Una herramienta de Simulación para la Planificación de Procesos

Mercedes Barrionuevo[1], Fabiana Piccoli[1], Ruben Apolloni[1]

1 Universidad Nacional de San Luis, San Luis, Argentina {mdbarrio ,mpiccoli, rubenga}@unsl.edu.ar

Resumen

El principal objetivo de los Sistemas Operativos (SOs) en un ambiente de computación es la gestión de los recursos del sistema. El recurso más importante de todo sistema es la CPU y por lo tanto administrarlo de forma eficiente implica realizar una cuidadosa planificación de su uso. Es por ello que realizar una correcta planificación constituye uno de los puntos centrales en el diseño de un buen SO.

Comprender la importancia de la administración de los recursos generalmente se basa en el análisis de conceptos teóricos con prácticas de escritorio. Ayudar a la metodología de enseñanza tradicional mediante la experimentación permitirá comprender las características y funcionalidades del administrador del procesador, objetivo de este trabajo.

SPPP es un Simulador del Planificador de Procesos; su objetivo es realizar una correcta simulación de la planificación de los procesos bajo distintas cargas del sistema, permitiéndole al usuario la definición del ambiente de trabajo a recrear. La utilización de SPPP permite comparar el desempeño de una política de planificación con respecto a otra, pudiéndose establecer cuál de ellas se comporta mejor bajo una determinada configuración.

En este trabajo se presentan las consideraciones principales tanto del diseño como de la implementación de SPPP, además de los resultados obtenidos en el dictado de materias relacionadas a los SOs.

Palabras clave: Operativo, Simulación, Planificador de Procesos, CPU, Software Libre.

Abstract

The Operating Systems (OSs) have a main task that is the management of computing system resources. One of these resources is the CPU, consequently its management must be efficient and carefully planned. This aspect represents one of central points of every good OS design.

Generally, the importance of resource management is teached through traditional desk practices based on theoretical concepts. Introducing this concept by means of experimentation will allows to understand better the characteristics and funcionalities processor manager.

In this work, we present to SPPP, a process manager simulator whose main objetive is to be a didactic tool to analyze the OS behavior in several situations. SPPP allows the user to stablish the system environment: planning policy, work load, process kinds, among others characteristics. SPPP enables to compare performance of different processor scheduling politics and to determine which is better.

This paper presents the main considerations for the design and the implementation of SPPP. Finally experimental results obtained of its application in different laboratories of OS are shown.

Keywords: Operating System, Simulation, Process Planning, CPU, free software

1. Introducción

La complejidad de un SO no queda sólo en su diseño ni en su implementación, sino también en la comprensión de su funcionamiento y de sus características. Entender los conceptos relacionados a los SOs y la interacción que existe entre ellos no es una tarea trivial para los alumnos de las materias afines a los SOs.

La enseñanza de los conceptos de los SOs incluye temas netamente teóricos, los cuales si bien son ampliamente tratados en variados y numerosos libros, no resulta simple encontrar entornos experimentales adecuados.

Es por ello que se planteó la necesidad de desarrollar una herramienta educativa, la cual permita simular

íntegramente el trabajo del Planificador de Procesos bajo distintas cargas de trabajo, pudiendo el usuario (alumno) definir las características del SO y evaluar su desempeño según parámetros preestablecidos. Este trabajo tuvo como objetivo principal cambiar las tradicionales prácticas de escritorio por buenas prácticas de laboratorio apoyándose en la simulación de sistemas. En [1, 2] se presentaron los primeros pasos en el desarrollo de SPPP.

En el desarrollo de SPPP se tuvieron en cuenta los conceptos de: SO con multiprogramación o sin ella, de tiempo compartido, monousuario o multiusuario, etc. En general, decimos que dos procesos son concurrentes si su ejecución se solapa en el tiempo. Si tales procesos están presentes en memoria principal al mismo tiempo y comparten el procesador, decimos que el sistema acepta multiprogramación. Esta idea surgió con el objetivo de maximizar el uso de la CPU, manteniendo siempre algún proceso en ejecución.

En este paper se describen las características del software desarrollado y se muestra cómo puede ayudar a los profesores y estudiantes en el proceso de enseñanza-aprendizaje. Para lograrlo, primero se realiza un análisis de las herramientas existentes en esta área, analizando sus ventajas y debilidades. Luego se detallan los aspectos de diseño y de implementación considerados para el desarrollo de SPPP y sus interfaces. Finalmente se explican los beneficios de la herramienta, las pruebas de campo desarrolladas, las conclusiones y futuros trabajos.

