Actas XVIII JENUI 2012, Ciudad Real, 10-13 de julio 2012
I.S.B.N. 10: 84-615-7157-6 | I.S.B.N. 13:978-84-615-7157-4
Páginas 365-372

LSscheduling: herramienta interdisciplinar de soporte docente para la asignatura de Sistemas Operativos

Joan Navarro, Xavi Canaleta y Xavi Salada
Departamento de Ingeniería
La Salle - Universitat Ramon Llull
C/ Quatre Camins 2
08022 Barcelona
{jnavarro, xavic, xsalada}@salleurl.edu

Resumen

La planificación y gestión de procesos, *scheduling*, es un concepto fundamental en la asignatura de Sistemas Operativos. Ésta es una de las asignaturas que encajan mejor en el paradigma del Aprendizaje Basado en Proyectos (ABP) dada la proximidad inherente que hay entre el temario y los casos reales. Sin embargo, esta parte de la materia es de difícil aplicación práctica puesto que en los escenarios de trabajo habituales—como Linux o Windows—no es posible alterar las políticas de *scheduling* de una forma sencilla. Además, con la implantación de los nuevos grados, esta asignatura acoge alumnos de especialidades tan variadas como telecomunicaciones o electrónica, lo que invita a ampliar los horizontes de los escenarios prácticos de la misma. En este artículo se presenta una plataforma de trabajo horizontal en forma de recurso docente, basada en un micro controlador PIC que permite desarrollar prácticas de Sistemas Operativos teniendo un control total de la arquitectura. Si bien existen otros entornos de desarrollo virtuales, la originalidad de esta propuesta reside en el enfoque interdisciplinar y real que se le da a la práctica.

Summary

Process scheduling and management is one of the most relevant topics in the Operating Systems course syllabus. Actually, this subject fits pretty well in the Problem Based Learning (PBL) teaching paradigm due to the proximity between its contents and real world use cases. However, this part of the program is very complex, and often not possible, to move into a practical scenario because current laboratory learning environments—such as Linux or Windows—do not allow modifying the system scheduling policies in a straightforward way. In addition, according to the latest directives stated by the introduction of the new degrees, this subject also covers several disciplines such as telecommunications or electronics far beyond computer science, which invites to break up with the traditional laboratory assignments and think in new landscapes. This work presents a practical teaching resource based on a PIC micro controller, aimed at providing an appealing horizontal framework to develop the Operating System practical assignments with a full control of the underlying system architecture. Although several approaches have been presented so far, the novelty of this proposal remains in its interdisciplinary and real-world characteristics.

Palabras clave

Aprendizaje basado en problemas, sistemas operativos, prácticas interdisciplinares, programación de micro controladores, PIC.

1. Motivación

Tradicionalmente, Sistemas Operativos ha sido uno de los pilares de la Ingeniería en Informática de Sistemas. Ciertamente, la gran cantidad de conocimientos que el alumno adquiere en un espacio acotado de tiempo y la proximidad real que tienen sus contenidos la hacen especialmente atractiva. Es por ello que, a lo largo del tiempo esta asignatura ha sido objeto de varias experiencias docentes orientadas a mejorar la calidad de los estudios de Informática.

En general, esta asignatura encaja fácilmente en el modelo ABP [9] puesto que (1) no requiere de complejas infraestructuras o laboratorios, (2) el alumno tiene múltiples fuentes de información disponibles y (3) los contenidos son fácilmente fraccionables en pequeños proyectos. El protagonista de estas mini prácticas suele ser un ordenador con una distribu-

ción basada en un sistema *open-source* como Linux, sobre el que se desarrollan una serie de experimentos y ejercicios que sirven al alumno para adquirir una experiencia real sobre muchos de los conceptos que se dan en las clases teóricas.

