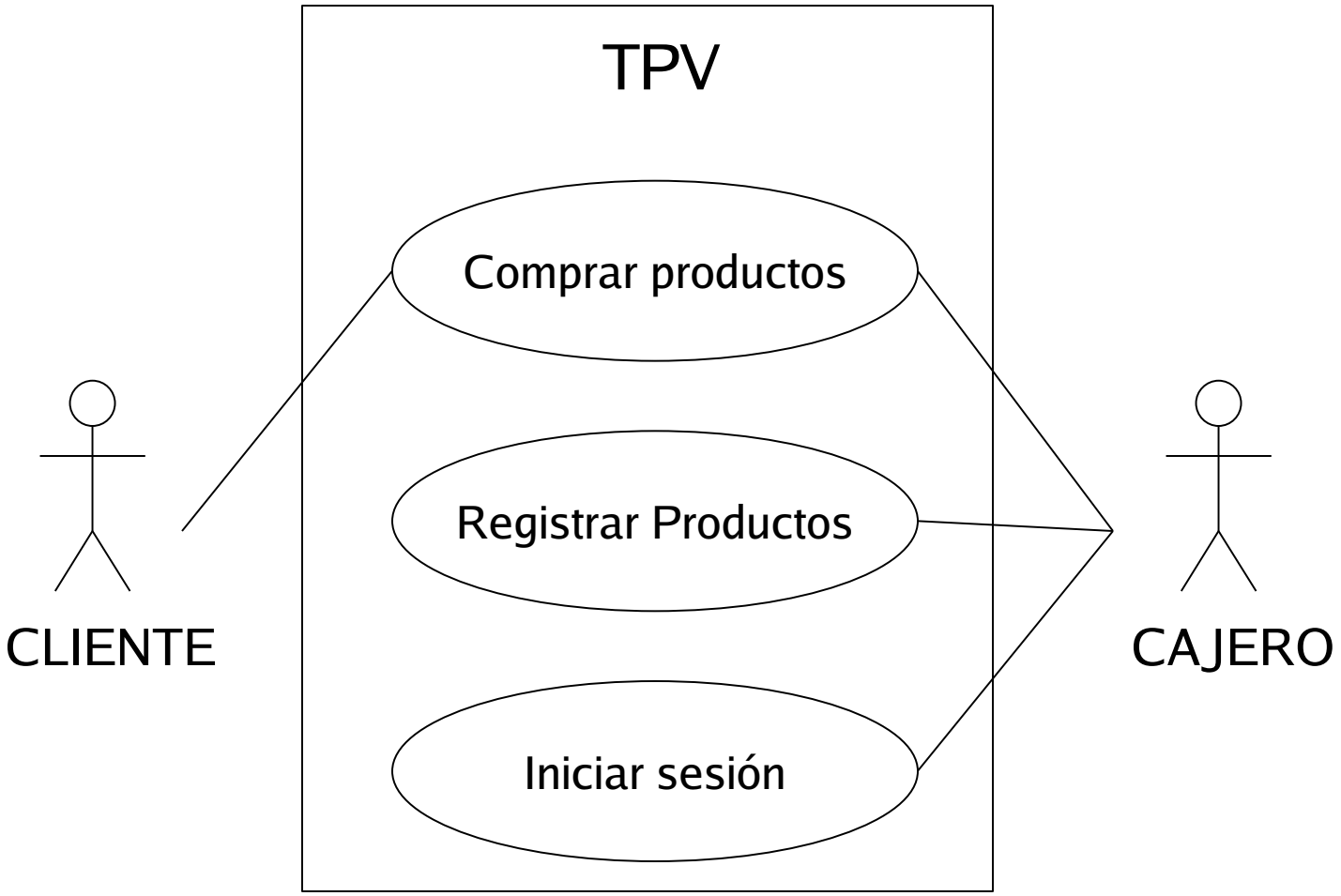
Ejemplo TPV: Diagrama de Casos de Uso



Ejemplo TPV: caso de uso real (1)

Caso de uso: **Comprar productos v1**

Actores: Cliente, Cajero (principal)

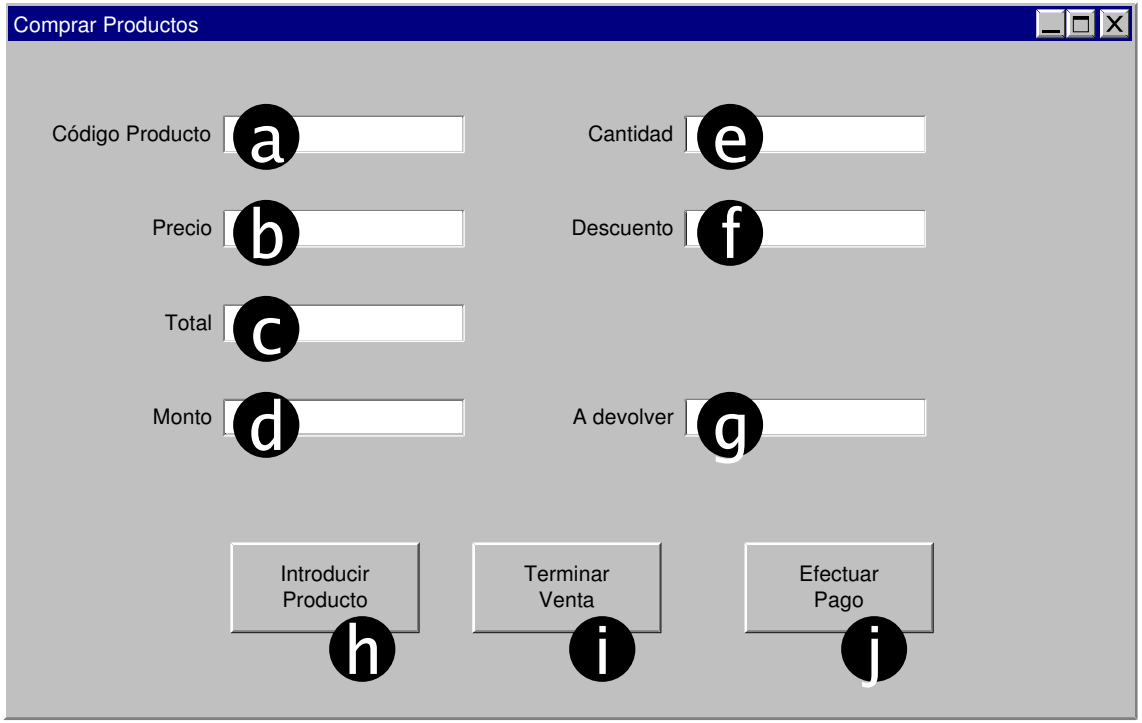
Resumen: Un Cliente llega a la caja registradora con los artículos que desea comprar. El Cajero registra los artículos y recibe un pago. Al terminar la operación, el Cliente se marcha con los productos comprados.

