Programación 4

EXAMEN FEBRERO 2009

Por favor siga las siguientes indicaciones:

- Escriba con lápiz.
- Escriba las hojas de un solo lado.
- Escriba su nombre y número de documento en todas las hojas que entregue.
- Numere las hojas e indique el total de hojas en la primera de ellas.
- Recuerde entregar su número de examen junto al examen.

Problema 1 (35 puntos)

a) Definir evento del sistema y operación del sistema.

b) Considere el modelo de dominio de la Figura 1, realizado durante la etapa de análisis de un sistema de gestión de pedidos.

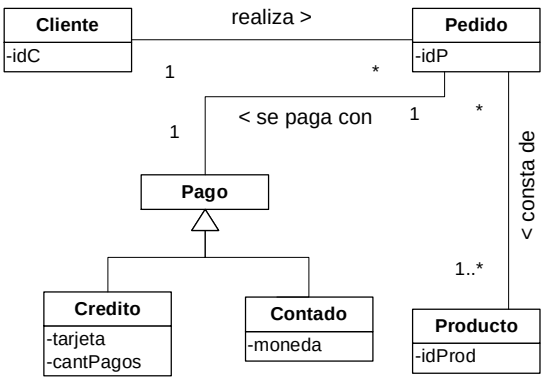

Figura 1

Se dispone además de la siguiente descripción del caso de uso Ingresar Pedido, modelado con el Diagrama de Secuencia del Sistema de la Figura 2:

El cliente se identifica ingresando su código y obtiene la lista de productos del sistema (dados por su código). El cliente agrega los productos (de a uno por vez) al pedido mediante su respectivo código. Una vez que ha agregado todos los productos deseados al pedido, lo puede cancelar o confirmar. En este último caso deberá ingresar además la información referente al pago, consistente en el nombre de tarjeta y cantidad de pagos en caso de ser a crédito, o tipo de moneda si es al contado. El sistema genera y asigna un código de forma automática para el pedido ingresado.

Se pide:

i. Escriba las pre y post condiciones de los contratos de todas las operaciones del sistema del diagrama de la Figura 2.
ii. Realice los Diagramas de Comunicación de las operaciones del sistema del diagrama de la Figura 2.

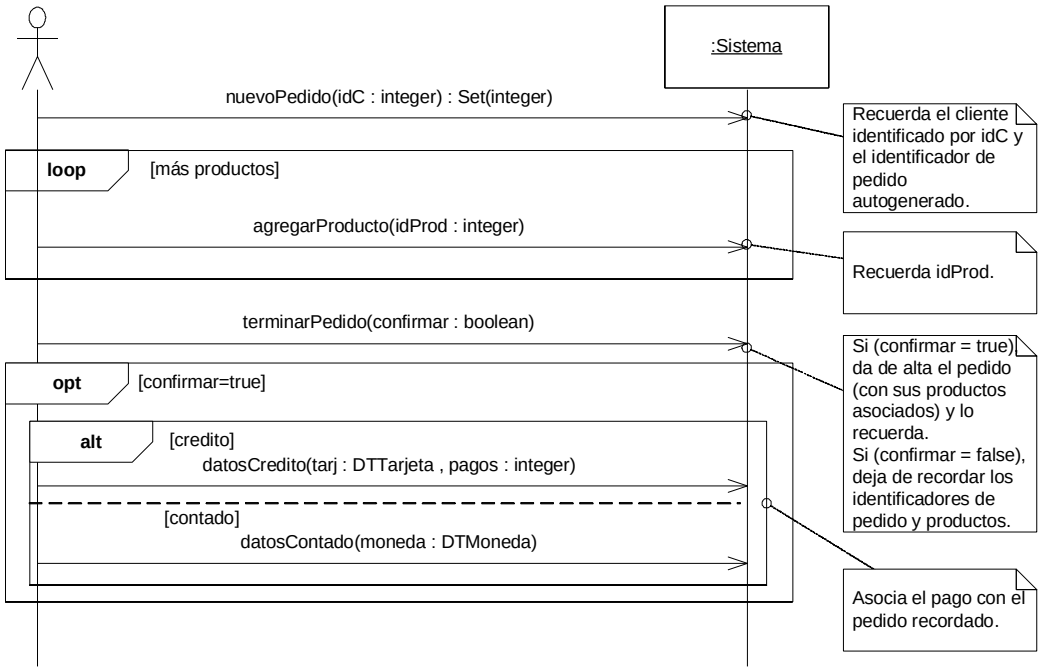

Figura 2

Problema 2 (30 puntos)

a) Mencione y describa muy brevemente las dos formas (técnicas) de identificación de conceptos candidatos para la construcción de un Modelo de Dominio vistas en el curso.

b) Su empresa (dedicada al desarrollo de aplicaciones para dispositivos móviles) le ha solicitado realizar el análisis de su próximo producto denominado ***iSoko***. Éste es un juego que consiste en que el jugador recorre pasillos (como si estuviera caminando) entre paredes y debe mover cajas que encontrará en su camino por esos pasillos (por el momento no interesa qué se debe hacer con ellas). El juego se beneficiará de la última tecnología en juegos interactivos que permite al usuario del dispositivo móvil mover dicho dispositivo y éste detecta el movimiento del usuario (utilizando un control electromecánico llamado ***accelerometer***).