2. Estado del Arte

La búsqueda de diversas metodologías de enseñanza para cubrir el problema de aprendizaje de los diversos conceptos relacionados a los SO se ve reflejada principalmente en las prácticas de aula o de laboratorio.

Analizando los laboratorios propuestos tanto en la UNSL como en otras universidades públicas y privadas nacionales e internacionales, se pudo observar que en la mayoría de ellos las prácticas se basan en la modificación del código fuente, usando el SO Linux, de alguna parte del SO. Si bien se obtienen resultados muy buenos, para lograrlos se necesita que los estudiantes posean un buen conocimiento de programación en C, Unix, de la arquitectura de la computadora y de la estructura general del SO.

Si bien, existen varios simuladores en el marco de los SOs, en particular del Planificador de Procesos, tales como: PlanP [3], Ovsos [4], Sosim [6], RCOS [5], NACHOS [7], etc., no existen trabajos más actuales en el área. Las herramientas mencionadas anteriormente fueron probadas y analizadas obteniendo las siguientes observaciones:

- *PlanP* : Es un simulador muy completo en la parte de configuración de los procesos, permitiendo establecer una comparación del comportamiento de las distintas políticas de planificación implementadas. Respecto a la visualización del progreso del sistema, si bien se puede observar el Diagrama de Gantt de la simulación, la animación no es muy clara. Se puede ver el cambio de estado de los procesos en el sistema pero no se puede establecer el momento en qué cada proceso entra a ejecución, se bloquea o finaliza. Además no permite llevar un registro de los eventos que provocan cambios de estados de los procesos.

Ovsos . Es una herramienta bastante completa a la hora de realizar la configuración del sistema a simular dado que se puede establecer: el algoritmo de planificación a simular, el tiempo de inicio y tiempo de vida en el sistema del proceso, el tamaño del quantum, etc. Sin embargo, no permite establecer en qué momento puede un proceso realizar una operación de E/S con lo cual es imposible comparar el comportamiento de dos políticas distintas con la misma carga de trabajo. Por otro lado, su animación presenta una interfaz poco clara a la hora de seguir lo que va ocurriendo en el sistema.

SoSim . Este simulador presenta una interfaz amigable al usuario pudiendo: crear procesos CPU-Bound, I/O Bound o Mix con sus respectivas prioridades, visualizar los distintos estados por los cuales va pasando cada proceso, establecer el tamaño del quantum, etc. Sin embargo su mayor inconveniente está en la recolección de estadísticas, dado que independientemente del estado de la simulación, el modulo responsable de hacerlo sigue calculando valores que no se pueden interpretar. Además de no permitir ver el comportamiento de los procesos ante una determinada política de planificación, sólo funciona en la plataforma windows.

RCOS : se centra demasiado en los aspectos visuales y no profundiza en la aplicación de aspectos teóricos.

NACHOS : La distribución de Nachos contiene el código base para un SO operacional y un simulador de una computadora genérica. Presenta una amplia funcionalidad, para asignaturas troncales básicas pero tiene como principal inconveniente su orientación hacia aspectos de diseño e implementación.

De lo antes expuesto, se puede observar que ninguna de las herramientas existentes logra ser completa, es decir que cumple con: ser fácil de instalar, poseer una interfaz gráfica amigable al usuario (alumno), permitir obtener resultados estadísticos correctos y claros de interpretar, contar con una buena animación, la cual permita especificar con exactitud el sistema a simular y, fundamentalmente, contar con la

documentación adecuada para su uso. Todo lo mencionado anteriormente es lo que se buscó desarrollar logrando nuestra herramienta de simulación: SPPP.

3. Generalidades

Los SOs son fundamentales en las disciplinas de las ciencias de la computación. Involucran altos niveles de teorías, diseños y abstracción. SPPP busca integrar lo anterior ayudando al estudiante a construir un modelo mental.

Los simuladores crean modelos simples y dinámicos del mundo real, pueden ser utilizados en diversas áreas en educación resultan de gran utilidad. Sin embargo, construir simuladores no es una tarea fácil ni trivial. Se debe construir una herramienta que muestre todos los eventos que suceden en el sistema, en nuestro caso todos los eventos que ocurren en la planificación de procesos, de una manera simple y compresiva por el alumno.

