PROGRAMACIÓN CONCURRENTE BASADA EN HILOS POSIX

José Manuel Rodríguez García[1], Juan Carlos Díaz Martín[2], Isidro Irala Veloso[3]

Escuela Politécnica. Departamento de Informática. Universidad de Extremadura
[1]e-mail: jmrodri@unex.es
[2]e-mail: juancarl@unex.es
[3]e-mail: iirala@unex.es

Resumen: El artículo identifica dos perspectivas diferentes para abordar la asignatura de programación concurrente: la perspectiva clásica en la que se presentan los mecanismos tradicionales de programación concurrente de alto nivel y se practica con algún lenguaje didáctico que implemente dichos mecanismos; y la perspectiva basada en hilos, en la que se presentan los mecanismos de tradicionales ya citados y se practica implementando dichos mecanismos. El artículo propone un programa de la asignatura programación concurrente centrado en el segundo enfoque, basándose para ello en los hilos POSIX. Se puede trascender de este modo al uso de herramientas exclusivamente didácticas, cuya utilidad fuera del ámbito de la iniciación es prácticamente nula. Como parte práctica de la asignatura se propone la construcción de una biblioteca que implemente mecanismos de sincronización similares a los mecanismos de sincronización clásicos.

1.- INTRODUCCIÓN

La programación concurrente viene, desde hace algún tiempo, tomando cuerpo de doctrina. Aunque clásicamente se estudiaba en los curricula de

informática como parte de la asignatura Sistemas Operativos, se observa la tendencia general a desgajarse de ese tronco y crear una disciplina con contenido propio, con entidad suficiente para alcanzar en numerosos planes de estudio la categoría de asignatura troncal. Estos aspectos de programación avanzada se estudian unas veces como asignatura troncal u obligatoria del ultimo curso del primer ciclo de los estudios de informática y, en otros casos, como asignatura optativa del segundo ciclo.

Se puede enfocar su tratamiento desde puntos de vista alternativos [5]:

- Enfoque matemático, con estudio estricto de los requisitos formales que debe cumplir un programa concurrente para ser correcto. Es necesario un planteamiento riguroso de este enfoque en algún punto del curriculum, pero parece que es adecuado este enfoque en ámbitos de especialización mas que en cursos introductorios.
- Enfoque intuitivo, que permite a los alumnos un acercamiento a la concurrencia más atractivo, facilitando la comprensión de la disciplina. Este será el punto de vista que tomemos en nuestro desarrollo.

2.- LENGUAJES E HILOS

Admitido el segundo enfoque como el más adecuado para una primera aproximación a la programación concurrente, es obvio que la parte práctica de la disciplina adquiere un papel tan importante como la parte teórica. Se plantea como elemento esencial de la programación docente qué herramientas son las más adecuadas a este propósito.

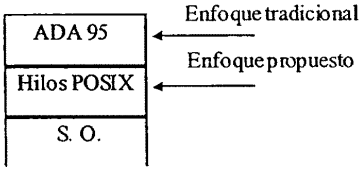
Históricamente la concurrencia en el sistema era soportada por el sistema operativo. Los sistemas operativos proporcionaban concurrencia a través de los procesos. Para poder comprender en profundidad los mecanismos de la concurrencia no quedaba otra opción que bucear en el núcleo de los sistemas operativos y utilizar las primitivas que ellos proporcionaban. Esta era, en cualquier caso, una tarea ardua y laboriosa, y en muchos casos imposible, al no disponer de la documentación pertinente ni de los programas fuente.

La necesidad de disponer de herramientas más flexibles y menos complejas llevó a muchos docentes a implementar solucciones didácticas que permitieran el acceso a la concurrencia de manera más cercana, más sencilla, más operativa. Esta opción, interesante y fructífera, ha dado pie a la

creación de herramientas tales como Pascon,, Pascal-FC, etc., cuya principal misión era y es iniciar a los estudiantes en la disciplina de la programación concurrente. El mayor inconveniente que presentan estas herramientas es que no se utilizan en entornos de desarrollo reales, lo que provoca, en muchos casos, que sean percibidas por el estudiante como meras curiosidades académicas.

