Metodología y Tecnología de Programación

Práctica Obligatoria - Curso 2010/2011

1. Enunciado

Una empresa "puntocom"necesita una aplicación para gestionar una página web con un sistema de subastas por internet.

La empresa presentará dos tipos de productos en la web: los de venta inmediata y los de subasta. Los productos de venta inmediata son ofertas que hace la empresa donde el precio de venta al público es fijo. Los productos de subasta tienen un precio de salida. Los distintos posibles compradores hacen su oferta para ese producto y al terminar el tiempo de subasta se determina el precio final del producto.

Cada producto, independientemente de si es de venta inmediata o de subasta, tiene una descripción del mismo. No se descarta que en un futuro se puedan añadir más características a los productos.

Se pide construir un sistema que implemente los requisitos que se describen más adelante, pero de forma suficientemente reutilizable como para albergar en un futuro otros requisitos, de forma que la ampliación suponga la menor cantidad posible de cambios en el diseño e implementación del sistema original. Para ello se necesita identificar los patrones de diseño necesarios y adaptarlos al problema actual.

Cada uno de los requisitos enunciados a continuación implica la implementación de al menos un patrón de diseño cuya aplicación se deberá justificar de forma obligatoria en el fichero de decisiones de diseño (`leeme.txt`). Sin embargo, esto no excluye la posibilidad de que se pueda identificar la necesidad de implementar patrones adicionales, cuya correcta justificación y realización se valorará positivamente.

1.1. Requisito R1

Los productos se identifican con un código en el formato AA-NNNNNN, donde AA es un código que identifica si el producto es de subasta o de venta directa, y NNNNNN es un número asignado secuencialmente. Por ejemplo, el producto PS-123456 es un producto para subasta (PS) cuyo número de secuencia es 123456. Sin embargo, no se descarta que el formato de identificación de los productos cambie en un futuro para, por ejemplo,

introducir el año y el mes en el que aparece un producto. Este cambio no debería afectar en gran medida al diseño e implementación de la aplicación.

Los códigos son únicos y pueden ser considerados como tal, aunque la implementación de esta característica queda fuera del alcance de la aplicación solicitada.

1.2. Requisito R2

La forma de organizar los productos en la web es por categoría. Las categorías pueden a su vez contener otras categorías. Por ejemplo, en la categoría "Antigüedades" existen las categorías "Muebles", "Libros", "Mapas", etc.

1.3. Requisito R3

Los productos de venta inmediata no tienen que tener siempre un precio fijo. Por ejemplo, si un producto es un lote de productos, el precio total será la suma de los precios de los productos que lo componen. Sin embargo, para la semana de rebajas, el precio puede llevar aplicado un porcentaje de descuento. Estos criterios para calcular el precio pueden variar, por lo que hay que dotar a la aplicación de la posibilidad de cambiar dinámicamente la forma de calcular el precio de un producto, dependiendo de un nuevo conjunto de criterios.

1.4. Requisito R4

La forma de manejar el sistema de subastas es la siguiente:

- Los usuarios conocen el precio de salida de un producto. A partir de ese precio, cada usuario puede mostrar su interés en ese producto haciendo una puja.
- Cada vez que un usuario hace una nueva puja sobre un determinado producto, se comunica al resto de usuarios interesados en el producto el valor con el que ha pujado.
- Al terminar el tiempo de subasta, el usuario con el valor de puja más alto es el que se lleva el producto.

1.5. Requisito R5

Las tareas que podemos llevar a cabo en nuestra aplicación de gestión son:

- NEWPROD: Crear un producto, asignandole los atributos que le caracterizan y el sitio dentro de nuestra organización donde debería crearse (categoría, subcategoría).

- SELL: Sacar a la venta inmediata un producto de venta inmediata.
- AUCTION: Sacar a subasta un producto de subasta.
- BID: Pujar para un determinado producto.
- ENDAUCTION: Simula el fin del tiempo de subasta para un determinado producto.
- SET XXX: Modificar los datos de un producto. A través de la segunda palabra del comando (XXX), debe ser posible modificar la clasificación de un producto, así como cualquier dato de los que los caracterizan, incluyendo su precio (SET PRICE), su descripción (SET DESC), etc., entre otros.
- GET XXX: Consultar los datos de un producto, con el mismo formato y capacidades que la opción SET.