El Simulador de Planificador de Procesos, SPPP, es una herramienta la cual muestra la evolución de una planificación previamente configurada con determinados conceptos significativos como: tipos de procesos, ráfagas de CPU necesarias para ejecutar cada proceso, política de planificación, entre otros.

Sus principales objetivos son:

- Mostrar los aspectos de la planificación de la CPU en un sistema de un único procesador. Para ello se ilustra la evolución de los procesos desde su ingreso al sistema hasta la finalización, teniendo en cuenta diferentes algoritmos de planificación.

Permitir al alumno analizar el comportamiento de las distintas políticas de planificación en un SO, proporcionándole así la capacidad de entender, modificar o agregar otra política, con el fin de observar y/o deducir los cambios consecuentes en el sistema.

Visualizar los resultados estadísticos obtenidos permitiendo al alumno elaborar sus propias conclusiones de qué algoritmo se comporta mejor bajo las mismas condiciones del sistema.

Durante el desarrollo de SPPP se tuvo en cuenta todo lo mencionado en los puntos anteriores.

En las siguientes secciones se detallan los aspectos principales del diseño e implementación de la herramienta desarrollada.

3.1. Aspectos de Diseño

El diseño de SPPP se basó en el objetivo de desarrollar una herramienta portable y escalable. Para ello se definió un conjunto de módulos autocontenidos y con interfaces bien diferenciadas. Como resultado, SPPP quedo compuesto por los siguientes módulos: *Administrador de Procesos, Administrador de la CPU, Administrador del Reloj y Administrador y Recolector de Estadísticas*.

Cada uno de los módulos cumple una función específica y bien definida. Dicha división en módulos permitió no sólo una mejor comprensión del funcionamiento del sistema, sino también posibilita su modificación o actualización, por ejemplo, incluir otras políticas de planificación no consideradas originalmente o políticas nuevas. Dicha modificación no alterará el funcionamiento de los otros módulos mientras se respete la interface definida entre ellos.

3.2. Aspectos de Implementación

Para la elección de las tecnologías de desarrollo, se tuvo en cuenta la necesidad de desarrollar una herramienta portable, de código libre y con una interface gráfica amigable al usuario. En función de ello, se optó respecto al:

- *Sistema Operativo*: las plataformas de desarrollo: Windows XP o posteriores (Vista, Seven) [8], MacOS [9] y Linux [10, 11] de la familia de los Sistemas Operativos Unix [12, 13]. De esta forma se cumple con el objetivo de obtener un software para todas estas plataformas, ampliamente utilizadas en la actualidad.

Lenguaje de Programación: Si bien existen diferentes lenguajes de programación, adecuados para la implementación de SPPP, se eligió utilizar Java [14, 15] por sus características tales como: licencia de uso (libre); portabilidad; facilidad para la creación de interfaces gráficas [16, 17]; velocidad de desarrollo; independencia de la plataforma de ejecución; posibilidad de Multithreading [18]; disponibilidad de herramientas; y por último, y no por eso menos importante, por ser un lenguaje Orientado a Objetos [19, 20], lo cual facilita la modularidad del sistema, permitiendo la incorporación de nuevos módulos, aspecto fundamental tenido en cuenta en la etapa de diseño de SPPP.

Gráficamente, la herramienta de simulación propuesta en este paper consiste de: una Ventana de Configuración, en la cual se establecen los parámetros necesarios para poder realizar la simulación; y una Ventana de Ejecución, donde se realiza la simulación (de forma animada o sin animación) y donde se presentan los resultados estadísticos.

Las interfaces antes mencionadas son explicadas de forma detallada en las siguientes secciones, mostrando además las prestaciones de cada una.

4. Interfaces

SPPP cuenta con dos interfaces principales, cada una de ellas con una función bien definida: interface de configuración e interface de simulación.

La pantalla de configuración es la base de la simulación, proporciona los datos sobre los cuales debe trabajar la simulación. Es a través de ella donde se define el ambiente del SO a simular.

Por su parte, la interface de simulación no sólo muestra la simulación y/o animación del proyecto sino también los resultados obtenidos de la misma. Muestra cómo los procesos creados competirán por el procesador según el algoritmo de planificación elegido y las características individuales de cada proceso.

A continuación se explica con más detalle cada una de las interfaces gráficas que componen a SPPP .