Si bien es relativamente fácil preparar mini proyectos para trabajar y reforzar áreas como las colas de mensajes, los *signals*, la concurrencia de procesos, los sistemas de ficheros, etc., existen pequeñas partes del temario, pero no por ello menos importantes, que son de muy difícil aplicación práctica y real en los entornos habituales. Concretamente, esas partes de la materia que describen los conceptos que residen en la zona más interna del núcleo (*kernel space*) del sistema operativo como por ejemplo la paginación y segmentación de la memoria o la planificación de procesos. La complejidad para diseñar un entorno práctico de estos últimos está en el hecho que (1) como norma general el sistema operativo no da mecanismos en el espacio de usuario (*user space*) para alterarlos y (2) que las consecuencias de una incorrecta alteración de esas partes del sistema puede acarrear consecuencias irreversibles.

Además, con la implantación de los nuevos grados del Espacio Europeo de Educación Superior (EEES) en nuestra universidad, la asignatura de Sistemas Operativos también se cursa en los Grados en Ingeniería Electrónica, Telecomunicación, Telemática e Informática. Esta es una situación muy interesante dado que es una de las pocas ocasiones en las que perfiles software y hardware convergen en una misma asignatura. Esto supone una ventaja y un reto a la vez, puesto que el hecho de tener ambos perfiles enriquecerá el contexto del aprendizaje basado en proyectos. Sin embargo, el equipo docente afirma que tener dos perfiles—típicamente—tan antagónicos entre el alumnado exige preparar un discurso neutral entre ambos para satisfacer sus expectativas y estimular su interés. Actualmente, la asignatura gira entorno al mundo del software, dejando el hardware relegado a un segundo plano.

Estas dos carencias en la asignatura— (1) inexistencia de temas importantes en los entornos prácticos y (2) olvido de algunos aspectos hardware vinculados al sistema operativo—dan lugar a la motivación de este trabajo. Concretamente se propone:

1. Reducir el alcance de las proyectos actuales que se desarrollan en la asignatura de Sistemas Operativos.
2. Aprovechar la disparidad de perfiles software y hardware del alumnado para enriquecer el modelo educativo que propone el ABP.
3. Proponer una práctica enmarcada en un entorno más cercano al hardware que permita trabajar esos aspectos de la asignatura que los entornos software tradicionales no permiten.

El resto del trabajo está organizado de la siguiente manera. La sección 2 describe el estado actual de la asignatura de Sistemas Operativos. La sección 3 recoge propuestas similares que se han hecho para esta asignatura. La sección 4 describe la plataforma de trabajo propuesta, sobre la que se montan los escenarios prácticos. La sección 5 presenta una propuesta de escenario práctico. La sección 6 recoge algunas reflexiones del personal docente. Finalmente, la sección 7 presenta las conclusiones de la ponencia.

2. Introducción

En el curso 2009/2010, la Escola Tècnica i Superior d'Enginyeria Electrònica i Informàtica (ETSEEI) de La Salle diseñó e implantó el uso de una herramienta interdisciplinar que acompaña a los alumnos a lo largo de su trayectoria universitaria: el LSMaker [7]. Esta herramienta consiste en un pequeño robot-oruga sobre el que los alumnos desarrollan los distintos proyectos asociados a una buena parte de las asignaturas que cursan.

Esta plataforma se ha ido instaurando paulatinamente en todos los programas de grado que se dan en el centro. Así, en el año académico 2010/2011 se introdujo en los primeros cursos, en el 2011/2012 se está introduciendo en los segundos cursos y en el 2012/2013 se introducirá transversalmente en los terceros cursos. Es tarea del equipo docente decidir la mejor manera de sacar provecho de esta plataforma y ver de qué modo encaja con los contenidos de cada asignatura.

