Apellidos: **Nombre:**

Diseño de Sistemas Informáticos

4º Ingeniería Informática

Examen Junio 2007

Parte I: Cuestiones Generales

1. Usando un diagrama de secuencia, presente la colaboración típica entre los participantes de un Iterador (*Iterator*).

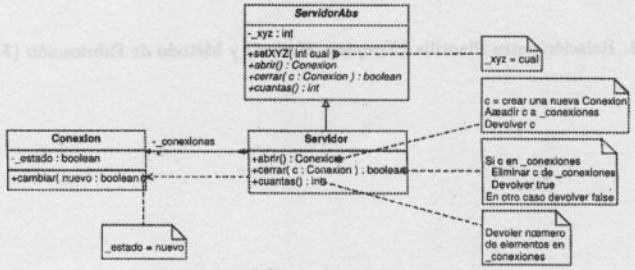
2. Usando un diagrama de secuencia, muestre las diferencias entre una Clase Adaptadora y un Objeto Adaptador en el patrón Adaptador (*Adapter*)

3. Relación entre Plantilla (*Template Method*) y Método de Fabricación (*Factory Method*)

4. Estructura del patrón Visitante (*Visitante*). Usando un diagrama de secuencia, presente la colaboración típica entre sus participantes.

5. Estructura del patrón Recuerdo (*Memento*). Comente brevemente la razón por la que se requiere una doble visibilidad en el Recuerdo

6. Implemente en Java (únicamente) los métodos adicionales necesarios para que *ServidorAbs* soporte el patrón prototipo (*Prototype*) (utilice una hoja adicional independiente)



Parte II: Diseño

Se desea construir un *Sistema de Gestión de Imágenes*, que permita organizar y servir imágenes para ser utilizadas en otras aplicaciones o directamente para ser visualizadas por usuarios a través de una interfaz de usuario. Una *Imagen* define un mapa de bits sin comprimir y una serie de propiedades como su dimensión (alto y ancho), fecha de creación o una serie de palabras clave (opcionales) para describir la imagen. Cada imagen puede, además, incoporar opcionalmente un *Pie* (un pequeño título) o un *Marco* alrededor de la imagen. Es interesante la posibilidad de poder definir un *Mosaico* a partir de otras imágenes; en este caso, se debería poder configurar de manera flexible la *Disposición* de las imágenes que componen un mosaico considerando, en principio, tres formas: una detrás de otra en forma de *Fila*, una detrás de otra en forma de *Columna*, de la forma más *Inteligente* posible dentro de unas dimensiones determinadas (aunque claramente surgirán nuevas maneras de disponer las imágenes).

Uno de los tipos de imagen que maneja el sistema es la *Imagen Plana*, que se construye directamente a partir de un color de relleno. El resto de posibles imágenes se construye a partir de imágenes comprimidas, y para ello se pretende aprovechar dos bibliotecas existentes: (a) *ImgJPEG* que define tres servicios a partir de un nombre de fichero (a.1) obtener mapa de bits (a.2) obtener alto de la imagen (a.3) obtener ancho de la imagen; y (b) *PNGLib* que define un único servicio que, partiendo de un nombre de fichero, devuelve un objeto (*PNGObject*) que encapsula mapa de bits, alto y ancho. En ambos casos se observa que leer y construir el fichero comprimido es un proceso costoso y que en algunos casos puede interesar asumir el coste de tener en memoria el mapa de bits evitando tener que leer más de una vez de fichero.

El sistema incluye diversos *Gestores de Imágenes* que conocen una serie de imágenes concretas y ofrecen un servicio de búsqueda que devuelve una réplica de todas aquellas imágenes que satisfacen un determinado *Criterio*: que contengan una determinada palabra clave, posteriores a una determinada fecha, etc. o combinaciones de estas. Es posible que un gestor de imágenes mantenga una conexión con otro gestor auxiliar que pueda ser usado para complementar la respuesta en el caso de que el gestor no localice imágenes que satisfagan el criterio del cliente. Los gestores pueden estar en distintos modos de funcionamiento: *Mantenimiento* (no pueden hacer búsquedas), *Online* (hacen búsquedas normalmente), *Saturado* (sólo devuelve la primera imagen que cumple el criterio) u *Offline* (no hacen búsquedas). Solo se puede añadir/eliminar imágenes a un gestor cuando se encuentra en modo mantenimiento. Inicialmente, un servidor se encuentra en modo offline hasta que explicitamente se cambia a modo online o mantenimiento. Estando en modo online, el gestor pasa a modo saturado si se atienden más de 10 peticiones en un segundo y no se vuelve a modo online hasta que el número de peticiones en un segundo baje de 2. Si en modo saturado la carga supera las 20 peticiones en un segundo se desconecta el gestor pasando a modo offline. Desde cualquier modo, se puede pasar explícitamente el gestor a modo offline.

Se desea construir una interfaz de usuario lo más independiente de plataforma posible. Esta debe poder conectarse a un gestor de imágenes concreto, debe permitir interrogar a dicho gestor con un criterio elegido dentro de una serie de criterios posibles y debe poder visualizar las imágenes resultantes. En todo momento, la interfaz debe mostrar el número de imágenes presentes en el gestor y este número debe actualizarse inmediatamente si el gestor incorpora o elimina imágenes. Además, si el gestor se elimina del sistema (se retira del metagestor) o se pone en modo offline, la interfaz debe desconectarse inmediatamente del gestor. Dado que la interacción con el sistema no se realizará únicamente desde esta interfaz de usuario, procure mantener la máxima independencia entre el sistema de gestión de imágenes y la interfaz de usuario.

Se pide:

- Diseño del sistema, justificando sus decisiones en base a los patrones empleados. Recuerde que el diseño no es únicamente un diagrama de clases (utilice todos los artefactos que considere necesario para describir adecuadamente la solución)