Un multicine tiene varias salas de cine; en cada sala se exhibe una película en varias funciones a lo largo del día. Una persona que quiere comprar una entrada para una película determinada en una función determinada va a la boletería del multicine; allí, le venden la entrada sólo si la sala correspondiente no está llena para esa función y la función aún no empieza.

Cuando una persona, con una entrada, llega a la portería del multicine, se le permite entrar sólo si la entrada es para una función que aún no empieza, y si tiene la edad suficiente según la calificación de la película, que puede ser "todo espectador", "mayores de 14", o "mayores de 18".

Cuando una persona, con una entrada, llega a la puerta de una sala del multicine, se le permite entrar sólo si la entrada corresponde a la película que se exhibe en esa sala y a la función que va a empezar, y si la función anterior ya terminó.

Para este proceso, dibuja un **modelo de dominio** (usando la notación del diagrama de clases de UML) que ilustre las principales clases y relaciones (asociaciones, generalizaciones) entre clases que se desprenden del enunciado. En las clases, incluye atributos; y en las asociaciones, nombres y multiplicidades. Procura aplicar las pautas estudiadas en clases para dibujar modelos de dominio, aunque no es necesario que los mencione explícitamente.

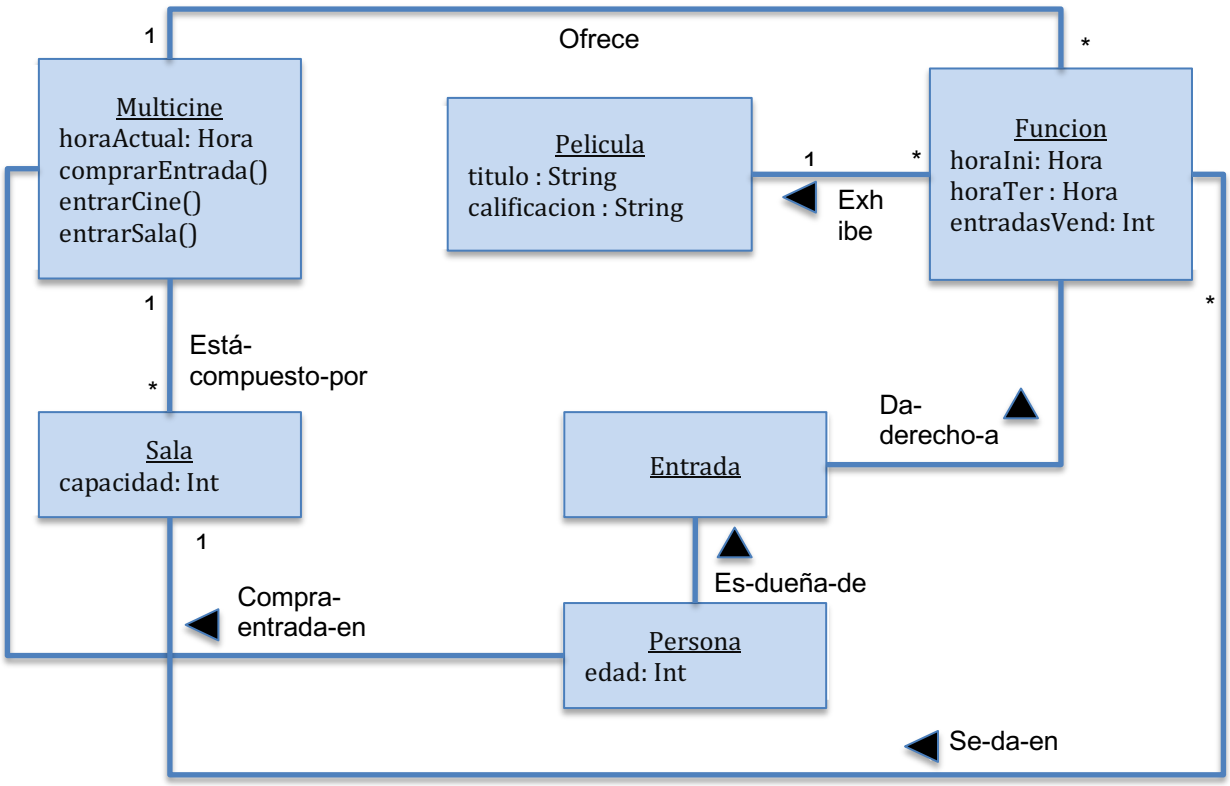
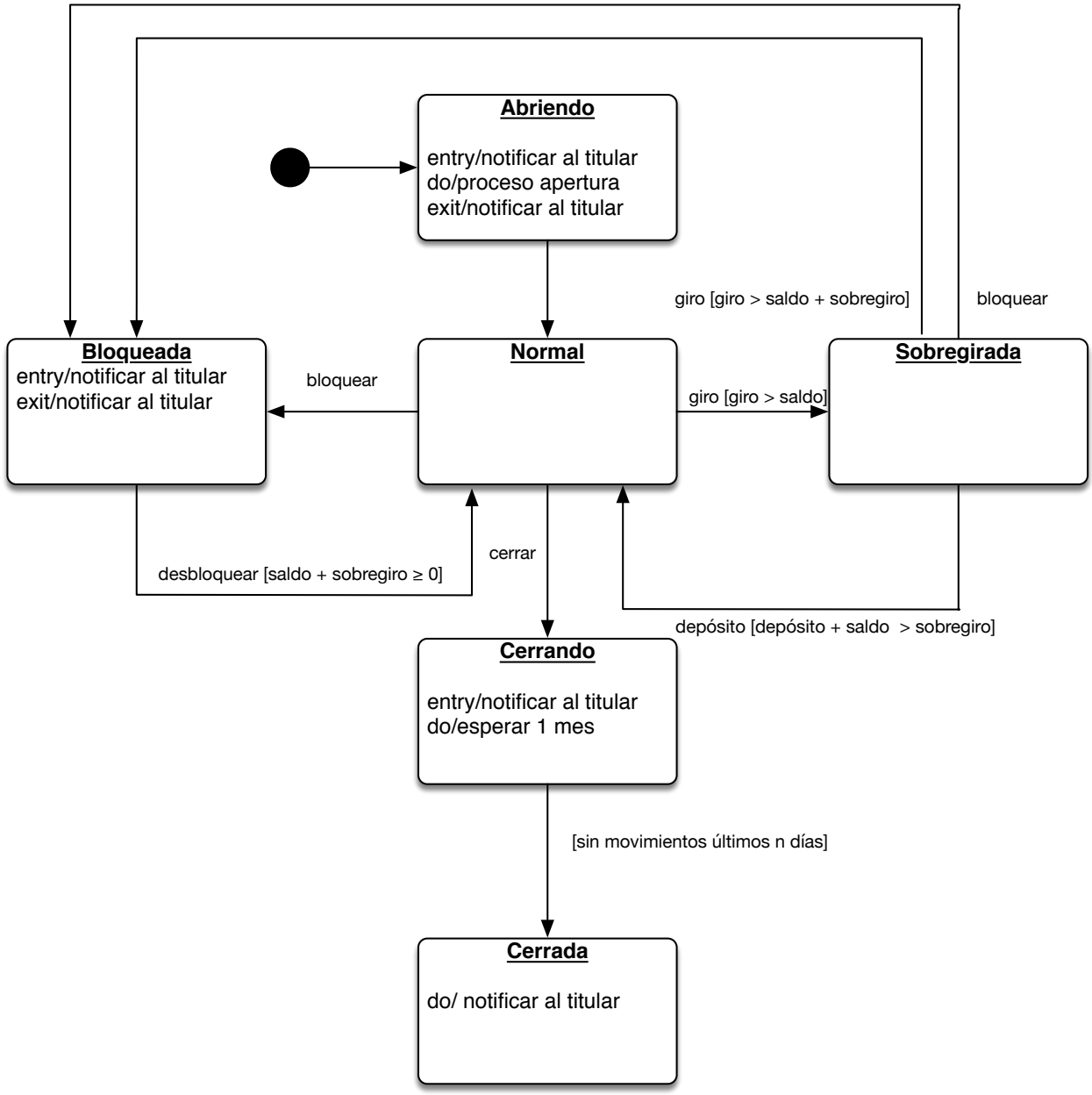


Diagrama de Estados

Deben aparecer al menos los siguientes estados



No es muy importante el que hayan puesto correctamente las actividades do o entry al interior de los estados
Es importante que aparezcan los guardias y los eventos que ocasionan la transición

Planificación

a) Primero ordenamos la tabla por prioridad y asignamos los story points (1 al de complejidad baja, 2 al mediano y 4 al alto)

Relato	Story Points	Prioridad
US03	1	1
US01	2	2
US10	1	3
US05	4	4
US04	2	5
US06	1	6
US09	4	7
US02	2	8
US08	2	9
US07	4	10
US12	2	11
US11	2	12
US16	1	13
US14	2	14
US15	2	15
US13	4	16
US17	2	17
US18	2	18
US20	4	19
US19	4	20