Precondiciones: El Cajero está identificado.

Postcondiciones: Se registra la venta completa, su importe y los impuestos. Se actualiza el inventario.

Referencias: R1.1, R1.2, R1.3, R1.4, R1.5, R1.7

Ejemplo TPV: caso de uso real (2)



Ejemplo TPV: caso de uso completo (2)

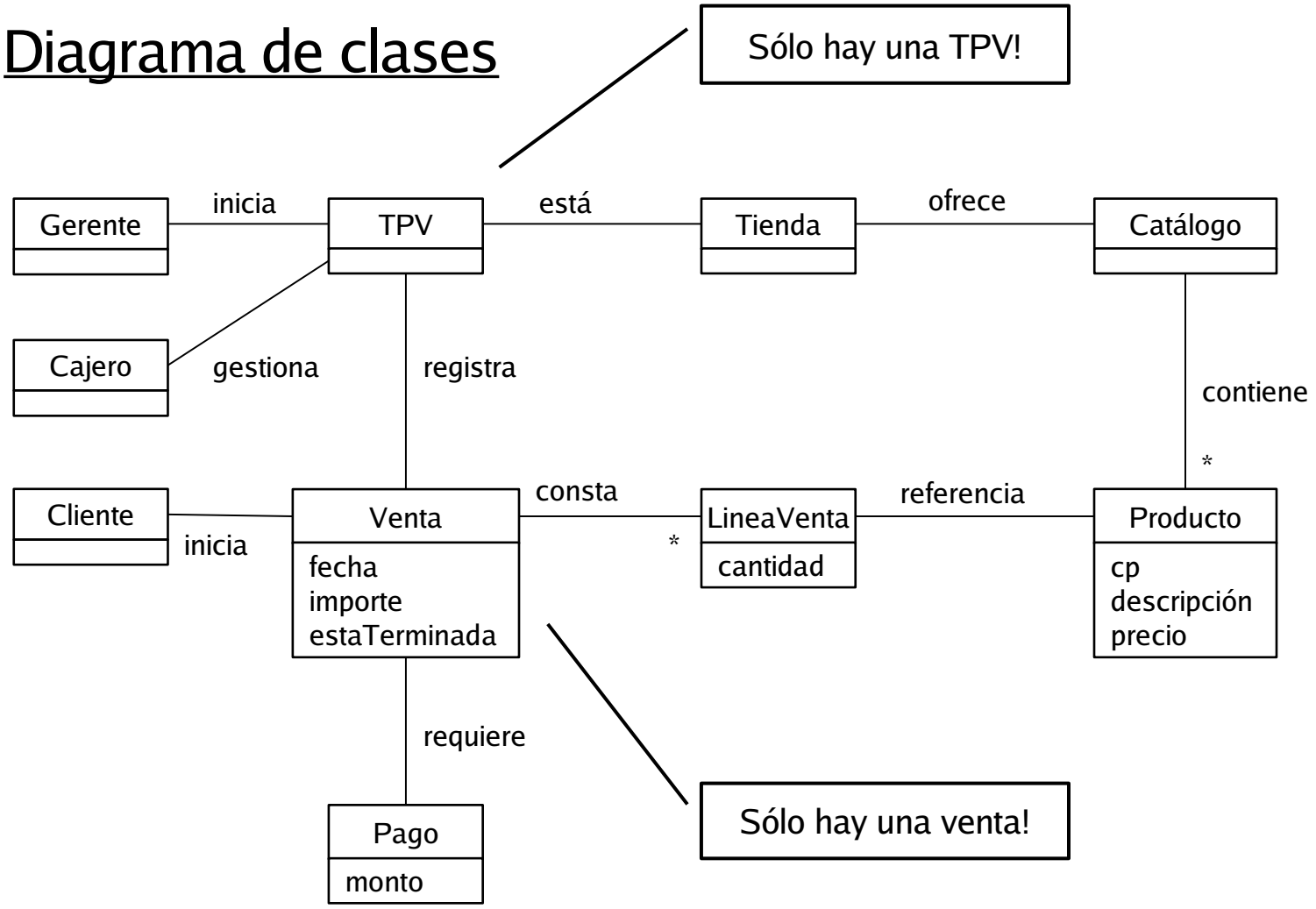
Escenario principal (o curso normal de los eventos):

1. **Cliente**: Llega a un TPV con productos que desea comprar.
2. **Cajero**: Comienza una nueva venta.
3. **Cajero**: Teclea el código de producto en **a** de la ventana-cp. Si hay más de un producto, es opcional añadir la cantidad en **e**. Se añade el producto con **h**.
4. **Sistema**: Registra la línea de la venta, y presenta el precio en **b** de la ventana-cp y la suma parcial en **c**.

