UNIVERSIDAD PONTIFICIA DE SALAMANCA

Campus de Madrid

Facultad de Informática

DOCTORADO EN INGENIERÍA INFORMÁTICA

Programa en Ingeniería del Software

BIENIO 2003-2005

ASIGNATURA:

Diseño Avanzado de Sistemas Operativos

TRABAJO DE INVESTIGACIÓN:

Modelado del funcionamiento del algoritmo "Bully" empleando Redes de Petri Coloreadas y el Software Renew.

PROFESOR:

Dr. D. Luis Antonio Miguel Quintales

AUTOR:

César Parejas Llanovarced

Madrid, Febrero de 2005

Especificaciones del Diseño

Se diseño el modelo con las siguientes consideraciones:

- Se han definido un conjunto de tuplas que representa a los procesos. El primer elemento de la tupla representa el número de identificación del proceso y a su vez su peso. El segundo elemento, indica si el proceso está activo (1) o inactivo (0). Y el tercer elemento representa el identificador del coordinador vigente. Ej. La tupla [1,1,6] representa al proceso identificado con el número uno, que se encuentra activo y el proceso coordinador vigente es el 6.

- Se ha inicializado el modelo con seis procesos, identificados secuencialmente del 1 al 6, todos habilitados y cuyo proceso coordinador es el sexto (por tener el mayor peso).
  Nuestra idea para inicializar estos procesos, era mediante la ejecución de un método java que generase el número deseado de proceso en función de un parámetro de entrada. Por cuestiones de tiempo, hemos concluido en esta solución alternativa.

- De igual forma, se inicializa al proceso coordinador vigente (estado en color rojo) con la tupla [6,1,6] y al primer proceso que lo verificará, con la tupla [1,1,6]. También nos planteamos inicialmente generar estas tuplas de forma automática.

- El modelo se ha diseñado con tiempos.

- El modelo provoca fallos aleatorios (deshabilitación del proceso) para el proceso coordinador. De igual forma, también habilita a los procesos Inactivos aleatoriamente.

- Nos surge un inconveniente en el momento que todos los procesos se caen, ya que al momento de habilitar a un proceso caído, necesitamos verificar si este proceso es de mayor peso que el coordinador vigente, para determinar si el nuevo proceso habilitado será o no el nuevo coordinador. Sin embargo, cuando el coordinador vigente también está caído, el bote de procesos activos (estado rojo) se encuentra totalmente vacío, con lo cual no se activa la condición de guarda "*guard x>w*", por ello, hemos añadido a esta condición la segunda regla "| *w.toString().length()==0"* con lo que tratamos de identificar si el bote coordinador está vacío, para directamente registrar el nuevo proceso habilitado, como proceso coordinador. Pero lamentablemente, está condición no responde como quisiéramos.

Referencias

- Bully Election Algorithm Workbench.
  http://cne.gmu.edu/workbenches/bully/bullyalg.html

- Material de la asignatura.

- Renew - The Reference Net Workshop
  http://www.renew.de/