SiSO: Un simulador integral del Sistema Operativo

M. Barrionuevo, R. Apolloni, F. Piccoli
LIDIC- Universidad Nacional de San Luis
Ejército de los Andes 950
Tel: 02652 420823, San Luis, Argentina
{*mdbarrio, rubenga, mpiccoli*}*@unsl.edu.ar*

1. Contexto

Esta propuesta de trabajo se lleva a cabo como trabajos finales de la Licenciatura en Ciencias de la Computación y su objetivo es la utilización de los productos desarrollados en la enseñanza de las materias donde se abordan los conceptos relacionados al diseño e implementación de los Sistemas Operativos.

Resumen

Comprender cada uno de los conceptos relacionados a los Sistemas Operativos(SO) y la interrelación de todas sus componentes, no es simple. Su estudio, generalmente se basa en el análisis de conceptos teóricos con prácticas de escritorio, cambiar éstas por buenas prácticas de laboratorio constituyó el disparador de esta línea de trabajo.

Si bien existen numerosos simuladores de SO, generalmente no constituyen una herramienta didáctica íntegra. La construcción de un simulador del SO tiene como objetivo proveer una herramienta de software, la cual simularía el funcionamiento de un SO y cada uno de sus componentes, en ambientes con condiciones controladas a través de la definición de las características del sistema, la selección de la políticas y la comparación del desempeño. En este trabajo se presentan las consideraciones principales tanto del diseño como de la implementación de SiSO(Simulador Integral del Sistema Operativo).

2. Introducción

Una computadora es un conjunto de componentes interconectados entre sí que sin ningún tipo de software son elementos sin mayor utilidad. Con el software adecuado, en cambio, una computadora puede almacenar, procesar y recuperar todo tipo de información. El software de computadora puede dividirse en dos tipos: programas de sistema, encargados de controlar la operación de la computadora misma; y programas de aplicación, responsables de realizar las tareas solicitadas por el usuario. Dentro del primer grupo, se destaca por su papel fundamental, el SO. El SO asegura una distribución ordenada y controlada de los distintos recursos: procesadores, memorias y dispositivos de Entrada/Salida (E/S) entre los distintos usuarios en el sistema[2, 3]. Además, el SO oculta toda la complejidad del hardware, brindando al programador un conjunto más conveniente de instrucciones para trabajar.

Los SOs modernos tienen diferentes estructuras y por su complejidad generalmente son divididos en partes, las cuales se encargan de administrar uno o más de los recuros antes mencionados.

La complejidad de un SO no queda sólo en su diseño ni en su implementación, sino también en la comprensión de su funcionamiento y sus características. Entender los conceptos relacionados a los SOs y la interacción que existen entre ellos no es una tarea trivial en la mayoría de los casos. La enseñanza acerca de conceptos de SO incluye temas netamente teóricos, los cuales si bien son ampliamente tratados en variados y numerosos libros, no resulta simple encontrar entornos adecuados, para el nivel de conocimiento del alumnado, que permitan realizar actividades donde poner en práctica la teoría.

El enfoque de simulación de sistemas[1], es una herramienta con la cual pueden modelarse distintos aspectos de los SO, tal como la administración de procesos, de dispositivos, de memoria principal y secundaria, la transmisión de datos e información, entre otros, pudiendo ser un enfoque alter-

nativo a la construcción de un SO, con fines de entender y experimentar en la parte de diseño.

La mayor cantidad de simuladores disponibles corresponden al Administrador del Procesador, más específicamente del Planificador de Procesos. Entre los simuladores utilizados en las prácticas de la materia podemos mencionar a Planp[7], sim[10], SoSim[6], Ovsos[8], RCOS[4]. Si bien su inclusión tuvo resultados satisfactorios, ninguno de ellos se adapta de manera completa a lo que se pretende trabajar en la materia. Es por ello que se plantea la necesidad de diseñar e implementar una herramienta de software para simular el funcionamiento de un SO a través de los distintos administradores, bajo distintas cargas de trabajo, permitiendo la definición de las características del sistema, la selección de las políticas de administración y la comparación del desempeño logrado según los parámetros seleccionados.