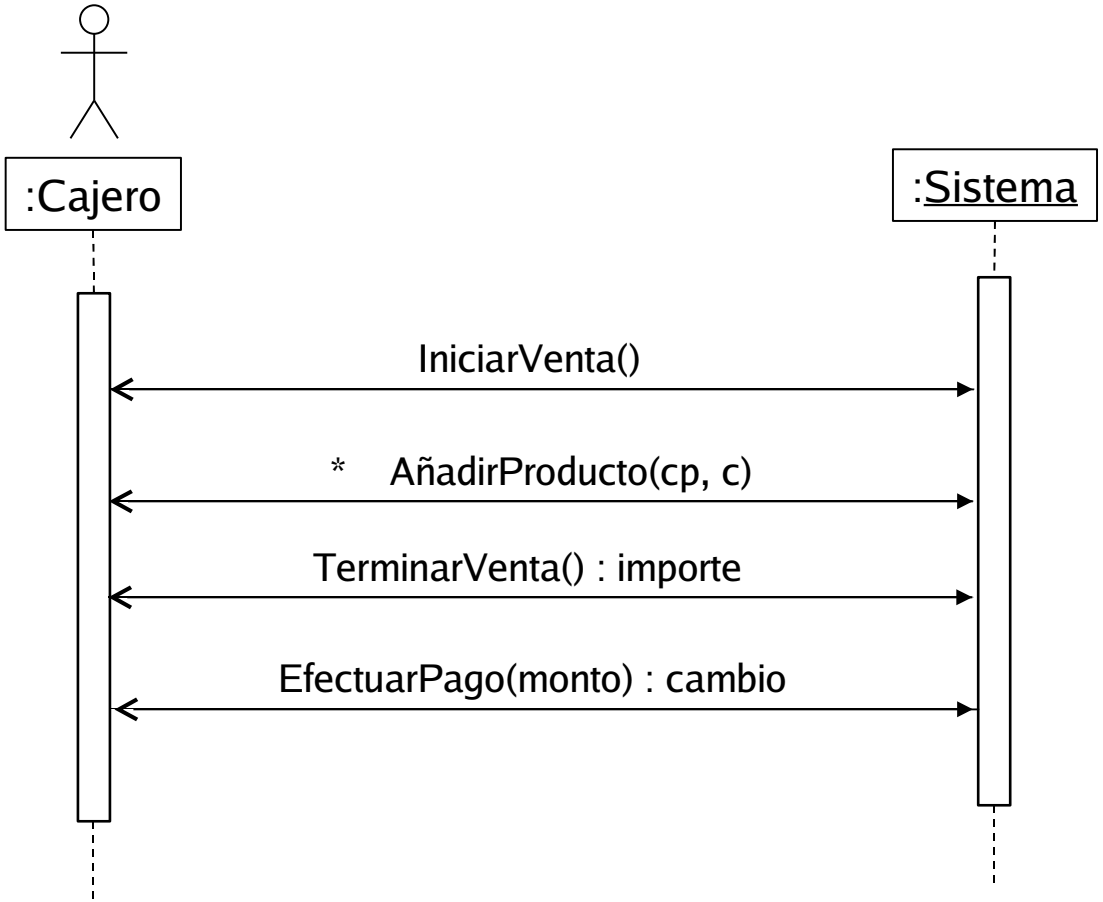
El Cajero repite los pasos 3 a 4 hasta terminar los artículos del Cliente.

5. **Cajero**: Indica al TPV que se concluyó la captura de productos pulsando **i**.
6. **Sistema**: Calcula y presenta el total con impuestos de la venta en **c**.
7. **Cajero**: Le indica el total de la venta al Cliente.
8. **Cliente**: Efectúa un pago.
9. **Cajero**: Gestiona el pago.
10. **Sistema**: Registra la venta. Genera un recibo.
11. **Cajero**: Da al Cliente el recibo impreso.
12. **Cliente**: Se marcha con los artículos comprados.

Diagrama de clases



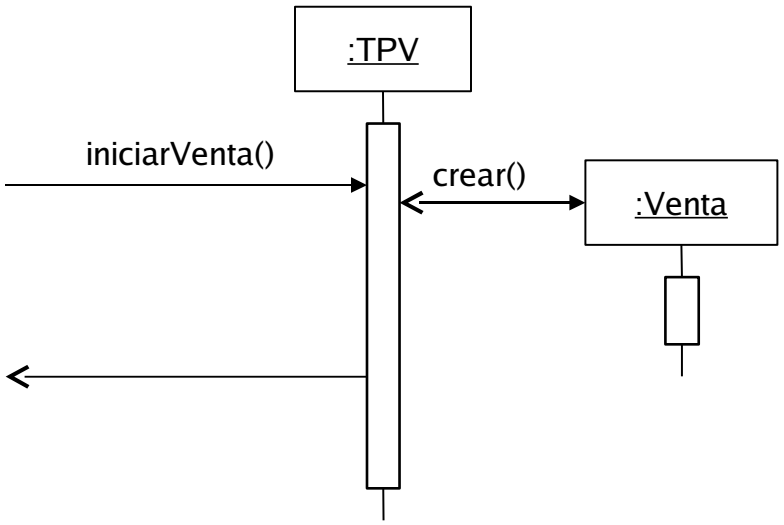
Ejemplo: Caso de uso Comprar productos



Contrato IniciarVenta

- **Name**: IniciarVenta()
- **Responsabilities**
  - Iniciar el registro de una venta
- **Preconditions**
  - No existe una venta ya iniciada
- **Postconditions**
  - Se dio de alta una instancia v de Venta con fecha actual
  - Se dio de alta una instancia de la asociación ‘tiene’ entre v y la instancia de TPV
- **Salida**