Los contenidos anuales de la asignatura de Sistemas Operativos de la antigua Ingeniería en Informática de Sistemas, se han reestructurado en dos asignaturas semestrales (Sistemas Operativos, objeto de este trabajo, y Sistemas Operativos Avanzados) dentro del nuevo plan de estudios del Grado en Ingeniería Informática. Mientras que la primera está orientada a dar una visión general de los sistemas operati-

Introducción a los Sistemas Operativos	**4.5 h.**
1. Introducción a los ordenadores	×
2. Evolución histórica	×
3. Componentes del Sistema Operativo	×
El núcleo del Sistema Operativo	**3 h.**
1. Representación de procesos PCB	×
2. Controlador de interrupciones FLIH	×
3. El *dispatcher*	×
4. Mecanismos de comunicación, sincronización y exclusión mutua	✓
Planificación de procesos	**6 h.**
1. Algoritmos de planificación básicos	×
2. Algoritmos de planificación en colas	×
3. Planificación en tiempo real	×
Mecanismos de comunicación, sincronización y exclusión mutua	**24 h.**
1. Comunicación entre procesos:	
1.1. *Pipes*	✓
1.2. Colas de mensajes	✓
1.3. *Sockets*	✓
1.4. Memoria compartida	✓
2. Herramientas de exclusión mutua y sincronización:	
2.1. Semáforos	✓
2.2. Monitores	×

Cuadro 1: Resumen de los contenidos de la asignatura. Horas lectivas dedicadas e inclusión en el modelo ABP (✓ incluído, × no incluído).

vos y acoge alumnos de otras titulaciones, la segunda es mucho más específica y trabaja los contenidos desde un punto de vista estrictamente informático tal y como se hacía en el antiguo programa.

Sistemas Operativos es una asignatura que se cursa en el primer semestre del tercer curso y tiene asignados 5 créditos ECTS. Semanalmente, los alumnos deben asistir a 3 horas de clase magistral y 1.5 horas de laboratorio. Las sesiones de laboratorio (o prácticas) consisten en resolver un pequeño proyecto sobre una máquina Linux tal y como recomienda la metodología ABP. El Cuadro 1 muestra el estado de la asignatura antes de aplicar la herramienta docente aquí propuesta. Se puede ver como la asignatura se divide en cuatro bloques:

- El primero, "Introducción a los Sistemas Operativos", da una descripción general de qué es un sistema operativo y lo contextualiza dentro de un computador. Este bloque sirve para contextualizar la asignatura y sentar las bases fundamentales de trabajao.
- El segundo, "El núcleo del Sistema Operativo", describe los módulos internos que componen un sistema operativo. Esta parte es de difícil aplicación práctica puesto que el tiempo que invertirían los alumnos en identificar cada entidad dentro del código fuente del sistema sería muy elevado. Sin embargo, sí que resulta factible trabajar los mecanismos de sincronización de procesos puesto que el sistema ya ofrece primitivas (*wait* y *signal*), en espacio de usuario, a tal efecto.
- El tercero, "Planificación de procesos", describe las distintas técnicas que usan los sistemas operativos para gestionar los procesos y dar la percepción de concurrencia en un entorno mono procesador. Nótese que este bloque no se trabaja desde el punto de vista práctico por la misma razón que el anterior. Además, ésta es una parte muy crítica del núcleo que podría ocasionar graves desperfectos si fuese erróneamente alterada.
- El cuarto, "Mecanismos de comunicación, sincronización y exclusión mutua", describe los problemas que se derivan de las interacciones entre múltiples procesos. Puesto que ésta es la parte que, proporcionalmente, más tiempo ocupa del temario (24 horas), la mayor parte de los proyectos prácticos van enfocados a reforzarla.

Entonces, se puede ver que hay una gran parte del temario (marcados con el símbolo "×" en el Cuadro 1) que pese a tener un porcentaje de dedicación significativamente menor que el cuarto bloque, no queda reflejada en las sesiones prácticas. A priori, se constata que la principal causa de ello radica en el hecho de que las sesiones prácticas de la asignatura están encorsetadas en un único entorno: Linux. Por eso, antes de describir nuestra propuesta docente, la siguiente sección evalúa la existencia de otros trabajos alineados a este problema.

3. Herramientas similares

Los robots siempre han sido una herramienta aparentemente atractiva para desarrollar herramientas docentes que faciliten el uso de metodologías ABP

en los estudios de ingeniería. Ciertamente, los robots poseen una naturaleza interdisciplinar que aglutina muchos conocimientos sobre una misma plataforma. Sin embargo, su uso en las aulas de Informática no siempre ha sido satisfactorio. Esta sección revisa algunos trabajos relevantes con el objetivo de recoger la experiencia de estudios previos y aplicarla en nuestro recurso docente.

