Tarea 3: Recuento en Votaciones Electrónicas Sincrónicas

Patricio Alegre D – 2623017-9

12 de abril del 2012

1. Abstract

La televisión digital emergente viene acompañada de interactividad, la cual permitiría por ejemplo, que un televidente pueda expresar su intención de voto por un candidato en el contexto de un debate político en vivo. Simular múltiples usuarios y un servidor que los atienda, es la intención de esta tarea.

Programas a desarrollar:

- acumulador
- televisor
- monitor
- testEsfuerzo

Siendo:

acumulador <puerto_votación> <puerto_monitor>

Es el servidor concurrente que acumula el número de preferencias por cada opción.

Recibe conexiones de clientes (televisores) en puerto <puerto_votación>

Por otro lado, para cada cliente que se conecta al puerto <puerto_monitor> reporta las preferencias de cada opción.

televisor <servidor_acumulador> <puerto_votación> <opción_seleccionada>

Es un cliente que establece una conexión TCP con el servidor acumulador mediante <servidor_acumulador> <puerto_votación>, y envía su preferencia ingresada en <opción _seleccionada> que se encuentra en el rango de cero a cinco.

monitor <servidor_acumulador> <puerto_monitor>

Este cliente permite consultar por el valor acumulado de las preferencias, luego muestra la estadística en la salida estándar y termina.

testEsfuerzo <servidor_acumulador> <puerto_votación> <n_clientes_simulados>

Es un simulador de clientes ó usuarios mediante hebras, cada una establece una conexión TCP con el servidor acumulador, y envía un número aleatorio entre cero y cinco que hará la parte de su preferencia ingresada.

2. Ejecución

Sintaxis:
java acumulador <puerto_votación> <puerto_monitor>

Al ejecutar el comando 'acumulador', se deben ingresar los siguientes parámetros:
<puerto_votación>
<puerto_monitor>

Donde <puerto_votación> es un puerto a abrir para permitir la conexión de los clientes y que estos ingresen su voto. Luego <puerto_monitor>, es un puerto especificado para entregar la información estadística asociada a las preferencias, que se encuentran entre cero y cinco inclusive.

Sintaxis:
java televisor <servidor_acumulador> <puerto_votación> <opción_seleccionada>

Al ejecutar el comando 'televisor', se deben ingresar los siguientes parámetros:
<servidor_acumulador>
<puerto_votación>
<opción_seleccionada>

Donde <servidor_acumulador> es el nombre de la maquina ó direccion IP del servidor que aceptará las conexiones en el puerto <puerto_votación>, y almacenara la preferencia indicada por el tercer argumento, <opción_seleccionada>.

Sintaxis:
java monitor <servidor_acumulador> <puerto_monitor>

Al ejecutar el comando 'monitor', se deben ingresar los siguientes parámetros:
<servidor_acumulador>
<puerto_monitor>

Donde <servidor_acumulador> es el nombre de la maquina ó direccion IP del servidor que aceptará las conexiones en el puerto <puerto_monitor>, y entregará la información estadística asociada a las preferencias.

Sintaxis:
java testEsfuerzo <servidor_acumulador> <puerto_votación>
<n_clientes_simulados>

Al ejecutar el comando 'testEsfuerzo', se deben ingresar los siguientes parámetros:
<servidor_acumulador> <puerto_votación> <n_clientes_simulados>

Donde <servidor_acumulador> es el nombre de la maquina que aceptará las conexiones en el puerto <puerto_votación>. El argumento <n_clientes_simulados> indica la cantidad de hebras a crear que serán las que se conectaran al <servidor_acumulador>, la preferencia de cada una será generada aleatoriamente (entre cero y cinco).