El ***iSoko*** cuenta con una serie de escenarios cada uno representando un nivel de dificultad. Cada escenario (dependiendo del nivel de dificultad que éste representa) tendrá más o menos espacios para caminar, más o menos cajas y más o menos paredes, aunque siempre serán escenarios rectangulares. Cada celda de la pantalla puede contener ya sea un lugar en blanco (parte de un pasillo) o una pared. Las cajas y jugadores están ubicados únicamente en las celdas en blanco, no pudiendo compartir una misma celda. El escenario se construye conociendo, para cada celda, aquellas que se encuentran en las cuatro direcciones (arriba, abajo, izquierda y derecha) dado que serán las celdas a las que podrá moverse un jugador.

Cada juego de ***iSoko*** es jugado por dos jugadores (conectados a través de una red celular 3G) los cuales en forma independiente eligen el escenario en el cual desean jugar y el ***iSoko*** debe controlar que se trate del mismo escenario (pues de lo contrario jugarían con diferentes niveles de dificultad). Cada jugador conoce a su contrincante (ya que, por ejemplo, ambos debieron comunicarse a través de la red 3G), e incluso su ubicación dentro del escenario a cada momento. Cada juego es independiente de los otros y no existe la necesidad de guardar ninguna información sobre éstos.

Se pide: A partir de la descripción anterior, realizar el Modelo de Dominio con restricciones en lenguaje natural y OCL.

Problema 3 (35 puntos)

Se está desarrollando un sistema que controlará el funcionamiento de las bocinas y alertas asociadas al marcador electrónico de un recinto deportivo, en este caso de basketball.

El marcador electrónico cuenta con un cronómetro principal que indica en todo momento el tiempo restante del cotejo deportivo, dicho tiempo cuenta regresivamente de N segundos hasta 0, donde N se determina al inicializar el cronómetro (en este caso N debe de ser 600 para cubrir la duración de cada tiempo). Finalmente luego de inicializado y estando en funcionamiento, el cronómetro puede resetearse al tiempo con el cual fue inicializado.

Las alertas son juegos de luces que se encuentran detrás de los tableros (existen 2 tableros en este tipo de recintos) y tienen como función encenderse por 1 segundo para notificar a los jueces del partido la llegada a 0 del cronómetro antes mencionado. Finalmente existe una bocina que solo debe accionarse en el caso de que el cronometro llegue a 0.

El equipo adquirido para el recinto deportivo cuenta con una biblioteca que posee las clases que se detallan en la Figura 3, cabe destacar que dichas clases no pueden modificarse ya que son la interfase con el hardware adquirido.

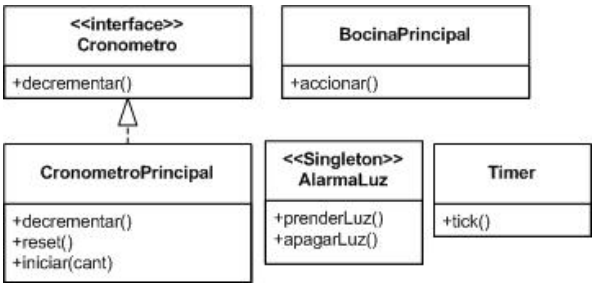

Figura 3

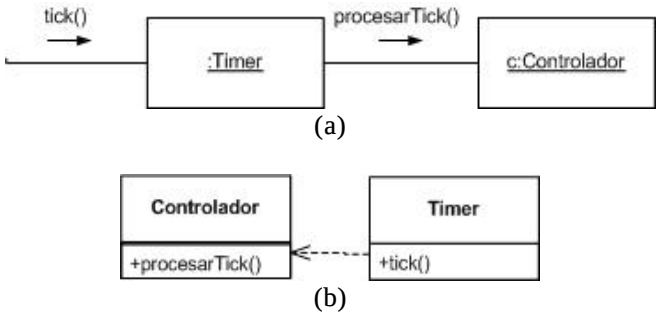

Figura 4

El funcionamiento general del sistema debe de cumplir que cuando el cronómetro llega a 0 automáticamente se accionan las luces de alarmas, así como también la bocina del estadio y se resetee el cronómetro para comenzar nuevamente con el funcionamiento. Para lograr eso el equipo de desarrollo planteó el diseño del funcionamiento de la clase Timer (Figura 4a) la cual recibe un mensaje tick() desde el hardware cada 1 segundo y un DCD (Figura 4b), ambos diseños están incompletos. La firma del método procesarTick() de la clase controlador no puede modificarse y se espera que a partir de este método se resuelva la lógica antes comentada.

El diseño debe permitir agregar en el futuro nuevos cronómetros sin modificar el comportamiento definido hasta el momento. Por ejemplo, se podría incluir cronómetros que controlen el tiempo máximo de la posición ofensiva de un equipo. La solución no debe incluir el diseño e implementación de estos cronómetros por el momento.

Se pide:

i. Completar el DCD de la Figura 4b para cumplir con los requerimientos de comportamiento solicitados, aplicando patrones de diseño.
ii. Explicar qué patrón(es) de diseño utilizó indicando (para cada patrón) las clases participantes y sus roles.
iii. Implementar en C++ el código completo de la clase Controlador, de la interfaz Cronometro y de todas las clases que se agreguen al diseño propuesto en la descripción. Asuma que posee todas las implementaciones de las colecciones necesarias en su solución.