En [3] se exponen los resultados de un estudio comparativo entre los alumnos que utilizan un robot Lego/Mindstorms para desarrollar sus prácticas de programación y los que utilizan las plataformas clásicas. Este estudio concluye que los alumnos que usaban el método tradicional sacaron mejores puntuaciones que los que utilizaban el robot. Según los autores, esto es debido a que (1) los alumnos no podían sacar el robot del laboratorio, (2) la poca afinidad entre la API del robot y la teoría de la asignatura, y (3) la poca experiencia del personal docente en esta nueva plataforma.

No obstante, en [5] se reflexiona sobre la necesidad de dar importancia al hardware en las carreras de Informática. Por eso, se propone de nuevo el uso de un robot, esta vez con un enfoque más transversal, como herramienta motivadora y vehicular para unificar distintas materias del plan de estudios y aumentar la motivación del alumnado. Sin embargo, el caso ABP que proponen está orientado a los alumnos del primer curso y enfocado únicamente a la especialidad de informática, lo que le priva de una experiencia interdisciplinar pura. Además, los autores apuntan que se debe cuantificar el volumen de trabajo asociado a la práctica puesto que el robot se podría convertir en una arma de doble filo y llegar a una situación en la que los alumnos rechazaran cualquier cosa relacionada con él.

La carencia de perfiles técnicos incorrelados entre los alumnos para realizar los trabajos propuestos por la metodología ABP también se refleja en [8], donde se habla sobre la dificultad de hacer unas prácticas interdisciplinares en la asignatura de Sistemas Operativos dada la poca homogeneidad de los perfiles y la carencia de tiempo. Por ello, se propone un entorno ABP centrado en la plataforma software MINIX sobre la que los alumnos trabajan los conceptos de la asignatura.

Por todo esto, la mayoría de trabajos orientados a mejorar la calidad docente de la asignatura de Sistemas Operativos pasan por desarrollar plataformas

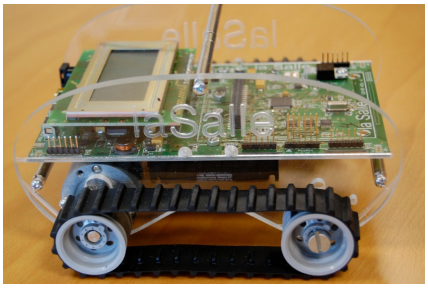

Figura 1: Robot LSMaker que reciben todos los alumnos.

software [1] que abstraen al alumno de la complejidad de trabajar a bajo nivel e impiden extrapolar los conceptos del temario de la asignatura a otros entornos.

Es evidente que el hardware no debe ser descuidado en los estudios de Informática, llámense Ingenierías o Grados [4]. Además, cabe recordar que uno de los principales objetivos del EEES es fomentar las interacciones entre alumnos con distintos perfiles. La siguiente sección tiene en cuenta las experiencias previas aquí resumidas—tanto positivas como negativas—y presenta el recurso docente objeto de la propuesta de este trabajo.

4. Descripción de la plataforma

Los planes de estudios de nuestra facultad establecen que el primer curso de los grados es común para todos los alumnos de ingeniería—grado en Ingeniería en Organización de las TIC, grado en Ingeniería Electrónica de Telecomunicación, grado en Ingeniería Informática, grado en Ingeniería Multimedia, grado en Ingeniería Telemática, grado en Ingeniería de Sistemas Audiovisuales y grado en Ingeniería de Sistemas de Telecomunicación. Esto invita a presentar el LSMaker a los alumnos como una herramienta vehicular entre las distintas especialidades y vertebradora a lo largo de sus estudios.

