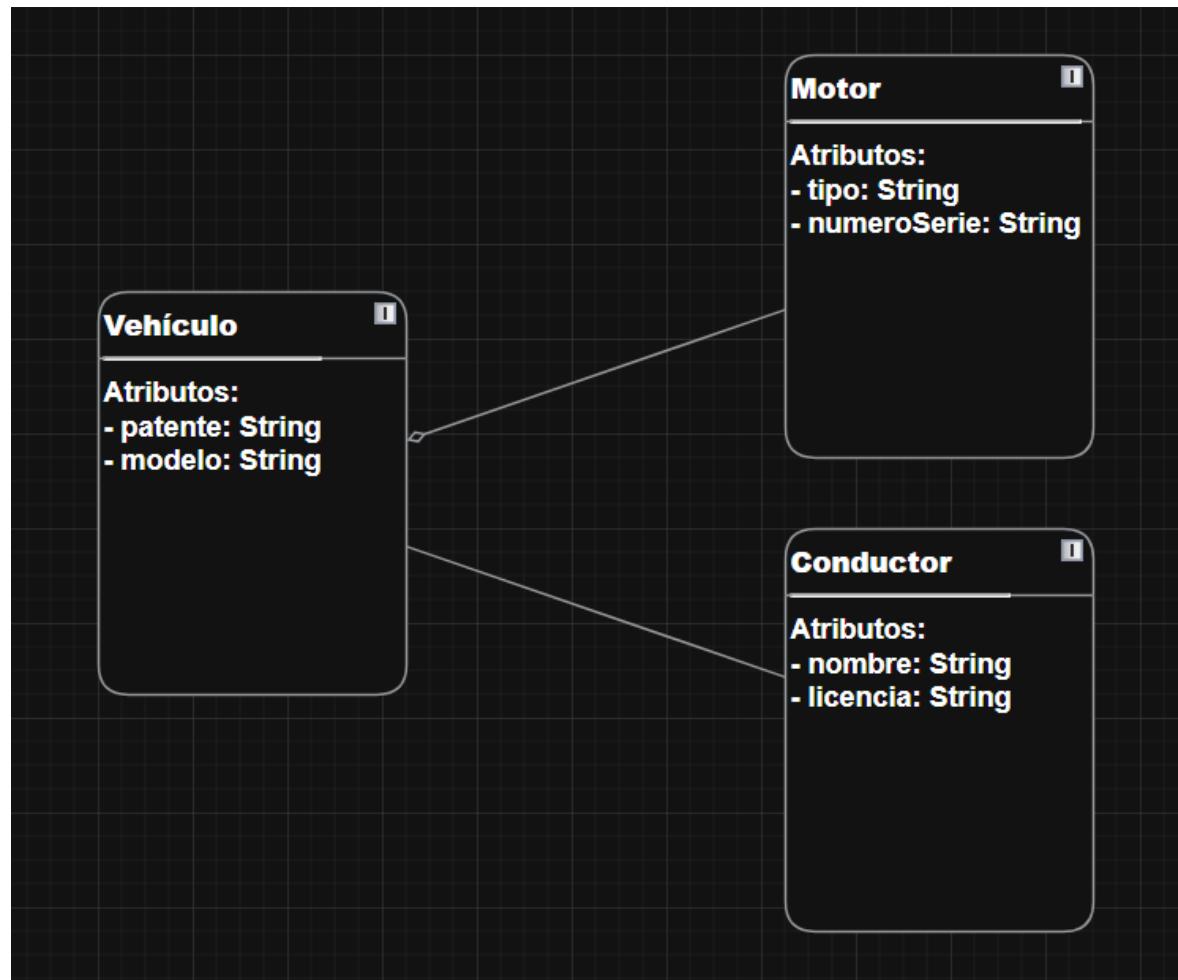
## 7) Vehículo — Motor — Conductor

a) Agregación: Vehículo  $\diamond\rightarrow$  Motor

b) Asociación bidireccional: Vehículo — Conductor

### UML

- Vehículo  $\diamond\rightarrow$  Motor
- Vehículo — Conductor



### Java

```

class MotorV { private String tipo, numeroSerie; public MotorV(String t, String ns){ this.tipo=t; this.numeroSerie=ns; } }

