Renata Carolina Castro Olmos
S22019640

Luis Gerardo León Salamanca
S22002209

Daniel Gutiérrez Contreras
S22019646

José Alexis González Chávez
S22002189

Othon Lozano Vidal
S22019632

Sistemas Operativos

M.C. Josué Shinoe Munguía Tiburcio

Proyecto Final
Manual Técnico

Veracruz, 20 de junio de 2025

ÍNDICE

Introducción

Propósito del Manual

Este manual técnico documenta la arquitectura, diseño e implementación del Sistema Operativo QueSO (MiniSO), un prototipo funcional desarrollado como proyecto académico para la comprensión de los componentes fundamentales de los sistemas operativos modernos.

Descripción General del Sistema

QueSO es un sistema operativo académico implementado en Java que simula las funcionalidades principales de un sistema operativo real mediante una interfaz gráfica intuitiva. El sistema está organizado en cuatro módulos principales:

- **Módulo de Procesos**: Simulación de algoritmos de planificación de CPU
- **Módulo de Memoria**: Gestión de memoria virtual y algoritmos de asignación
- **Módulo de Archivos**: Sistema de archivos jerárquico virtual
- **Módulo de Seguridad**: Gestión de usuarios, autenticación y control de acceso

Objetivos del Sistema

El sistema QueSO tiene como objetivos principales:

1. **Educativo**: Proporcionar una plataforma visual para comprender conceptos de sistemas operativos
2. **Interactivo**: Permitir la simulación en tiempo real de algoritmos fundamentales
3. **Modular**: Presentar cada componente del SO como un módulo independiente
4. **Extensible**: Facilitar la adición de nuevas funcionalidades y algoritmos

Características Principales

- Interfaz gráfica de usuario desarrollada con Java Swing
- Simulación de algoritmos de planificación (FIFO, Round Robin, Prioridades)
- Gestión de memoria con algoritmos First Fit, Best Fit, Worst Fit y Paginación
- Sistema de archivos jerárquico con operaciones CRUD completas
- Sistema de seguridad con autenticación por contraseñas hash
- Arquitectura modular y extensible

