Universidad Nacional de Ingeniería

Área de Conocimiento de Tecnología de la Información y Comunicación

Trabajo final de curso, Monitor de recursos del sistema.

Arquitectura de Sistemas Operativos.

Elaborado por:

José Antonio Marín Zelaya.

Carnet: 2021-0056U.

Harvin Gabriel Gutiérrez Guillen.

Carnet: 2021-0127U.

Kevin Geovanni Cerpas González.

Carnet: 2020-0441U.

Docente:

Jacqueline Janette Montes López

Grupo: 4T3-COM-S

22 de octubre de 2025
Managua, Nicaragua.

Tabla de contenido

Resumen

El presente informe técnico describe el desarrollo e implementación de InsanOS, una aplicación en Python diseñada para simular un entorno básico de Sistema Operativo que integra un Monitor de Recursos del Sistema en tiempo real.

El proyecto fue desarrollado para la asignatura Arquitectura de Sistemas Operativos, empleando la librería psutil para interactuar con las métricas del *kernel* (CPU, RAM, Disco, Red) y ttkbootstrap para la interfaz gráfica de usuario (GUI) bajo una estética minimalista.

El monitor proporciona un análisis de la Planificación de CPU mediante una tabla de procesos ordenada y un gráfico histórico, además de reflejar la Gestión de Memoria y E/S. La simulación de este entorno de escritorio demuestra la comprensión de la arquitectura del SO no solo a nivel de *kernel*, sino también en su capa de interacción con el usuario.

Introducción

El estudio de la Arquitectura de Sistemas Operativos requiere la comprensión de cómo el sistema administra sus componentes fundamentales. El monitoreo de recursos es la ventana de diagnóstico que permite observar la implementación de los algoritmos de Planificación de CPU, Gestión de Memoria y Control de E/S del *kernel*.

Este proyecto de Trabajo final de curso para la asignatura de Arquitectura de Sistemas Operativos, simula dicho entorno a través de InsanOS, rudimentario que aloja el monitor, justificando su desarrollo como una herramienta práctica para visualizar la teoría académica.

Objetivos

Objetivo general:

- Desarrollar una aplicación en Python que simule un entorno de Sistema Operativo minimalista (InsanOS) y visualice el uso de los recursos del sistema en tiempo real.

Objetivos específicos:

- Recopilar el uso de CPU, Memoria RAM, Almacenamiento y Red mediante la interacción con los servicios del sistema operativo.

- Implementar tres vistas modulares (Escritorio, Presentación, Monitor) y una interfaz gráfica minimalista basada en la paleta de colores oscuros de la distribución simulada.

- Mostrar una tabla de procesos ordenada por el porcentaje de uso de CPU y un registro histórico de la carga de CPU en tiempo real.

Alcance y Limitaciones

El proyecto se centra en el monitoreo en tiempo real de métricas clave y la simulación del entorno. No incluye la gestión activa de procesos (ej. finalizar tareas) ni la persistencia de datos históricos (*logging* a largo plazo).

Fundamentos de Arquitectura de SO

- **Proceso y Subproceso:** La unidad de trabajo que el SO administra. La tabla de procesos refleja la Tabla de Procesos del *Kernel*.
- **Planificación de la CPU (*CPU Scheduling*):** El monitor muestra el resultado de los algoritmos de planificación a través de las métricas de uso de CPU por proceso y el Tiempo de Ocio del Sistema (*System Idle Process*).
- **Gestión de Memoria:** El porcentaje de RAM utilizada indica la eficacia de los mecanismos de asignación y liberación de páginas.
- **Gestión de E/S:** Las velocidades de Enviar/Recibir (Red) y la actividad del disco son la medida del rendimiento de las operaciones de Entrada/Salida del SO.

Herramientas y Requisitos Técnicos

La elección de las librerías fue crucial para lograr la funcionalidad de monitoreo y el diseño de la interfaz InsanOS:

Requerimiento	Propósito Específico en InsanOS	Implicación en Arquitectura de SO
Python 3.x	Lenguaje de desarrollo.	Plataforma que ejecuta las llamadas a las funciones de las librerías del sistema.
psutil (≥ 5.9.8)	Capa de Recolección de Datos. Permite la interacción directa y multiplataforma con las métricas del kernel del SO para obtener datos de CPU, RAM, Disco, Red y la lista de procesos.	Actúa como el puente que lee las estructuras internas de datos del kernel (tablas de procesos, descriptores de recursos).
ttkbootstrap (≥ 1.10.1)	Capa de Presentación (GUI) y Diseño. Usado para construir la interfaz de las 3 vistas y lograr el diseño minimalista, flat y oscuro de InsanOS.	Define la experiencia de usuario que interactúa con el kernel y presenta sus resultados.
Pillow (≥ 10.0.0)	Manejo de Activos Visuales. Utilizado para cargar, redimensionar y gestionar las imágenes (logo, iconos, fondo de pantalla) dentro de la interfaz.	Soporte a la estética y coherencia visual del entorno InsanOS.

