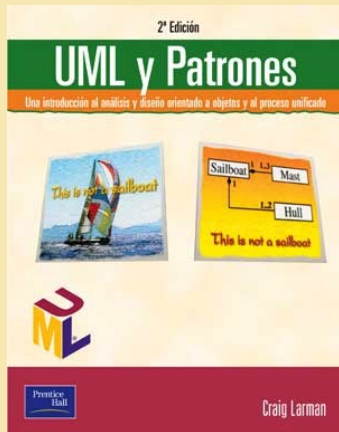


Capítulo 17

Modelo de Diseño: realización de los casos de uso con los patrones GRASP



UML y Patrones

Una introducción al análisis y diseño orientado a objetos y al proceso unificado

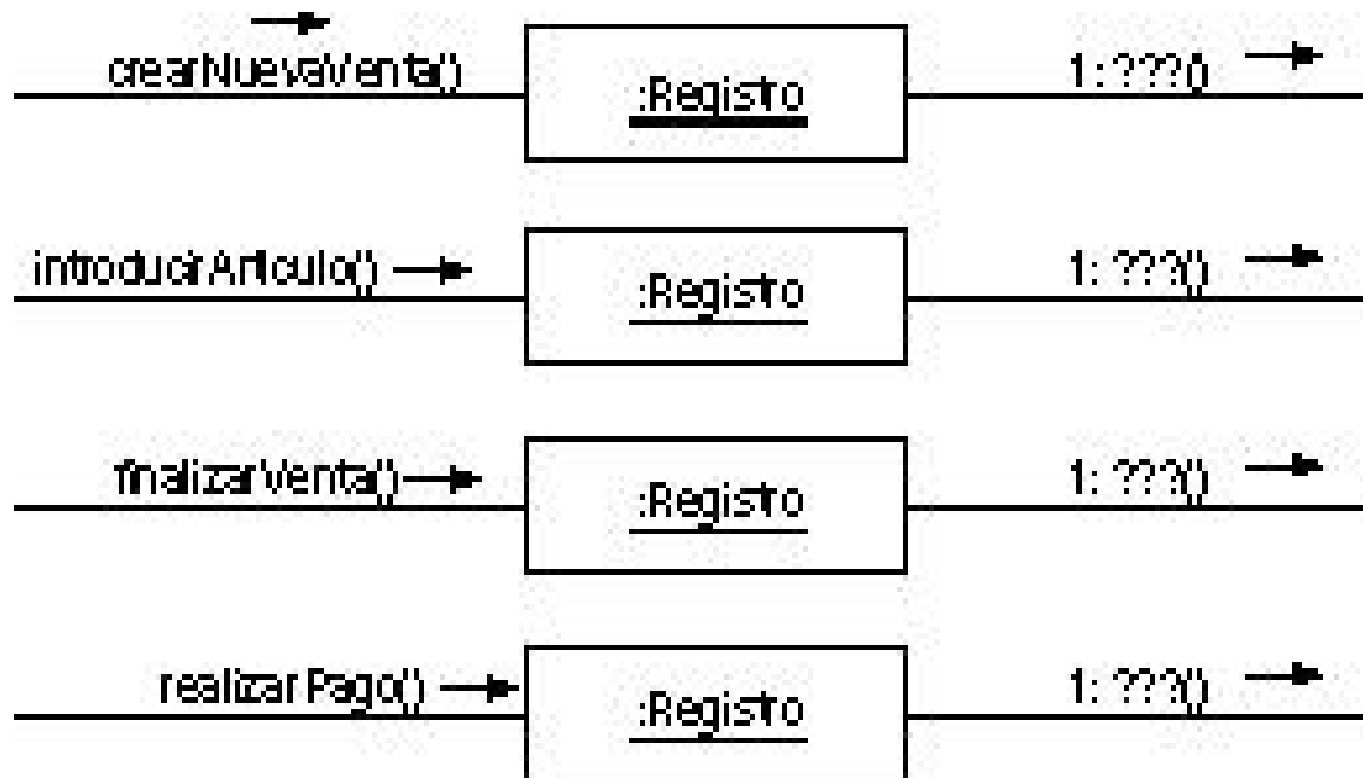
Craig Larman

2ª Edición

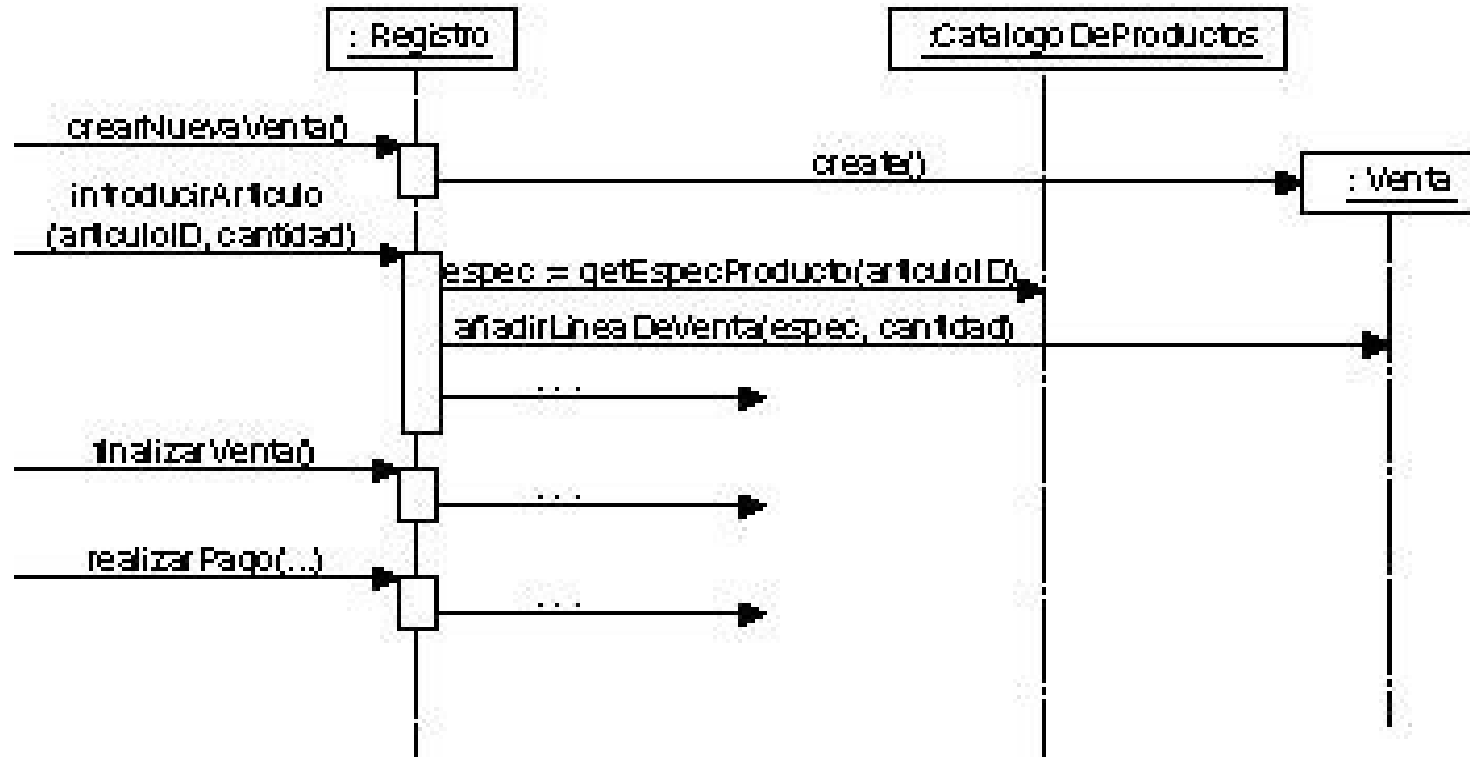
Realización de casos de uso

- La asignación de responsabilidades y el diseño de colaboraciones son etapas muy importantes y creativas durante el diseño, mientras se elaboran los diagramas o mientras se programa.
- La realización de un caso de uso describe cómo se realiza un caso de uso particular en el modelo de diseño, en función de los objetos que colaboran.

