TEC | Tecnológico de Costa Rica

Instituto Tecnológico de Costa Rica
Área Académica de Ingeniería en Computadores
CE 4303 — Principios de Sistemas Operativos

Proyecto 1
Semáforos en el Buzón Circular

Fecha de asignación:	6 de abril, 2020	Fecha de entrega:	27 de Abril, 2021
Grupos:	Parejas	Profesor:	Jason Leitón

1. Objetivo

Aplicar las técnicas de sincronización de los algoritmos básicos, así como abstraer las soluciones de tal manera que sea lo más eficiente posible, con el fin de solucionar problemas de sincronización entre procesos.

2. Atributos a evaluar

- Aprendizaje continuo. Se requiere que el estudiante valore las estrategias y el conocimiento adquirido para alcanzar el objetivo.

- Herramientas de Ingeniería. Se requiere que el estudiante sea capaz de adaptar técnicas, recursos y herramientas modernas para la solución de problemas.

3. Motivación

El problema del consumidor y productor es un clásico de los conflictos en los ámbitos de los sistemas operativos, muchos de los problemas que se encuentran en la informática y otras áreas se puede modelar de la misma manera, y con ello se podría solucionar aplicando las mismas técnicas. La idea fundamental de este proyecto es crear un buffer que sirva de buzón circular y tamaño finito. Este buffer debe inicializarse y crearse en un proceso independiente del consumidor y productor, además existirá un agente externo (Proceso) encargado de liberar todos los recursos de manera elegante y cerrar de manera idónea cada proceso.

4. Descripción

En el proyecto existen 4 tipos de procesos (*Heavy Process*), los cuales son descritos a continuación:

4.1. Inicializador

El inicializador es un programa que tiene como fin aceptar los parámetros iniciales de los recursos. Este proceso será el responsable de crear el buffer compartido (lugar donde se almacenan los mensajes) y de inicializar todas las variables que se utilizaran en la solución del problema (semáforos, banderas de control, contador de consumidores, productores ...). El nombre del buffer, el tamaño del mismo y cualquier otro parámetro deberá ser proporcionados por la línea de comando y haciendo todas las validaciones para que el programa no provoque un error.

4.2. Productor

Todos los productores utilizan el mismo código, que se asocian al buffer creado por el inicializador. Se administra de manera circular con tiempo de espera aleatorios. Este proceso debe recibir por consola el nombre del buffer que se le debe asociar, así como la media en segundos de los tiempos aleatorios siguiendo una distribución exponencial, estos tiempos son los que debe esperar el productor para colocar un mensaje en el buffer. Cabe resaltar que el acceso al buffer debe sincronizarse ya que puede haber n productores produciendo e introduciendo mensajes, además que el buffer es memoria compartida. El usuario puede realizar la cantidad de instancias de este programa como desee, pero en procesos diferentes y compitiendo el mismo buffer. Cada vez que se ejecute un productor (para un mismo buffer) se incrementará el contador global de productores. Los productores repiten un ciclo de fabricación de mensajes hasta que algún tipo de bandera global en memoria compartida indique que el sistema se debe suspender. En este caso, los productores terminan, decrementan el contador de productores vivos, y muestran en pantalla de una manera agradable al usuario el identificador, y características básicas (número de mensajes producidos, acumulado de tiempo de esperado, acumulado de tiempo bloqueado por semáforos, Tiempo en kernel). En caso de que no haya espacio en el buffer, el productor queda suspendido hasta que se libere espacio. No se puede utilizar busy waiting. El formato de los mensajes será creatividad de cada grupo, sin embargo, deberá contener como mínimo el identificador del productor, fecha y hora de creación, número mágico entre 0-6. Cada vez que un mensaje logra ser puesto en el buffer, se deberá mostrar un mensaje describiendo que se ingresó un mensaje, incluyendo el índice de entrada donde de almacenó el mensaje y la cantidad de consumidores y productores vivos en ese momento que ocurre el evento.

Los productores tendrán dos modos de operación: el automático que es que se mencionó anteriormente y el manual, que produce un mensaje cuando el usuario presiona enter. El modo se debe indicar como parámetro cuando se crea el productor, cabe resaltar que este parámetro debe ser por consola, como todo los demás.

4.3. Consumidor

Estos procesos ejecutarán el mismo código siempre, los cuales serán vinculados a un buffer que previamente tuvo que haberse inicializado por el proceso correspondiente. Estos procesos consumen mensajes del buffer con tiempos de espera aleatorios.

Cuando se inicia el proceso se le deberá proveer por consola el nombre del buffer compartido y un parámetro que indique la media en segundos de los tiempos aleatorios, siguiendo una distribución de Poisson. Estos lapsos de tiempo corresponden a los que se deben de esperar para consumir un mensaje, cabe resaltar que los consumidores administrarán el consumo de manera circular y sincronizada para no causar conflicto con otros procesos.