```

```

class Conductor {
    private String nombre, licencia; private Vehiculo vehiculo;
    public Conductor(String n, String l){ this.nombre=n; this.licencia=l;
}
    public void setVehiculo(Vehiculo v){ this.vehiculo=v; if(v!=null &&
v.getConductor()!=this) v.setConductor(this); }
    public Vehiculo getVehiculo(){ return vehiculo; }
}
class Vehiculo {
    private String patente, modelo; private MotorV motor; private
Conductor conductor;
    public Vehiculo(String patente, String modelo, MotorV motor){
this.patente=patente; this.modelo=modelo; this.motor=motor; }
    public void setConductor(Conductor c){ this.conductor=c; if(c!=null
&& c.getVehiculo()!=this) c.setVehiculo(this); }
    public Conductor getConductor(){ return conductor; }
}

```

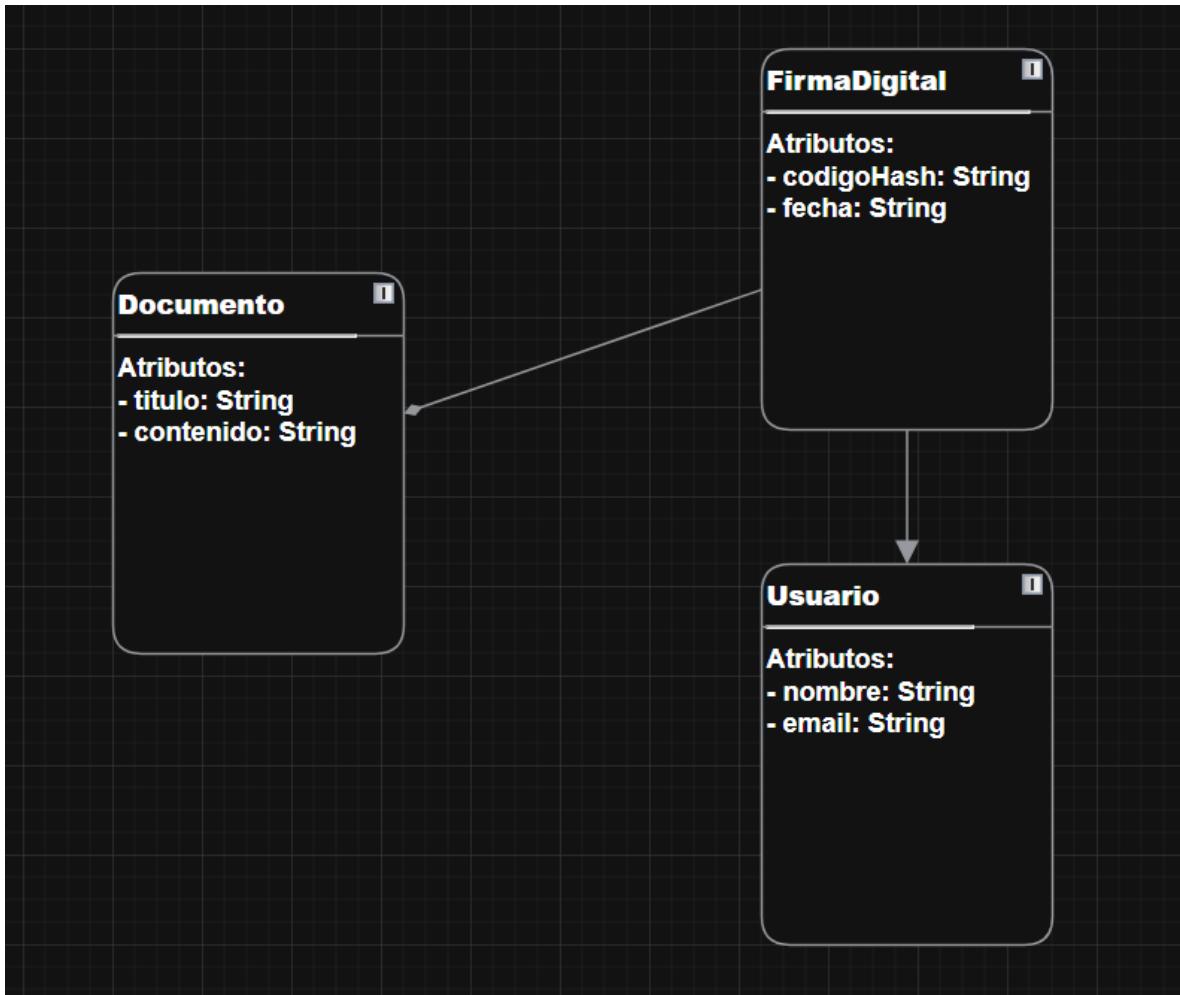
---

## 8) Documento — FirmaDigital — Usuario

- a) Composición: Documento ◆→ FirmaDigital
- b) Agregación: FirmaDigital ◇→ Usuario

### UML

- Documento ◆→ FirmaDigital
- FirmaDigital ◇→ Usuario



## Java

```

class UsuarioFD { private String nombre, email; public UsuarioFD(String n, String e){ this.nombre=n; this.email=e; } }
class FirmaDigital {
    private String codigoHash, fecha; private UsuarioFD usuario; // agregación hacia Usuario
    public FirmaDigital(String codigoHash, String fecha, UsuarioFD u){
        this.codigoHash=codigoHash; this.fecha=fecha; this.usuario=u; }
}
class Documento {
    private String titulo, contenido; private FirmaDigital firma; // composición
    public Documento(String titulo, String contenido, String hash, String fecha, UsuarioFD u){
        this.titulo=titulo; this.contenido=contenido; this.firma=new FirmaDigital(hash, fecha, u);
    }
}