Operaciones del sistema

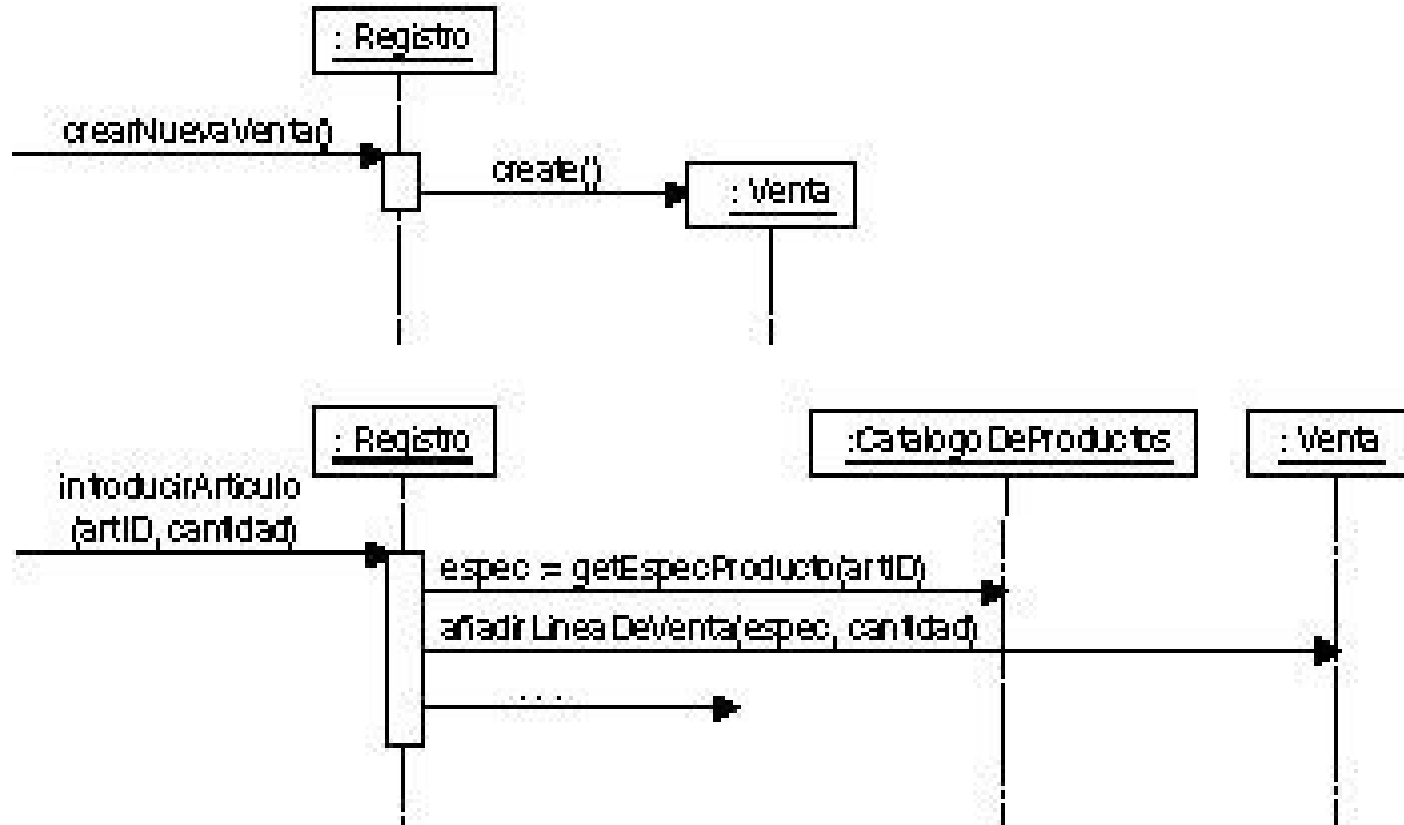


Realización de casos de uso



- No aspiramos a describirlos todos en un sólo diagrama

Figura 17.3. Múltiples diagramas de secuencia y manejo de los mensajes de eventos del sistema.



Realización de casos de uso

- Para algunas operaciones de sistema escribiremos su contrato

Contrato CONT02: IntroducirArticulo

Operación: IntroducirArticulo(articuloID:ArticuloID,cantidad:integer)

Referencias: caso de uso Procesar Venta

Precondiciones: Hay una venta en curso

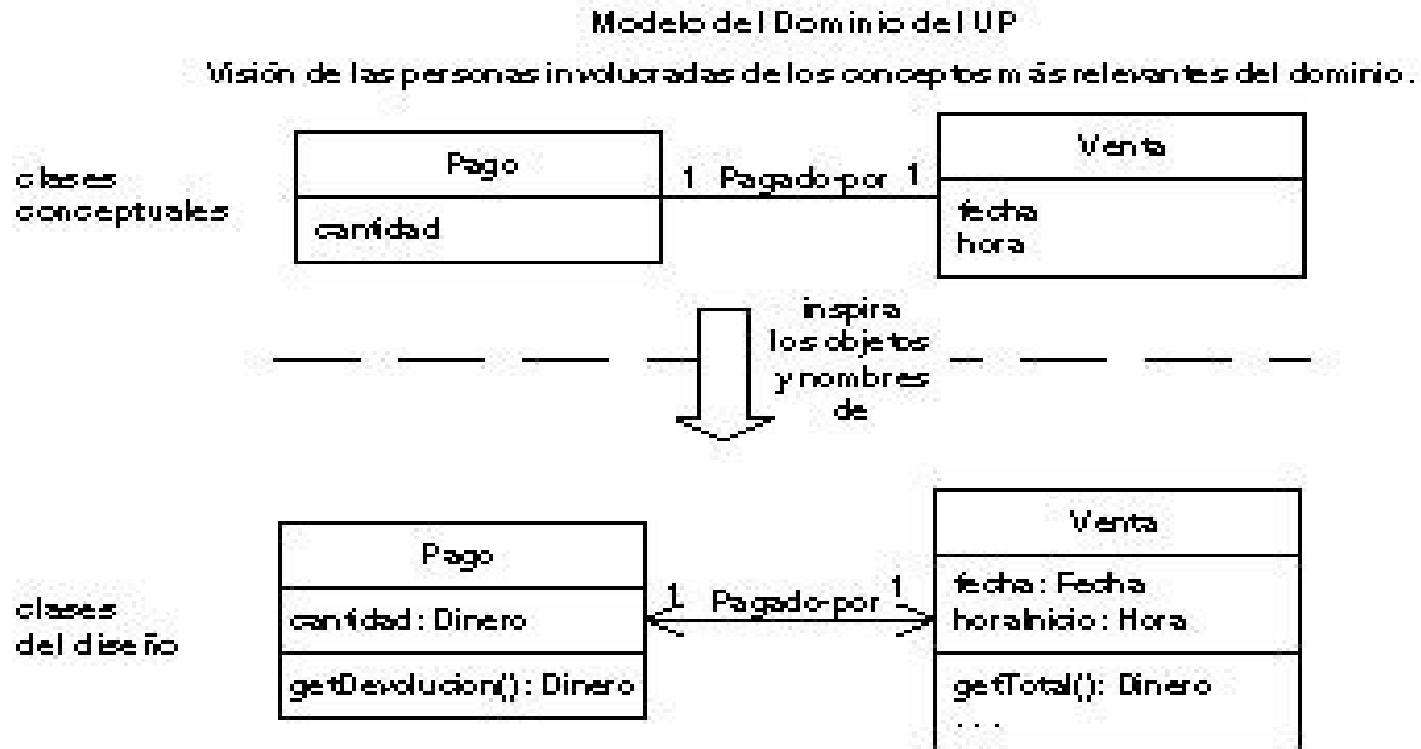
Postcondiciones:

- Se creó una instancia de LineaDeVenta ldv
- Se asoció ldv con la Venta actual
- Se asoció ldv con una EspecificaciónDelProducto, en base al valor del articuloID

Realización de casos de uso

- Las clases del modelo del sominio no son clases software
- Durante la realización de los casos de uso iremos inspirándonos en estas clases para proponer clases software con nombres parecidos,
- Iremos descubriendo nuevas clases del domino del problema,
- y proponiendo clases software sin ninguna correspondencia clara en el modelo del dominio

Figura 17.5. Disminución del salto en la representación nombrando las clases del diseño a partir de las clases conceptuales.