El usuario puede ejecutar las instancias de este programa como lo desee y el diseñador deberá sincronizar el acceso al buffer compartido. Además cuando un consumidor es creado se deberá aumentar el contador global de consumidores activos.

Los consumidores repiten un ciclo de espera aleatoria y consumen mensajes hasta que lean un mensaje especial que indique que el sistema se deba de suspender o cuando al leer un mensaje este incluya una llave (número de 0-6) que sea igual al process id o PID del consumidor módulo 6. En cualquiera de estos casos, el consumidor termina de manera elegante, decrementa el contador de consumidores activos, muestra su identificación, razón de suspensión, y algunas estadísticas básicas (número de mensajes consumidos, acumulado de tiempos esperados, acumulado de tiempo que estuvo bloqueado, tiempo de usuario . . .).

Si no hay mensajes en el buffer, el consumidor queda suspendido hasta que aparezca uno. No se puede utilizar busy waiting.

Cada vez que un mensaje logra ser leído del buffer, se mostrará en pantalla de una manera amigable con el usuario (diferencia de colores y en línea recta, al menos) el índice de la entrada del buffer que estaba, la cantidad de productores y consumidores vivos a momento del evento, así como el contenido del mensaje.

Los consumidores tendrán dos modos de operación: el automático que es que se mencionó anteriormente y el manual que consume un mensaje cuando el usuario presiona enter. El modo se debe indicar como parámetro cuando se crea el consumidor, cabe resaltar que este parámetro debe ser por consola, como todo los demás.

4.4. Finalizador

La función de este programa es cancelar todo el sistema de procesos, enviando mensajes de finalización a cada consumidor vivo usando el buffer, e indicándole a los productores que se detengan de producir por medio de alguna bandera global compartida. Una vez que no haya consumidores ni productores vivos, el buffer deberá ser liberado, deberá mostrar todas las estadísticas, como mínimo debe mostrar: Mensajes totales, mensajes en el buffer, total de productores, total de consumidores, consumidores eliminados por llave, Tiempo esperando total, tiempo bloqueado total, tiempo de usuario total, tiempo de kernel total.

5. Requerimientos técnicos

- Este proyecto se debe realizar en el lenguaje de programación C.
- Debe ser implementado en Linux (no máquina virtual) y se debe proporcionar un makefile.
- No se permite soluciones .ªlambradas".
- Se debe prestar especial atención a los errores de acceso a memoria o utilización de recursos. Es inaceptable el error *segmentation fault* o *core dumped*.
- No se permite busy waiting como se mencionó anteriormente.

6. Documentación- Estilo IEEE-Trans (máximo 5 páginas)

- Introducción: Teoría necesaria, breve descripción del proyecto y qué es lo que se espera en el escrito.
- Ambiente de desarrollo: Todos los detalles de implementación y herramienta durante el desarrollo del proyecto.
- Atributos: Esta sección deben de describirse cuales atributos fueron reforzados durante el desarrollo del proyecto. Para el atributo de de Aprendizaje Continuo se debe mencionar sobre el valor del conocimiento adquirido y las estrategias implementadas para satisfacer las necesidades de aprendizaje. Mientras que para el atributo de Herramientas de Ingeniería se debe mencionar las herramientas ingenieriles que se crearon o que se adaptaron en la solución del problema.
- Detalles del diseño del programa desarrollo, tanto del software como del hardware (en caso de que aplique): Diagramas UML (obligatorio), diagrama de arquitectura (obligatorio), diagrama de funcionalidades (obligatorio), imágenes, descripciones entre otros, todo lo que sea necesario para entender de una mejor manera el diseño y funcionamiento del proyecto.
- Instrucciones de cómo se utiliza el proyecto.
- Tabla de actividades por cada estudiante: bitácora con el total de horas trabajadas.
- Conclusiones
- Sugerencias y recomendaciones.
- Referencias

7. Entregables

- Código fuente con documentación interna.
- Documentación.
- Archivos necesarios para ejecutar el programa.

8. Evaluación

- Sincronización con semáforos entre los 4 procesos 30 %
- Integración 20 %
- Estadísticas 20 %
- Control de eventos y datos mostrados con elegancia 10 %
- Documentación 20 %

9. Fecha de entrega

- 27 de abril 23:55 por tecdigital.

10. Otros aspectos administrativos

- Para la revisión del proyecto se debe de entregar tanto la documentación como la implementación del software.
- No se reciben trabajos después de la hora indicada.
- En la revisión del proyecto pueden estar presentes el coordinador y asistente.
- Es responsabilidad del estudiante proveer los medios para poder revisar la funcionalidad del software, por ejemplo, si no se realiza la interfaz, se debe de proporcionar otro medio para la verificación, de lo contrario la nota será cero en los rubros correspondientes a la funcionalidad faltante.