Diagrama de secuencia: IniciarVenta



Código Java IniciarVenta: TPV

```
package TPV; import java.util.*;

class TPV {
    private Catalogo catalogo;
    private Venta venta;
    ...

    private boolean esNuevaVenta() { return (venta==null) || (venta.estaTerminada())}
    public void IniciarVenta () {
        if (esNuevaVenta()) {
            venta = new Venta();
        }
    }
}
```

Código Java AñadirProducto: Venta

```
package TPV; import java.util.*;

class Venta {
    private Vector lineaVenta = new Vector();
    private Date fecha = new Date();
    private float importe = 0;
    private boolean estaTerminada = false;
    ...
}
```

Contrato AñadirProducto

- **Name**:AñadirProducto(cp, c)
- **Responsabilities**
  - Registrar una línea de venta.
- **Preconditions**
  - Existe un producto p con p.cp = cp
  - c >= 0
  - Existe una Venta v asociada a TPV
- **Postconditions**
  - Se dio de alta una instancia de LíneaVenta lv con lv.cantidad = c
  - Se dio de alta una instancia de la asociación 'consta' entre lv y la Venta v
  - Se dio de alta una instancia de la asociación 'referencia' que asocia lv y el producto p con p.cp = cp
- **Salida**
  - p.descripción y p.precio

