Ingeniería del Software (ISW)

Evaluación Teoría. Acto 2.

16-12-2013 ETSInf-UPV

Nombre:

Cuestiones *(4 puntos)* **Tiempo: 2 horas 30 min**

1. **(***1 punto)* Indica si las siguientes afirmaciones son verdaderas o falsas y razona tu respuesta.

a) En una arquitectura de tres capas, la capa lógica contendrá una interfaz DAO por cada clase del dominio con las operaciones CRUD (Create, Read, Update, Delete) u otras que sean necesarias.

b) Cuando una clase A está relacionada con otra clase B con multiplicidad mínima 1, en el constructor de A siempre habrá que pasar un objeto que sea una instancia de B.

c) Los métodos de prueba de caja blanca y caja negra son fácilmente intercambiables y lo que se puede demostrar con un método se puede demostrar también con el otro.

d) La técnica de la partición equivalente sigue el criterio de cobertura de sentencias y de condiciones.

Ingeniería del Software (ISW)

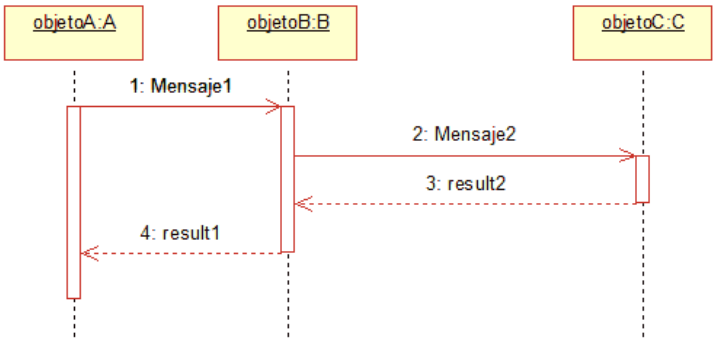
Evaluación Teoría. Acto 2.

16-12-2013 ETSInf-UPV

2. *(1 punto)* Si desarrollamos la interfaz gráfica de usuario en Java, ¿en qué se basa su modelo de eventos? ¿Para qué sirven las clases adaptadoras en el modelo de eventos de Java?

3. *(1 punto)* ¿Qué relación hay entre el modelo de casos de uso y el Objeto de Control de la capa de lógica? Razona tu respuesta.

4. *(1 punto)* Dado el siguiente fragmento de un diagrama de secuencia, ¿aporta algún tipo de información para el diseño/implementación de las clases A, B, y C? ¿Cuál?



Ingeniería del Software (ISW)

Evaluación Teoría. Acto 2.

