Ingeniería del Software II - Enunciado de la práctica del segundo cuatrimestre – Curso 2007/2008

La empresa de juegos de azar del primer cuatrimestre ha quedado muy satisfecha con nuestro trabajo, y desea encargarnos la ampliación de los sistemas que hemos desarrollado. Por ello, nos ha encargado crear un sistema que permita realizar algunas operaciones mediante un lenguaje de comandos y una interfaz de usuario construida con web.

El lenguajes de comandos es el siguiente:

```
comando        : apostar | calcular
apostar        : ‘apostar’ combinaciónM | ‘apostar’ combinaciónM apostar
combinaciónM   : {[‘1’ | ‘X’ | ‘2’]{1,3} ‘.’}{15} | {[1..49] ‘.’}{6,11}
calcular       : ‘calcular’ combinacionS
combinaciónS   : {[‘1’ | ‘X’ | ‘2’] ‘.’}{15} | {[1..49] ‘.’}{6}
```

Mediante *apostar*, el usuario puede realizar apuestas como en el primer parcial y recibirá, en la página web, la misma información que al apostar desde un terminal del primer cuatrimestre (Id del boleto, etc.). Mediante *calcular*, el sistema responde al usuario dándole el número de boletos de quiniela o primitiva con 11, 12, 13, 14 o 15 aciertos (caso de las quinielas) o 3, 4, 5 o 6 aciertos (caso de las quinielas), en cadenas de texto de la siguiente forma:

```
15->2;14->6;13->1200;12->18071;11->45000
```

…en donde el sistema está diciendo que hay 2 acertantes de 15, 6 de 14, etc.

Como se observa en la gramática, no hay tokens como “P” o “Q” que distingan el tipo de juego al que se hace referencia, sino que éste se deduce en función de la cadena que se pase en el texto del comando.

El analizador del lenguaje debe construirse utilizando el patrón Intérprete y se probará con casos de prueba de este estilo (no sólo de este estilo, sino que estos tres casos de prueba que se ponen como ejemplo deberán ser superados):

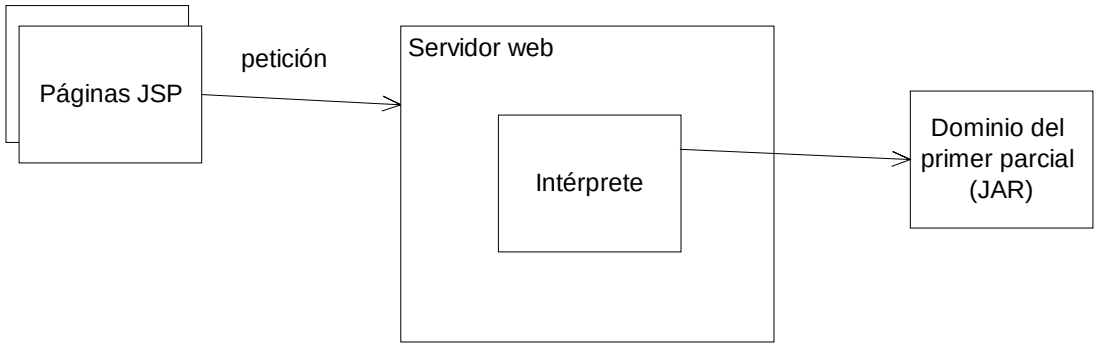
```
public void test1() {
	String textoComando=”apostar”;
	try {
		Parser p=new Parser();
		p.ejecutar(textoComando);
		fail(“Se esperaba excepción”);
	}
	catch (SintaxisIncorrectaException e) {
	}
}

public void test2() {
	String comando=”apostar 1X.1.2.1.1.1.1.1.1.1.1.1.1.1.”;
	try {
		Parser p=new Parser();
		p.ejecutar(textoComando);
		assertTrue(p.getRespuesta().equals(“Boleto aceptado”);
	}
	catch (SintaxisIncorrectaException e) {
		fail(“No se esperaba excepción”);
	}
}
```

```
// Ojo a este: concatena dos apuestas
public void test3() {
	String comando=”apostar 1X.1.2.1.1.1.1.1.1.1.1.1.1. “ +
		“apostar 5.6.7.8.9.10.”;
	try {
		Parser p=new Parser();
		p.ejecutar(textoComando);
		assertTrue(p.getRespuesta().equals(“Dos boletos aceptados”);
	}
	catch (SintaxisIncorrectaException e) {
		fail(“No se esperaba excepción”);
	}
}
```

La interfaz de usuario estará implementada mediante una página web y se utilizará tecnología JSP.

El grupo debe entregar un pequeño plan de iteraciones y comprometer unas fechas de entrega de material (como en el primer cuatrimestre).

Se propone basarse en el siguiente esquema:



El intérprete deberá tener un *Parser* que realice el análisis del texto del comando y que instancie los objetos que correspondan.

La utilización de las siguientes características serán valoradas en la evaluación de la práctica:

1) Ajax.
2) Struts.
3) Patrón Estado en el intérprete para delegar el comportamiento de la realización de apuestas y cálculos.
4) Otros patrones (su uso debe explicitarse en la documentación).