El hecho de entregar este robot, mostrado en la Figura 1, a los alumnos del primer curso de todos los grados, soslaya significativamente los posibles efectos negativos anteriormente descritos [3, 5, 8] que este pueda tener sobre el rendimiento académico del alumnado: por un lado se familiarizan con el robot desde el inicio de sus estudios y por el otro,

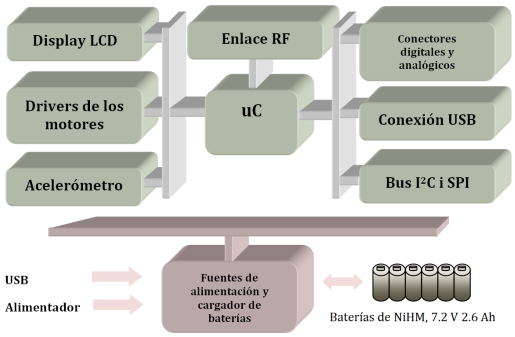

Figura 2: Arquitectura hardware del LSMaker.

viendo a sus compañeros de otras especialidades, lo aceptan y lo trabajan como una herramienta verdaderamente multidisciplinar.

Desde el punto visto pedagógico, el robot puede emplearse de dos modos: por un lado, el hecho de entregar una plataforma en funcionamiento permite trabajar el concepto de ingeniería inversa [2]—esto es especialmente útil para los alumnos de los primeros cursos que aún no poseen suficientes conocimientos técnicos. Por el otro lado, los alumnos de cursos más avanzados que ya poseen más conocimientos, pueden desarrollar nuevas mejoras y *gadgets* sobre la plataforma usando el diseño clásico *bottom-up*.

El robot ha sido diseñado por personal docente de la facultad, lo que facilita la difusión de conocimiento al resto de profesorado y minimiza los inconvenientes hallados en [3]. Además, se ha construido con componentes *low-cost* para hacerlo realmente asequible. En la Figura 2 se muestran los componentes hardware que conforman el robot[1].

Además, también se da a los alumnos una capa software de alto nivel (API)[2] para que puedan abstraerse, si quieren y lo necesitan, de las complejidades de los periféricos del robot. Por ejemplo, para hacer mover el robot hacia adelante pueden usar el siguiente método:

```
unsigned int MTLineal(unsigned int Time, int
Speed, int StopBits, int * StopReasons);
```

Nótese que los alumnos sólo tienen que elegir una velocidad, un tiempo durante el que el robot va a avanzar y elegir por qué razones el robot debería parar. El método devolverá el tiempo durante el cual el robot ha estado avanzando. Esto abstrae al alumno de la complejidad de la electrónica asociada al robot y le permite desarrollar su software con una mayor agilidad.

La siguiente sección propone el entorno de prácticas—objeto de este artículo—basado en esta plataforma hardware, el cual permite aplicar el método ABP en las sesiones de laboratorio correspondientes a los bloques “El núcleo del sistema operativo” y “Planificación de procesos” del Cuadro 1 y así hacer que la asignatura de Sistemas Operativos pueda impartirse mediante ABP en prácticamente su totalidad.

5. Propuesta práctica

La API que se entrega a los alumnos para que realicen prácticas de otras asignaturas implementa un paradigma de planificación no apropiativo [6]. Esto permite abstraerlos de las complejidades de la concurrencia de procesos, objeto de estudio únicamente en la asignatura de Sistemas Operativos, y concentrar sus esfuerzos en los objetivos de cada materia. Entonces, es en la asignatura de Sistemas Operativos cuando los alumnos (1) comprenden el funcionamiento interno de la API, (2) trabajan el paradigma de planificación apropiativo para (3) poder comparar el comportamiento desde un punto de vista funcional de ambos *schedulers* (apropiativo *versus* no apropiativo).