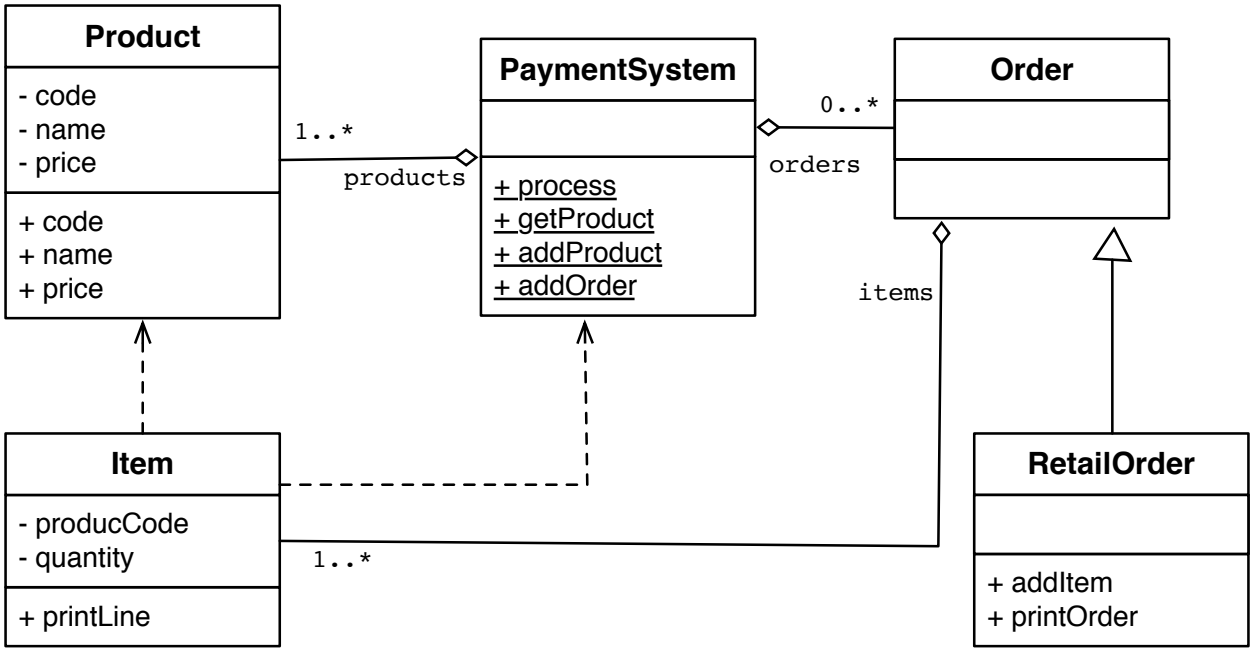
De acuerdo al primer sprint, la productividad del equipo es de 8 SPs por sprint (1+2+1+4) Es decir solo 4 SPs por semana. Usando el valor de 8 SPs por sprint tendríamos
Sprint 1 (terminado) US03, US01, US10, US05 - 8 SP
Sprint 2 US04, US06, US09, US02 - 9 SP
Sprint 3 US07, US12 - 8 SP
Sprint 4 US11, US16, US14, US15 - 7 SP
Sprint 5 US13, US17, US18 - 8 SP
Sprint 6 US20, US19 - 8 SP

a) Si es posible negociar el calendario, dado que el proyecto requiere de 6 sprints (12 semanas) y ya han pasado las dos primeras, se necesitarían 10 semanas.
Entonces una negociación por calendario implicaría lograr una ampliación del plazo de entrega de 4 semanas adicionales.
La opción de negociar por alcance implica comprometerse a entregar a tiempo (las 8 semanas pactadas). Esto significa que solo podemos hacer 4 sprints en total dejando para un release futuro los relatos de baja prioridad US13, US17, US18, US20, y US19.
Generalmente la estrategia por alcance es mejor recibida por el usuario si es que las prioridades fueron bien asignadas.

b) Dada la información disponible, y suponiendo ajuste por calendario tendríamos:
tamaño = 48 sp (si alguien lo pone en relatos (20) tambien es aceptable)
duración = 10 semanas
esfuerzo = 5 personas x 10 semanas = 50 semanas hombre