4.1. Interface de Configuración

La ventana de configuración tiene como marco de trabajo a los *Proyectos*. Un proyecto es una configuración de un sistema específico, tiene la información de la política de planificación elegida y las características de cada uno de los procesos.

Figura 1. SPPP: Interfase de Configuración

Brinda la posibilidad de generar un nuevo proyecto, abrir uno existente o guardar los datos del proyecto actual. Esto último permite poder trabajar con los mismos procesos pero con políticas de planificación diferentes, posibilitando la comparación de las políticas en un sistema con las mismas condiciones.

La figura 1 muestra la ventana de configuración de datos. El usuario (alumno) puede: Seleccionar una determinada política de planificación, y Configurar los procesos que intervienen en la planificación. Para llevarlo a cabo, la interfaz provee las siguientes facilidades:

Para los proyectos: Cada vez que se desee ejecutar una simulación, se trabaja a nivel de proyecto. Las acciones permitidas son:

- Nuevo: Crear un nuevo ambiente de configuración.

Abrir: Abre un proyecto anterior.

Guardar: Almacena de forma permanente los datos generados de cada proceso y la información necesaria de la política a simular. Los archivos de SPPP se almacenan con extension *.mdb.

Para las políticas de planificación: Aquí se selecciona todo lo relacionado a la política de planificación.

Configuración de los datos de la política: Cuál política se va a aplicar, puede ser:

  - FIFO: El primer proceso en solicitar la CPU es el primero en recibirla.

Round Robin: A cada proceso se le asigna un intervalo de tiempo, llamado quantum, durante el cual se le permite ejecutar. Si el proceso en ejecución agota su quantum, el SO le expropia la CPU y se la asigna al próximo proceso ubicado en la cola de listos. Si el proceso se bloquea o termina antes de finalizar el quantum, la conmutación de CPU se efectúa naturalmente cuando el proceso se bloquea.

Por Prioridad: A cada proceso se le asigna una prioridad, el proceso con prioridad más alta es seleccionado para hacer uso de la CPU.

- Tipo:

  - No Apropiativa: Escoge el proceso que se ejecutará y luego, simplemente, le permite ejecutarse hasta que se bloquee (ya sea por una operación de E/S o en espera de otro proceso) o ceda voluntariamente la CPU. Aunque el proceso se ejecute durante horas, no se lo suspenderá. En este caso no se toman decisiones de planificación durante las interrupciones de reloj, siempre se reanuda el proceso que se estaba ejecutando antes de ella.

Apropiativa: Escoge un proceso, el cual se ejecutará durante un durante un tiempo establecido. Si finaliza dicho tiempo y el proceso continúa en ejecución, se le retira la CPU, el proceso pasa a la cola de listos y el planificador escoge otro proceso para ejecutarse (si hay uno disponible).

- Quantum: Este parámetro es necesario para la política Round Robin, su valor debe ser siempre estrictamente mayor a cero.

Para un proceso se debe especificar:

- Prioridad: Importancia asignada al proceso configurado.

Tiempo de Arribo: Tiempo en el cual el proceso arriba al sistema.

Cantidad de E/S: Es el número de E/S a realizar por el proceso. A cada E/S se le asigna un tiempo de duración.

Ráfagas de CPU mínima y máxima: Tiempo de CPU necesario para que el proceso complete su trabajo. Este valor se puede generar dos formas: Aleatoria o Fija.

Duración de cada E/S o tiempo de E/S: tiempo durante el cual un proceso se encuentra realizando una operación de E/S. Cada operación de E/S tiene un tiempo de duración asociado.

Porcentaje de tiempo de CPU: Es el porcentaje del tiempo total del proceso (CPU+E/S) asignado al uso de la CPU. Si el porcentaje es superior al 50 %, el proceso es considerado CPU-Bound.

Porcentaje de tiempo de E/S: Es el porcentaje del tiempo total del proceso (CPU+E/S) asignado a realizar operaciones de E/S. Si este porcentaje es superior al 50%, el proceso es considerado E/S-Bound.

Tipo de Proceso: Los procesos se clasifican en tres tipos: CPU-Bound, I/O-Bound y Mix. El tipo de proceso se determina en función del porcentaje de uso de CPU: Mayor al 50% es CPU-Bound, igual al 50% es Mix y menor al 50% es I/O-Bound.