Modelo de Diseño del IUP

El desarrollador de objetos se ha inspirado en el dominio del mundo real al crear las clases software. Por tanto, se disminuye el salto en la representación entre el modo en el que las personas involucradas en el proyecto conciben el dominio y su representación en el software.

Realización de casos de uso

Contrato CONT01: CrearNuevaVenta

Operación: crearNuevaVenta()

Referencias: caso de uso Procesar Venta

Precondiciones: ninguna

Postcondiciones:

- Se creó una instancia de Venta v
- Se asoció v con el registro
- Se inicializaron los atributos de v

Realización de casos de uso

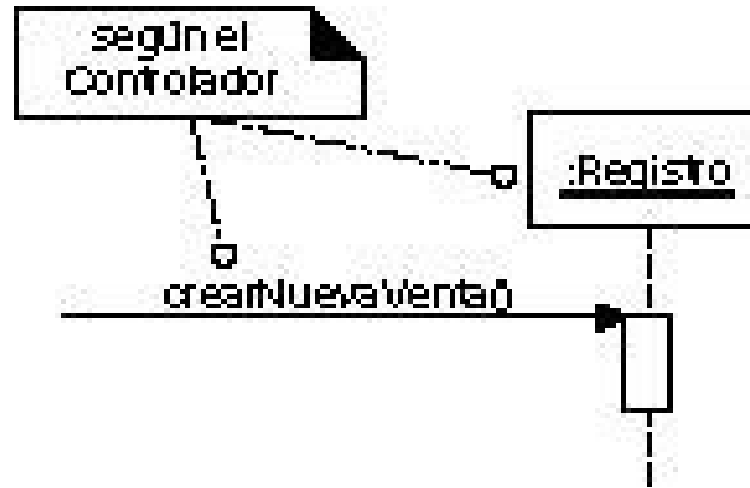
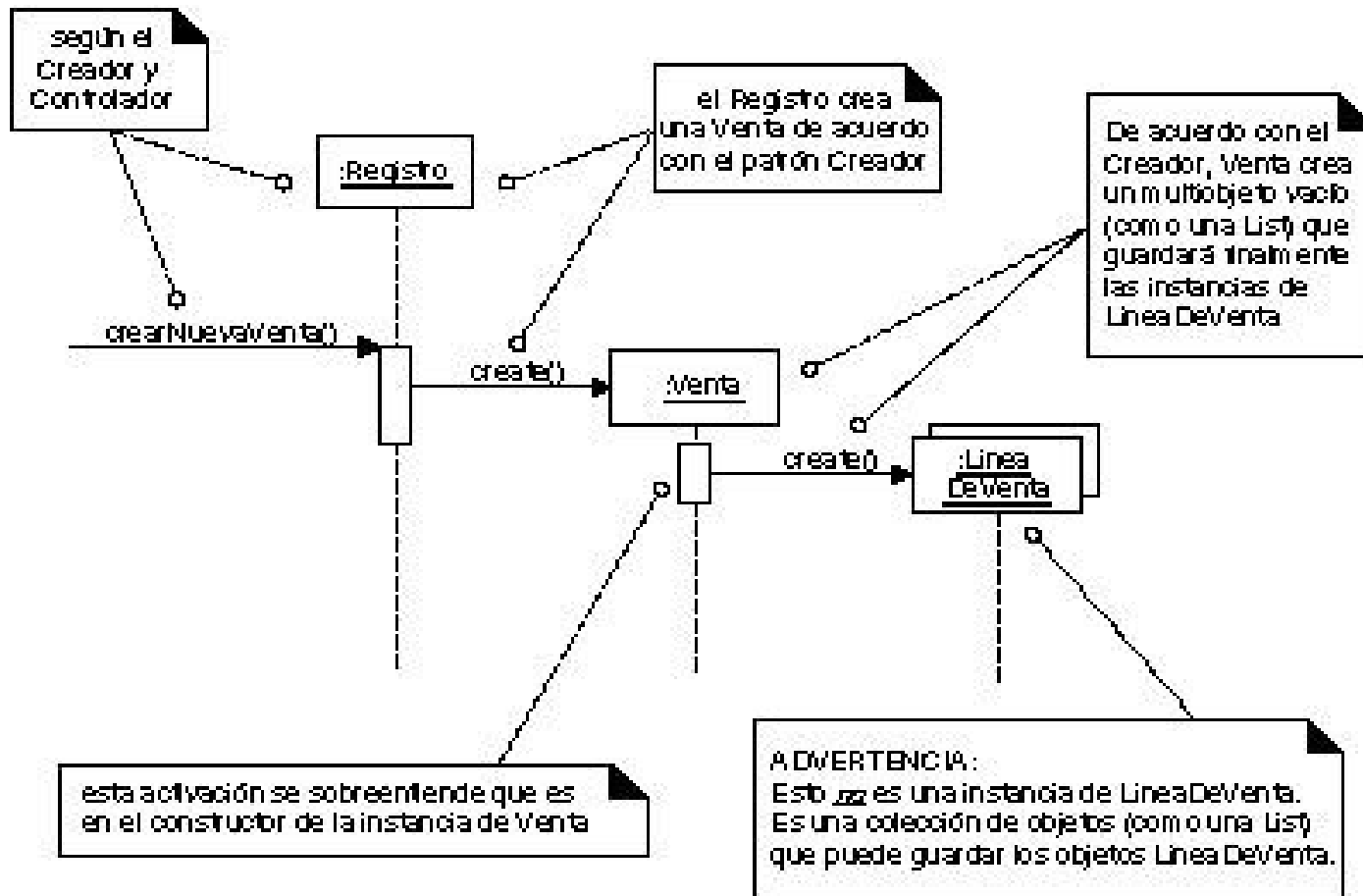


Figura 17.6. Aplicación del patrón GRASP Controlador.

Figura 17.7. Creación de Venta y multiobjeto.