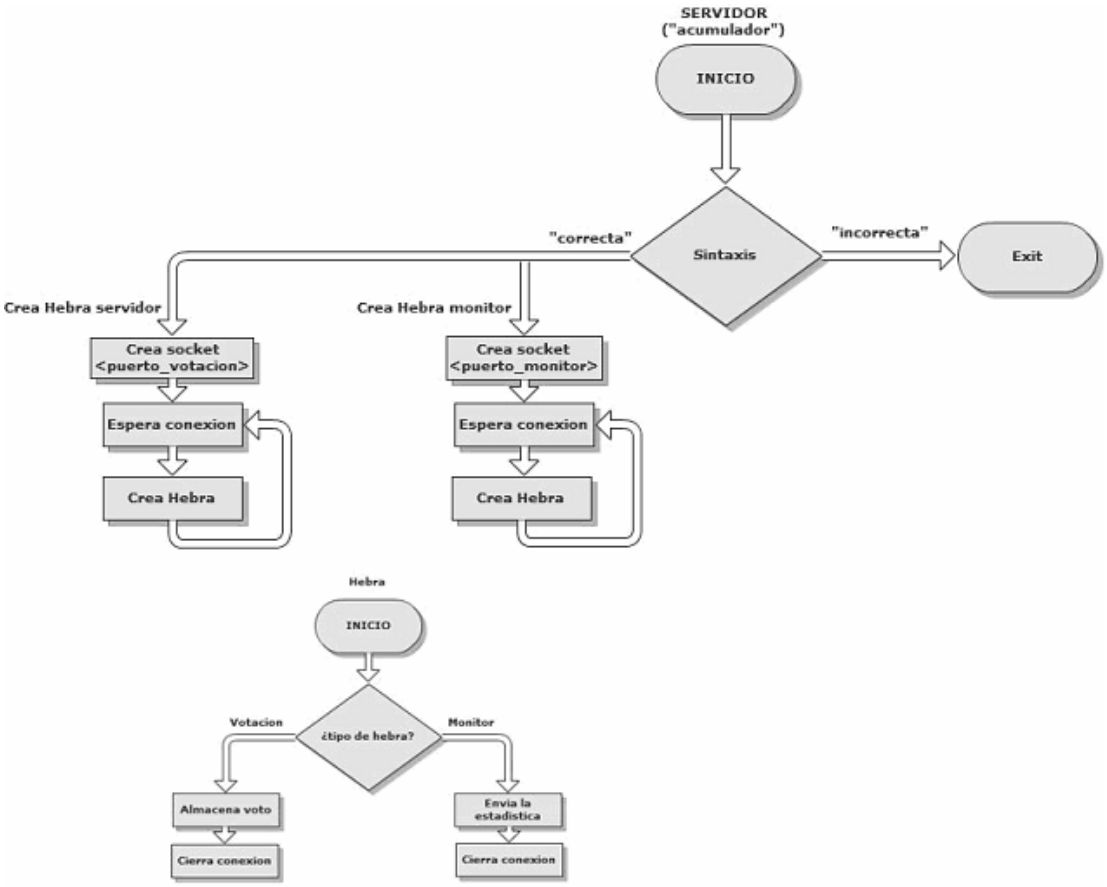
3. Funcionamiento

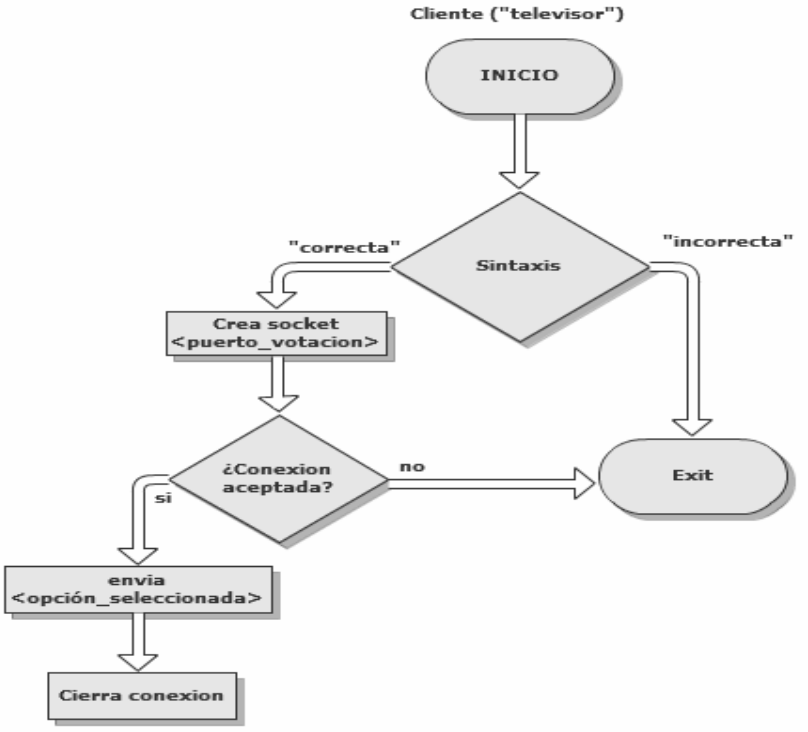
Primero que todo debemos montar el servidor “acumulador” mediante los comandos descritos anteriormente, el programa creado valida la cantidad de argumentos ingresados, que en este caso son dos (<puerto_votación> <puerto_monitor>). Una vez en funcionamiento el servidor retornara por la salida estándar su nombre y su direccion IP, a la cual podrán conectarse los clientes.

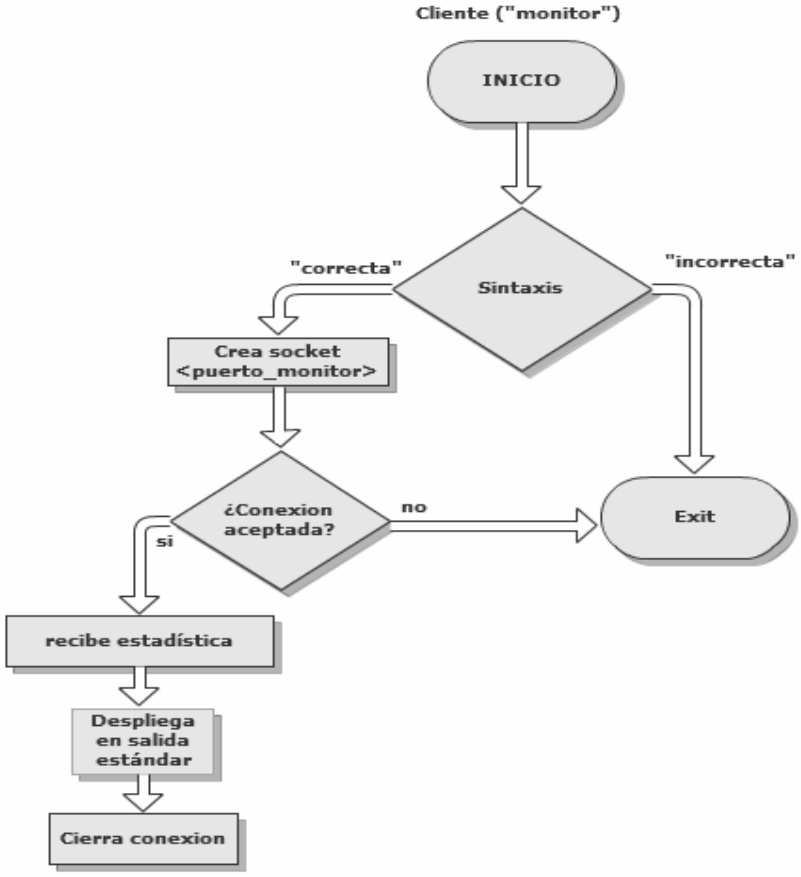
Para lograr una votación determinada, será necesario utilizar “televisor”. Debemos ingresar el parámetro <servidor_acumulador>, que fue entregado por el programa anterior, luego especificamos <puerto_votación> y finalmente nuestra preferencia. Otra forma de ingresar votos, es mediante el uso de “testEsfuerzo”, que presenta los mismos parámetros que “televisor”, con la excepción de que agrega <n_clientes_simulados>, otra salvedad es que esta vez no podremos indicar una preferencia, ya que estas serán generadas al azar.