Tiempo mínimo necesario para finalizar: Es la cantidad de tiempo requerido por un proceso para completar su tarea. Éste es calculado automáticamente por el sistema considerando las ráfagas de CPU, la cantidad de E/S y la duración de cada E/S.

Para los procesos en el sistema: Una vez dadas las características de un proceso, se puede:

- Generar Más: Calcula, en base a la cantidad de E/S y las ráfagas de CPU, datos adicionales del proceso no ingresado por el usuario, tales como: tipo de proceso (CPU-Bound, I/O-Bound o Mix), porcentaje de uso de CPU, porcentaje de E/S y tiempo mínimo necesario para que un proceso finalice su ejecución.

Agregar: Una vez configurado los datos de un proceso particular se agrega a la tabla de procesos para confirmarlo en el sistema.

Eliminar: Cuando un proceso no va a formar parte del sistema se lo elimina, previa selección en la tabla.

Modificar: Permite ajustar los datos de configuración de un proceso.

Tabla de Procesos Generados: Muestra la totalidad de los procesos generados y confirmados hasta el momento para la simulación.

Limpiar Datos: Elimina la información de la tabla de procesos generados.

Tabla de información de E/S: Esta tabla mantiene la información relevante sobre el número de E/S, la duración de dicha E/S y el tiempo de ocurrencia de la misma.

4.2. Interface de Simulación

Una vez establecida la configuración de la política y los procesos, se puede comenzar la simulación de la planificación. En cada paso de tiempo (clock) se muestra el estado de cada proceso como también el estado de las colas y de la CPU. Tras concluir la planificación todos los procesos han finalizado su ejecución. Finalmente, SPPP calcula y muestra los resultados estadísticos de la planificación ejecutada. Para mostrar tanto la simulación como los resultados obtenidos se utiliza la misma interface, la cual tiene dos momentos de visualización: el desarrollo de la simulación y los resultados estadísticos. Las características de la interface anteriormente mencionada son las siguientes:

Visualización de la Simulación

La ventana de simulación mostrada en la figura 2 tiene los siguientes elementos con sus respectivas funciones:

- Ejecutar Proyecto: Realiza la simulación sin la visualización del sistema y lo que ocurre en él.

Ejecución de la Simulación con animación: Inicia la simulación conjuntamente con la visualización de la misma.

Detener la Animación: Finaliza la ejecución de la simulación y la animación. No obtiene resultados estadísticos.

Pausar/Reanudar la Animación: Detiene la simulación y la visualización (sin finalizarla) durante el tiempo que el usuario decide, reanudándolo al presionar nuevamente dicho botón.

Velocidad de la animación: Acelera o ralentiza la visualización de la animación.

Volver: Regresa a la ventana.

Resultados de la ejecución: Muestra los resultados estadísticos del planificador para el sistema ejecutado.

Cola de procesos Listos: Muestra los procesos listos para ser ejecutados por la CPU, ordenados según la política seleccionada.

CPU: Distingue al proceso siendo atendido.

Procesos que finalizaron sus tareas.

Lista de procesos bloqueados: Muestra los procesos en espera de algún evento: la finalización de una operación de E/S. Dicha estructura no mantiene un orden en particular, los procesos se retiran de la misma según el tiempo de finalización de la operación de E/S causante del bloqueo del proceso.

Clock: Muestra el tiempo de simulación transcurrido, el cual es discreto (y no uniforme). El avance del tiempo es en función del tiempo de ocurrencia de los eventos.

Histórico: Muestra todos los eventos producidos en el sistema, el tiempo de ocurrencia y quién lo generó.

Visualización de Resultados

Al finalizar la simulación se muestran los resultados estadísticos obtenidos por SPPP (ver sector 7 de la figura 2). Los indicadores considerados para analizar el desempeño de una política de planificación son:

- Tiempo de Respuesta Promedio: Es la media de los tiempos de respuesta de cada proceso. El tiempo de respuesta de un proceso se calcula como la diferencia entre su tiempo de finalización y su tiempo de arribo.

Tiempo de Espera Promedio: Es el promedio de los tiempos de espera de cada proceso (Diferencia entre el tiempo de respuesta del proceso y la cantidad de Ráfagas de CPU requeridas).

Eficiencia Promedio: Es la media de la eficiencia de todos los proceso creados y ejecutados en el sistema. La eficiencia de un proceso se calcula mediante el cociente entre el tiempo requerido por sus ráfagas de CPU y su tiempo de respuesta.