Realización de casos de uso

Contrato CONT02: IntroducirArticulo

Operación: IntroducirArticulo(articuloID:ArticuloID,cantidad:integer)

Referencias: caso de uso Procesar Venta

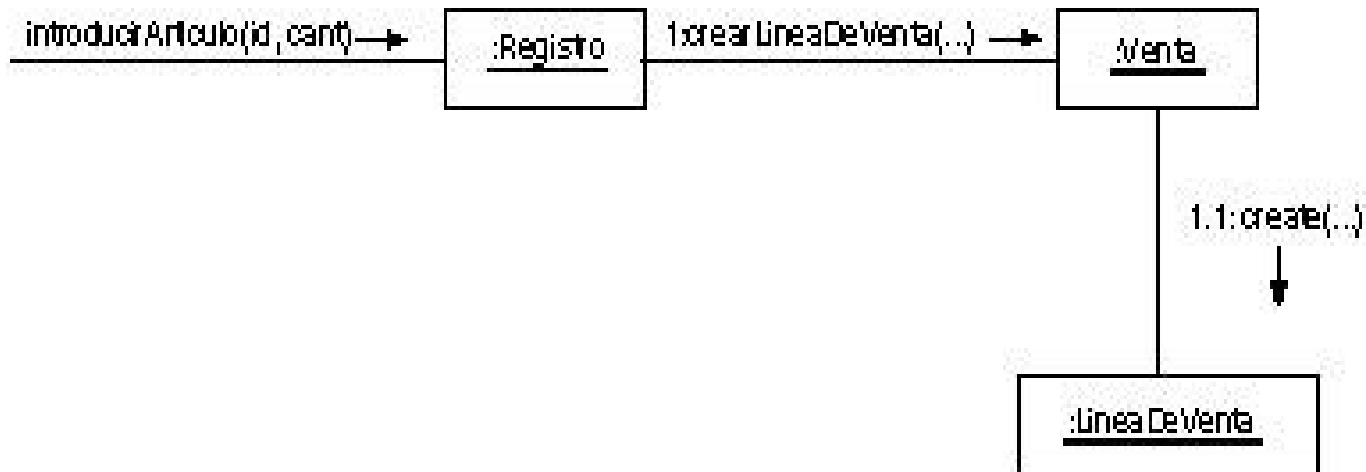
Precondiciones: Hay una venta en curso

Postcondiciones:

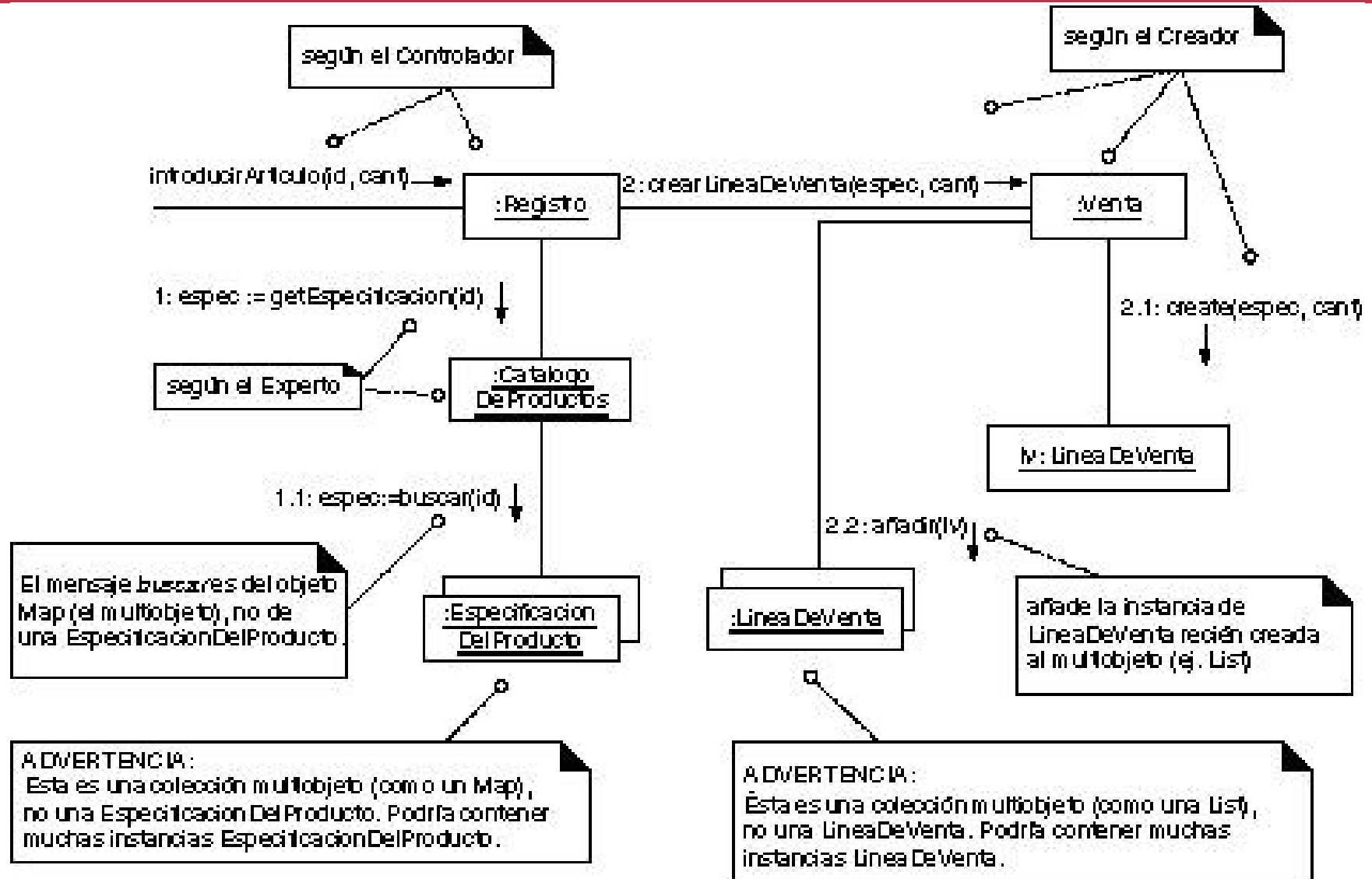
- Se creó una instancia de LineaDeVenta Idv
- Se asoció Idv con la Venta actual
- Idv.cantidad pasó a ser cantidad
- Se asoció Idv con una EspecificaciónDelProducto, en base al valor del articuloID

Realización de casos de uso

- Este diagrama cumple parte de las condiciones



Realización de casos de uso



Realización de casos de uso

- Es razonable asumir que durante el caso de uso inicial **Poner en marcha** se creó un **Registro** y un **CatalogoDeProductos**, y que éste es accesible desde **Registro**
- Será el **Registro** el que llame a **getEspecificacion()**
- El **CatálogoDeProductos** debe ser visible desde el **Registro**

Realización de casos de uso

- finalmente el **CatálogoDeProductos** residirá en una base de datos, pero de momento ese aspecto no lo consideramos
- También hemos obviado, por el momento, la presentación por pantalla de la información de la línea de venta, esta será una responsabilidad de la interface

Realización de casos de uso

Contrato CONT3: finalizarVenta

Operación: finalizarVenta()

Referencias: caso de uso Procesar Venta

Precondiciones: Hay una venta en curso

Postcondiciones:

→ `venta.esCompleta` pasó a ser `true`

Figura 17.9. Finalización de la entrada de un artículo.