Diagrama de secuencia: AñadirProducto

- Selección de la clase controlador

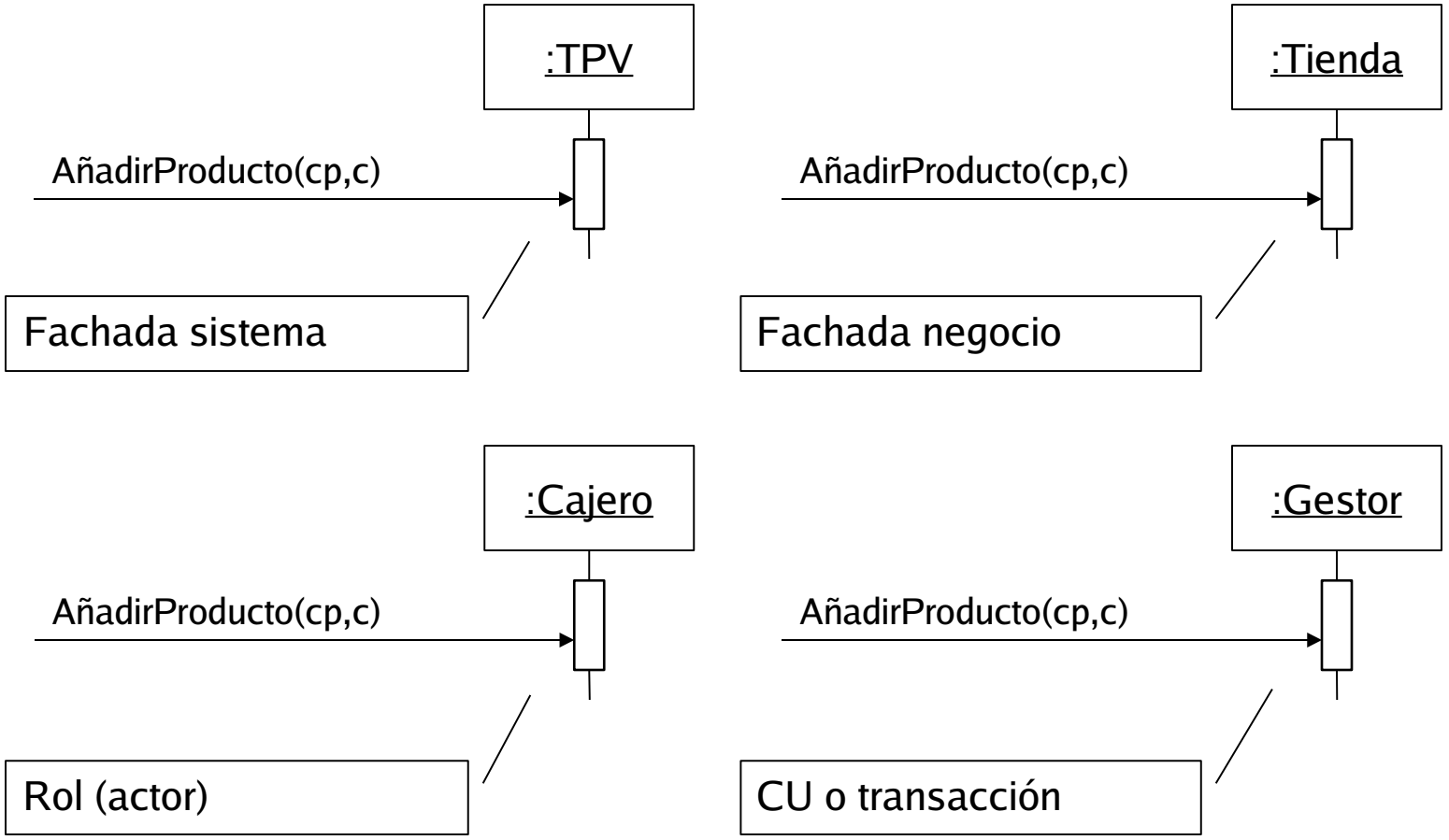


Diagrama de secuencia: AñadirProducto

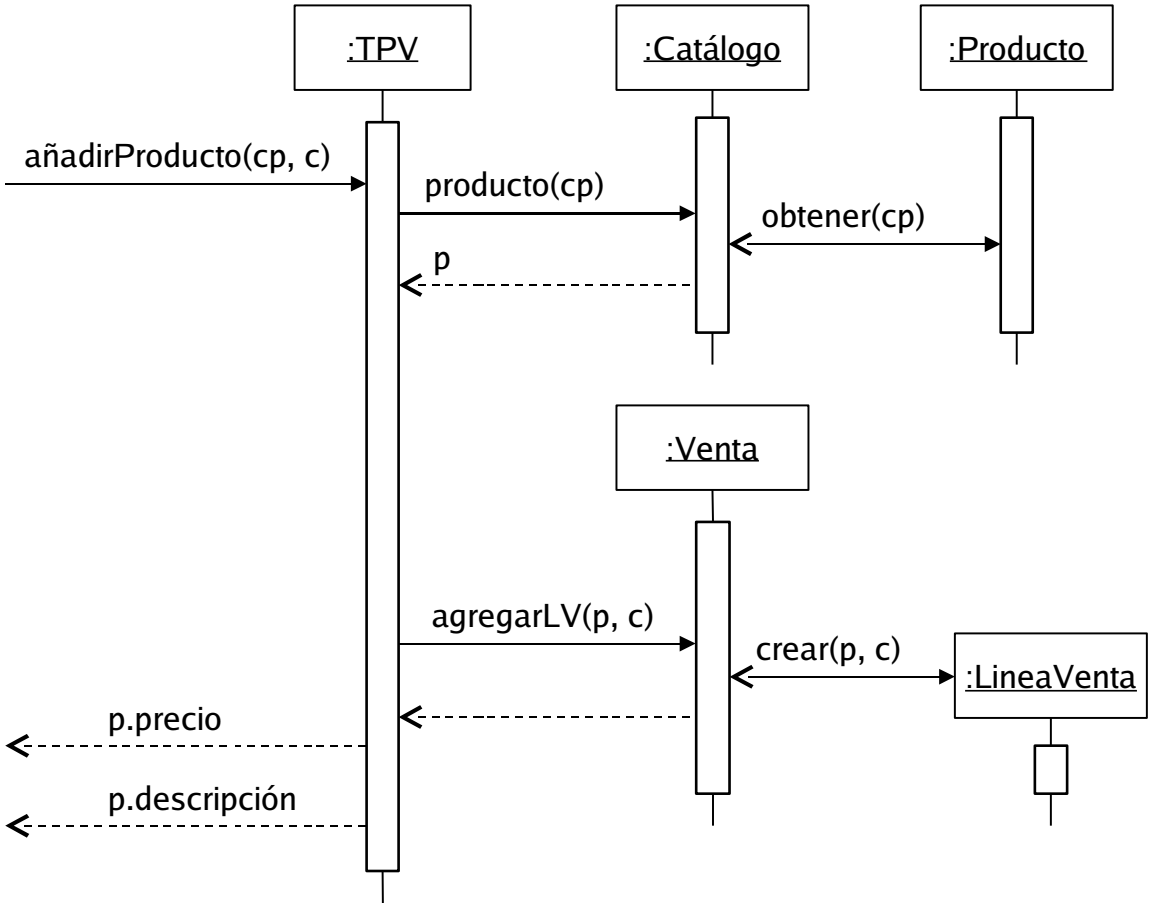
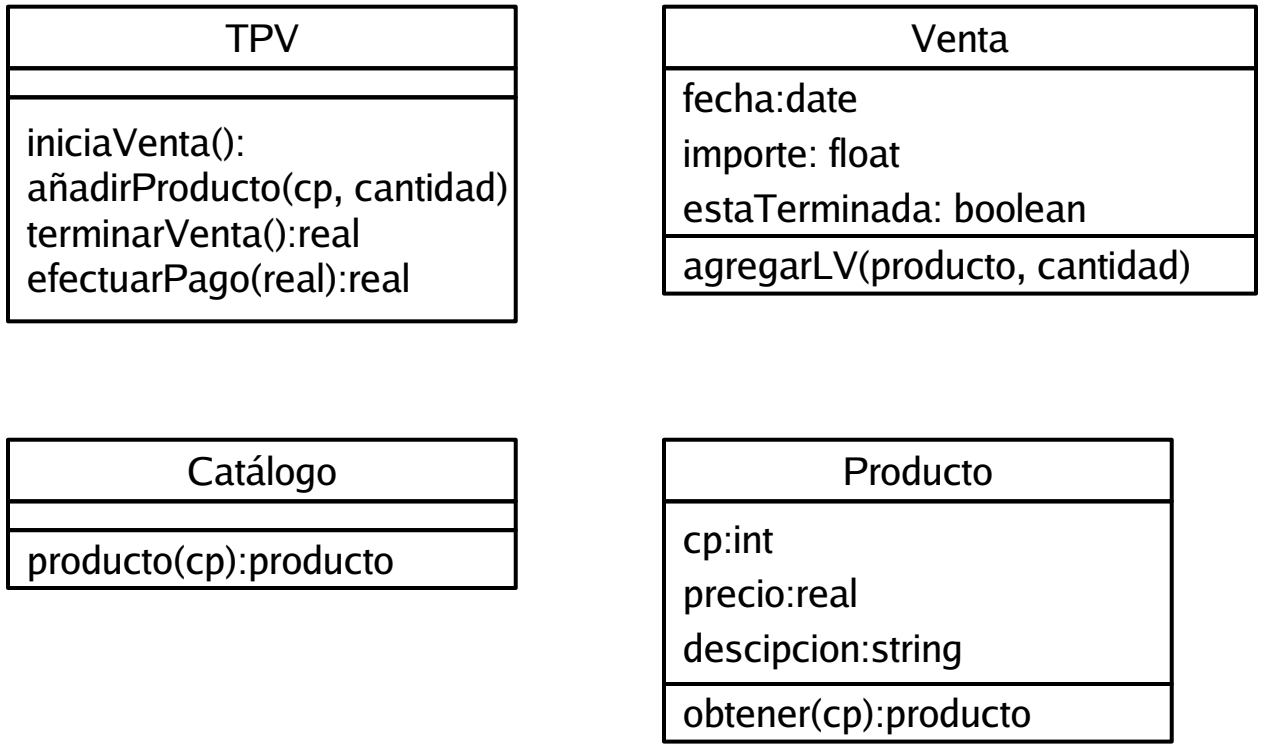


