SPPP: Simulador del Planificador de Procesos

Mercedes Barrionuevo, Rubén Apolloni, and Fabiana Piccoli

Universidad Nacional de San Luis
Ejército de los Andes 950
5700, San Luis, Argentina
{mdbarrio.rubenga,mpiccoli}@unsl.edu.ar
http://www.unsl.edu.ar

Resumen El principal objetivo de los sistemas operativos en un ambiente de computación es la gestión de los recursos del sistema. Administrarlos de forma eficiente implica realizar una cuidadosa planificación de su uso. Un sistema de computación tiene muchos recursos, el más importante: la CPU. Su correcta planificación constituye uno de los puntos centrales en el diseño de un buen Sistema Operativo (SO).
Comprender y entender la importancia de la administración de los recursos, generalmente se basa en el análisis de conceptos teóricos con prácticas de escritorio. Ayudar a dicha 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 proveer una herramienta de software para simular la planificación de los procesos, permitiendo al usuario la definición de las características del sistema, la selección de la política de planificación y la comparación del desempeño de cada una de las políticas implementadas. En este trabajo se presentan las consideraciones principales tanto del diseño como de la implementación de SPPP, además de los resultados obtenidos de una primera aplicación en el dictado de materias relacionadas a los SOs.

Keywords: Sistema Operativo. Recursos. CPU. Planificador de Procesos. Simulador

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 sus características. Entender los conceptos relacionados a los SOs y la interacción que existen entre ellos no es una tarea trivial para los alumnos de las materias afines a los SOs.

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 experimentales adecuados, para el nivel de conocimiento de los alumnos.

Es por ello que se planteó la necesidad de desarrollar un sistema, el 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 bajo los 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 [2] se presentaron los primeros pasos para el desarrollo de SPPP.

Entre las consideraciones a tener en cuenta para el desarrollo de SPPP 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.

Este trabajo se organiza de la siguiente manera, en la sección 2 se explican los objetivos y responsabilidades de SPPP. En las siguientes secciones se detallan los aspectos de diseño y de implementación tenidos en cuenta para el desarrollo del simulador y sus interfaces. Finalmente se explican las pruebas de campo desarrolladas, las conclusiones y futuros trabajos.

2. SPPP: Generalidades

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.

Y sus responsabilidades:

- Determinar los posibles estados en los cuales un proceso puede estar durante su ciclo de vida dentro del sistema.
- Calcular la cantidad de ráfagas de CPU restantes durante la ejecución de un proceso.
- Administrar las estructuras que alojen procesos en espera por la CPU o por algún dispositivo de E/S.

En el desarrollo de SPPP se tuvo en cuenta todo esto. En las siguientes secciones se detallan los aspectos principales del diseño e implementación.

2.1. Aspectos de Diseño

El diseño de SPPP tuvo como objetivo el desarrollo de una herramienta portable y escalable. Por ello su diseño se realizó como un conjunto de módulos autocontenidos y con interfaces bien definidas. Como resultado, SPPP está 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. La división en módulos permite 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 interfaz definida entre ellos.

2.2. Aspectos de Implementación

Para la implementación de SPPP y la elección de las tecnologías de desarrollo, se tuvo presente la necesidad de desarrollar una herramienta portable, de código libre y con una interfaz gráfica amigable al usuario. En función de ellas, se optó respecto al:

- Sistema Operativo: las plataformas de desarrollo: Windows XP o posteriores (vista, Seven) [11], MacOS [4] y Linux [1,3] de la familia de los Sistemas Operativos Unix [9] [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 a Java [12] [7] por sus características tales como: licencia de uso (libre); portabilidad; facilidad para la creación de interfaces gráficas [6] [8]; velocidad de desarrollo; independencia de la plataforma de ejecución; posibilidad de Multithreading [10]; disponibilidad de herramientas; y por último, y no por eso menos importante, por ser un lenguaje Orientado a Objetos [5] [14], 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.

SPPP fue desarrollado en base a dos interfaces gráficas, las cuales son explicadas en las siguientes secciones, mostrando además las prestaciones de cada una.

3. SPPP: Interfaces

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

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

Por su parte la interfaz 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.