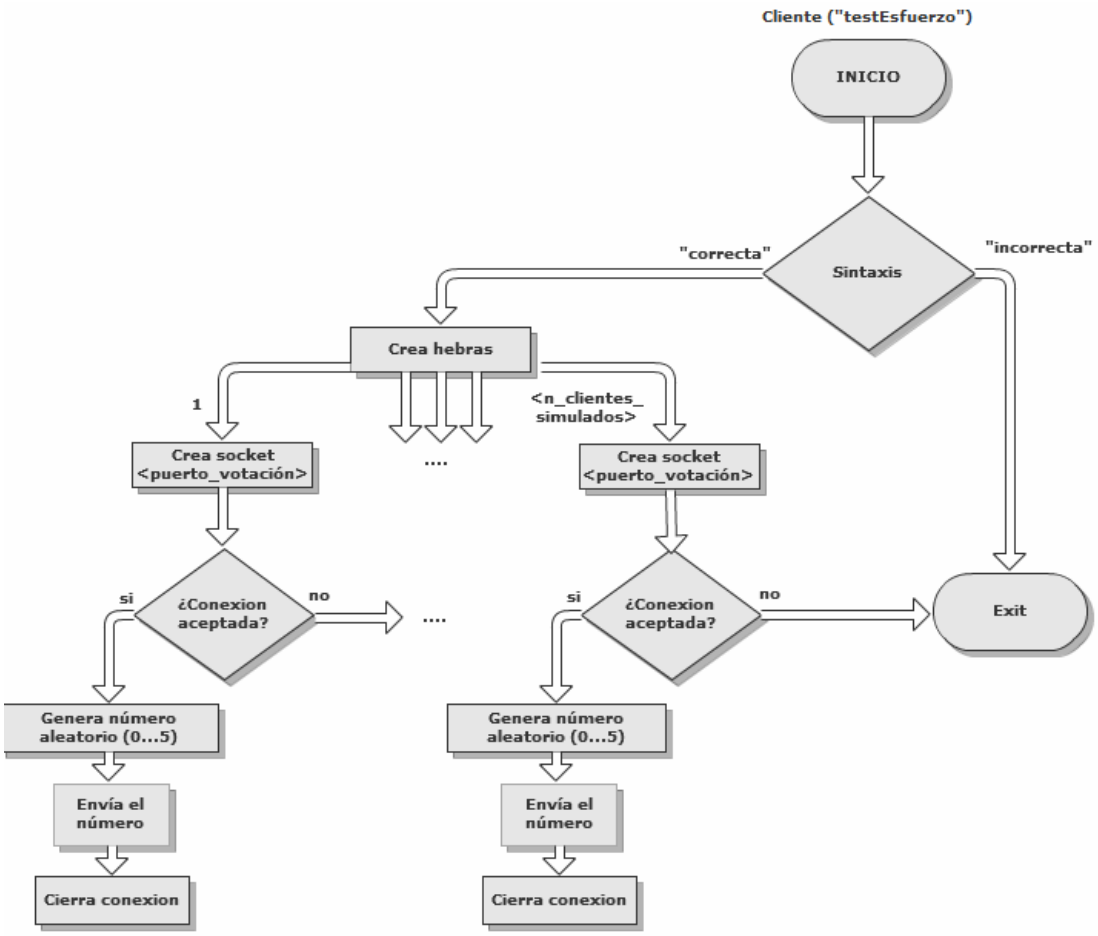
Finalmente podemos obtener la estadística de las votaciones realizadas por los usuarios a través de “monitor”, este necesita de los argumentos de entrada <servidor_acumulador> y <puerto_monitor>, que los entregará el servidor “acumulador”, la información adquirida se despliega en un formato de matriz.