```

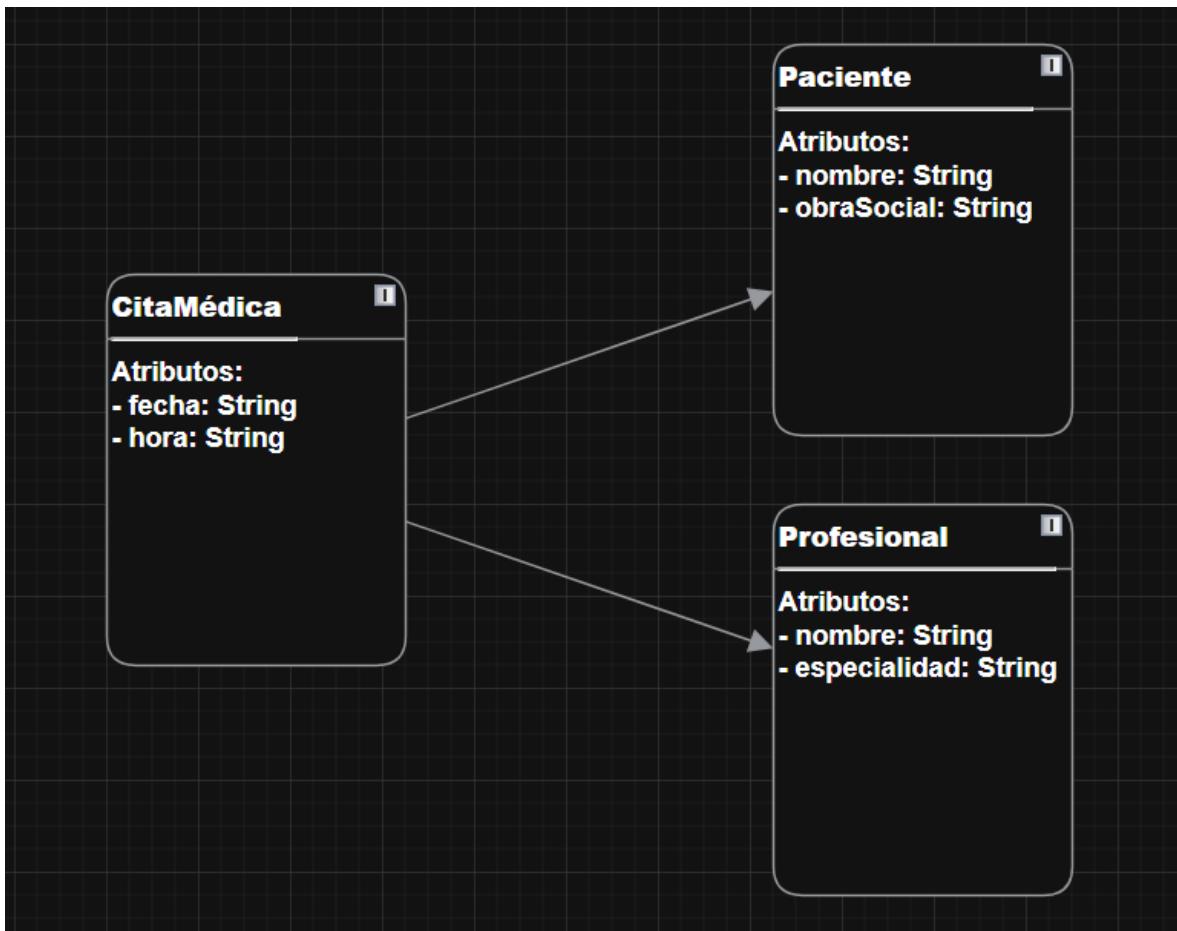
---

## 9) CitaMédica — Paciente — Profesional

- a) Asociación unidireccional: CitaMédica → Paciente
- b) Asociación unidireccional: CitaMédica → Profesional

### UML

- Cita → Paciente
- Cita → Profesional



### Java

```
class Paciente { private String nombre, obraSocial; public
Paciente(String n,String o){ this.nombre=n; this.obraSocial=o; } }
class Profesional { private String nombre, especialidad; public
Profesional(String n,String e){ this.nombre=n; this.especialidad=e; } }
class CitaMedica {
    private String fecha, hora; private Paciente paciente; private
Profesional profesional;
    public CitaMedica(String fecha, String hora, Paciente p, Profesional
pr){ this.fecha=fecha; this.hora=hora; this.paciente=p;
this.profesional=pr; }
}
```