3.1. Interfaz de Con guración

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

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:

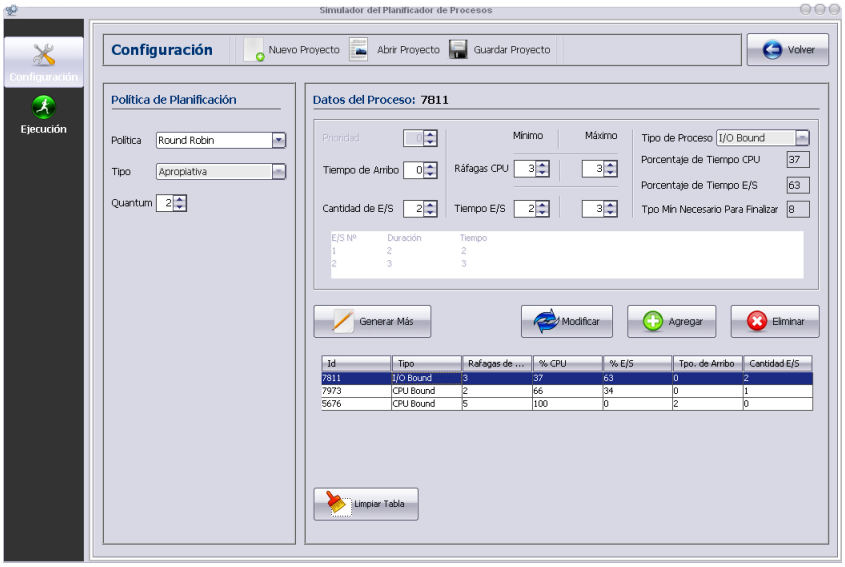

Figura 1. SPPP: Interfaz de Configuración.

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 extensión *.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:
  - *F FO:* 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 se apropia de la CPU y se la asigna al próximo proceso 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 mas 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 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 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:* Porcentaje de E/S que puede realizar el proceso. Los procesos CPU-Bound realizan la mínima cantidad de E/S, mientras que los procesos I/O-Bound efectúan un alto porcentaje de E/S.
- *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.
- *Tipo de Proceso:* Se determina en función del porcentaje de uso de CPU: Mayor al 50 % es CPU-Bound, igual al 50 % es Mix o menor al 50 % es I/O-Bound
- *Porcentaje de tiempo de CPU:* Los procesos cuyo uso de la CPU es intensivo tendrán un valor superior al 50 %.
- *Porcentaje de tiempo de E/S:* Los procesos que realizan varias operaciones de E/S tendrán un valor superior al 50 %.
- *Tiempo mínimo necesario para finali ar:* Es la cantidad de tiempo requerido por un proceso para completar su tarea. Éste se obtiene de las ráfagas de CPU y la cantidad y duración de 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.

3.2. Interfaz de imulación

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. Finalmente, tras concluir la planificación, todos los procesos han finalizado su ejecución, SPPP calcula y muestra los resultados estadísticos de la planificación ejecutada. Para mostrar tanto la simulación como los resultados obtenidos, se utiliza la misma interfaz, la cual tiene dos momentos de visualización: el desarrollo de la simulación y los resultados estadísticos. Cada una con las siguientes características:

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

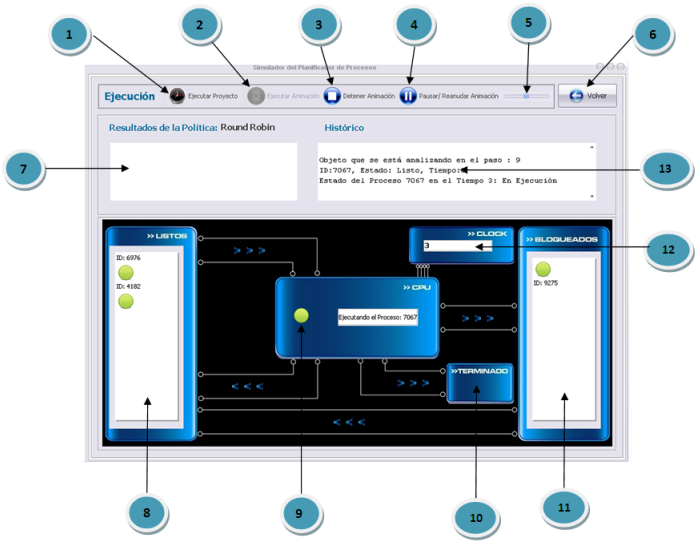

Figura 2. SPPP: Interfaz de Ejecución.

