Programa 1: Problema de los cinco filósofos de E. Dikjstra

Fundamentos de Sistemas Embebidos

Autor: José Mauricio Matamoros de Maria y Campos

Entrega: Lunes 4 de Mayo, 2020

> The question whether machines can think is
> itself too meaningless to deserve discussion.
> —Alan Turing

Los programas a desarrollar están enfocados a que ustedes aprendan:

a) Cuál es el problema de la cena de los filósofos.

b) La diferencia que existe entre un conjunto de procesos que se ejecutan de manera concurrente y otro cuyos procesos se ejecutan en paralelo.

c) Cuáles son los principales problemas que se presentan cuando se realizan programas en paralelo, particularmente corrupción de datos y *deadlocks*.

d) La diferencia entre semáforos y monitores.

e) Un poco más de la historia del cómputo, y que los problemas del cómputo paralelo, *multi-threading*, y *Big Data* se han estudiado desde hace más de 50 años.

Conocer los desafíos que presenta el diseño y desarrollo de programas concurrentes y paralelos, asi como las herramientas básicas disponibles para la sincronización de procesos y el acceso a datos es fundamental en el desarrollo de sistemas embebidos en plataformas con procesadores multi-núcleo, además de servir como requisito para poder comprender sistemas de sistemas embebidos distribuídos, en enjambre y, por extensión, *Internet of Things*.

Para la realización de estos programas, el alumno deberá estudiar los artículos *Communicating Sequential Processes* de Sir Charles Antony Richard Hoare, donde se familiarizará y aprenderá posibles soluciones y problemas relacionados con el «problema de la cena de los filósofos»; y *Co-operating Sequential Processes* de Edsger W. Dijkstra, donde se describe la teoría fundamental del modelado de programas de cómputo para solución de problemas en paralelo o de forma concurrente.

1. El Problema de la cena de los filósofos

Cinco filósofos se sientan alrededor de una mesa redonda para comer un plato de fideos. Cada filósofo cuenta con un tenedor a la izquierda de su plato, pero para comer los fideos necesita usar dos tenedores (cada filósofo sólo puede tomar los tenedores que están a su izquierda y derecha). Cuando un filósofo tiene hambre, éste coge primero el tenedor de la izquierda y luego el tenedor de la derecha, pero si alguno de los tenedores estuviese ocupado, se queda con el tenedor que tomó en la mano esperando a que su vecino termine de comer para poder coger el otro tenedor.

Los filósofos comerán durante un determinado tiempo y luego de comer se sentirán llenos. Un filósofo lleno no come, sino que piensa hasta que le de hambre de nuevo, entonces intentará tomar ambos tenedores y comer, pues no puede pensar con el estómago vacío. Después de un tiempo sin comer, un filósofo hambriento muere de hambre.

El problema consiste en encontrar un algoritmo que permita que los filósofos nunca se mueran de hambre.

Este problema es extensible para n filósofos. Por cada filósofo habrá un plato de fideos (de suministro infinito) y un tenedor a la izquierda de su plato, es decir, el número de tenedores es igual al número de filósofos.

2. Especificaciones del programa

Desarrolle un programa en que resuelva el problema de los cinco filósofos en dos lenguajes de programación: Python y cualquiera otro[1] entre C/C++, C#, J#, Java, VisualBasic.NET. El programa deberá cumplir con las siguientes especificaciones:

- El programa acepta 2 argumento por línea de comandos: el tiempo de ejecución en segundos y el número de filósofos.
- Si los parámetros ingresado por línea de comandos no son números o está fuera del intervalo válido, se despliega la ayuda y concluye la ejecución del programa.
- Si no se proporciona argumento por línea de comandos, el programa se ejecutará con valores predeterminados, es decir 30 segundos y 5 filósofos.
- Los filosofos piensan durante un tiempo aleatorio entre 100 y 800 milisegundos.
- Los filosofos comen durante un tiempo aleatorio entre 100 y 200 milisegundos.
- El programa debe mostrar a la salida el tiempo y cada acción que ejecuta un filósofo y por cuanto tiempo la realizará.
  - `500ms filósofo 5 come por 230ms`
  - `503ms filósofo 3 intenta comer. Toma tenedor derecho`
  - `520ms filósofo 4 piensa por 560ms`
  - `520ms filósofo 3 come por 150ms`
- El script de Python deberá estar en un sólo archivo de nombre `filosofos.py`.
- El programa en lenguaje imperativo deberá incluír archivos `Makefile` y `readme.MD` para su compilación en Linux.
- (Opcional para punto extra) El programa lee un archivo XML del mismo nombre del cual se leen los valores predeterminados y los nombres de los filósofos. Si este archivo no existiere, el programa deberá crearlo.

[1] Otros lenguajes de programación imperativos, compilados y con soporte de ejecución paralela de hilos son aceptables previa aprovación del profesor.