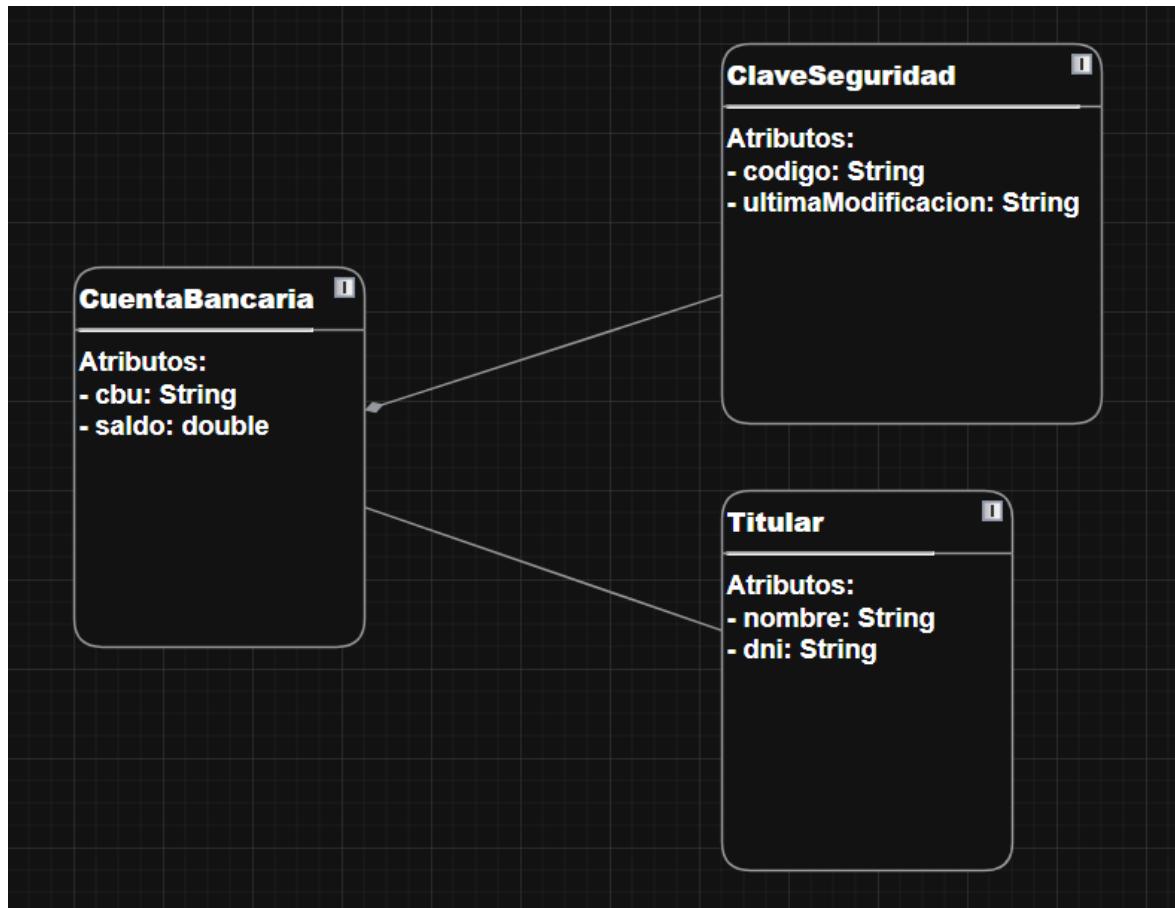
---

## 10) CuentaBancaria — ClaveSeguridad — Titular

- a) Composición: CuentaBancaria  $\blacklozenge\!\!\rightarrow$  ClaveSeguridad  
b) Asociación bidireccional: CuentaBancaria — Titular

UML

- Cuenta  $\blacklozenge\!\!\rightarrow$  Clave
- Cuenta — Titular



Java

```
class ClaveSeguridad { private String codigo, ultimaModificacion; public
ClaveSeguridad(String c, String f){ this.codigo=c;
this.ultimaModificacion=f; }
class TitularCB {
    private String nombre, dni; private CuentaBancaria cuenta;
    public TitularCB(String n, String d){ this.nombre=n; this.dni=d; }
    public void setCuenta(CuentaBancaria c){ this.cuenta=c; if(c!=null &&
