Proyecto N° 1

Experimentación de Procesos y Threads con los Sistemas Operativos

Comisión: Escudero Johanna, LU: 105868
Pesce Danilo, LU: 104936

Informe – Proyecto Nro 1 Sistemas Operativos

Experimentación de Procesos y Threads con los Sistemas Operativos

Verificador de Sudoku

Para la resolución de este problema, se consideró al sudoku como una matriz de 9x9 la cual es inicializada con los valores de un archivo llamado “sudoku.txt”. Para maximizar la concurrencia son utilizados 27 procesos de los cuales cada uno controla una fila, una columna o un cuadrante. Como los procesos solo tienen acceso para lectura en la matriz, no se hay problema alguno en que trabajen todos concurrentemente.

Para resolver el problemas con hilos, es utilizada la librería pthread la cual provee funcionalidades para utilizar hilos. La forma de resolver el problema es la misma que con procesos; son utilizados 27 hilos que realizan cada uno una tarea: verificar una fila, una columna o un cuadrante.

La mencionada verificación se encarga de decidir si una fila o una columna o un cuadrante tienen valores repetidos, si no los tienen, la verificación es exitosa.

Mini Shell

Para la implementación de la mini shell se tuvieron en cuenta factores muy importantes como la extensibilidad y sobre todo, el funcionamiento de la linea de comandos de Linux, la cual ejecuta sus comandos mediante un proceso independiente, por lo tanto, cada uno fue implementado en un archivo a parte (con excepción al comando “exit”). Estos archivos son cargados por la shell utilizando la llamada al sistema execv() en el momento que son solicitados, la cual recibe el nombre del archivo y los parámetros correspondientes. Gracias a esto, resulta muy sencillo agregarle comandos extra a la mini shell sin tener la necesidad de recompilar el código en su totalidad.

Los comandos implementados son :

-mkdir(nombre) : Crea un directorio.
-rmdir(nombre) : Elimina un directorio.
-touch(nombre) : Crea un archivo.
-ls : Imprime el contenido del directorio actual.
-help: Imprime un texto de ayuda por pantalla.
-cat(nombre) : Imprime el contenido de un archivo por pantalla.
-pwd : Muestra la ruta actual.
-exit : finaliza el proceso de la shell.

Sincronización

Para resolver el problema, considerando que cada usuario es un thread, se utiliza la librería semaphore.h la cual provee los semáforos que son utilizados. La resolución es simple, hay una rutina que modela el comportamiento del usuario, en la cual, él mismo requiere el uso de la impresora y luego avisa cuando la termina de utilizar a través de los métodos requerir_impresora() y liberar_impresora(int).

int requerir_impresora(): espera por impresoras disponibles, si las hay, concede el permiso de imprimir en una de ellas realizando un wait por la impresora correspondiente, luego devuelve el numero de impresora que esta siendo utilizada.

liberar_impresora(int) : recibe un entero que identifica a la impresora a desocupar, luego realiza un signal a la impresora correspondiente.

En esta solución siempre que esté disponible la impresora1, esta será la utilizada. Ya que cuando hay impresoras disponibles primero intenta utilizar la mencionada y si está ocupada, la impresora disponible es la impresora2.

Impresora con prioridades:

Para resolver este problema, ya que los hilos tienen prioridades, se considera:
Según lo hablado en la cátedra, se asumie que las prioridades pueden ser modeladas de manera dinámica, también que en vez de utilizar prioridades diferentes era posible utilizar tres (3) tipos de prioridades como “alta”, “media” y “baja” en este caso , la prioridad “alta” corresponde a 0, la “media” a 1 y la “baja” a 2.

Modelado de la solución:

El usuario y las impresoras son modelados como hilos. Cada impresora espera una señal del usuario que quiere imprimir, el que tenga mayor prioridad esperará por una impresora disponible, si la hay, avisa que la va a utilizar, luego imprime y espera que la impresora envíe la señal de finalización, la cual espera que el usuario la desocupe, para finalmente, avisar que está disponible nuevamente.