Diagrama de clases: AñadirProducto



Código Java AñadirProducto: TPV

```
package TPV; import java.util.*;

class TPV {
    private Catalogo catalogo;
    private Venta venta;
    ...

    public void añadirProducto (int cp, int cantidad) {
        Producto p = catalogo.producto(cp);
        venta.agregarLV(p, cantidad);
    }
}
```

Contrato AñadirProducto

- **Name**:AñadirProducto(cp, c)
- **Responsabilities**
  - Registrar una línea de venta.
- **Preconditions**
  - Existe un producto p con p.cp = cp
  - c >= 0
  - Existe una Venta v asociada a TPV
- **Postconditions**
  - Se dio de alta una instancia de LíneaVenta lv con lv.cantidad = c
  - Se dio de alta una instancia de la asociación 'consta' entre lv y la Venta v
  - Se dio de alta una instancia de la asociación 'referencia' que asocia lv y el producto p con p.cp = cp
- **Salida**
  - p.descripción y p.precio

Quién es el responsable de comprobar esto?

Código Java AñadirProducto: Catalogo

```
package TPV; import java.util.*;

public class Catalogo {
    private Hashtable catalogo = new Hashtable();

    public Catalogo() {
        catalogo c = new producto(100, 2, “producto 100”);
        catalogo.put(new Integer(100), c);
        catalogo c = new producto(200, 1, “producto 200”);
        catalogo.put(new Integer(200), c);
    }

    public producto producto (int cp) {
        return (producto) catalogo.get(new Integer (cp));
    }
}
```

Código Java AñadirProducto: Producto

```
package TPV;

public class Producto {
    private int cp = 0;
    private float precio = 0;
    private String descripcion = “”;

    public producto (int cp, float precio, String descripcion) {
        this.cp = cp;
        this.precio = precio;
        this.descripcion = descripcion;
    }
    ...
}
```

Código Java AñadirProducto: Venta

```
package TPV; import java.util.*;

class Venta {
    private Vector lineaVenta = new Vector();
    private Date fecha = new Date();
    private float importe = 0;
    private boolean estaTerminada = false;
    ...
    public void agregarLV (producto p, int c){
        lineaVenta.addElement(new LineaVenta(p, c));
    }
}
```

Código Java AñadirProducto: LineaVenta

```
package TPV;

class LineaVenta {
    private int cantidad;
    private Producto producto;
    ...
    public LineaVenta (producto p, int cantidad) {
        this.producto = p;
        this.cantidad = cantidad;
    }
}
```

Contrato TerminarVenta

- **Name**:TerminarVenta() : importe
- **Responsabilities**
  - Finalizar el registro de una venta y mostrar el importe pagado
- **Preconditions**
  - Existe una Venta v asociada a TPV
- **Postconditions**
  - v.importe = suma de los subtotales de todas las lineas de venta asociadas a la Venta v
  - v.terminada = verdadero
- **Salida**
  - importe = v.importe