Los hilos proporcionan concurrencia a nivel de proceso, con más potencia y flexibilidad que la permitida por los mecanismos de comunicación de procesos clásicos UNIX, sin necesidad de hacer uso de llamadas al sistema de una manera explícita;. Proporcionan, pues, las mismas facilidades que los lenguajes de didácticos de programación concurrente.

Las implementaciones de paquetes de hilos en forma de bibliotecas han evolucionado hasta llegar a un nivel de desarrollo elevado. La aparición del estándar POSIX Pthreads, con los consecuentes desarrollos basados en él, ha enriquecido el panorama de la programación concurrente. Bibliotecas como FSU-Pthreads o LinuxThreads constituyen implementaciones del estándar de libre distribución que han demostrado la suficiente potencia y flexibilidad para constituir la base del núcleo de ejecución de ADA 95 [10].



Asimismo, las bibliotecas pueden servir como herramientas de aprendizaje de la programación concurrente. La migración de FSU-Pthreads al entorno didáctico evolucionado MINIX ([8], [12]), responde a este enfoque. La disponibilidad del código fuente, tanto del sistema operativo como de la biblioteca, permite, si se desea y se cuenta con el tiempo necesario, un estudio exhaustivo de la concurrencia a bajo nivel.

3.- LA PROGRAMACION CONCURRENTE EN LA UNIVERSIDAD ESPAÑOLA

Al objeto de conocer el estado de la cuestión, nos propusimos realizar un muestreo informal sobre los planes de estudios y los programas de la asignatura en alguna de las mas representativas universidades españolas.

Como resultado del mismo hemos obtenido las siguientes conclusiones:

- En la mayoría de las universidades hay algún tipo de asignatura de programación concurrente en alguno de los tres ciclos.
- Hay dos tendencias diferenciadas:
  1 Estructurar los estudios de esta materia como una asignatura troncal u obligatoria en el tercer curso del primer ciclo.
  2 Estructurar los estudios de esta materia como una asignatura optativa de segundo ciclo.
- En algunos casos hay materias de tercer ciclo con contenidos de programación concurrente avanzada, orientación a objeto, estudio de modelos concretos, etc.
- En los casos en los que se indicaban los contenidos de la parte práctica, se incluyen prácticas con Pascal-FC; en algunos casos se incluían asimismo prácticas con PVM.
- Existen varios casos de asignaturas en que se asocia la programación concurrente bien con el "paralelismo" o bien con la "distribución".

Los programas responden en la mayoría de los casos a una estructura típica en la que aparecen temas introductorios, temas de estudio de los mecanismos clásicos de sincronización y de los diferentes paradigmas existentes, y análisis de algún lenguaje de programación concurrente con soporte de concurrencia, habitualmente lenguajes didácticos.

4.- UNA PROPUESTA DE ASIGNATURA

A la vista de las consideraciones anteriores, proponemos una asignatura de programación concurrente basada en hilos POSIX, cuya carga docente sea tres créditos teóricos y tres prácticos. Veamos por separado los cuatro aspectos básicos de todo programa: objetivos, contenidos, criterios de evaluación y bibliografía.

a) Objetivos

Los objetivos que se pretende alcanzar con la asignatura son:

- Introducir al alumno en el concepto de concurrencia y sus ventajas sobre la programación secuencial.
- Identificar los problemas inherentes de la programación concurrente: condiciones de carrera, precedencias, ...

