

Auxiliar V

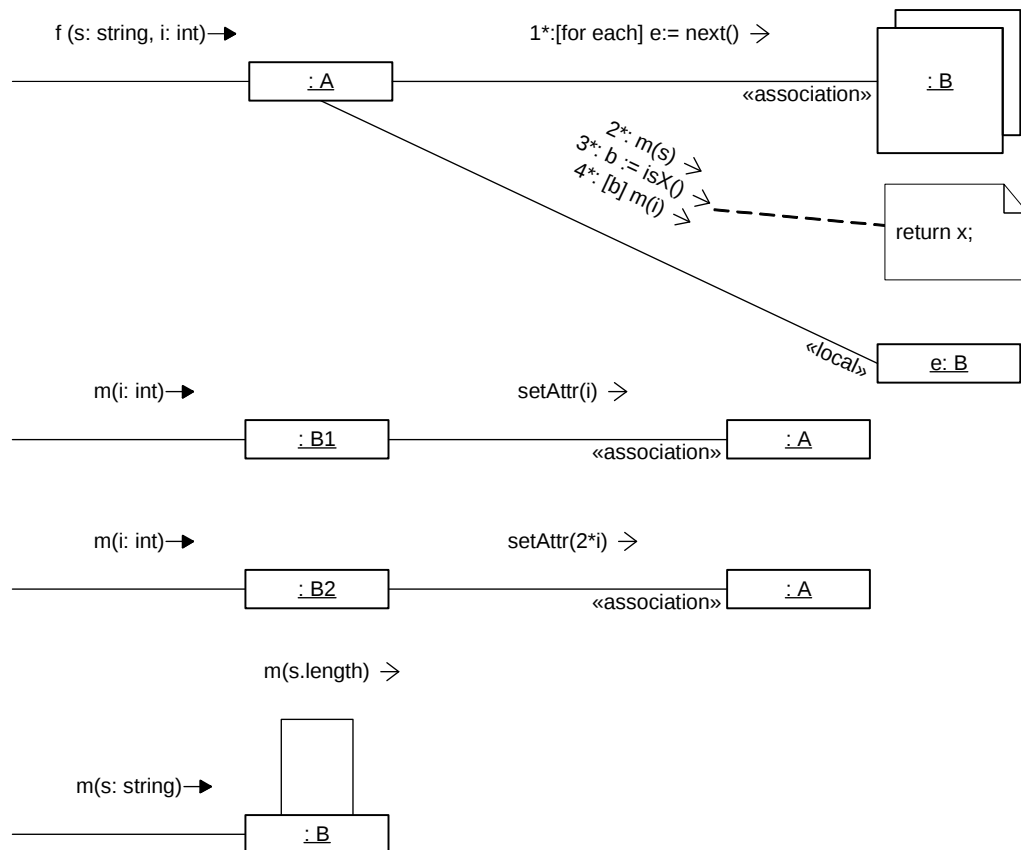
METODOLOGÍAS DE DISEÑO Y PROGRAMACIÓN CC3002 @ 2009

Ejercicio 1

Realizar la estructura que permita las interacciones diseñadas en los Ejercicios 1, 2, 3 y 4 del Auxiliar IV.

Ejercicio 2

Diseñar la estructura que permita las siguientes interacciones:



Ejercicio 3

Analizar y diseñar las siguientes estructuras de datos considerando sus operaciones habituales:

- a) Lista enlazada
- b) Lista doblemente enlazada
- c) Pila
- d) Cola
- e) Árbol binario de búsqueda
- f) Tabla de dispersión

Ejercicio 4

Considerar la realidad del Ejercicio 10 del Auxiliar II referente al sistema de información de un videoclub. Basándose en el Modelo de Dominio construido, diseñar la operación *alquilarPelícula* teniendo en cuenta la siguiente sinopsis del caso de uso:

CU-AP: Alquilar Película

El videoclub tiene como política que debe existir una reserva previa al alquiler de una película. Cuando un socio quiere alquilar una película, el sistema chequea que exista la reserva adecuada, y en caso de que la haya, se borra la reserva y se registra el alquiler. En caso de que la reserva no exista, la operación no debe tener efecto sobre el sistema.

Ejercicio 5

Se desea construir un sistema de simulación de transporte urbano colectivo. El mismo debe modelar la interacción entre buses y pasajeros. Se trabaja sobre un conjunto de paradas que representan todos los destinos posibles de los pasajeros. Una parada tiene un identificador, una descripción (por ejemplo, su ubicación) y los pasajeros esperando en la misma. De cada pasajero interesa el destino (parada) al cual se dirige y la línea que toma para llegar al mismo. Se considera un conjunto de buses donde cada uno tiene asignada una línea, cuyo recorrido se debe realizar. Una línea consiste en dos recorridos, uno de ida y otro de vuelta, y un recorrido consiste en una secuencia de paradas. Los buses tienen determinadas capacidades para pasajeros sentados y parados, las cuales se deben tener en cuenta en el momento de subir pasajeros al bus.

- a) Construir el Modelo de Dominio y presentarlo en un diagrama utilizando UML.
- b) Diseñar la operación *realizarUnRecorrido(Recorrido rec)* de la clase correspondiente a los buses que efectúa el recorrido indicado por el parámetro *rec*. La operación debe visitar las paradas del recorrido en el orden correspondiente y en cada una de las mismas efectuar la subida y bajada de pasajeros.