Diagrama de secuencia: TerminarVenta

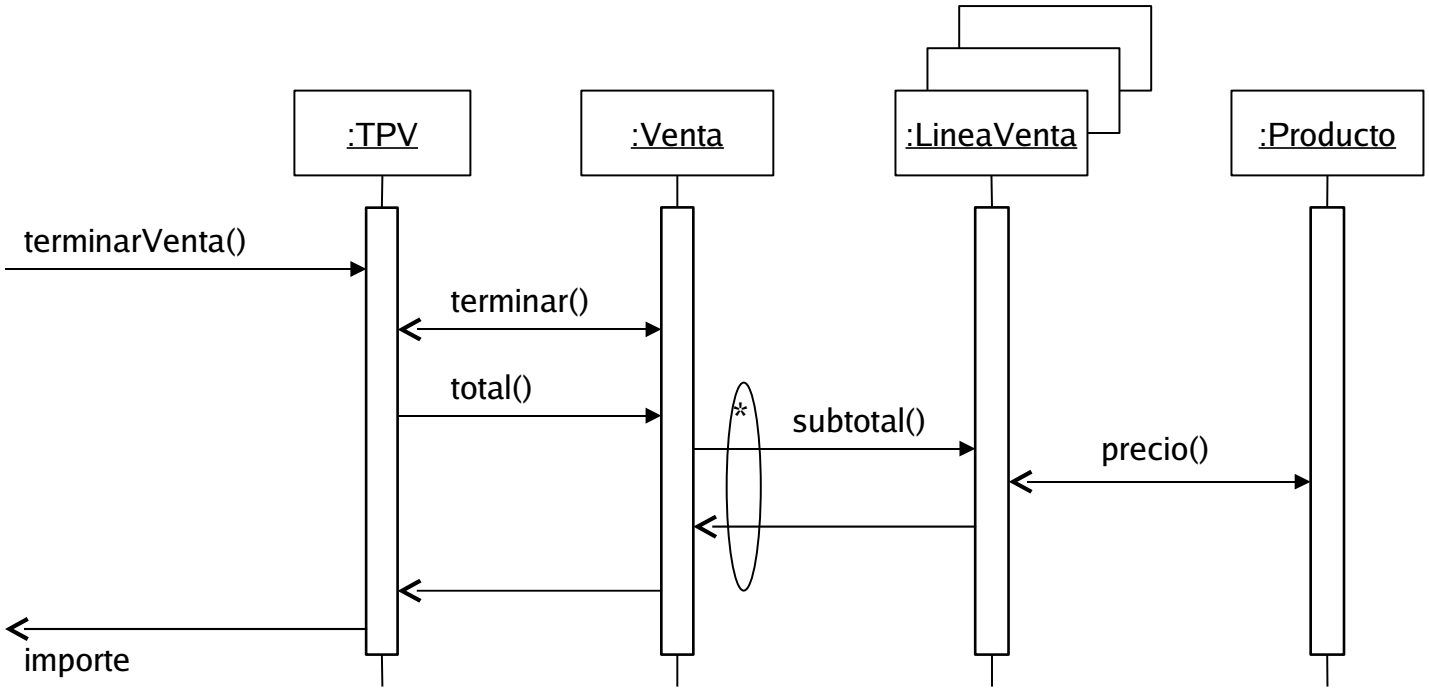
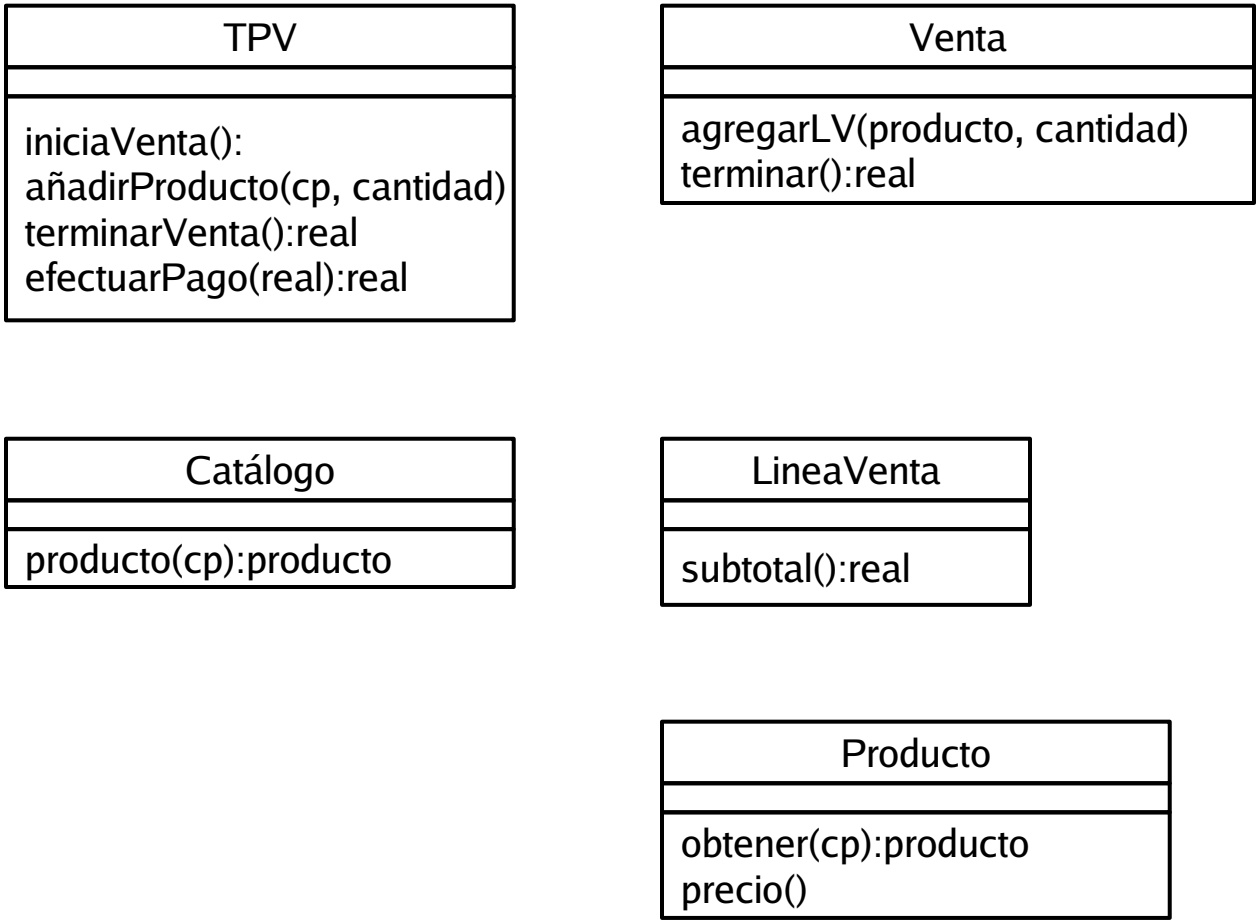


Diagrama de clases: TerminarVenta



Código Java TerminarVenta: TPV

```
package TPV; import java.util.*;

class TPV {
    private Catalogo catalogo;
    private Venta venta;
    ...

    public float terminarVenta () {
        venta.terminar();
        return (float) venta.total();
    }
}
```

Código Java TerminarVenta: Venta

```
package TPV; import java.util.*;

class Venta {
    private Vector lineaVenta = new Vector();
    private Date fecha = new Date();
    private float importe = 0;
    private boolean estaTerminada = false;
            ...
    public void terminar () { estaTerminada = true; }
    public float total() {
            float total = 0;
            enumeration e = lineaVenta.elements();
            while (e.hasMoreElements()) {
                    total += ( (LineaVenta) e.nextElement() ).subtotal();
            }
            importe = total;
            return total;
    }
}
```

Código Java TerminarVenta: LineaVenta

```
package TPV;

class LineaVenta {
    private int cantidad;
    private Producto producto;
    ...
    public float subtotal () {
        return cantidad * producto.precio();
    }
}
```