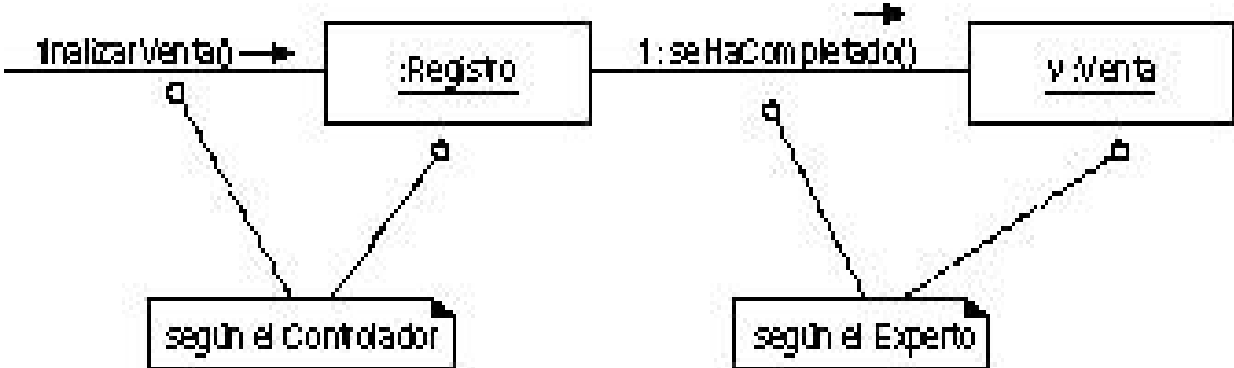


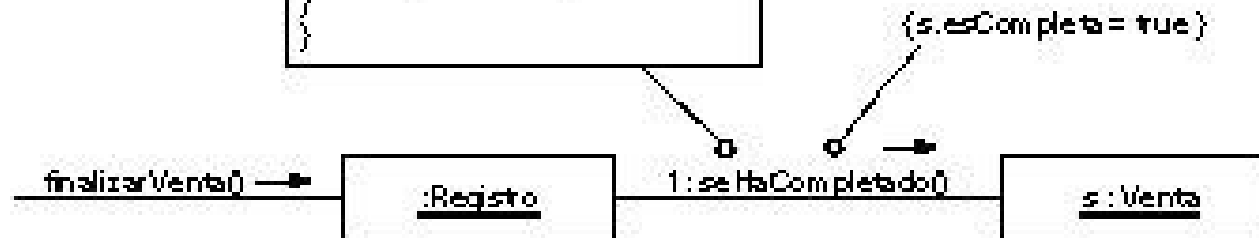
Figura 17.10. Restricciones y notas.

la implementación de una restricción en un cuadro de nota

obsérvese la llave más externa que rodea el método, significa que se trata de una restricción en un cuadro de nota

```
{  
  public void setHaCompletado()  
  {  
    esCompleta = true;  
  }  
}
```

una restricción que no define el algoritmo, pero especifica lo que se tiene que cumplir



// nota creada por Craig

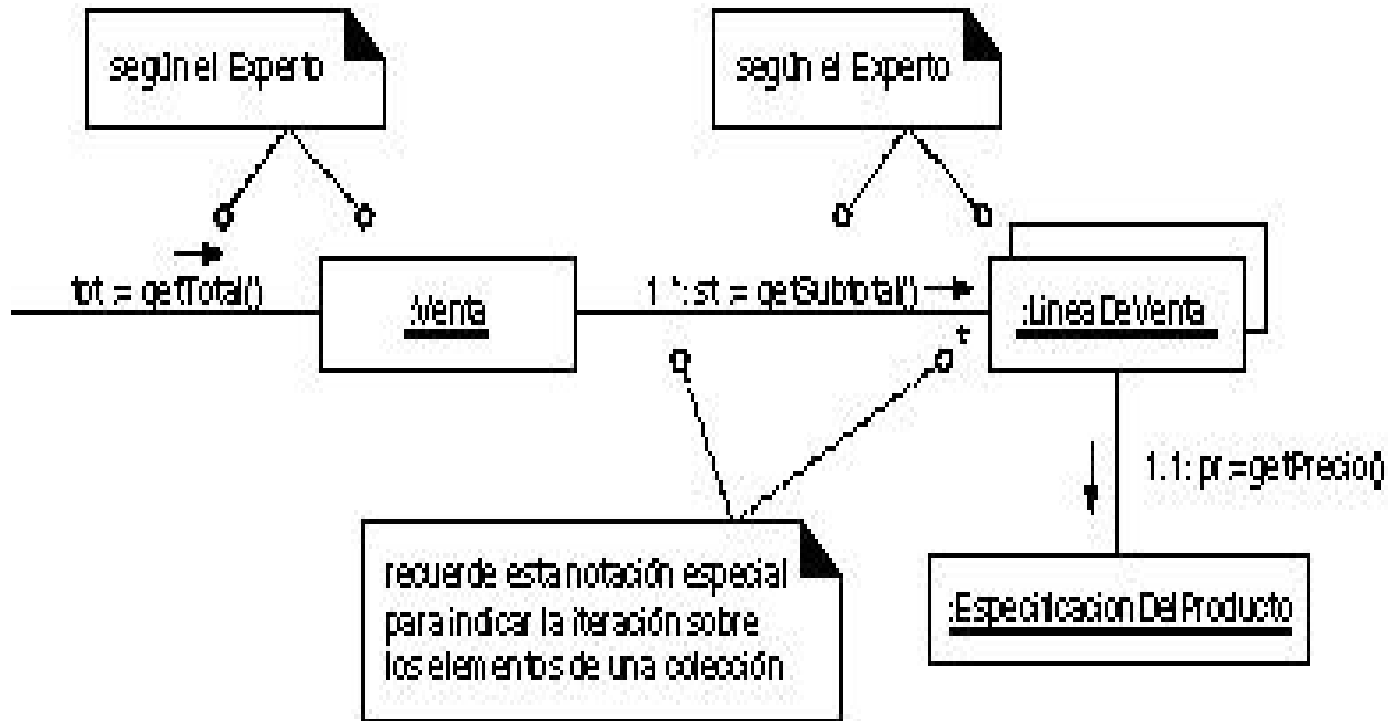
Realización de casos de uso

Caso de uso: Alquilar

Curso normal

Actor	Sistema
...	...
4. Para cada video o juego, el empleado graba en el sistema su número de identificación.	5. Presenta una lista de los títulos alquilados, fechas de devolución, precio total del alquiler y cargos por retraso en la devolución.
6. El empleado informa al cliente de la cantidad a abonar y le pide el pago..	
...	© Pearson Educación, S.A. Todos los derechos reservados.

Figura 17.11. Diagrama de interacción Venta--getTotal.



Clase	Responsabilidad
Venta	Calcular el importe total de una venta
LineaDeVenta	Calcular el subtotal de la línea de venta
EspecificaciónDelProducto	Conocer el precio del artículo

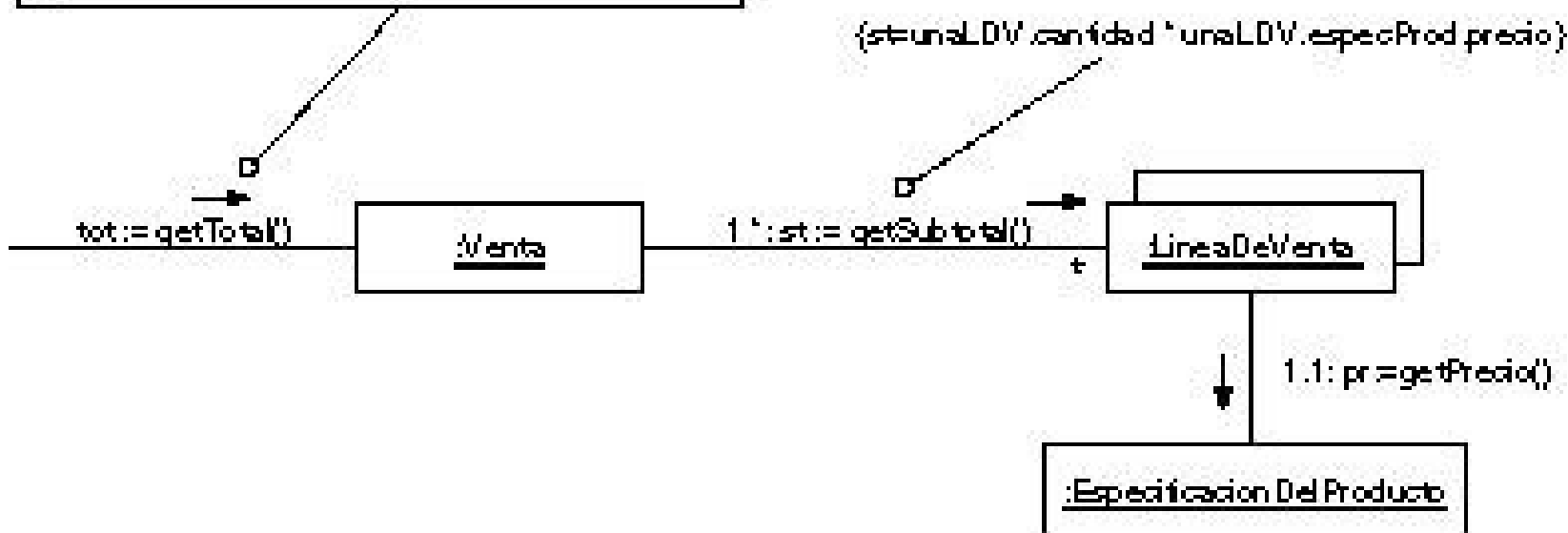
Experto (en información)