c.getTitular()!=this) c.setTitular(this); }
    public CuentaBancaria getCuenta(){ return cuenta; } 
```

```

}
class CuentaBancaria {
    private String cbu; private double saldo; private ClaveSeguridad
    clave; private TitularCB titular;
    public CuentaBancaria(String cbu, double saldo, String codigo, String
    fecha){
        this.cbu=cbu; this.saldo=saldo; this.clave=new
        ClaveSeguridad(codigo, fecha); // composición
    }
    public void setTitular(TitularCB t){ this.titular=t; if(t!=null &&
    t.getCUENTA()!<this> t.setCuenta(this);}
    public TitularCB getTitular(){ return titular; }
}

```

---

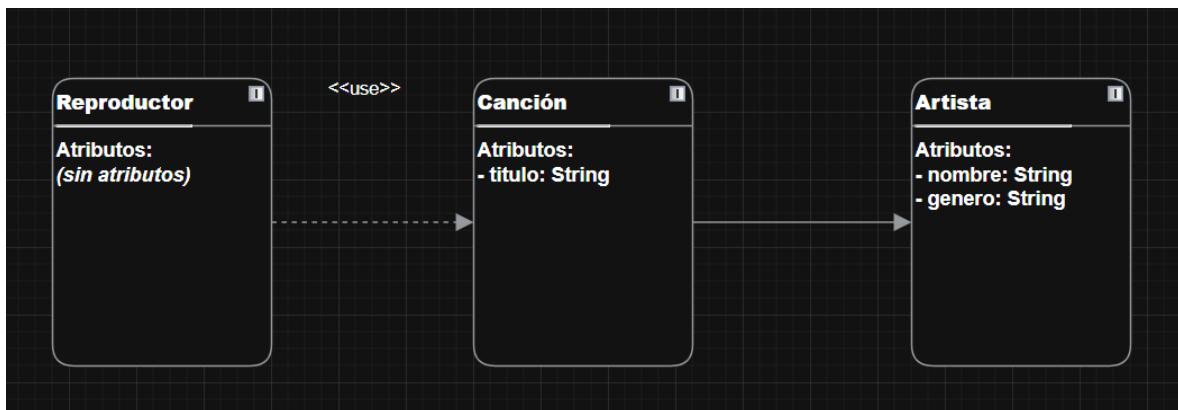
## PARTE B — Dependencias (Uso y Creación)