Ejercicio 6

En los Ejercicios 5 a 10 del Auxiliar IV se pedía el diseño de las *interacciones* (utilizando diagramas de colaboración de UML) para distintas realidades. Completar las *colaboraciones* correspondientes a cada ejercicio construyendo la *estructura* (utilizando diagramas de clases de UML).

Ejercicio 7

Se desea modelar el funcionamiento de un sencillo reloj digital de dos botones. El mismo permite visualizar la hora actual y modificarla. Presionando el botón A, se cambia a uno de los siguientes modos (en forma circular): *set hora*, *set minutos* y *normal*. El botón B incrementa (también en forma circular) el valor de la hora cuando se está en modo *set horas*, el valor de los minutos cuando se está en modo *set minutos* y no realiza ninguna acción en el modo *normal*.

Realizar análisis y diseño completos del caso descrito.

Ejercicio 8

Se quiere diseñar una colección genérica que permita:

- que la colección contenga instancias de cualquier tipo, y
- que la estructura de datos de la colección pueda ser de distintas maneras, por ejemplo como las estructuras de datos diseñadas en el Ejercicio 3.

- a) Considerando en primer lugar el tipo de colección más simple, esto es, una que sólo permite *add()*, *member()* y *remove()*, construir la colaboración.

El diseño construido debe permitir responder las siguientes preguntas:

- i. ¿Qué se le exige a una instancia para poder guardarse en la colección?
- ii. ¿De qué manera se logra la independencia de la implementación?

- b) Considerar ahora que la colección debe proporcionar un servicio de iteración sobre sus elementos. Aplicar el Design Pattern *Iterator* para resolver este problema.

- c) Considerando ahora que se requiere diseñar también la operación *find()*, que recibe una clave y devuelve el primer miembro de la colección con dicha clave, construir la colaboración.

El diseño construido debe permitir responder las siguientes preguntas:

- i. ¿De qué manera fue necesario cambiar el diseño para cumplir con los nuevos requerimientos?
- ii. ¿Qué nuevas exigencias se tienen sobre las instancias a guardar en la colección?
- iii. ¿Cómo sería posible permitir la coexistencia de ambos tipos de colección?

- d) Se quiere diseñar ahora un tipo particular de colección *Sequence*, en el que los elementos están ordenados en una secuencia. Reconsiderar su diseño para permitir este nuevo tipo de colección, considerando que se debe proveer además las operaciones *first()* y *last()*. Contestar nuevamente las preguntas anteriores para este nuevo diseño.

Ejercicio 9

Se quiere diseñar la operación $f()$ de una clase A.

- a) Se sabe que el comportamiento de $f()$ corresponde a un algoritmo complejo. Se ha identificado tres funciones básicas $o1()$, $o2()$ y $o3()$, dentro de este algoritmo, las cuales son invocadas en el orden dado. Se sabe también que A tiene un atributo x que determina dos estados posibles para A, A1 y A2, y que $f()$ se comporta de maneras distintas según A esté en el estado A1 o A2. Pero, se sabe también que el algoritmo es el mismo en ambos casos, debiendo modificarse únicamente el funcionamiento de $o1()$, $o2()$ y $o3()$.

Diseñar la operación $f()$ de la clase A para cumplir con estos requerimientos. Justificar el diseño construido.

- b) Al continuar con el diseño de la clase A se detectó que la función $o3()$ para el estado A2 es una función muy difícil de implementar eficientemente. Por este motivo se pretende diseñarla de tal manera que resulte sencillo cambiar la implementación de $o3()$ muchas veces durante el funcionamiento del sistema.

Diseñar la operación $o3()$, y justificar el diseño construido.

- c) Se desea reutilizar A en un contexto distinto del original. En el nuevo contexto se tiene una clase C que cuenta con una operación $h()$ y con un constructor recibe una instancia cuya clase realiza la interfaz I. Se sabe que el único uso que se le da a la instancia de I recibida es en el método para $h()$ en donde se invoca la operación $oper()$ definida en la interfaz I. Se sabe que la clase A y su operación $f()$ se ajusta a lo requerido por la operación $oper()$ de I.

Diseñar una solución para este problema considerando que las clases A y C **no** deben ser modificadas, y justificar el diseño construido.

Ejercicio 10

Una empresa maneja información respecto a sus empleados a los efectos de automatizar el proceso de liquidación de sueldos. De éstos, se sabe que existen varias categorías:

- gerente: se conocen el nombre, número, sueldo mensual y comisión sobre ventas.
- vendedor: se conocen el nombre, número, sueldo mensual y ventas hechas en el mes (se tiene monto y fecha de cada venta).
- jornalero: se conocen el nombre, número y el valor de la hora a pagar, además para cada día del mes se tiene un registro de las horas de entrada y salida.

Para los empleados que cobran sueldo mensual, se conoce su carga horaria mensual, actualmente fijada en 240.

Usuarios del sistema: El sistema es operado por un único tipo de usuario (operador del sistema de liquidación).

Proceso de cálculo de la liquidación mensual de sueldos: Mensualmente el operador calcula el monto a pagar a cada empleado y suma el total de sueldos a pagar de todo el conjunto de empleados. Para el gerente se suman el sueldo mensual y la comisión sobre ventas, para el vendedor se suma su sueldo y el 10% de las ventas realizadas en el mes y para los jornaleros se suman las horas registradas (en todo momento se tienen los registros del mes en curso únicamente).

Realizar el análisis y diseño completo para la realidad descrita.