Figura 17.12. Notas de algoritmos y restricciones.

```
//observe el siguiente estilo de pseudocódigo  
{  
public void getTotal()  
{  
int tot = 0;  
para cada LíneaDeVenta, ldv  
tot = tot + ldv.getSubtotal();  
return tot  
}
```

Obsérvese el estilo semi-formal de la restricción. No se define formalmente "unaLDV", pero la mayoría de los desarrolladores entenderán razonablemente que significa una instancia de LíneaDeVenta. Lo mismo ocurre con la expresión unaLDV.especProd.precio.

Lo importante es que el lenguaje de restricción puede ser informal, para permitir una escritura rápida y fácil, si se desea.



Realización de casos de uso

Contrato CONT4: realizarPago

Operación: realizarPago(cantidad: Dinero)

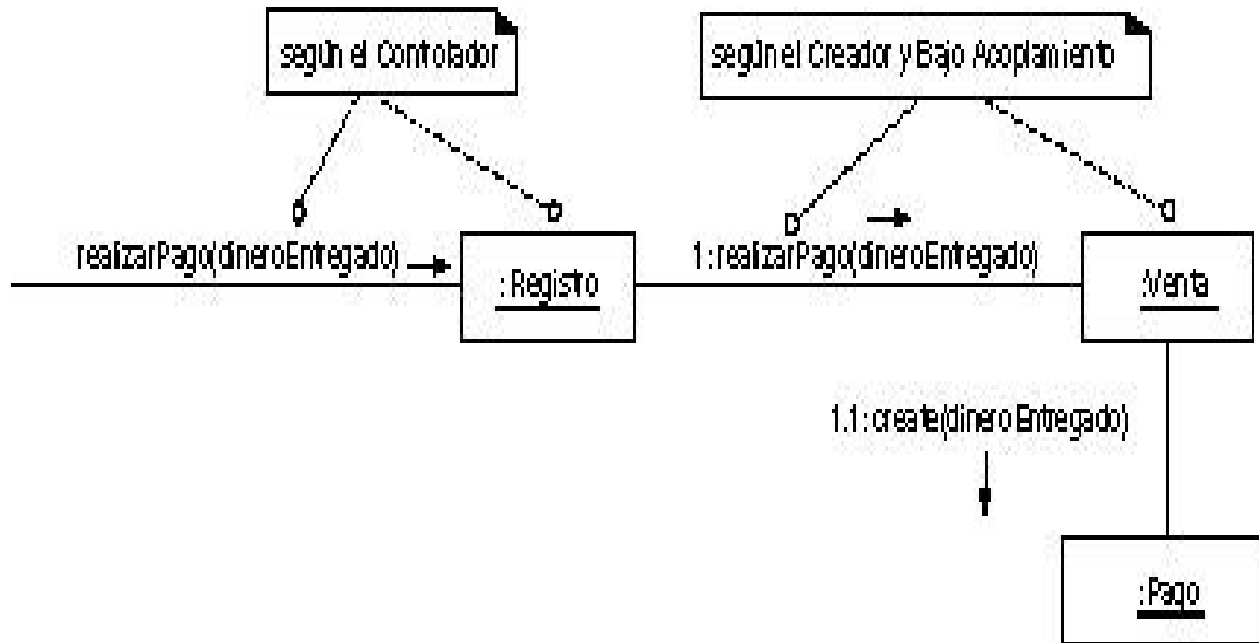
Referencias: caso de uso Procesar Venta

Precondiciones: Hay una venta en curso

Postcondiciones:

- Se creó una instancia de Pago p
- p.cantidadEntregada pasó a ser cantidad
- P se asoció con la Venta actual
- La Venta actual se asoció con la Tienda (para añadirlo al registro histórico de las ventas completadas)

Figura 17.13. Diagrama de interacción Registro--realizarPago.



- parte de las postcondiciones

Figura 17.14. ¿Quién debería ser responsable de conocer la venta completada?



La tienda es la responsable de conocer y añadir las ventas completadas.

Es aceptable en los primeros ciclos de desarrollo si la Tienda tiene pocas responsabilidades.



El LibroMayor De Ventas es el responsable de conocer la venta y añadir las ventas completadas.

Es adecuado cuando el diseño crece y la Tienda pierde la cohesión.

Figura 17.15. Registro de una venta completada.