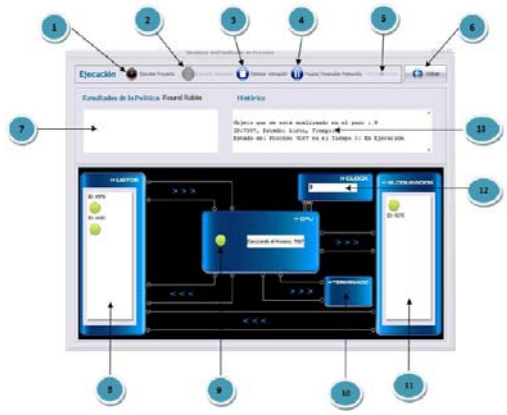

Figura 2. SPPP: Interfase de Visualización de la Simulación

Durante el diseño de la herramienta, se puso como objetivo que esta ventana sea lo más simple y clara posible para que el alumno (usuario), quien pueda seguir los eventos del sistema sin muchos inconvenientes.

A continuación se detalla uno de los últimos pasos en el ciclo de vida del desarrollo del software: las pruebas.

5. Pruebas de Campo

SPPP fue sometido a exhaustivas pruebas para su evaluación, demostrando en cada una de ellas ser capaz de realizar las simulaciones de forma correcta y con los resultados estadísticos esperados.

En primer lugar, la evaluación fue llevada a cabo por los profesores de las materias afines y sus respectivos equipos de cátedra.

Una vez aprobado por los equipos de cátedra, SPPP fue presentado a tres grupos de alumnos con tres niveles de grado de avance en las carreras del Departamento de Informática:

- Grupo Inicial: Alumnos de los primeros años de la carrera, sin haber cursado ninguna materia relacionada al diseño de los SO's. Con ellos se trabajaron los conceptos básicos de los SO's, y principalmente la claridad y didáctica de las interfaces. El número de alumnos en este grupo fue de 52.

Grupo Medio: Alumnos que están cursando la materia Sistemas Operativos. Es este grupo se trabajaron los conceptos de diseño de los SO's y las consecuencias de las decisiones tomadas en dicha etapa, por ejemplo tipo de política de planificación. En este caso fueron 46 los alumnos que trabajaron con SPPP.

Grupo Avanzado: Alumnos avanzados de la carrera cursando los últimos años. Con ellos, aparte de analizar el funcionamiento del simulador y la evaluación de sus características positivas y negativas, se evaluó la modificación de código agregando nuevas políticas de planificación. La cantidad de alumnos de dicho grupo fue 31.

Los grupos *Inicial* y *Medio* realizaron dos actividades prácticas. La primera siguió las consignas propuestas en la práctica tradicional de la materia (Prácticas en lápiz y papel), mientras que la segunda consistió de actividades de laboratorio utilizando SPPP. Dicha práctica consistió de tres ejercicios con las siguientes características:

1. En el primero se debía configurar un sistema que simulara la política de planificación

Round Robin con un quantum de 3 unidades de tiempo, 4 procesos donde los tiempos de arribos y ráfagas de CPU fueran distintos y sin operaciones de E/S. Luego se pedía realizar la animación y una vez obtenidos los resultados compararlos con los obtenidos mediante el uso de diagramas de Gantt.

El segundo ejercicio consistía en simular los algoritmos de planificación FIFO y Round Robin con un quantum de 2 unidades de tiempo. Para ello se contaba con 4 procesos arribados todos en el mismo instante de tiempo, con distintas ráfagas de CPU y cantidad de operaciones de E/S, dando lugar a la coéxistencia de procesos CPU-Bound e I/O-Bound. Una vez terminada la simulación se pedía observar los resultados obtenidos, verificando si dichos resultados son correctos y comparar los dos algoritmos simulados para analizar cuál de ellos se comporta mejor ante la situación planteada.

El tercer ejercicio proponía realizar la simulación de un sistema con 4 procesos con igual tiempo de arribo, distintas prioridades y sin operaciones de E/S. Bajo esas hipótesis se pedía planificar dichos procesos, registrando los resultados obtenidos, usando las políticas Por Prioridad (No Apropiativo), FIFO y Round Robin con quantum=2, para finalmente definir cuál de ellos presentaba una mejor eficiencia explicando el por qué de ello.

