Programa 2 y Reporte 2

Simular el Proceso por Lotes

Salvador Castañeda Andrade

Sistemas Operativos

Índice

Introducción

El simulador de procesamiento por lotes en Python con la interfaz gráfica tkinter es una herramienta diseñada para simular el procesamiento de múltiples procesos en lotes. Este programa proporciona una representación visual en tiempo real del estado de los procesos en ejecución, en espera y finalizados.

El código fuente del simulador está diseñado de manera modular y es fácilmente comprensible, lo que facilita a los desarrolladores entender y modificar el programa según sea necesario. Con esta herramienta, los usuarios pueden experimentar y comprender mejor los conceptos de procesamiento por lotes y la gestión de procesos en un entorno controlado y visualmente intuitivo.

El simulador utiliza funciones para generar procesos aleatorios con diferentes operaciones matemáticas y tiempos de ejecución. Además, incluye funciones para actualizar la interfaz gráfica en función del estado de los procesos y para generar informes de resultados una vez que los procesos han sido completados. También incorpora funciones para manejar errores durante la ejecución de los procesos (error_process) y para interrumpir un proceso en curso y colocarlo en espera (break_process), lo que añade flexibilidad y robustez al sistema de procesamiento por lotes simulado.

Desarrollo

Función error_process:

```python
def error_process():
    global current_process
    if current_process:
        current_process.result = "Error: Process interrupted"
        finished_processes.append(current_process)
        current_process = None
    update_GUI_process_lbl_txt_box()
```

- **Propósito**: Esta función maneja los errores que pueden ocurrir durante la ejecución de un proceso y marca el proceso como interrumpido.
- **Implementación**:
    - Cuando se llama a esta función, marca el proceso actual como interrumpido cambiando su estado o su resultado a un mensaje de error.
    - Luego, agrega el proceso interrumpido a la lista de procesos finalizados.
- **Importancia**:
    - Permite al sistema manejar de manera adecuada los errores que puedan surgir durante la ejecución de los procesos.
    - Proporciona información útil sobre los procesos que no se completaron correctamente, lo que puede ser útil para análisis y depuración.

Función break_process:

```python
def break_process():
    global current_process, pending_processes
    if current_process:
        n_process = calculate_current_batch_processes()
        current_process.original_position = n_process
        pending_processes.insert(n_process, current_process)
        current_process = None
    update_GUI_process_lbl_txt_box()
```

- **Propósito**: Esta función simula la interrupción de un proceso y lo mueve a la lista de procesos en espera.
- **Implementación**:
    - Cuando se llama a esta función, el proceso actual se mueve de la lista de procesos en ejecución a la lista de procesos en espera.
    - Esta función puede ser útil para simular situaciones en las que un proceso debe ser interrumpido temporalmente y luego continuado más tarde.
- **Importancia**:
    - Brinda flexibilidad al sistema para controlar la ejecución de los procesos, permitiendo la interrupción y reanudación de procesos según sea necesario.
    - Puede ser útil en situaciones donde se requiere priorización dinámica de procesos o ajustes en el flujo de trabajo según las condiciones del sistema.

Conclusiones

El simulador de procesamiento por lotes desarrollado en Python con tkinter es una herramienta eficaz para comprender y visualizar el funcionamiento del procesamiento por lotes. A través de su interfaz gráfica intuitiva, los usuarios pueden observar el estado de los procesos en tiempo real, lo que facilita la comprensión de conceptos complejos relacionados con la gestión de procesos.

La estructura modular del código facilita su comprensión y modificación, lo que permite a los desarrolladores adaptar el simulador a diferentes necesidades y escenarios. Además, la inclusión de funciones como error_process y break_process añade versatilidad al simulador, permitiendo simular situaciones de error y la capacidad de interrumpir procesos en curso.

Bibliografía

Bibliografía básica

Stallings, W. (2011), Operating Systems: Internals and Design Principles. Prentice Hall. 7th Edition.

Silberschatz, A., Galvin, P. B., & Gagne, G. (2010) Ingeniería del Software: Un enfoque práctico, McGraw Hill, México.

Bibliografía complementaria

Peters, James F. & Pedrycz, Witold (2008) Operating System Concepts. John Wiley & Sons Inc. 8th Edition.

Tanenbaum, A. S. (2008) Modern Operating Systems Pearson Educación, México.