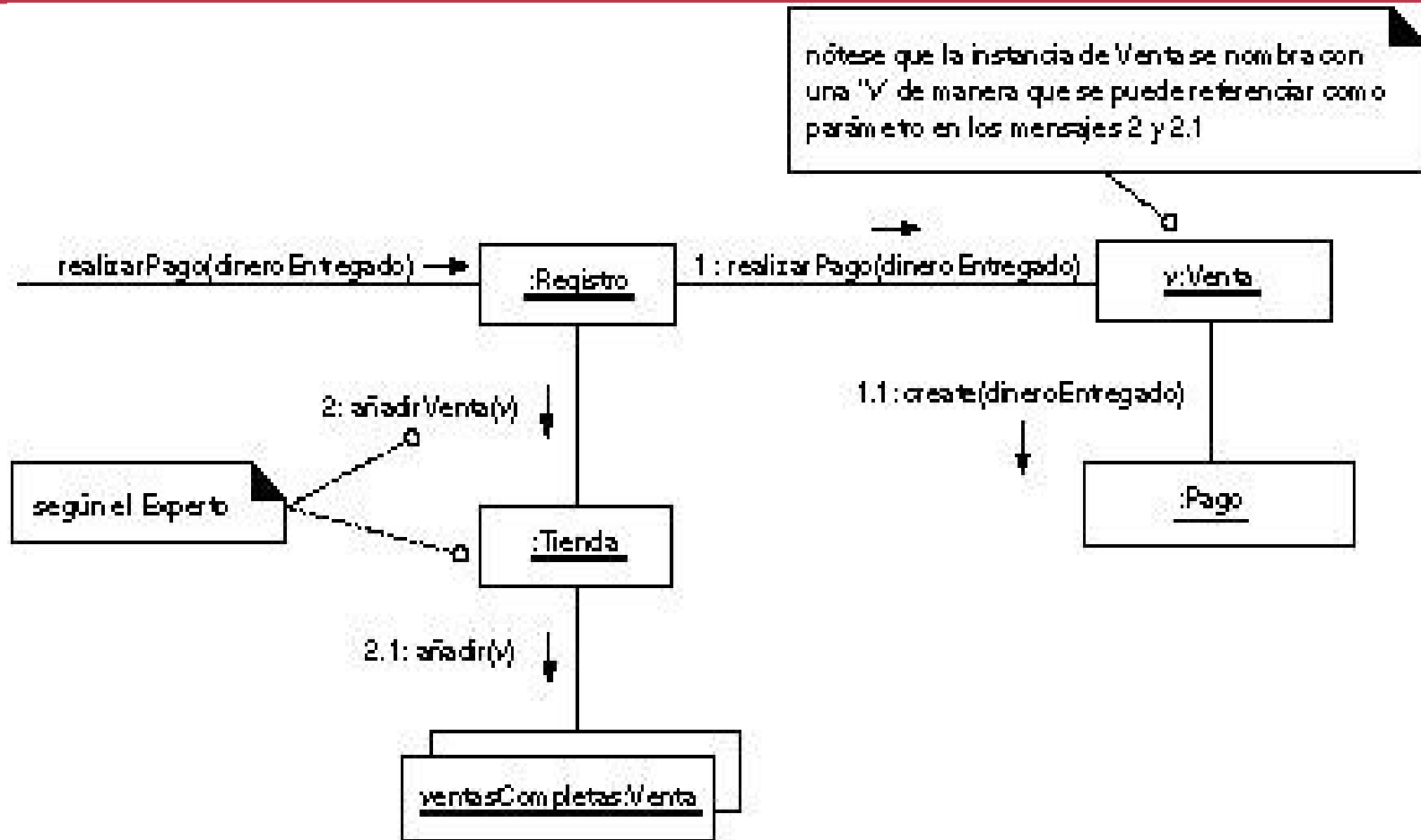
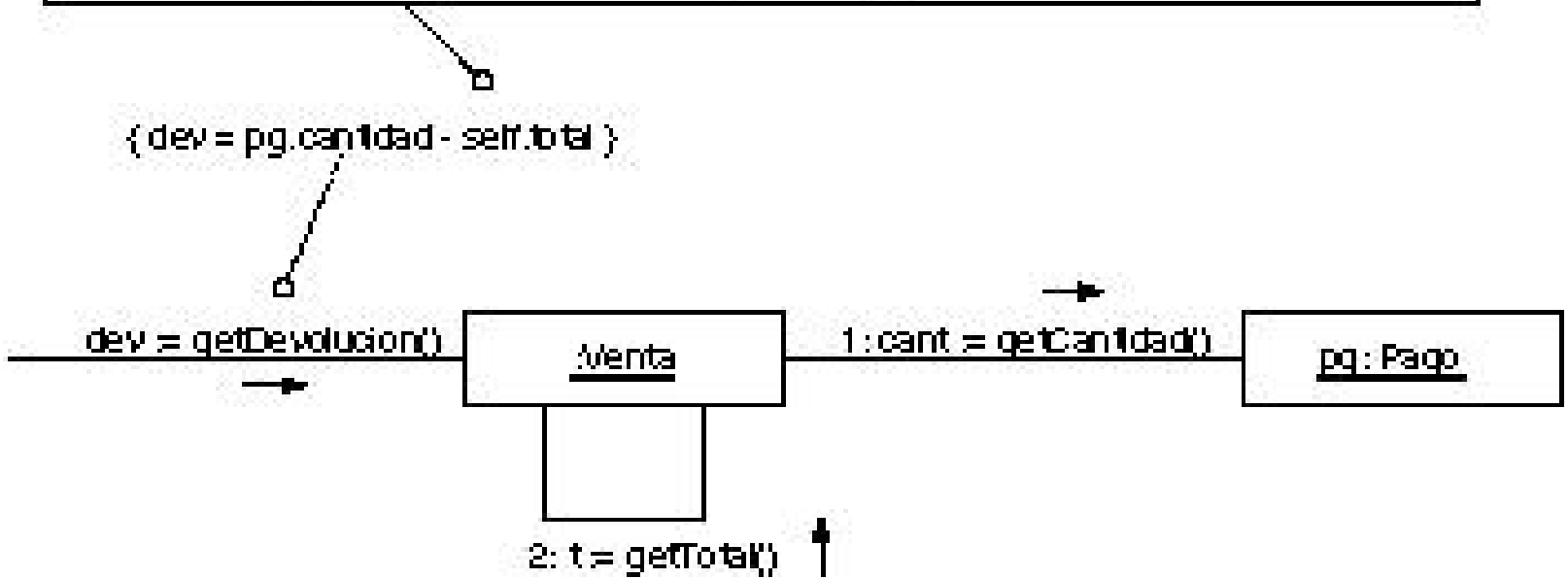


Figura 17.16. Diagrama de interacción Venta--getDevolucion.

Obsérvese el uso de "self" en la restricción. El lenguaje formal OCL utiliza la variable especial "self" como el "this" (en Java y C++). "self" en esta restricción implica la instancia de la Venta.

Aunque no se está utilizando el OCL oficial, este estilo se asemeja.

Una restricción puede expresarse en cualquier lenguaje formal o informal.



Experto (en información)
Bajo Acoplamiento

Figura 17.17. Creación del objeto del dominio inicial y los objetos siguientes.



Figura 17.18. Conexión de las capas de UI y del dominio.

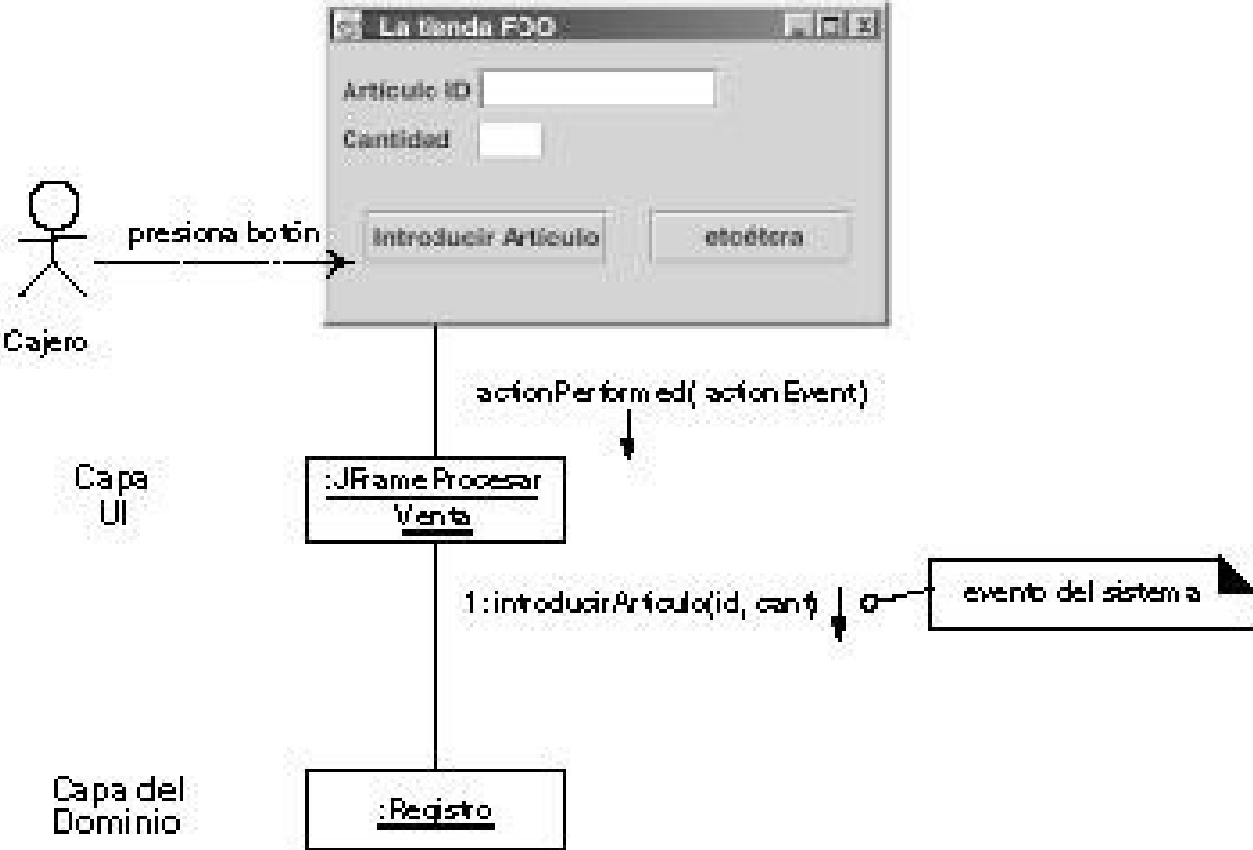


Figura 17.19. Conexión de las capas de la UI y del dominio.