Entre las cosideraciones a tener en cuenta para el desarrollo de un buen simulador, se encuentran las diferentes características de un SO, se puede hablar de SOs con multiprogramación o sin ella, de tiempo compartido, monousuario o multiusuario, etc. Si el sistema es multiprogramado, varios procesos están presentes en memoria principal al mismo tiempo y la CPU se alterna entre ellos, esto surgió con la idea de maximizar el uso de la CPU, manteniendo siempre algún proceso en ejecución.

Un simulador integral del SO, *SiSO*, debe tener en cuenta las funciones de éste, las más reevantes son:

- Administrar el procesador: Asignar el procesador a los distintos procesos a fin de obtener una buena ejecución en el sistema.
- Administrar la memoria: Gestionar el espacio de memoria asignado a cada proceso y a cada usuario. Cuando la memoria física es insuficiente, el SO será responsable de administrar la "memoria secundaria" o "virtual".
- Gestionar las E/S: Unificar y controlar el acceso de los procesos a los recursos. El SO debe gestionar el almacenamiento temporal de E/S y atender las interrupciones de los dispositivos.
- Administrar la Seguridad: Garantizar el uso de recursos sólo a aquellos procesos que tengan las autorizaciones para hacerlo.
- Gestionar los Archivos: Manejar la lectura y escritura en el sistema de archivos y las autorizaciones de acceso a los archivos.

En este trabajo se especifican las consideraciones a seguir para el desarrollo de *SiSO*.

3. Líneas de Desarrollo

Comprender las características SO, los distintos aspectos involucrados en su diseño y su funcionamiento, no es un proceso simple. Contar con una herramienta que lo facilite, permitiría afianzar e integrar todos los conceptos teóricos.

Si bien existen numerosos simuladores de SO, generalmente no constituyen una herramienta didáctica íntegra. La construcción de un simulador global permitirá visualizar y analizar el funcionamiento de los distintos componentes de un SO en condiciones controladas del sistema.

El desarrollo de un buen simulador global del SO no es simple. Por la complejidad del problema, se lo puede dividir en distintas líneas de desarrollo, las cuales se corresponden con los administradores de recursos principales, como son el procesador y la memoria[2, 3, 11, 12, 13]. Por todo lo expuesto, las líneas propuestas a seguir son:

- *Administrador del Procesador*

  Un programa es un conjunto de instrucciones que serán ejecutadas por la CPU. Por su parte un proceso es un programa en ejecución, con un contador de programa especificando la próxima instrucción a ejecutar y un conjunto de recursos asociados. Dependiendo de las operaciones de E/S que realizan los procesos se pueden clasificar en CPU-Bound o I/O Bound. Si un proceso hace uso intensivo de E/S se lo denomina I/O Bound (limitados por E/S); si hace uso intensivo de CPU, se lo conoce como CPU Bound (limitados por la CPU), una mezcla de ambos es también válida.

  En un sistema de computadora, el procesador (CPU) ejecuta una gran cantidad de procesos. A medida que un proceso avanza en su ejecución necesita diversos recursos: procesador, memoria, archivos, dispositivos de E/S, etc. Un proceso, durante su ciclo de vida, puede estar en distintos estados, permanecer o no en un estado depende

de la ocurrencia de uno o más eventos. Los principales estados en los que puede estar un proceso son: Listo(Permanece a la espera de la asignación del procesador), Ejecutando(Posee el procesador) y Bloqueado(Espera por la ocurrencia de un evento externo o de E/S).

El SO debe asegurar que cada proceso reciba una cantidad suficiente y justa de tiempo de CPU. Es por ello que es necesario administrar eficientemente el procesador entre los procesos. El Administrador de Procesos es el responsable de administrar los procesos, sus interrelaciones y sus actividades en el SO. Entre las tareas a realizar está la asignación del procesador a los distintos procesos en el sistema, conocida como Planificación de Procesos. A la hora de diseñar un administrador de procesos se debe tener encuenta los distintos estados de un proceso.

La planificación de procesos en un SO puede ocurrir en distintos niveles. Existen tres niveles de planificación bien determinados: *Planificación de Alto Nivel* (Administra los programas que ingresan al sistema), *Planificación de Nivel Medio* (Es responsable de mantener en condiciones equilibradas al sistema) y la *Planificación de Bajo Nivel* (o Panificador del Procesador, es su responsabilidad decidir quién, cuándo, cómo y por cuánto tiempo recibe el procesador un proceso listo). En este momento estamos abocados a este último tipo de Planificador.