Código Java TerminarVenta: Producto

```
package TPV;

public class Producto {
    private int cp = 0;
    private float precio = 0;
    private String descripcion = “”;

    public float precio () { return precio;}
    ...
}
```

Diagrama de secuencia: EfectuarPago

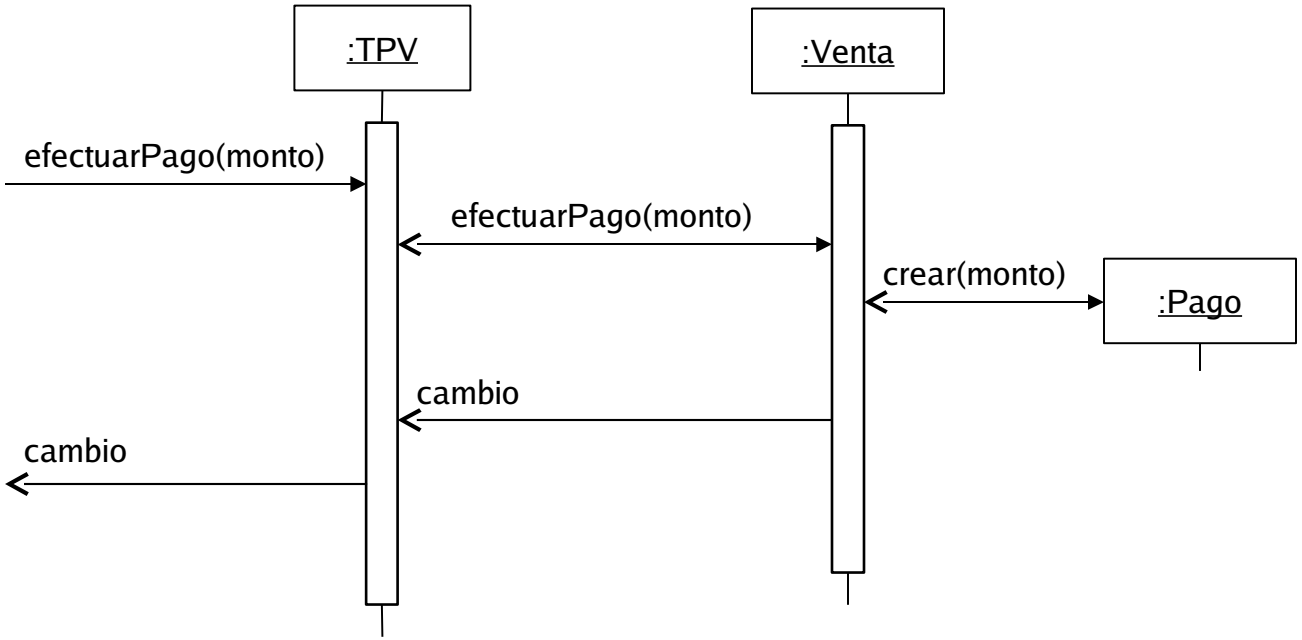
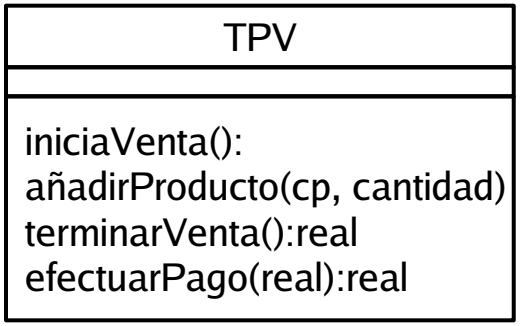
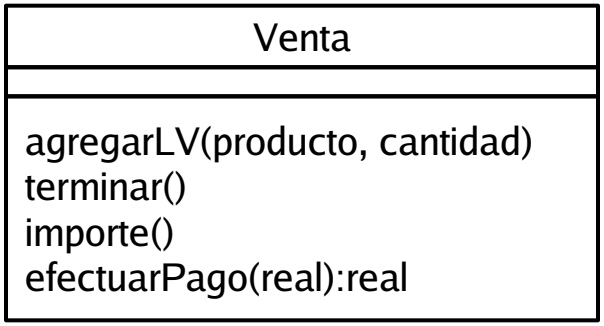
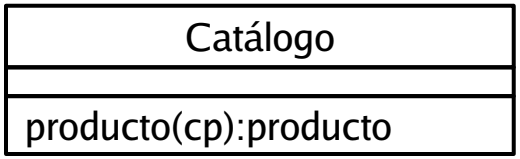
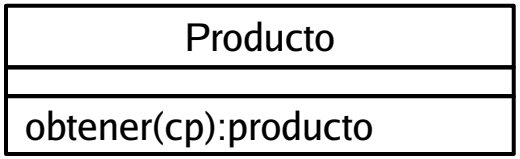


Diagrama de clases: EfectuarPago









Código Java EfectuarPago: TPV

```
package TPV; import java.util.*;

class TPV {
    private Catalogo catalogo;
    private Venta venta;
    ...

    public float efectuarPago (float monto) {
        return venta.efectuarPago(monto);
    }
}
```

Código Java EfectuarPago: Venta

```
package TPV; import java.util.*;

class Venta {
    private Vector lineaVenta = new Vector();
    private Date fecha = new Date();
    private float importe = 0;
    private boolean estaTerminada = false;
    private Pago pago;

    public void efectuarPago (float monto) {
        pago = new Pago(monto)
        return pago.monto - importe
    }

}
```

Código Java EfectuarPago: Pago

```
package TPV;

public class Pago {
    private float monto;

    public Pago (float monto) {
        this.monto = monto;
    }

}
```