1.6. Requisito R6

No siempre se desea mostrar todos los productos y no siempre ordenados de la misma forma, de modo que se plantea la inclusión de todo un nuevo grupo de comandos de como pueden ser:

- `SHOWLESS`, que recibe como parámetro una cifra y muestra los productos cuyo valor de salida está por debajo de esa cantidad.
- `SHOWGROUPS`, que no recibe parámetros y muestra solo las categorías y subcategorías de productos.

Es previsible que en el futuro se desee añadir más comandos de este tipo destinados a recuperar productos en un determinado orden, o solo aquellos que cumplan una determinada condición.

2. Ejecución de la aplicación

Las tareas se realizarán desde una interfaz simple basada en comandos que se ejecutarán desde una línea de comandos. Para cada una de los posibles formatos de ejecución se creará un *target* en ant con prefijo `run` (`run1, run2, run3`, etc.).

Formato 1 de ejecución. En este formato de ejecución la aplicación no recibe parámetros y la comunicación con el usuario tiene lugar desde la línea de comandos.

```
C:\MtpJun2011-grupoXX> ant run1
CMD:> NEWPROD producto1 venta antigüedades mapas datos-restantes
      Producto PV-000001 creado satisfactoriamente:
        Descripción: producto1
        Tipo: venta
        Categoría: antigüedades
        Subcategoría: mapas
        datos-restantes
CMD:> SET PRICE producto1 300
      Asignado un precio de 300 euros al producto1
CMD:> SELL producto1
      producto1 PV-000001 a la venta
CMD:> GET PRICE producto1
      El precio del producto1 es de 300 euros
CMD:> GET PRICE producto1 semanadescuento 10
      El precio del producto1 es de 270 euros
CMD:> NEWPROD producto2 subasta antigüedades libros datos-restantes
      Proyecto PS-000001 creado satisfactoriamente:
        Descripción: producto2
        Tipo: subasta
        Categoría: antigüedades
        Subcategoría: libros
        datos-restantes
CMD:> SET PRICE producto2 200
      Asignado un precio de 200 euros al producto2
CMD:> AUCTION producto2
      producto2 PS-000001 a subasta con un precio inicial de 200 euros
CMD:> BID producto2 usuario1 210
      Usuario1 ha pujado por el producto producto2 PS-000001    con 210 euros
      Usuario1 incluido en los interesados por el producto2
      Precio actualizado del producto producto2: 210 euros
      Notificación a usuarios:
        Usuario1: Nuevo precio producto producto2 es 210 euros
CMD:> BID producto2 usuario2 215
      usuario2 ha pujado por el producto producto2 PS-000001    con 215 euros
      Usuario2 incluido en los interesados por el producto2
```

```
      Precio actualizado del producto producto2: 215 euros
      Notificación a usuarios:
        Usuario1: Nuevo precio producto producto2 es 215 euros
        Usuario2: Nuevo precio producto producto2 es 215 euros
CMD:> ENDAUCTION producto2
      producto2 PS-000001 vendido a usuario Usuario2 con un precio de 215 euros
```

Notas:

- Los nombres de los comandos son obligatorios. Sin embargo, los parámetros son ilustrativos, habiéndose de elegir éstos en función de lo solicitado en todos los requisitos obligatorios enunciados.
- En el comando GET PRICE producto1 semanadescuento 10, el número 10 indica el porcentaje de descuento que lleva el producto.