Existen numerosas políticas de asignación de procesos listos a la CPU. Las poíticas más conocidas son:

- Primero en llegar, Primero en ser Servido (FCFS): en este algoritmo el primer proceso en solicitar la CPU es el primero en recibirla.
- Round Robin(RR): La CPU es asignada por un mismo periodo de tiempo (quantum) a cada proceso listo. Si el proceso no finaliza en dicho intervalo de tiempo vuelve al final de la cola listo (*ready*) para seguir ejecutando.
- Planificación con múltiples colas: la cola ready es dividida en varias colas, los procesos son asignados a alguna de ellas. Cada cola tiene su propio algoritmo de planificación.

Por todo lo mencionado anteriormente, se puede observar que un buen desempeño del Planificador de proceso permitirá aprovechar la CPU al máximo, y en consecuencia, se tendrá un rendimiento óptimo. Esta propiedad es la que nos interesa que el alumno comprenda a través de prácticas de laboratorio. Es por ello que proponemos un Simulador del Planificador de Procesos, SPPP.

- *Administrador de la Memoria:*

  Generalmente un proceso reside en un archivo binario en disco. Para poder ejecutar un proceso, éste debe estar, al menos n forma parcial, presente en memoria principal (en cosecuencia es necesario ubicarlo en ella)[].

  La ubicación de un proceso en memoria implica no sólo cargar el archivo, sino también, dependiendo del tipo de administración, la traducción o enlace de las direcciones lógicas a físicas.

  A través de la historia, se ha demostrado que los procesos crecen en requerimientos de memoria tan rápido como crece la memoria. Por ello se puede dividir al administrador de la memoria en dos: Administrador de memoria principal y el administrador de memoria virtual. Por el momento nos dedicaremos a la administración de la memoria principal.

  Así como evolucionaron los SO y sus características, también lo hizo la administración de la memoria principal. Podemos citar como íconos de la administración de la memoria a: Una única partición de memoria, Múltiples particiones fijas con múltiples colas de procesos, Múltiples particiones fijas con una única cola de procesos y Múltiples particiones variables. Puede observarse que la evolución de la administración de la memoria coincide con la incorporación de propiedades en los SO, tal como la multiprogramación, direccionamiento lógico, entre otros.

  El simulador del SO, considerará una administración de la memoria principal de múltiples particiones, con dos filosofías distintas:

particiones fijas no necesariamente del mismo tamaño, y particiones variables. En ambos casos permitirá estudiar: Las estrategias de ubicación de procesos en memoria y la fragmentación de la memoria.

Para ubicar un proceso en memoria, el SO se basa en distintas estrategias de ubicación. Para satisfacer una solicitud de memoria de tamaño n, independiente de si el particionado es fijo o variable, el SO decide cuál partición es la adecuada para el procesos en cuestión. Existen distintas estrategias, las mas conocidas son: Primer ajuste (Asigna la primer partición capaz de contener al proceso), Mejor Ajuste (Selecciona la partición más pequeña capaz de contener al proceso) y Peor Ajuste (Asigna el proceso a la partición más grande).

Dependiendo de la forma en que se generan las particiones y las políticas de asignación de memoria a procesos, el alumno podrá visualizar y/o analizar el concepto de fragmentación: Interna (Espacio libre de memoria dentro de una partición) y Externa (Existe suficiente lugar para alojar un proceso, pero éste no es contiguo).

La administración de memoria con particiones dinámicas, además de la fragmentación externa, trae aparejado otros conceptos a tener en cuenta, ellos son: La administración de espacios libres, unificación de espacios libres y garbage collection.

SiSO permitirá al alumno seleccionar las políticas de administrador de procesador y la memoria principal en cada simulación, inicializando los parámetros requeridos, pudiendo realizar varias simulaciones con distintos parámetros a fin de evaluar y comparar el comportamiento del SO[5, 9]. Para poder realizar la simulación, *SiSO* necesitará establecer el estado del sistema:

- Cantidad y tipo de procesos.
- Política de planificación a utilizar, con su respectiva configuración de parámetros.
- Tiempo de CPU que necesitara un proceso para ejecutar (ráfagas de CPU).
- Cantidad de Memoria requerida por cada proceso.
- Estrategia de Ubicación en memoria.