Asistente:

El asistente y los estudiantes son modelados como hilos. Se modela una cantidad aleatoria de estudiantes y los tiempos de espera en casi todos los casos también son aleatorios

- Asistente : Si no hay estudiantes esperando, se duerme hasta que un estudiante lo despierta. Luego el comportamiento del asistente es el mismo siempre, avisa que hay una silla disponible y que su oficina esta disponible. Luego de haber dado su consulta, avisa que terminó

- Estudiante : Si hay sillas disponibles para esperar al asistente, el estudiante se sienta y ocupa un lugar, avisando que llegó, luego si el asistente esta durmiendo lo despierta y procede con la consulta. Si ya estaba despierto quiere decir que esta atendiendo a otro estudiante y éste debe esperar a que se desocupe la oficina para ser atendido. Cuando el estudiante realiza su consulta, espera que el asistente finalice su explicación y se va.
  Si no hay sillas disponibles el estudiante se va, y vuelve más tarde.

Best Practices for Operating System Deployment

a)

El desafío principal del negocio se centra en la correcta administración de la implementación del sistema operativo, la cual tuvo que adaptarse debido a la evolución de la industria hacia un modelo de SO-como-servicio. Este nuevo modelo involucra actualizaciones y entregas con nuevas características de forma continua por lo que requirió hacer uso de nuevas prácticas de desarrollo o de una adaptación de las existentes, haciendo foco en que las mismas sean útiles para mejorar la flexibilidad y agilidad del desarrollo. También fue necesario considerar otros cambios en el desarrollo tales como: la toma de decisiones a la hora de si es vital el desarrollo de un nuevo SO o

sólo de nuevas características, lograr un ambiente de testeo permanente, llevar registro de cada cambio realizado en capas de aplicaciones y configuraciones en la plataforma de manera que permita realizar cambios frecuentes en el sistema operativo subyacente sin interrumpir la productividad de los empleados o la experiencia del usuario, verificar constantemente la sincronización con los proveedores y la aprobación de las últimas actualizaciones realizadas.

A grandes rasgos, el conjunto de prácticas realizadas para adaptarse al nuevo modelo cae en tres categorías:

- **Evaluación del sistema operativo:** Es necesario evaluar cada nuevo sistema operativo antes de que esté disponible, cuantificando costos y beneficios, y analizando el impacto que producirá en Intel como en sus clientes. Esta evaluación incluye:
    1. *Testear el Sistema Operativo:* haciendo foco en estabilidad, performance y experiencia del usuario, aspectos que tienen impacto directo en la satisfacción y productividad de los empleados. En el modelo anterior de software los testeos eran anuales, en este nuevo modelo de software-como-servicio las pruebas se vuelven casi continuas.
    2. *Establecer el valor comercial del Sistema Operativo*: Evaluación del sistema operativo en términos de costo de productividad, retorno de la inversión y seguridad.
    3. Considerar la demanda del usuario
- **Preparación de la migración:** Después del desarrollo es necesario verificar que se tiene todas las condiciones requeridas para implementar, integrar y soportar el nuevo sistema operativo. La preparación de la migración incluye:
    1. *Plan de preparación operacional:* Verificando que la entrega del sistema operativo sea apta para actualizaciones o una instalación nueva. Se considera qué cambios deben realizarse para soportar el nuevo sistema durante y después de la migración, teniendo en cuenta que la misma puede tomar semanas o meses, y que los agentes de soporte pueden necesitar acceder al viejo y nuevo sistema operativo.
    2. *Verificar la preparación de la aplicación:* Verificando que todas las aplicaciones funcionen con un nuevo sistema operativo y mitiguen el impacto de aquellas que no son inmediatamente compatibles.
       Para realizar las pruebas se tienen en cuenta diferentes aspectos:
        - Incluir participantes para las pruebas: usuarios aptos, propietarios de las aplicaciones que conocen el entorno de desarrollo.
        - Asignar prioridades a las aplicaciones: Distinguir entre aquellas críticas y ampliamente utilizadas.
        - Descentralizar las pruebas, pero mantener la comunicación, para probar aplicaciones en lugar de construir un gran equipo de pruebas relacionándose con representantes de cada grupo. Los grupos empresariales y los propietarios de las aplicaciones se responsabilizan de probar sus aplicaciones.
        - Colaborar con los equipos de desarrollo de aplicaciones, tal que se pueda confirmar que actualicen sus estándares empresariales para reflejar los cambios en el nuevo sistema operativo.
        - Realizar un seguimiento de qué aplicaciones están certificadas para el nuevo sistema operativo, publicarlo a modo que los empleados puedan validar si las aplicaciones que utilizan están certificadas antes de que decidan migrar.
        - Usar la infraestructura de virtualización existente por seguridad, si falla uno de los escenarios de prueba se prueba en entornos alternativos. Preparar el ecosistema como plataformas, seguridad y modelos de soporte.
    3. *Completar las tareas de ingenierías:* Las mismas caen en varias categorías: imagen del sistema operativo, seguro de calidad, ingeniería de seguridad (la cual incluye la ingeniería de la seguridad y componentes de administración y políticas de la imagen e infraestructura de seguridad), se evalúa la experiencia de usuario de la imagen y hacer mejoras si es necesario.