Formato 2 de ejecución. Adicionalmente al formato 1 de ejecución, el programa podrá ser llamado pasándole como argumento el nombre de un fichero de comandos de prueba, uno por línea. Siguiendo el ejemplo anterior, la ejecución sería en este caso:

```
C:\MtpJun2011-grupoXX> ant run2
```

Y el contenido del fichero de comandos será el siguiente:

```
NEWPROD producto1 venta antigüedades mapas datos-restantes
SET PRICE producto1 300
SELL producto1
GET PRICE producto1
GET PRICE producto1 semanadescuento 10
NEWPROD producto2 subasta antigüedades libros datos-restantes
SET PRICE producto2 200
AUCTION producto2
BID producto2 usuario1 210
BID producto2 usuario2 215
ENDAUCTION producto2
```

Formato 3 de ejecución. Si se añade un segundo parámetro (opcional) a la línea de comandos, éste se interpretará como el nombre del fichero de salida, donde depositar el resultado de la ejecución de todos los comandos del fichero de comandos, consistente en la

descripción de cada comando ejecutado, seguida de los mensajes resultado de su ejecución, y en líneas separadas. Por ejemplo:

```
C:\MtpJun2011-grupoXX> ant run3
```

Esta orden genera un fichero de salida con la estructura:

```
Comando 1:
Salida del comando 1

Comando 2:
Salida del comando 2

etc...
```

En cualquier caso, se pase o no este tercer parámetro en la línea de órdenes, el programa debe generar un fichero con extensión `.log` con el resultado de la ejecución, cuyo formato y contenido coincida con la descripción recién realizada.

3. Entrega

1. La fecha límite de entrega será el día del examen final de la convocatoria de junio.
2. La práctica se encontrará en su totalidad contenida en un directorio llamado **MtpJun2011-grupoXX/** que contendrá **obligatoriamente** los siguientes ficheros/subcarpetas:
   - Fichero `build.xml` – Buildfile para ant que permita:
     - compilar, con `ant compile`
     - generar la documentación javadoc, con `ant doc`
     - pasar casos de prueba junit, con `ant test`
     - ejecutar, con `ant run1`, `ant run2`,...
     - todo lo anterior, con `ant`
   - Carpeta `src` – ficheros fuente
   - Carpeta `classes` – bytecode
   - Carpeta `doc` – documentación javadoc generada
   - Carpeta `lib` – bibliotecas .jar externas que fuesen necesarios para que la práctica funcione, en caso de que se necesiten

- Carpeta `etc` – ficheros de datos (con extensión .dat o .xml) y de configuración (con extensión .conf o .props), en caso de que se necesiten. Esta carpeta debe contener uno o varios ficheros de texto que incluyan los comandos de prueba.
- Carpeta `log` – ficheros con el registro de las acciones que haya ido realizando el programa al ejecutar los comandos. El resto de información de registro generada hará uso de las facilidades que ofrece Log4J.
- Carpeta `img` – ficheros con imágenes (.gif o .jpeg exclusivamente) de los diagramas UML que muestren el diseño realizado de la práctica
- Fichero `leeme.txt` – nombres y apellidos de los componentes del grupo, las decisiones de diseño tomadas y cualquier aclaración adicional sobre la ejecución de la práctica.

3. La entrega se realizará a través de moodle.

4. La versión final de la práctica se entregará dentro de un fichero comprimido que almacene el contenido del directorio `MtpJun2011-grupoXX`.

   El código de cada práctica deberá incluir los nombre de los autores de la misma utilizando el *tag* `@author` de la documentación javadoc.

4. Consideraciones, Aclaraciones, Observaciones, ...

1. Es **condición imprescindible** para aprobar la práctica que los ficheros *.java entregados compilen correctamente, es decir, sin generar errores.

2. Es **condición imprescindible** para aprobar la práctica que los ficheros *.class entregados ejecuten correctamente, es decir, no den errores de ejecución y ofrezcan la funcionalidad requerida.

3. Es **condición imprescindible** para aprobar la práctica la realización y entrega de la documentación correctamente generada con la herramienta `javadoc`, es decir, habiéndola generado haciendo uso de las opciones necesarias para que se genere documentación de todas las clases, atributos y métodos.

4. Es **condición imprescindible** para aprobar la práctica la realización y entrega de los casos de prueba JUnit para cada una de las clases desarrolladas.

5. Es **condición imprescindible** para aprobar la práctica que ésta funcione correctamente tanto en plataforma linux como en windows, es decir, la lectura de ficheros no podrá ser dependiente del editor de texto utilizado para generar los ficheros de entrada.