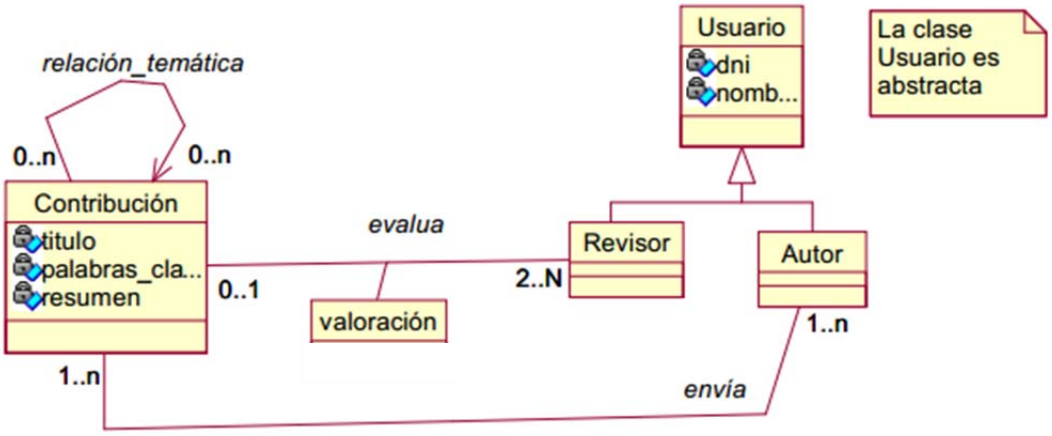
16-12-2013 ETSInf-UPV

Problemas *(6 puntos)*

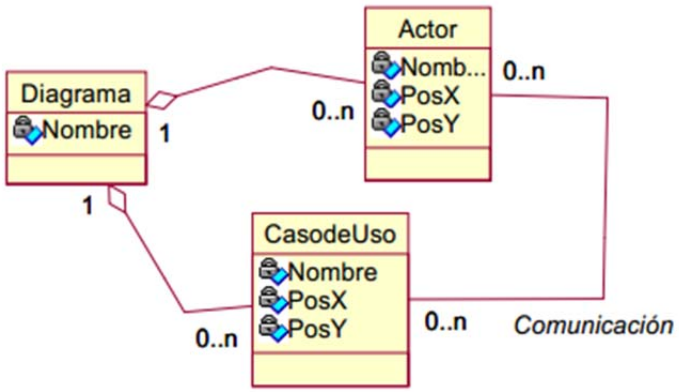
Problema 1. *(2 puntos)* ISWSoft va a desarrollar una aplicación para gestionar el envío y revisión de contribuciones científicas en un congreso. Partiendo del diagrama de clases que ha realizado el equipo de desarrollo, se pide:

a. *(1'25 puntos)* Obtener el diseño en Java, siguiendo las técnicas de diseño vistas en clase (No es necesario indicar ningún método en las clases diseñadas).

b. *(0'75 puntos)* Escribe el código en Java para invocar los constructores que consideres necesarios, de forma que el sistema quede inicializado en un estado correcto y consistente (debes crear al menos una instancia de cada clase). Puedes utilizar los valores que desees.



Problema 2. *(2 puntos)* Se desea construir un editor gráfico simplificado para definir casos de uso. La interacción con el sistema se inicia cuando el usuario indica que quiere crear un nuevo diagrama de casos de uso proporcionando el nombre del mismo. La interacción termina cuando el usuario indica que desea salir, mientras tanto el usuario puede crear tantos actores, casos de uso y relaciones de comunicación como desee. Cuando el usuario quiera crear un nuevo actor o un nuevo caso de uso debe proporcionar el nombre del mismo, el sistema comprobará que no existen con anterioridad generando en su caso un mensaje de error. Para definir una nueva relación de comunicación, el sistema muestra los actores para que el usuario pueda selecciona uno, a continuación los casos de uso del diagrama, para que el usuario seleccione uno y finalmente se crea la relación entre ambos. El editor utilizará el siguiente diagrama de clases, donde PosX, PosY son las coordenadas del centro de la figura, la relación de comunicación se dibuja de centro de actor a centro de caso de uso. Observe que el editor no admite relaciones de inclusión, extensión y herencia entre casos de uso. Tampoco herencia entre actores.



Se pide:

a. Construir el/los diagrama(s) de secuencia del editor.

Ingeniería del Software (ISW)

Evaluación Teoría. Acto 2.

16-12-2013 ETSInf-UPV

Problema 3. *(2 puntos)* El siguiente fragmento de código cuenta el número de veces que aparece un número de tipo long “num” en el fichero “myNumbers” en posiciones múltiplos enteros de “position” hasta un máximo número de ocurrencias “maxtimes”. Si se llega al máximo número de ocurrencias la función para de contar. La clase Scanner convierte el contenido de un archivo en una lista de tokens y es capaz de filtrarlos según distintos criterios. En particular, posee un método nextLong() que devuelve el siguiente token de tipo long que encuentra.

Diseñar los casos de prueba siguiendo la técnica del camino básico: dibuje el grafo de flujo, calcule la complejidad ciclomática, especifique los caminos independientes y los casos de prueba asociados a cada camino.

```
import java.io.File;
import java.io.FileNotFoundException;
import java.util.Scanner;

public int Scanning(long num, int position, int maxtimes){
      int times,index;

      times=0;
      index=0;
      try {
             Scanner sc = new Scanner(new File("myNumbers"));
             while (sc.hasNextLong()) {
             long aLong = sc.nextLong();
             if ((aLong==num) && (index % position==0)) times++;
             if (times==maxtimes) break;
             index++;
             }
              return times;

           }catch (FileNotFoundException e){
              e.printStackTrace();
              return 0;
            }
      }
```