5.1. Objetivos

El objetivo de esta propuesta es cubrir desde un punto de vista práctico aquellas partes del temario que no es posible trabajar en los laboratorios de Informática tradicionales (véase Cuadro 1). Concretamente, se propone usar esta plataforma para trabajar los conceptos inherentes a la planificación de procesos:

1. Modelado de procesos. Información asociada a los procesos (también llamada*Process Control Block* o PCB).

[1] El detalle del hardware y los esquemáticos completos pueden descargarse de `http://blogs.salleurl.edu/LSMaker/zona-de-descargas/#hardware`

[2] La API completa puede descargarse de `http://blogs.salleurl.edu/LSMaker/zona-de-descargas/#apis`

2. Paradigmas de planificación. Planificadores *preemptive* y *no preemptive*, diferencias, ventajas e inconvenientes.
3. Algoritmos de planificación. Implementación, dominios de aplicación y análisis funcional.
4. Parámetros de ajuste y métricas. *Quantum*, tamaño del *stack*, tiempo de cambio de proceso.
5. El *dispatcher*. Implementación y consecuencias del cambio y restauración de procesos.

Estos objetivos se pueden trabajar sobre la plataforma LSMaker enfocando la práctica o miniproyecto desde dos posibles vertientes: el diseño *top-down* o ingeniería inversa, o bien el diseño *bottom-up*. En función de las restricciones temporales, necesidades y objetivos pedagógicos de cada asignatura, el personal docente deberá orientar el escenario a un modelo u otro. En adelante, por claridad, se asume un diseño *bottom-up*, considerando que el modelo de práctica basado en la ingeniería inversa debería seguir los pasos aquí descritos en orden opuesto.

5.2. Escenario

Para el correcto desarrollo de la práctica, únicamente se asume que (1) los alumnos tienen conocimientos básicos del lenguaje de programación C y (2) que entienden el concepto de concurrencia de procesos en un entorno mono procesador—hechos perfectamente asumibles dentro de la asignatura de sistemas operativos.

A partir de ahí, se le propone al alumno un escenario elemental en el que 4 procesos, los cuales deben implementar de forma transparente al planificador, compiten por usar el micro controlador del LSMaker:

1. Tracción. Este proceso activa los motores de tracción del robot durante 10 segundos, los para 5 segundos, y luego vuelve a activar.
2. LEDs. Este proceso enciende y apaga los LEDs del LSMaker a una frecuencia de 10 Hz.
3. Display LCD. Este proceso escribe y actualiza en la pantalla de cristal líquido la hora actual del sistema.
4. Canal série. Este proceso manda por el Hyperterminal la hora actual del sistema cada 5 segundos.

Los procesos se pueden implementar usando la API o sin usarla. El hecho de usar la API permite

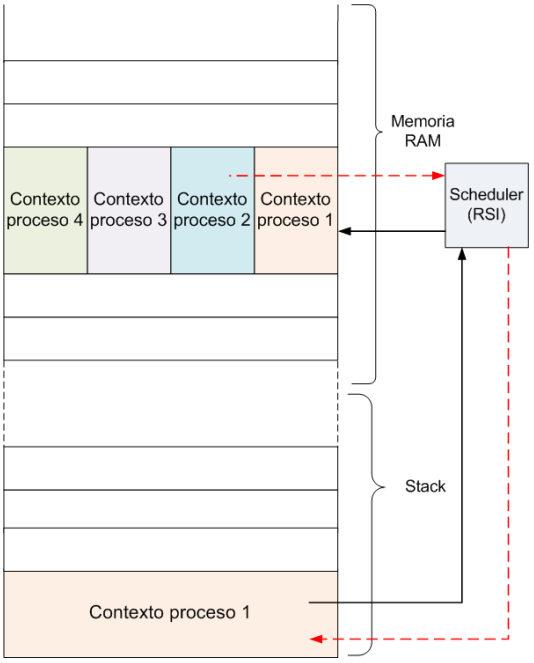

Figura 3: Primera aproximación al cambio de procesos. El PCB se almacena en la memoria RAM y la RSI lo copia oportunamente en la zona de *stack*.

a los alumnos escribir el código fuente de cada proceso con mucha facilidad, sin embargo complica el entorno desde el punto de vista conceptual ya que la API por sí misma ya implementa un planificador no apropiativo. En este caso los alumnos verían un modelo de *scheduler* híbrido—apropiativo y no apropiativo al mismo tiempo—que podría ser interesante estudiar en determinados contextos (especialmente bajo el modelo de la ingeniería inversa). En cambio, si no se permite que los alumnos usen la API, estos deberán comprender la arquitectura interna del micro controlador—que ya han trabajo en otras asignaturas del plan de estudios—antes de implementar cada proceso. Esta solución da un control total de cada proceso al alumno, lo que facilita la comprensión de lo que está pasando en todo momento.