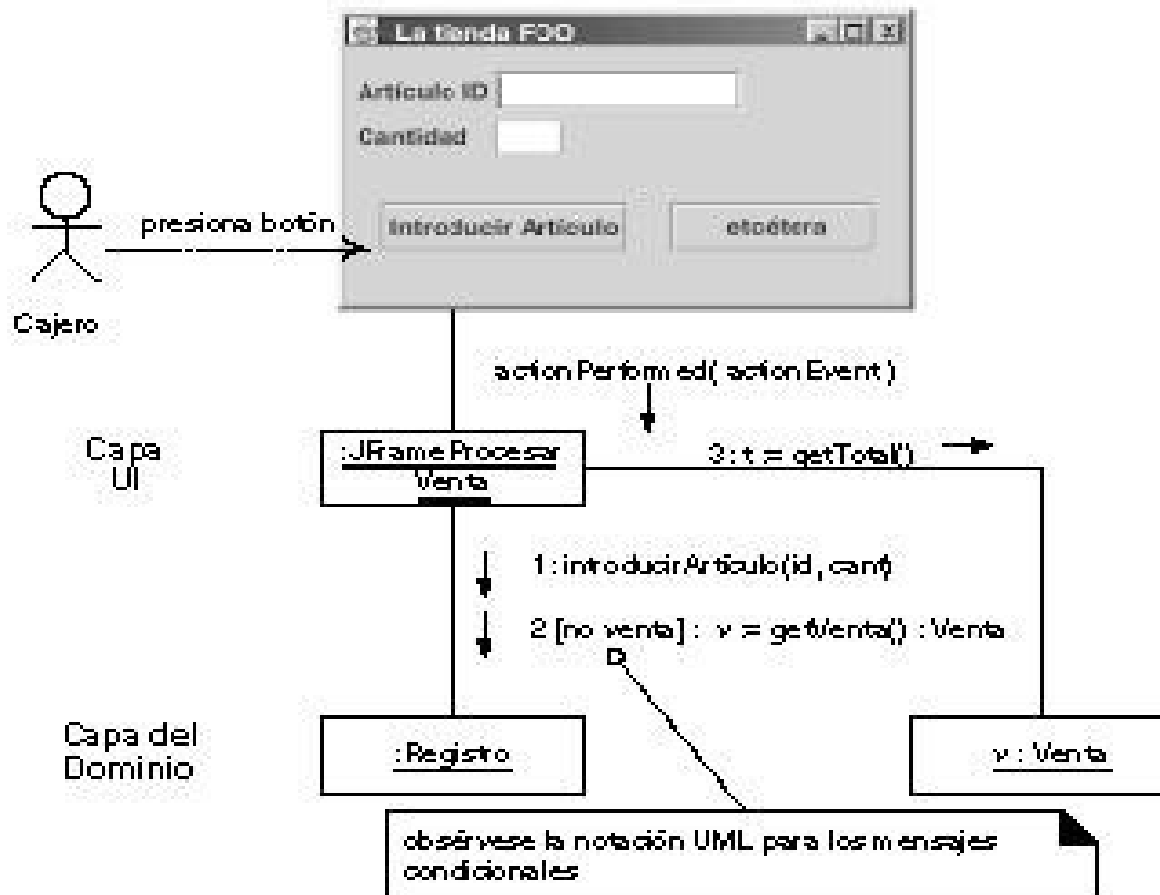


Figura 17.20. Muestra de la influencia entre los artefactos UP.

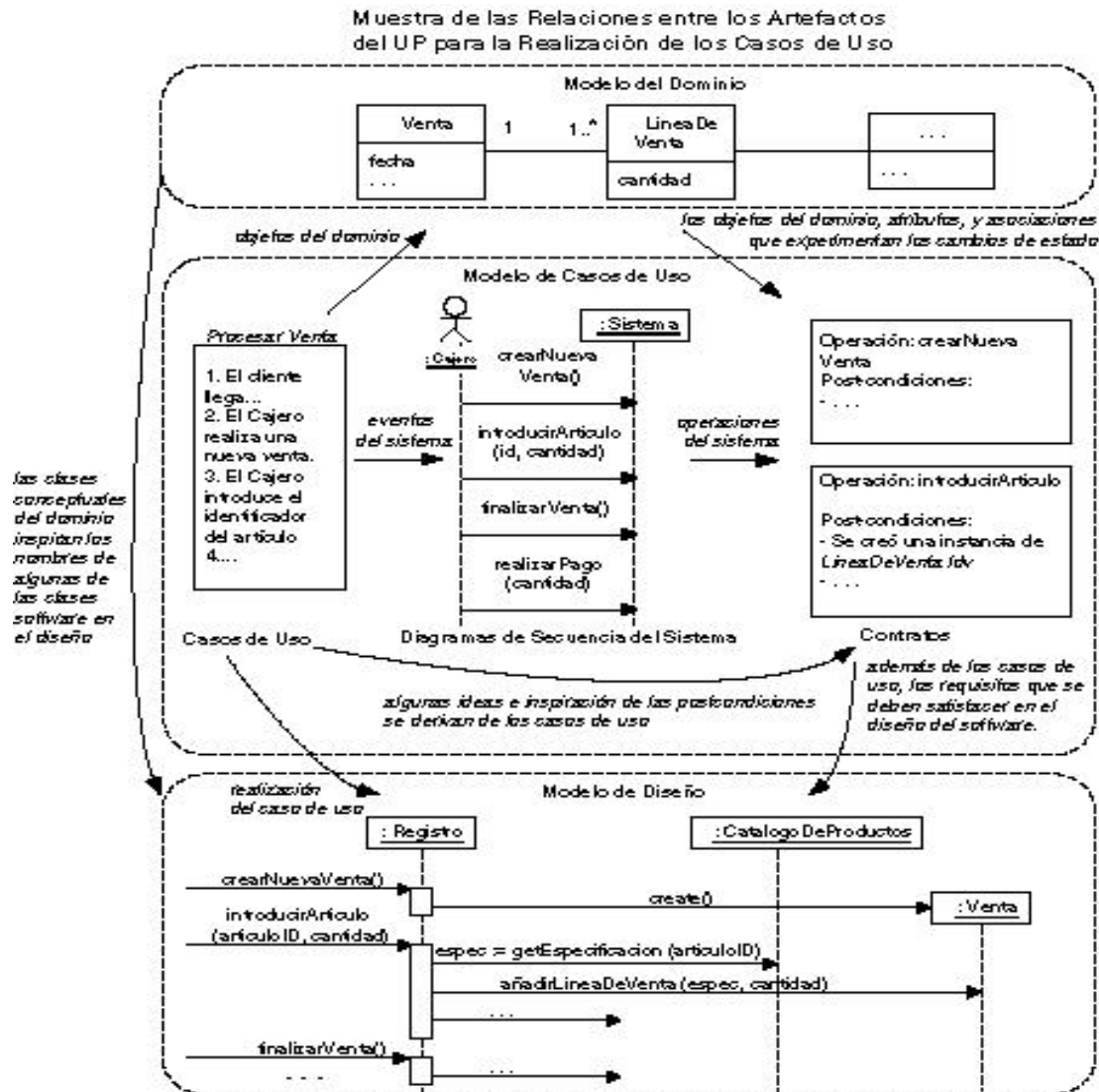


Figura 17.21. Proceso y establecimiento del contexto.