4. *Iniciar el entrenamiento:* La capacitación abarca tanto a los usuarios finales a través del soporte comunitario en línea, como al personal de soporte realizando una capacitación temprana en línea y publicando artículos de conocimiento. También se le permite el acceso temprano al nuevo sistema operativo ya sea por medio de máquinas virtuales o computadoras portátiles. El personal de soporte que ya esta familiarizado con el nuevo sistema operativo puede suavizar la transición para el resto de los usuarios.
5. *Comunicación regularmente:* Se asignan responsables a cada sector de empleados mediante los cuales se mantendrá la comunicación de información relevante, de esta forma, se crea un flujo medido de comunicaciones.
6. *Integrar herramientas de inteligencia empresarial:* Utilizar herramientas para proporcionar información oportuna y relevante que ayude a rastrear el progreso y colabore con la toma de decisiones durante todo el proceso de entrega del sistema operativo.
7. *Aplicar capas al entorno informático:* Los cambios en el sistema operativo se realizarán con frecuencia, y tenemos que acomodar esos cambios sin interrumpir la productividad de los empleados. El enfoque en capas permite realizar cambios en una capa sin alterar otras.

- **Despliegue:** Cuando la empresa está lista para migrar se realizan prácticas que ayudan a que la operación sea lo más rentable posible e involucre un efecto negativo mínimo en la productividad de los empleados y la experiencia del usuario. Tales como: implementar en fases (Fase 1: implementar para los primeros usuarios, Fase 2: implementar para desarrolladores de aplicaciones adicionales y adoptantes tempranos, Fase 3: implementar la opción estándar de actualización de PC, Fase 4: implementar como problema estándar en dispositivos primarios).

B)

Los aspectos del artículo que se relacionan con la carrera que estamos estudiando son varios, se hace referencia a sistemas operativos, se habla de arquitecturas, usuarios, participantes de diferentes roles dentro del desarrollo de un sistema operativo, pruebas y verificación. Se relaciona principalmente con el proceso de desarrollo de software en el cual vemos como se dividen las tareas de cada una de las partes del proceso, el análisis de requerimientos, su especificación, la verificación, la implementación y el testeo; todas estas etapas se realizan con un enfoque modular adaptándose a un modelo de software como servicio, o en este caso un Sistema Operativo como servicio.

Seguir los procedimientos vistos en la carrera, resultaría en aplicar un procedimiento ideal para producir software. Aunque en la práctica muchas ideas no se respeten en su totalidad, es sumamente importante mantener estoy hábitos de trabajo y respetar el proceso, que gracias a el, obtenemos un software de mejor calidad desde un principio, feedback en cada una de las etapas y también por parte de los usuarios, que son solo tres de los beneficios mas importantes que esto conlleva. Muchos de estos aspectos fueron estudiados en la materia Requerimientos de Sistemas en la que aprendimos diferentes técnicas para capturar, especificar y verificar requerimientos a la hora de atender a la demanda de un sistema de software.