Luego, los alumnos deben implementar un algoritmo de planificación apropiativo, por ejemplo un *round-robin* con *quantum* [6], sobre el LSMaker y que relacionen el comportamiento funcional del robot con el algoritmo de planificación.

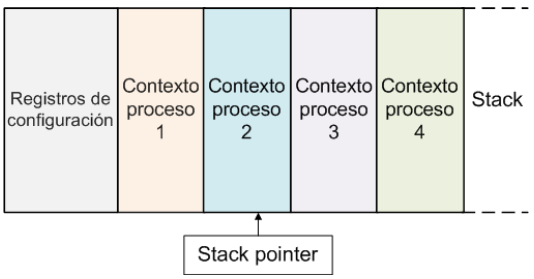

Figura 4: Segunda aproximación al cambio de procesos. El *stack pointer* recorre la tabla de PCBs.

5.3. Propuesta de solución

El elemento clave para llevar a cabo un cambio de proceso síncronamente, es la rutina de servicio de interrupción (RSI). Cuando el procesador está dando servicio a un determinado proceso y se activa una interrupción, ésta, por su naturaleza intrínseca, guarda todo lo relativo al proceso: variables locales, registros de estado, instrucción concreta que se estaba ejecutando, etc. En el *stack*, y una vez ha finalizado la interrupción, se restaura todo el contenido guardado para proseguir con el proceso que se había interrumpido previamente. Nótese que las acciones derivadas de gestionar la entrada de la interrupción y restaurar el *stack* lo resuelve internamente el micro controlador.

Entonces, se orienta al alumno para que desarrolle un *scheduler*, con su correspondiente *dispatcher*, que mediante la interrupción del *timer* pueda cambiar entre procesos tal y como muestra la Figura 3.

Nótese que cuando dicha interrupción guarde el contexto del proceso saliente en el *stack*, el *scheduler*, programado por los alumnos, deberá desapilar su PCB y guardarlo en la memoria RAM. A continuación, buscará el PCB del siguiente proceso a ejecutar en la memoria RAM del microcontrolador, y lo apilará sobre el *stack* para que cuando finalice la interrupción sea este proceso el que se restaure.

Con esta primera aproximación se consigue que los 4 procesos se ejecuten concurrentemente sobre un único procesador. Sin embargo, cuando los alumnos reducen el *quantum* del sistema se dan cuenta de que el cambio de proceso no es óptimo puesto que el hecho de ir apilando y desapilando los procesos en el *stack*, además de leer y escribir en la memoria RAM continuamente, requiere un tiempo de ejecución muy elevado que además es directamente proporcional al tamaño del PCB. En este punto se invita a los alumnos a optimizar el funcionamiento del *dispatcher* tal y como se describe a continuación.

La segunda aproximación, consiste en asignar un espacio de memoria RAM para el PCB de cada proceso para que ésta actúe como *stack* particular. La idea principal de esta implementación es mover el puntero *stack pointer*—en vez de todo el PCB—para que apunte al proceso que se está ejecutando en ese momento tal y como muestra la Figura 4. Entonces, cada vez que entra la interrupción del *timer* la única acción que se debe llevar a cabo es mover el *stack pointer* para que apunte al PCB del siguiente proceso a ejecutar. Así, cuando salga de la interrupción, el sistema restaurará automáticamente el proceso apuntado por el *stack pointer*.

6. Resultados e impacto observado

A fecha de hoy los resultados que se han obtenido, aunque parciales dada la planificación propuesta, han sido totalmente satisfactorios. El objetivo durante el presente curso académico, dada la falta de tiempo de preparación, ha sido la de utilizar LSscheduling no como una práctica dentro de la asignatura semestral de Sistema Operativos sino como un recurso docente de ejemplo. De este modo se diseñó una sesión lectiva en formato taller a los 35 alumnos. La duración de la misma fue de 90 minutos de duración. En el taller se presentó a la clase el escenario y la problemática, se animó al alumno a proponer soluciones al problema y, finalmente, se explicó la solución implementada.