1. *Ejecutar Proyecto:* Realiza la simulación sin la visualización del sistema y lo que ocurre en él.
2. *Ejecución de la Simulación con animación:* Inicia la simulación conjuntamente con la visualización de la misma.
3. *Detener la Animación:* Finaliza la ejecución de la simulación y la animación. No obtiene resultados estadísticos.
4. *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.
5. *Velocidad de la animación:* Acelera o ralentiza la visualización de la animación.
6. *Volver:* Regresa a la ventana inicial.
7. *Resultados de la ejecución:* Muestra los resultados estadísticos del planificador para el sistema ejecutado.
8. *Cola de procesos Listos:* Muestra los procesos listos para ser ejecutados por la CPU, ordenados según la política seleccionada
9. *CPU:* Distingue al proceso siendo atendido.
10. *Procesos que finali aron sus tareas.*
11. *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.

12. *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.
13. *Histórico:* Muestra todos los eventos producidos en el sistema, el tiempo de ocurrencia y quién lo generó.

- **Visualización de esultados:** Al finalizar la simulación se muestran los resultados estadísticos obtenidos por SPPP en el sector 7 de la interfaz mostrada en la figura 2. Las estadísticas recolectadas son:
  1. *Tiempo de Respuesta Promedio:* Es la media entre el tiempo de finalización de cada proceso y su tiempo de arribo.
  2. *Tiempo de Espera Promedio:* Es el promedio entre el tiempo de respuesta de cada proceso menos la cantidad de Ráfagas de CPU del mismo.
  3. *Eficiencia Promedio:* Se calcula en base a la cantidad de Ráfagas de CPU necesarias de cada proceso dividido el tiempo de respuesta.

4. Prueba de Campo

SPPP fue sometido a exhaustivas prueba para su evaluación, demostrando en cada una de ellas ser capaz de realizar las simulaciones de forma correcta y con los resultados estadísticos esperados. Esta etapa de 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 dos grupos de alumnos, el primer grupo (treinta alumnos) fueron alumnos del tercer año de la Licenciatura en Ciencias de la Computación, los cuales estaban cursando la materia Sistemas Operativos. Los mismos realizaron dos tipos de prácticas, una la tradicional y la otra en laboratorio. La práctica de laboratorio tenía las mismas características que la de aula, sólo que el alumno debía configurar y obtener los resultados estadísticos utilizando SPPP. El segundo grupo (veinte alumnos): alumnos avanzados de cuarto y quinto año, quienes ya tenían aprobada dicha materia. A este grupo se le presentó la segunda parte del práctico.

Finalmente, todos los alumnos realizaron una evaluación personal del simulador, exponiendo las ventajas y desventajas encontradas. En todos los casos las críticas fueron muy constructivas, permitiendo en muchos casos la mejora o el planteo de trabajos futuros.

5. Conclusiones y Trabajos Futuros

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 constituyo 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 sólo a mayores prestaciones de la herramienta desarrollada, sino al hecho de desarrollar un producto integral para la enseñanza de los SO.

Referencias

1. H. F. Arena. La Biblia de Linux. M. P. Ediciones (2003).
2. M. Barrionuevo, R. Apolloni y M. F. Piccoli. El Planificador de Procesos a través de un Simulador. XV Congreso Argentino en Ciencias de la Computación. CACIC 2009. Jujuy. Argentina, pages 444-453, 2009.
3. M. Beck, H. Bohme, M. Dziadzka, U. Kunitz, R. Magnums, and D. Verworner. Linux Kernel Internal. Addison-Wesley, 1998.
4. J. Feiler. La Biblia de Mac OS JAGUAR. Anaya Multimedia, 2003.
5. Ángel García Beltrán and José M. Arranz Santamaría. Programación Orientada a Objetos con Java. ETS Ingenieros Industriales, 2005.
6. S. Holzner. La Biblia de Java 2. Anaya Multimedia, 2000.
7. Joyanes. Programación En Java 2. Mcgraw Hill - España, 2002.
8. M. Loy and R. Eckstein. Java Swing. O'reilly Media, 2003.
9. S. Sánchez Prieto and O. García Población. Unix y Linux: Guía Práctica. Ra Ma, 2004.
10. O. Scott and W. Henry. Java Threads. O'reilly and Associates Inc - Estados Unidos.
11. Wi. R. Stanek. Microsoft Windows XP Profesional. Manual del administrador. McGraw-Hill / Interamericana de España, S.A., 2002.
12. Sun. The source for java developers, `http://java.sun.com`
13. D. Taylor. Unix. Anaya Multimedia-Anaya Interactiva, 2006.
14. D. Warnes and M. Kolling. Programación Orientada a Objetos con Java. Prentice Hall, 2004.