4. Diagramas de flujo




Cliente ("televisor")
INICIO
Sintaxis
"correcta"
"incorrecta"
Crea socket
<puerto_votacion>
¿Conexion
aceptada?
no
Exit
si
envia
<opción_seleccionada>
Cierra conexion


Cliente ("monitor")
INICIO
Sintaxis
"correcta"
"incorrecta"
Crea socket
<puerto_monitor>
¿Conexion
aceptada?
no
Exit
si
recibe estadística
Despliega
en salida
estándar
Cierra conexion



5. Resultados

Primero debemos compilar los programas generados, mediante el comando make:

```
palegred@aragorn:~/t.sistemas/T4$ make
javac acumulador.java
javac monitor.java
javac televisor.java
javac testEsfuerzo.java
```

Ahora montamos el servidor, podemos imprimir la sintaxis correcta ingresando simplemente el comando de cualquier programa, en este caso es “acumulador”:

```
palegred@aragorn:~/t.sistemas/T4$ java acumulador
Usar: java acumulador <puerto_votacion> <puerto_monitor>
```

Por lo que ingresamos los parámetros pedidos por el servidor:

```
palegred@aragorn:~/t.sistemas/T4$ java acumulador 58888 60000
Server Running at: aragorn.elo.utfsm.cl (200.1.17.195)
Server Port Votacion: 58888
Server Port Monitor: 60000
```

Se conecta un cliente “televisor”, al servidor e indicando su preferencia (número 1):

```
palegred@aragorn:~/t.sistemas/T4$ java televisor aragorn 58888 1
Connected to TCP/Server: aragorn.elo.utfsm.cl (200.1.17.195:58888)
Enviando: 1 ...DONE
```

Luego, otro usuario le sigue, con otra opción (número 2):

```
palegred@aragorn:~/t.sistemas/T4$ java televisor aragorn 58888 2
Connected to TCP/Server: aragorn.elo.utfsm.cl (200.1.17.195:58888)
Enviando: 2 ...DONE
```

Posibles errores de sintaxis: voto fuera de rango (número 10)

```
palegred@aragorn:~/t.sistemas/T4$ java televisor 200.1.17.195 58888 10
<opcion_seleccionada> debe encontrarse entre 0 y 5
```

Verificaremos que los votos fueron ingresados en el servidor "acumulador", mediante "monitor":

```
palegred@aragorn:~/t.sistemas/T4$ java monitor aragorn 60000
Connected to TCP/Server: aragorn.elo.utfsm.cl (200.1.17.195:60000)

|       op      |       pref    |       %       |
|       0       |       0       |       0       |
|       1       |       1       |       50      |
|       2       |       1       |       50      |
|       3       |       0       |       0       |
|       4       |       0       |       0       |
|       5       |       0       |       0       |
```

Como ejemplo haremos un test de esfuerzo con 10 clientes simultáneos, gracias a "testEstfuerzo":

```
palegred@aragorn:~/t.sistemas/T4$ java testEsfuerzo aragorn 58888 10
Enviando: 1 ...DONE
Enviando: 4 ...DONE
Enviando: 5 ...DONE
Enviando: 0 ...DONE
Enviando: 4 ...DONE
Enviando: 4 ...DONE
Enviando: 0 ...DONE
Enviando: 0 ...DONE
Enviando: 4 ...DONE
Enviando: 2 ...DONE
```

Nuevamente, verificamos listando la información del servidor "acumulador"

```
palegred@aragorn:~/t.sistemas/T4$ java monitor aragorn 60000
Connected to TCP/Server: aragorn.elo.utfsm.cl (200.1.17.195:60000)

|       op      |       pref    |       %       |
|       0       |       3       |       25      |
|       1       |       2       |       16      |
|       2       |       2       |       16      |
|       3       |       0       |       0       |
|       4       |       4       |       33      |
|       5       |       1       |       8       |
```

Por lo que se concluye, que las votaciones fueron registradas con éxito.

6. Otros y las no-especificaciones de la tarea:

El formato de la estadística entregada por el servidor se desarrollo para que sea entendible a simple vista.