Diagramas de Clases y Secuencia

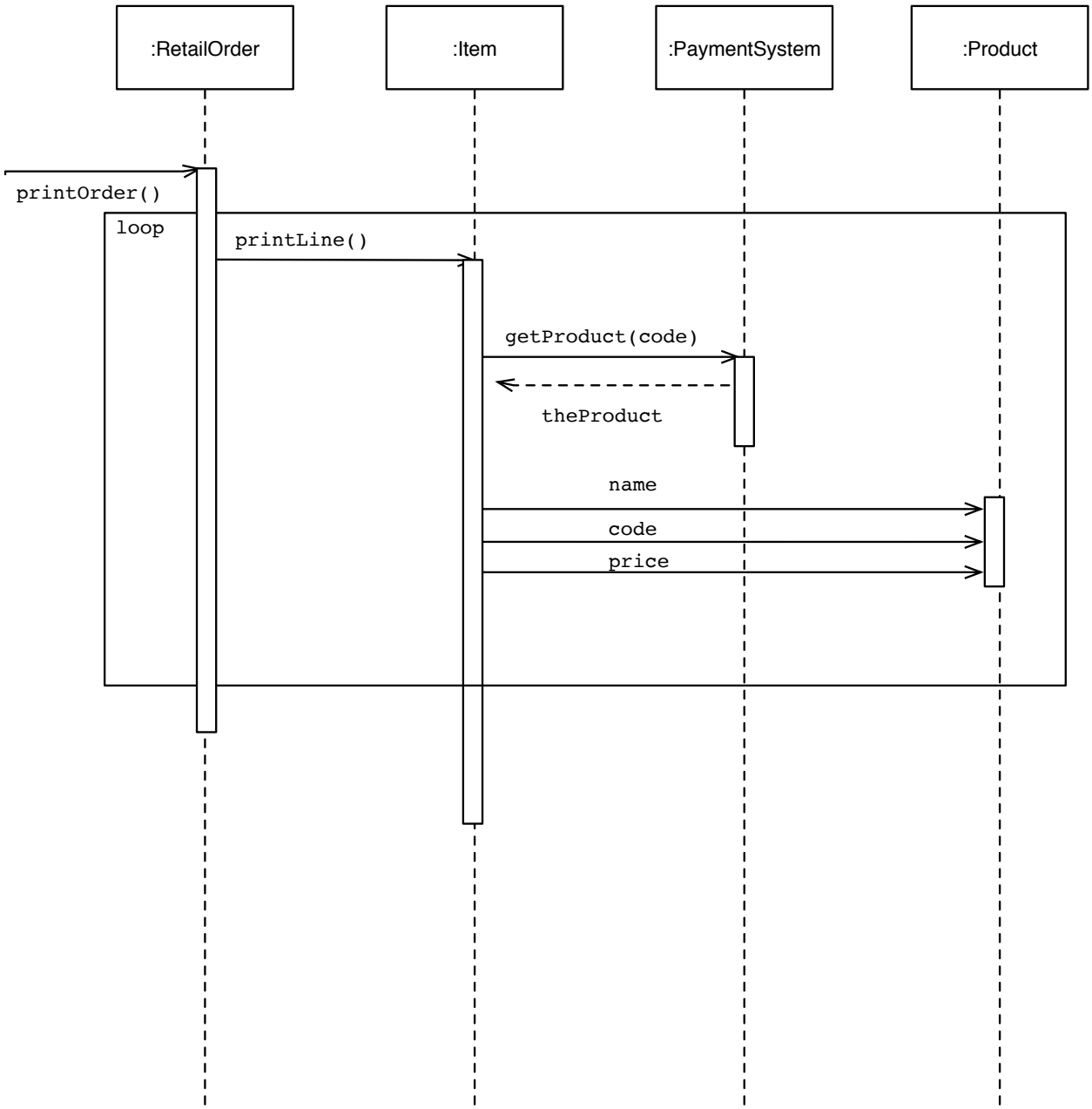
a) Diagrama de Clases



Observaciones:

- Product tiene métodos llamados de igual forma que sus atributos gracias a la declaración attr reader
- Aunque la clase PaymentSystem tiene como atributos la lista de ordenes y la lista de productos ellos se dibujan como asociaciones de composición
- La clase Order mantiene la lista de Items pero se representa como una asociación (igual al caso anterior)
- Hay flechas de dependencia entre Item y Product y entre Item y PaymentSystem porque el método printLine depende de ambos
- Finalmente RetailOrder es una especialización de una Order

b) Diagrama de secuencia para RetailOrder recibiendo mensaje printOrder



Observaciones:

- No es importante si no sabía como se pone un loop repetitivo en el UML (el rectángulo)
- Es importante que figure la obtención del producto desde el item pasando el código al PaymenSystem
- El producto figura con una flecha de regreso
- Debe aparecer el acceso a las características del producto por medio de sus métodos de acceso