### 11) Reproductor — Canción — Artista

- **Asociación (uni):** Canción → Artista
- **Dependencia de uso:** Reproductor - - -> Canción <<call>>

#### UML

- Reproductor - - -> Canción <<call>>
- Canción → Artista



#### Java

```

class Artista { private String nombre, genero; public Artista(String
n, String g){ this.nombre=n; this.genero=g; } }
class Cancion { private String titulo; private Artista artista; public
Cancion(String t, Artista a){ this.titulo=t; this.artista=a; } public
String getTitulo(){ return titulo; } }
class Reproductor {
    public void reproducir(Cancion c){ // dependencia de uso: solo por
}

```

```

    parámetro
        System.out.println("Reproduciendo: " + c.getTitulo());
    }
}

```

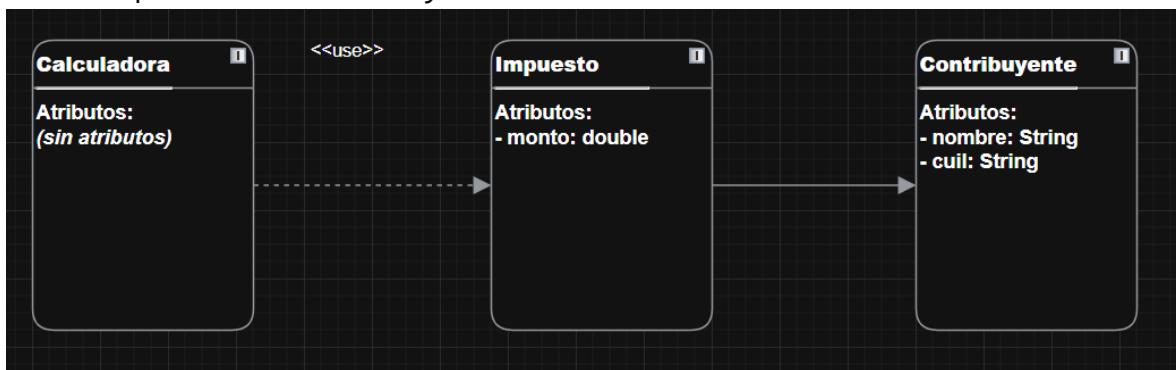
---

## 12) Impuesto — Contribuyente — Calculadora

- **Asociación (uni):** Impuesto → Contribuyente
- **Dependencia de uso:** Calculadora - - -> Impuesto <<call>>

### UML

- Calculadora - - -> Impuesto <<call>>
- Impuesto → Contribuyente



### Java

```

class Contribuyente { private String nombre, cuil; public
Contribuyente(String n, String c){ this.nombre=n; this.cuil=c; } }
class Impuesto { private double monto; private Contribuyente
contribuyente; public Impuesto(double m, Contribuyente c){ this.monto=m;
this.contribuyente=c; } public double getMonto(){ return monto; } }
class Calculadora {
    public void calcular(Impuesto imp){ // uso temporal
        double total = imp.getMonto() * 1.21; // ej. aplica IVA a modo
        didáctico
        System.out.println("Total calculado: " + total);
    }
}