Junto al práctico de máquina se le entregó a cada alumno involucrado en la práctica de laboratorio una encuesta cuyo objetivo fue evaluar el impacto de SPPP en cada grupo.

La encuesta se realizó en forma no anónima para lograr un mayor nivel de compromiso por parte de los alumnos y obtener resultados fehacientes.

Las preguntas se realizaron en función de poder responder de entre cuatro opciones: Malo, Regular, Bueno o Muy Bueno.

Las preguntas que se presentaron fueron las siguientes:

1. ¿Cómo considera a SPPP como herramienta didáctica para la comprensión de la Planificación de Procesos?

Establezca la facilidad de uso de SPPP y la interfaz gráfica presentada.

¿De qué manera el uso del simulador le permite aprender conceptos teóricos respecto a la Planificación de Procesos?

¿Qué resultados estadísticos cree que sería importante agregar?

Mencione fortalezas y debilidades qué, según su criterio, posee el simulador de procesos.

Las respuestas de las dos últimas preguntas fueron libres, el objetivo fue que los alumnos expongan aquellos aspectos que sería interesante agregar y/o modificar. Los resultados obtenidos fueron la sugerencia de incluir entre las medidas de performance del sistema informadas al througput y al porcentaje de tiempo de CPU ocioso. Además surgió como principal fortaleza de nuestra herramienta su facilidad de uso y la simplicidad de la animación, permitiendo un perfecto entendimiento del funcionamiento de las políticas de planificación.

Como principal debilidad la mayoría observó la falta de un módulo que permitiese ver el Diagrama de Gantt generado por la planificación simulada, motivo del cual se plantea como trabajo futuro.

En la Tabla 1 se muestran los correspondientes porcentajes obtenidos en la encuesta.

Pregunta	Mala	Regular	Buena	Muy Buena
1			27%	73%
2		6%	13%	81%
3			14%	86%

Tabla 1. Resultados de la Encuesta hecha a los alumnos después de finalizar el práctico.

Conclusiones

Comprender los conceptos relacionados a los SO y sus interrelaciones no es una tarea simple para los alumnos de materias afines a los SO, generalmente la enseñanza se plantea de forma teórica y con prácticas de escritorio. Cambiar éstas por buenas prácticas de laboratorio constituyó la motivación para este trabajo.

SPPP es una herramienta portable, de software libre y multiplataforma. Permite presentar, estudiar y comprender el planificador de procesos en un sistema con condiciones reguladas. Constituye un buen recurso didáctico para los docentes y una herramienta útil para los alumnos, quienes pueden recrear sistemas con características específicas, visualizar y analizar el comportamiento del sistema según el planificador seleccionado, realizar comparaciones, elaborar conclusiones y, fundamentalmente, entender el funcionamiento de los diferentes algoritmos de planificación de los procesos desde una práctica amigable.

SPPP fue evaluado en la práctica por los alumnos de tercer, cuarto y quinto año de la carrera Licenciatura en Ciencias de la Computación, de los cuales se obtuvieron resultados positivos en cuanto a su

aceptación como así también críticas constructivas a tener en cuenta para futuras modificaciones.

Varias son las posibles extensiones a SPPP, entre ellas se encuentran:

- Mejorar la simulación desarrollada, automatizando los tiempos entre arribos de los procesos y la duración de las E/S usando mejores distribuciones de probabilidad.

Adaptar a SPPP de forma tal de poder simular los niveles de planificación de mediano y largo plazo.

Ampliar las capacidades de SPPP para visualizar diagramas de N estados y N planificadores.

Agregar un modulo de visualización que permita visualizar los Diagramas de Gantt.

Desarrollar un simulador integral del Sistema Operativo, lo cual implica incluir los Administradores de Memoria Real y Virtual, el Administrador de Archivos y el Administrador de Entrada/Salida.

Adaptar a SPPP a la nueva generación de procesadores para poder realizar simulaciones de planificación de procesos en entornos multicore, determinando cuáles procesos son independientes y la forma en la que se van asignando a cada core.

Ampliar las capacidades funcionales del SPPP para permitir la simulación de Planificación de Hilos (threads).

Todas estas extensiones están relacionadas no solo a mayores prestaciones de la herramienta desarrollada, sino al hecho de desarrollar un producto integral para la enseñanza de los SO

Referencias

[1] L. Frazier, J. D. Fodor, The sausage machine: A new two-stage parsing model. Cognition, 6 (1978), pp. 291-325

[1] M. Barrionuevo, R. Apolloni, and M. F. Piccoli. El Planificador de Procesos a través de un Simulador. XV Congreso Argentino en Ciencias de la Computación. CACIC 2009. Jujuy. Argentina. Paginas: 444-453. 2009.