Una vez establecido el ambiente del SO e iniciada la simulación, los procesos serán ubicados en memoria principal según la estrategia elegida y competirán por el procesador según el planificador seleccionado. Para concluir su ejecución, cada proceso debe consumir el total de ráfagas de CPU asignada al momento de su creación. Una vez finalizado el proceso, se lo retira de memoria liberando el espacio ocupado.

4. Resultados obtenidos / esperados

El principal aporte de esta propuesta será la incorporación de las herramientas desarrolladas en las prácticas de laboratorio de la materia Sistemas Operativos de las carreras Licenciado en Ciencias de la Computación, Ingenieria en Computación e Ingenieria en Informática. Además, como se desarrolla sobre software libre podrá ser utilizado en otros centros educativos.

El simulador permitirá al alumno analizar el desempeño de un SO según las características del sistema y las políticas seleccionadas. El análisis se hará según algún criterio de comparación y recolección de datos estadísticos [3, 11]. Alguno de los criterios a tener en cuenta son:

- Tiempo de respuesta: es el tiempo transcurrido en obtener un resultado.
- Turnaround Time: Mide cuanto tiempo demora en ejecutar un proceso, incluyendo el tiempo que el proceso se demora en ingresar al sistema y los que permanece ocioso.
- Tiempo de espera: Mide el tiempo que el proceso está en la cola ready.
- Fragmentación de interna y/o externa de la memoria.
- Eficiencia en el uso de los recursos.
- Porcentaje de Utilización de la CPU por procesos de usuario.
- Throughput: Numero de procesos completados por unidad de tiempo.

Respecto a la Formación de Recursos Humanos, como resultado de las tareas realizadas se están desarrollando dos trabajos de fin de carrera

de la Licenciatura en Ciencias de la Computación, una de las cuales está en la fase final de desarrollo.

Además se debe destacar la posibilidad de extensión del trabajo, entre las extensiones posibles se encuentran: el administrador de la memoria virtual y sus distintas métodos: paginado, segmentado, segmentado/paginado; considerar la existencia de más de un procesador; entre otros.

Referencias

[1] J. Banks, J. Carson, B.L. Nelson, D. Nicol. Discrete-Event System Simulation, 4/E. Prentice Hall (2004). ISBN: 0131446797.

[2] H.M. Deitel, P.J. Deitel, D.R. Choffnes. Operating Systems, 3/E. Prentice Hall (2003). ISBN: 0131828274.

[3] P.B. Galvin, G. Gagne, A. Silberschatz. Fundamentos de Sistemas Operativos, 7/E. McGraw-Hill (2006). ISBN: 8448146417

[4] D. Jones, A. Newman. A constructivist-based tools for operating systems education, Proceedings of EdMedia'2002, Denver, Colorado, Jun. 2002.

[5] A.M. Law. Simulation Modeling and Analysis, 4/E. McGraw-Hill (2007). ISBN: 0073294411.

[6] L.P. Maia, A.C. Pacheco. A Simulator Supporting Lectures on Operating Systems. 33rd ASEE/IEEE Frontiers In Education Conference 2003 Boulder, Colorado.

[7] H.M. Mérida. Planp: Herramienta para la simulación de políticas de planificación de procesos. Simulador de planificación de CPU

[8] OVSOS: Other Visual Simulation of an O.S. http://sourceforge.net/projects/ovsos/

[9] J.A. Payne. Introduction to Simulation Programming Techniques and Methods of Analysis. McGraw-Hill College (1982). ISBN: 9780070489455.

[10] Software para la Simulación de un Planificador de Procesos. Escuela Universitaria Politécnica de Teruel, Universidad de Zaragoza. España. http://eupt2.unizar.es.

[11] W. Stallings. Operating Systems: Internals and Design Principles, 4/E. Prentice Hall (2001). ISBN: 0130319996.

[12] A.S. Tanenbaum. Modern Operating Systems, 3/E. Prentice Hall (2007). ISBN: 0136006639.

[13] A.S. Tanenbaum, A.S Woodhull. Operating Systems Design and Implementation, 3/E. Prentice Hall (2006). ISBN: 0131429388.