- Distinguir entre concurrencia a nivel de sistema operativo y a nivel de proceso de usuario.
- Analizar los diferentes paradigmas de sincronización: cita Ada, monitores, variables de condición.
- Presentar al alumno los mecanismos tradicionales de sincronización.
- Proporcionar al alumno los conceptos básicos sobre hilos.
- Presentar el interfaz de hilos POSIX 1003.1c
- Aprender a construir programas multihilo POSIX sobre un sistema operativo: Linux.
- Presentar dos opciones diferentes de implementación de hilos sobre Linux: LinuxThreads, FSU-Pthreads.
- Descubrir al alumno las técnicas de implementación de una biblioteca de hilos POSIX mediante un caso de estudio: FSU-Pthreads.

b) Contenidos

Los contenidos teóricos de la asignatura son los que a continuación se exponen:

- Introducción a la programación concurrente. Motivación de la programación concurrente.
- El concepto de proceso y el concepto de hilo en el contexto POSIX. Similitudes y diferencias.
- Implementación del concepto de hilo en el proceso clásico UNIX: saltos largos, descriptor de hilo, pila de hilo.
- El problema de la exclusión mutua. Soluciones planteadas.
- Los mecanismos de sincronización tradicionales: Las regiones críticas, las regiones críticas condicionales, los monitores, los mecanismos de paso de mensaje.
- El estándar POSIX 1003.1c (Pthreads). Gestión, sincronización y aspectos avanzados.
- Implementaciones de hilos. LinuxThreads y FSU-Pthreads.

En la parte práctica, se trabajará con las herramientas básicas de sincronización Ptrehas a través de las bibliotecas de hilos LinuxThreads y FSU-Pthreads. El objetivo final de las prácticas es la construcción de una biblioteca de funciones C sobre Pthreads que implemente mecanismos de sincronización de mayor nivel que los proporcionados por los hilos POSIX, similares a las que ofrecen los lenguajes concurrentes didácticos.

c) Criterios de evaluación

La evaluación debe contemplar las dos partes en que se divide la asignatura. Por lo que respecta a la parte teórica, un examen escrito sobre los conceptos abordados a lo largo de la asignatura parece un modo adecuado de valorar hasta qué punto los alumnos han asimilado los conceptos.

Por lo que respecta a la parte práctica, el examen consistirá en construir un programa que responda a un enunciado propuesto utilizando para ello las bibliotecas que el alumno ha ido construyendo en las clases prácticas de la asignatura. De este modo se valora no sólo el esfuerzo del examen, sino además el esfuerzo realizado por el alumno en las clases prácticas.

d) Bibliografía

La bibliografía básica coincide en esencia con las referencias del artículo, razón por la que no vamos a repetirla en este epígrafe.

5.- REFERENCIAS

[1] Andrews, Gregory R., Concurrent Programming: Principles and practice. Ed Prentice-Hall, 1991.

[2] Barnes, J.G.P., Programación en Ada . Ed. Diaz de Santos, 1987.

[3] Ben-Ari, M., Principles of concurrent and distributed programming. Ed. Prentice Hall 1990.

[4] Blazquez Entonado F., González Bravo, T., Terrón González, J., Coordinadores. Materiales para la enseñanza universitaria. Instituto de Ciencias de la Edcación Universidad de Extremadura, 1997.

[5] Burns, Alan, Davies, Geoff, Concurrent programming. Ed. Addison Wesley 1990.

[6] Burns, Alan, Wellings, A., Real time systems and programming languages. Ed. Addison-Wesley, 1997.

[7] Gropp, W., Lusk, E., Skjellum, A., Using MPI, portable parallel programming with the message-passing interface. Ed. The MIT Press, 1995.

[8] http://atc.unex.es/jdiaz/anatome/contrib/champollion/champollion.html

[9] ISO/IEC 9945-1 ANSI/IEEE Std 1003.1, Information technology - Portable Operating System Interface (POSIX). Part 1: System Application Program Interface (API) (C Languaje). IEEE 1996.

[10] Mueller, F., "A library implementation of POSIX threads under UNIX". Proceedings of the USENIX Conference, pages 29-41, (1993)

[11] Pérez Martínez, Programación concurrente. Ed. Rueda 1990.

[12] Reinoso Peinado, A. *et al.* "Migracion de FSU-Pthreads a MINIX 2.0", IV Jornadas de Informática. Universidad de Las Palmas de Gran Canaria, Las Palmas de Gran Canaria, 13 a 17 de julio, (1998).

[13] Silberrschatz, A., Peterson, J., Galvin, P., Sistemas operativos. Conceptos fundamentales. Ed. Addison-Wesley Iberoamericana.