```

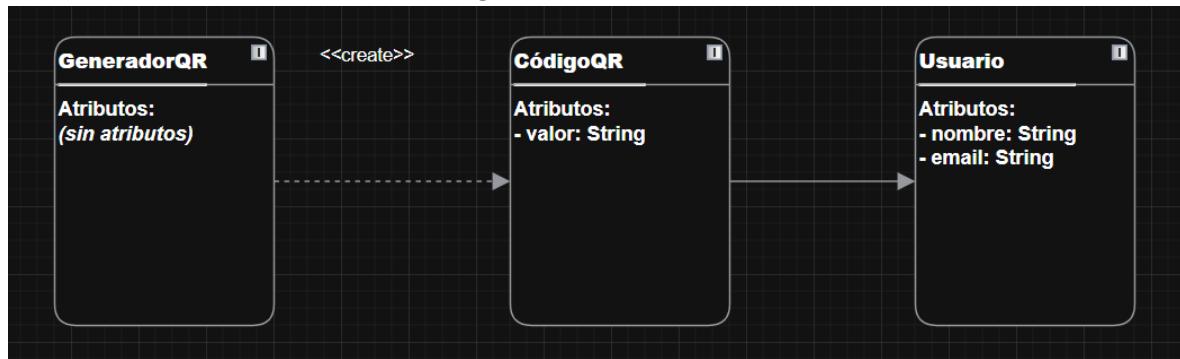
---

## 13) GeneradorQR — Usuario — CódigoQR

- **Asociación (uni):** CódigoQR → Usuario
- **Dependencia de creación:** GeneradorQR - - -> CódigoQR <<create>>

## UML

- GeneradorQR - - -> CódigoQR <<create>>



## Java

```
class UsuarioQR { private String nombre, email; public UsuarioQR(String n, String e){ this.nombre=n; this.email=e; } }
class CódigoQR { private String valor; private UsuarioQR usuario; public CódigoQR(String v, UsuarioQR u){ this.valor=v; this.usuario=u; } }
class GeneradorQR {
    public void generar(String valor, UsuarioQR usuario){ // crea y usa
        sin guardar
        CódigoQR qr = new CódigoQR(valor, usuario); // dependencia de
        creación
        System.out.println("QR generado para: " + valor);
    }
}
```

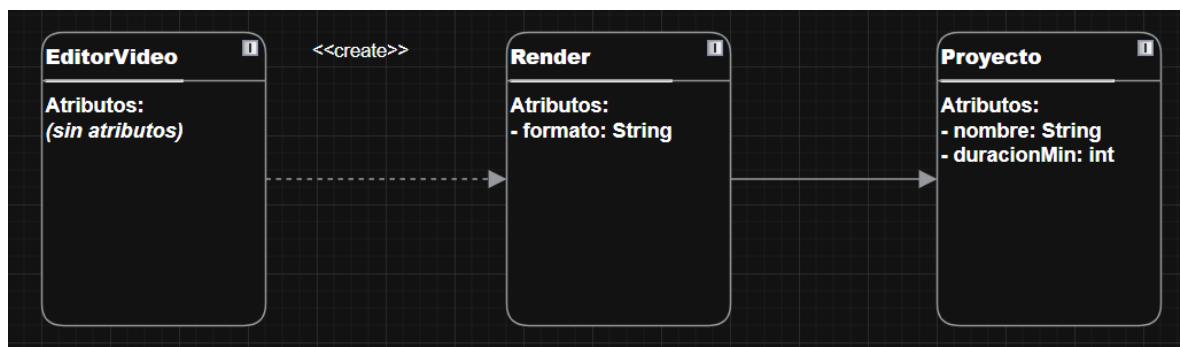
---

## 14) EditorVideo — Proyecto — Render

- **Asociación (uni):** Render → Proyecto
- **Dependencia de creación:** EditorVideo - - -> Render <<create>>

## UML

- EditorVideo - - -> Render <<create>>



## Java

```
class Proyecto { private String nombre; private int duracionMin; public  
Proyecto(String n,int d){ this.nombre=n; this.duracionMin=d; } }  
class Render { private String formato; private Proyecto proyecto; public  
Render(String f, Proyecto p){ this.formato=f; this.proyecto=p; } }  
class EditorVideo {  
    public void exportar(String formato, Proyecto proyecto){  
        Render r = new Render(formato, proyecto); // dependencia de  
creación  
        System.out.println("Exportando en formato: " + formato);  
    }  
}
```

---