Arquitectura de Vistas (InsanOS)

El proyecto se diseñó modularmente con una arquitectura de tres vistas dentro de la ventana principal, simulando un entorno de escritorio:

- **Vista "Escritorio"** *(Figura 1: Captura de la vista Escritorio*): Actúa como el *launcher* del sistema, mostrando el logo y los iconos (Presentación y Monitor).
- **Vista "Presentación"** *(Figura 2: Captura de la vista Presentacion*): Contiene la información del equipo de trabajo y la asignatura (simula la sección "Acerca de este SO").
- **Vista "Monitor"** *(Figura 3: Captura de la vista Monitor*): El núcleo del trabajo, que presenta las métricas en tiempo real.

Implementación del Tiempo Real (*Real-Time Polling*)

La funcionalidad de tiempo real se basa en un ciclo de actualización constante y automatizada (*polling*):

- Un temporizador interno llama a una función de actualización cada [Mencionar el intervalo de actualización, ej: 1000 milisegundos].
- Esta función consulta a psutil para todas las métricas.
- Los nuevos valores (CPU, RAM, E/S) se procesan.
- La GUI de ttkbootstrap se actualiza: se mueven las barras de progreso, se refrescan los valores de la tabla de procesos y se añade un nuevo punto al gráfico histórico.

Resultados y Análisis de la Simulación

La vista "Monitor" *(Figura 3: Captura de la vista Monitor)* es el principal resultado del proyecto y demuestra la comprensión de la Arquitectura de SO.

Análisis de la Planificación de la CPU

- **Tabla de Procesos (Ordenada por CPU%):** Esta característica muestra directamente la eficiencia del Planificador de CPU. El ordenamiento por CPU% (de mayor a menor) identifica inmediatamente los procesos que consumen más tiempo de ejecución (*time slices*).
- **Proceso Inactivo (*System Idle Process* / PID 0):** Como se observa en el resultado, este proceso es clave. El porcentaje que ocupa (ej. 75.0%) no es un consumo, sino el tiempo ocioso que el SO tiene disponible. Es la métrica más importante para evaluar la capacidad libre del sistema.
- **Gráfico de Historial (Tiempo Real):** El gráfico de las últimas 60 muestras simula un registro de eventos a corto plazo, permitiendo identificar picos de carga repentinos causados por la activación o finalización de procesos.

Análisis de la Gestión de Memoria y E/S

- **Memoria RAM:** Se visualiza el uso total y disponible (ej. 46.6% de 15.78 GB). Esto refleja la asignación de memoria por el SO. Si un proceso tiene un alto MEM%, indica una aplicación intensiva en datos o una posible fuga de memoria.
- **Almacenamiento (Disco Root):** Muestra el porcentaje de espacio utilizado y la actividad de E/S. En un entorno real, una alta actividad de lectura/escritura es un indicador de cuellos de botella en la E/S.
- **Red (Enviar/Recibir):** Las velocidades de tráfico (ej. Enviar: 0.2 KB/s | Recibir: 0.1 KB/s) miden la interacción con el subsistema de red del *kernel*, reflejando el rendimiento de las operaciones de E/S de red.

Coherencia del Entorno InsanOS

El diseño minimalista con colores oscuros logra el objetivo de simular un SO. Además, la barra de tareas que muestra la hora y fecha del sistema (tomadas del SO) refuerza la idea de que InsanOS es un entorno que obtiene servicios fundamentales del *kernel* subyacente.

Conclusión

El proyecto InsanOS cumplió cabalmente su objetivo de simular un entorno de Sistema Operativo y visualizar en tiempo real los mecanismos de la Arquitectura de SO.

La integración exitosa de la librería psutil permitió obtener métricas precisas del *kernel* (CPU, Memoria, E/S), traduciendo conceptos abstractos como la Planificación de CPU (evidente en la tabla de procesos ordenada por consumo) y la Gestión de Recursos a una representación visual.

El diseño de InsanOS mediante ttkbootstrap, con su enfoque minimalista y la presentación del historial gráfico, demostró la capacidad de construir una interfaz funcional que, a su vez, refuerza la comprensión de cómo el SO administra y reporta su propio estado. En suma, el monitor desarrollado no solo es una aplicación funcional, sino una herramienta de simulación que valida y aterriza los fundamentos teóricos de la asignatura.