La respuesta de los alumnos fue altamente positiva, valorando especialmente el hecho de poder observar un caso real de aplicación de los algoritmos de planificación con un grado de complejidad asumible dado el conocimiento que tiene el alumnado.

Cabe también mencionar que la presentación de dicho recurso en una jornada interna de innovación docente también tuvo un impacto claramente favorable en el resto de profesores de grado asistentes a dicho evento.

7. Conclusiones y lineas de futuro

Este artículo propone una plataforma de soporte docente para la asignatura de Sistemas Operativos que permite trabajar, desde un punto de vista práctico, aquellas partes del temario que típicamente no se pueden trabajar en los entornos tradicionales Linux o Windows. Especialmente, se recomienda su uso para reforzar aquellos conceptos que no se pueden trabajar con capas de software intermedias—dado que éstas pueden alterar los resultados—como son la gestión de interrupciones, la planificación y/o la gestión de la concurrencia de procesos en un sistema operativo.

Dada la novedad que supone utilizar una plataforma hardware en una de las asignaturas que históricamente se ha centrado en el software, el presente artículo recoge la experiencia de otras propuestas semejantes y propone el uso del LSMaker, un robot oruga diseñado a medida por nuestra universidad, que ha demostrado—en las pruebas piloto que se han realizado—carecer de los defectos apuntados por otros autores de experiencias docentes similares en el pasado.

Esta herramienta es de fácil aplicación en otras universidades dada la simplicidad del hardware utilizado y la extensa documentación del software asociado. No obstante, a pesar del éxito que ha tenido la prueba piloto, se debe tener en cuenta que la implantación de esta propuesta conlleva (1) redimensionar la carga de las prácticas actuales y (2) evaluar qué efectos pedagógicos tiene este redimensionado.

Agradecimientos

Los autores quieren mostrar su agradecimiento a Cristina Borda y Andreu Sancho por sus revisiones y comentarios. Y en especial a Francesc Escudero por su soporte incondicional al proyecto.

Referencias

[1] Canaleta, X., Navarro, J., *Herramientas de soporte al aprendizaje de sistemas de ficheros*, Actas XVI Jornadas de Enseñanza Universitaria de la Informática, 2010.

[2] Chikofsky, E. J., Cross, J.H., *Reverse Engineering and Design Recovery: A Taxonomy*, IEEE Software 7(1): 13-17, 1990.

[3] Fagin, B., Merkle, L., *Measuring the effectiveness of robots in teaching computer sicence*. SIGCSE Bulletin, vol 35. 2003 pp 307-311.

[4] Jimenez, A., Paz, R., Dominguez, M., Cerezuela E., Rodriguez, M. A., Villar, J. A., *Sistema de co-diseño hardware/software basado en FPGA para la captura de video analógico a través del bus serie USB*, Actas XVI Jornadas de Enseñanza Universitaria de la Informática, 2011.

[5] Oliver, J., Toledo, R., Pujol, J., Sorribes, J., Valderrama, E., *Un ABP basado en la robótica para las ingenierías informáticas*, Actas XV Jornadas de Enseñanza Universitaria de la Informática, 2009.

[6] Silverschatz A., Galvin P., Gagne, G., *Sistemas Operativos*, Editorial Limusa, ISBN: 968- 18-6168-X, 2002.

[7] Vernet, D., Canaleta, X., *LSMaker: un proyecto interdisciplinar*, Actas del Simposio-Taller de XVII Jornadas de Enseñanza Universitaria de la Informática, 2011.

[8] Vivaracho, C., *Aplicación del aprendizaje basado en problemas a la parte de laboratorio (MINIX) de una asignatura de Sistemas Operativos*, Actas XV Jornadas de Enseñanza Universitaria de la Informática, 2009.

[9] Walters, L., *Four Leading Models*, Harvard Education Letter's Research Online, 2000. 1992.