[2] SPPP: Simulador del Planificador de Procesos. Mercedes Barrionuevo, Rubén Apolloni, Fabiana Piccoli. XVII CONGRESO ARGENTINO DE CIENCIAS DE LA COMPUTACION. CACIC 2011. La Plata. Argentina. Páginas: 610-619. ISBN 978-950-34-0756-1

[3] PLANP: Simulador de planificación de CPU. Resultado del Proyecto Fin de Carrera titulado "PLANP: herramienta para la simulación de políticas de planificación de procesos" realizado por Honorio Mateo Mérida y dirigido por Mónica Trella López. Diciembre 2005.

[4] Ovsos: is a free open source of Visual Operating System. http://ovsos.sourceforge.net/ . Agosto 2007.

[5] RCOS: Yet Another Teaching Operating System. Ron Chernich, Bruce Jamieson, David Jones.

http://davidtjones.wordpress.com/publications/rcosyet-another-teaching-operating-system. 1996.

[6] SoSim. Luiz Paulo Maia. M.Sc. thesis in Federal University of Rio de Janeiro (NCE/UFRJ), Brazil, in 2001. http://www.training.com.br/sosim/indexen.htm

[7] Nachos. W. A. Christopher et a. (1993), The Nachos Instructional Operating System. Proceedings of the Winter 1993 Usenix Technical Conference, pp 481-489.

[8] Wi. R. Stanek. Microsoft Windows XP Profesional. Manual del administrador. McGraw-Hill / Interamericana de España, S.A., 2002.

[9] J. Feiler. La Biblia de Mac OS JAGUAR. Anaya Multimedia, 2003.

[10] H. F. Arena. La Biblia de Linux. M. P. Ediciones, 2003.

[11] M. Beck, H. Bohme, M. Dziadzka, U. Kunitz, R. Magnums, and D. Verworner. Linux Kernel Internal. Addison-Wesley, 1998.

[12] S. Sánchez Prieto and O. García Población. Unix y Linux: Guía Práctica. Ra Ma, 2004.

[13] D. Taylor. Unix. Anaya Multimedia-Anaya Interactiva, 2006.

[14] Joyanes. Programación En Java 2. Mcgraw Hill - España, 2002.

[15] Sun. The source for java developers. http://java.sun.com

[16] Holzner. La Biblia de Java 2. Anaya Multimedia, 2000.

[17] M. Loy and R. Eckstein. Java Swing. O'reilly Media, 2003.

[18] O. Scott and W. Henry. Java Threads. O'reilly and Associates Inc - Estados Unidos.

[19] Ángel García Beltrán and José M. Arranz Santamaría. Programación Orientada a Objetos con Java. ETS Ingenieros Industriales, 2005.

[20] D. Warnes and M. Kolling. Programación Orientada a Objetos con Java. Prentice Hall, 2004.

Dirección de Contacto del Autor/es:

Mercedes Deolinda Barrionuevo
LIDIC
Ejército de los Andes 950
San Luis-Argentina
e-mail: mdbarrio@unsl.edu.ar
Sitio web: http://www.unsl.edu.ar

María Fabiana Piccoli
LIDIC
Ejército de los Andes 950
San Luis
Argentina
e-mail: mpiccoli@unsl.edu.ar
sitio web: http://www.unsl.edu.ar

Rubén Gerardo Apolloni
LIDIC
Ejército de los Andes 950
San Luis
Argentina
e-mail: rubenga@unsl.edu.ar
sitio web: http://www.unsl.edu.ar

Mercedes Deolinda Barrionuevo. Lic. en Cs de la Computación. Auxilar-Docente de la UNSL, Dpto. de Informática en el área de Sistemas de Computación. Investigador en el área de HPC

María Fabiana Piccoli. Doctora en Cs. de la Computación. Profesor Adjunto de la UNSL Dpto. de Informática en el área de Sistemas de Computación. Directora de la línea de investigador HPC.

Rubén Gerardo Apolloni. Doctor en Ciencias de la Computación. Profesor Adjunto de la UNSL, Dpto. de Informática en el área de Sistemas de Computación. Investigador en el área de HPC