Anexos

Figura 3: Captura de la vista Escritorio

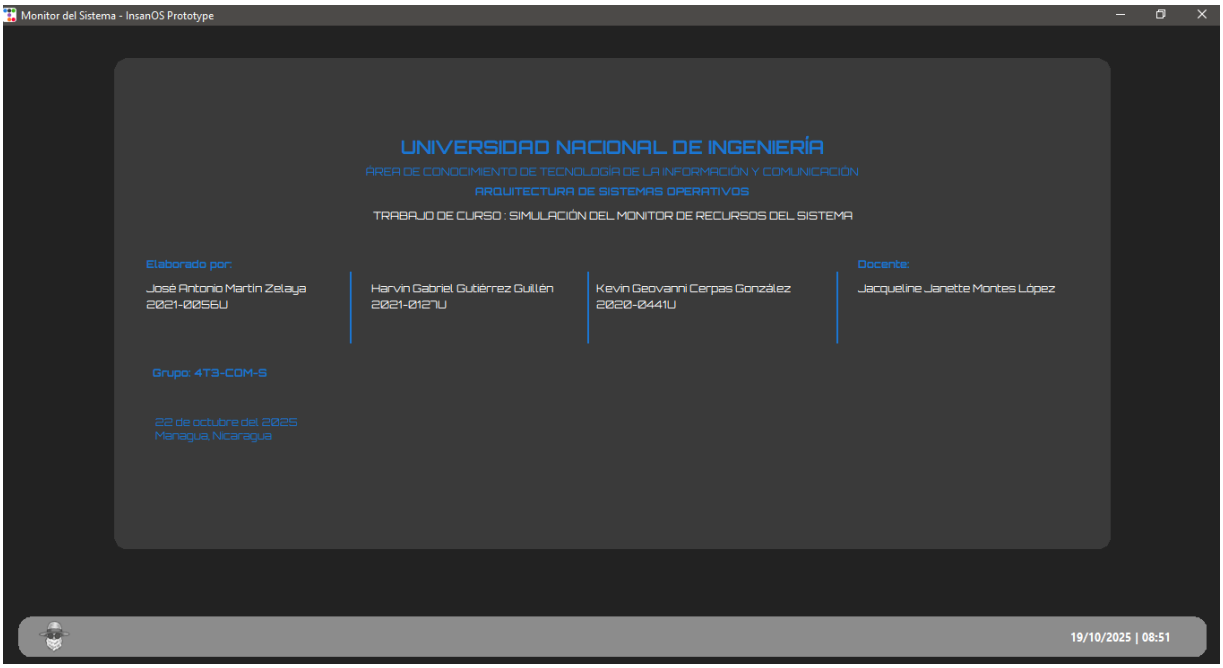

Figura 2: Captura de la vista Presentacion

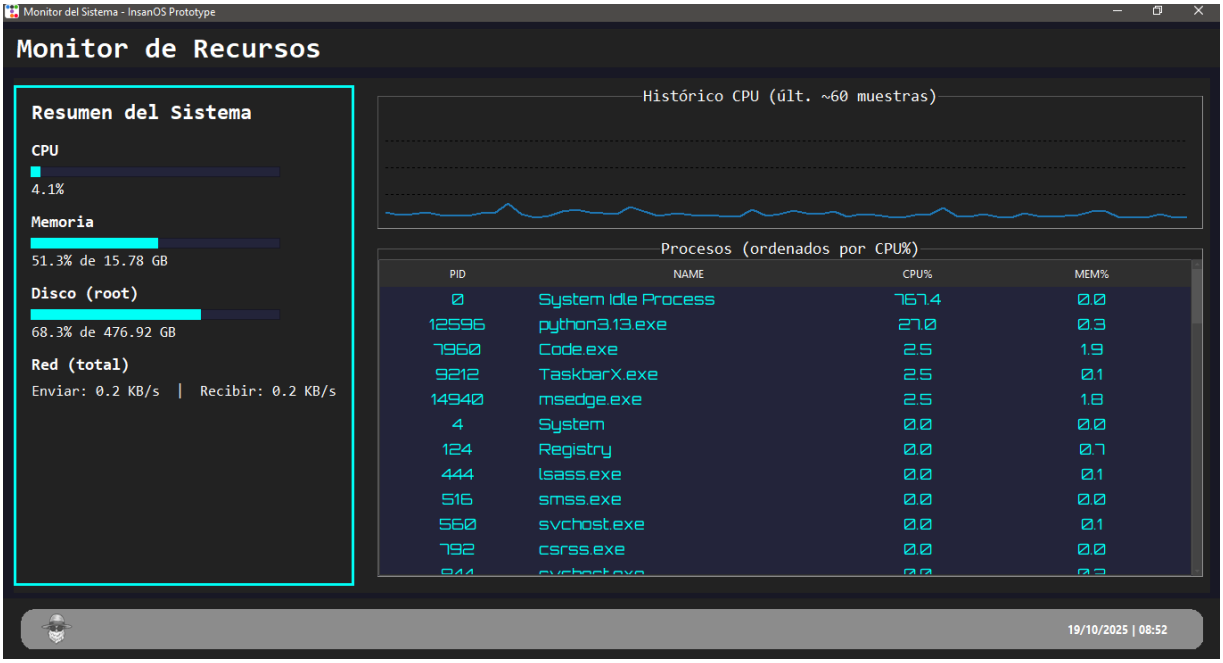

Figura 3: Captura de la vista Monitor

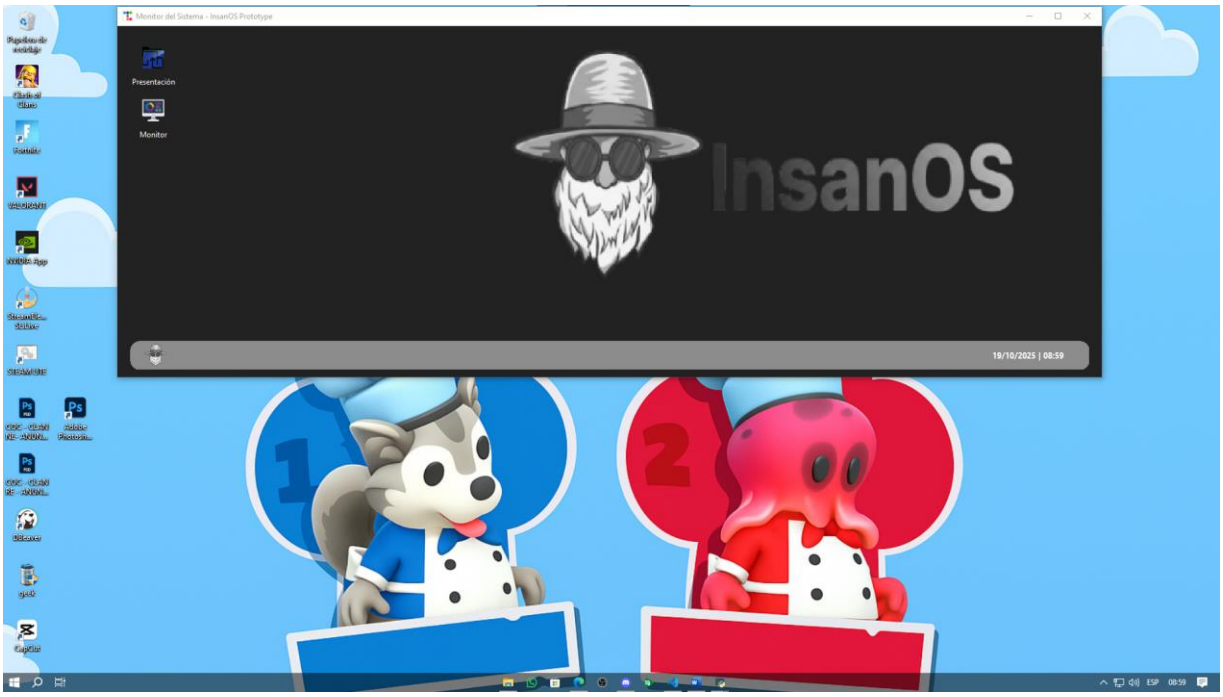

Figura: Captura de la vista del InsanOS siendo ejecutado en Windows10

Bibliografía

Silberschatz, A., Galvin, P. B., & Gagne, G. Fundamentos de Sistemas Operativos (*Operating System Concepts*). John Wiley & Sons.

Tanenbaum, A. S., & Van Steen, M. Sistemas Operativos Modernos (*Modern Operating Systems*). Pearson Educación.

G. M. (*psutil developers*) (*[Año de consulta o última versión]*). psutil documentation: Cross-platform lib for process and system monitoring. Recuperado de https://psutil.readthedocs.io/en/latest/.

ttkbootstrap developers. ttkbootstrap Documentation. Recuperado de https://ttkbootstrap.readthedocs.io/en/latest/.

Lutz, Mark. Learning Python: Powerful Object-Oriented Programming. O'Reilly Media.