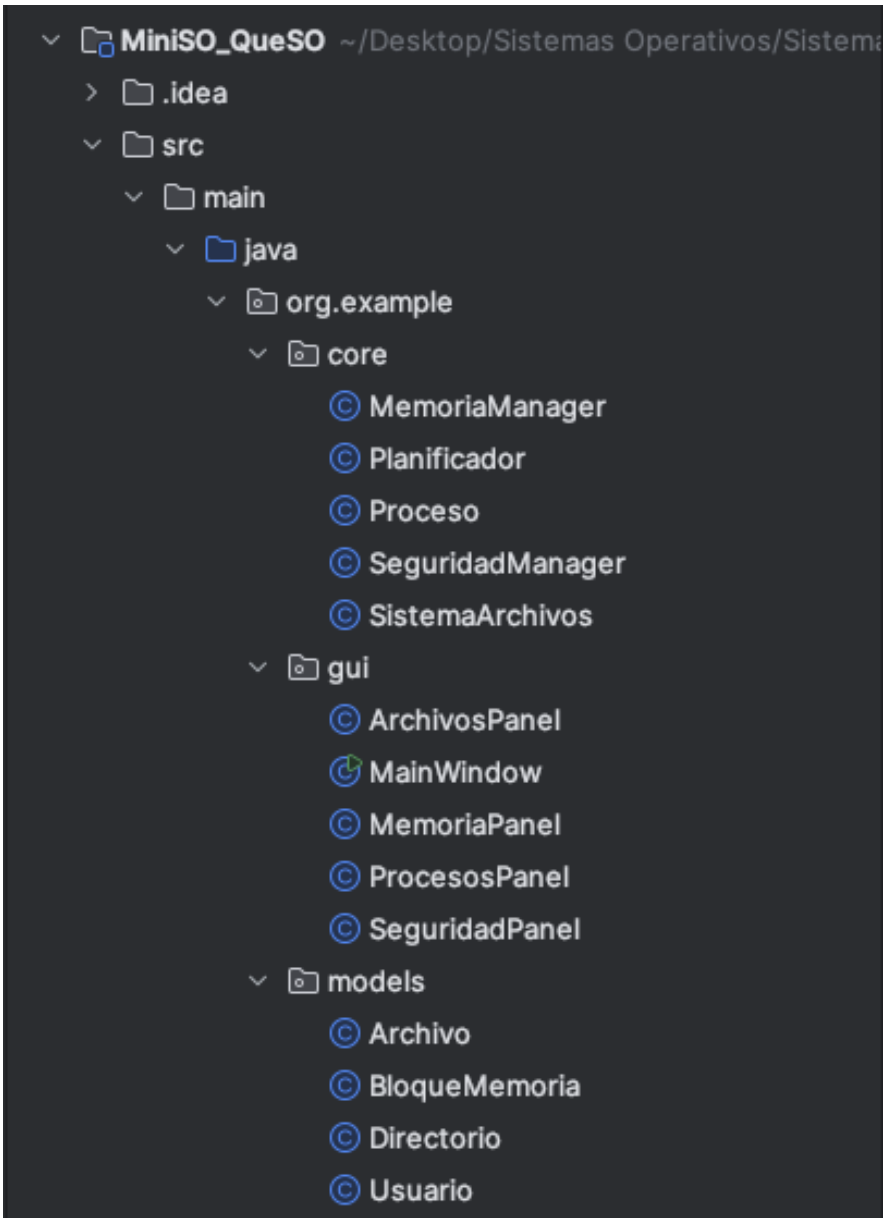
Arquitectura del Sistema

Patrón Arquitectónico

QueSO implementa una arquitectura modular basada en el patrón **MVC (Modelo-Vista-Controlador)** con separación clara de responsabilidades:

- **Vista (GUI)**: Interfaces gráficas implementadas en javax.swing
- **Modelo (Core)**: Lógica de negocio y algoritmos fundamentales
- **Controlador**: Gestores que coordinan la interacción entre vista y modelo

Estructura de Paquetes



Módulo de Procesos

Implementación del Planificador

La clase Planificador implementa tres algoritmos fundamentales de planificación de CPU:

Algoritmos Implementados:

- **FIFO/FCFS (First Come First Served)**: Los procesos se ejecutan en orden de llegada
- **Round Robin**: Asignación cíclica con quantum configurable
- **Planificación por Prioridades**: Procesos con mayor prioridad se ejecutan primero

Gestión de Estados de Procesos

Los procesos pueden encontrarse en los siguientes estados:

- **NUEVO**: Proceso recién creado
- **LISTO**: Preparado para ejecución
- **EJECUTANDO**: Actualmente en CPU
- **TERMINADO**: Ejecución completada

Módulo de Memoria

Gestor de Memoria

La clase MemoriaManager implementa algoritmos de asignación de memoria:

Algoritmos de Asignación:

- **First Fit**: Asigna el primer bloque libre suficiente
- **Best Fit**: Asigna el bloque libre más pequeño que sea suficiente
- **Worst Fit**: Asigna el bloque libre más grande disponible
- **Paginación**: División de memoria en páginas de tamaño fijo

Visualización de Memoria

El sistema proporciona una representación visual en tiempo real del estado de la memoria:

- **Verde**: Memoria libre/disponible
- **Rojo**: Memoria ocupada por procesos
- **Azul**: Memoria reservada del sistema

- **Amarillo**: Fragmentación externa

Módulo de Sistema de Archivos

Estructura Jerárquica

El SistemaArchivos implementa una estructura de árbol similar a sistemas Unix/Linux:
Características del Sistema de Archivos:

- Directorio raíz único ("/")
- Navegación por rutas absolutas y relativas
- Operaciones CRUD (Create, Read, Update, Delete)
- Búsqueda recursiva por nombre y tipo
- Sistema de permisos integrado

Módulo de Seguridad

Gestión de Usuarios

El SeguridadManager implementa un sistema completo de autenticación y autorización:
Tipos de Usuario:

- **ADMINISTRADOR**: Acceso completo al sistema
- **USUARIO_ESTANDAR**: Acceso limitado a recursos
- **INVITADO**: Acceso de solo lectura

Sistema de Autenticación

- Validación de contraseñas mediante hash SHA-256
- Control de acceso basado en roles
- Registro de auditoría de todas las acciones
- Políticas de seguridad configurables

Interfaz Gráfica de Usuario

Ventana Principal

La clase MainWindow implementa la interfaz principal usando JTabbedPane:

- tabbedPane

- procesosPanel
- memoriaPanel
- archivosPanel
- seguridadPanel;

Diseño Visual

- **Dimensiones**: 1000x700 píxeles
- **Tema**: Interfaz moderna con iconos emoji descriptivos
- **Navegación**: Sistema de pestañas intuitivo
- **Responsividad**: Paneles redimensionables con JSplitPane

Conclusión

Logros Alcanzados

El desarrollo del Sistema Operativo QueSO ha cumplido exitosamente con los objetivos planteados:

1. **Implementación Completa**: Se han desarrollado todos los módulos fundamentales de un sistema operativo (procesos, memoria, archivos y seguridad) con funcionalidad completa.
2. **Arquitectura Sólida**: La implementación sigue principios de ingeniería de software con separación clara de responsabilidades, alta cohesión y bajo acoplamiento.
3. **Interfaz Intuitiva**: La GUI desarrollada en Java Swing proporciona una experiencia de usuario moderna e intuitiva que facilita la comprensión de conceptos complejos.
4. **Simulación Realista**: Los algoritmos implementados reflejan fielmente el comportamiento de sistemas operativos reales, proporcionando un entorno de aprendizaje auténtico.

Contribuciones Técnicas

- **Innovaciones en la Implementación**
  - **Visualización en Tiempo Real**: La representación gráfica de la memoria y el estado de los procesos permite una comprensión inmediata de los algoritmos.
  - **Modularidad Extrema**: Cada componente puede funcionar independientemente, facilitando el mantenimiento y la extensión del sistema.
  - **Simulación Integrada**: Todos los módulos interactúan cohesivamente, simulando un sistema operativo real.

- **Calidad del Código**
  - **Documentación Exhaustiva**: Cada clase y método está completamente documentado con JavaDoc
  - **Manejo de Errores**: Implementación robusta de manejo de excepciones
  - **Estándares de Codificación**: Adherencia a convenciones de nomenclatura Java
  - **Arquitectura Escalable**: Diseño que permite fácil adición de nuevas funcionalidades

Impacto Educativo

QueSO demuestra que es posible crear herramientas educativas efectivas que:

1. **Simplifican Conceptos Complejos**: Los algoritmos de sistemas operativos se vuelven tangibles y comprensibles
2. **Fomentan la Experimentación**: Los estudiantes pueden modificar parámetros y observar resultados inmediatos
3. **Proporcionan Experiencia Práctica**: Trabajo directo con conceptos teóricos en un entorno controlado

3.4 Limitaciones y Trabajo Futuro

- **Limitaciones Actuales**
  - **Simulación vs Realidad**: Aunque precisa, la simulación no puede replicar completamente la complejidad de sistemas reales
  - **Escalabilidad**: El sistema está optimizado para propósitos educativos, no para carga de producción
  - **Persistencia**: Los datos no se mantienen entre sesiones (por diseño educativo)

- **Oportunidades de Mejora**
  1. **Algoritmos Adicionales**: Implementación de más algoritmos de planificación y gestión de memoria
  2. **Métricas Avanzadas**: Incorporación de análisis estadísticos más profundos
  3. **Networking**: Simulación de aspectos de red y comunicación entre procesos
  4. **Concurrencia**: Implementación de sincronización y mutex

Referencias Bibliográficas

1. Bloch, J. (2017). *Effective Java* (3rd ed.). Addison-Wesley.
2. Fowler, M. (2002). *Patterns of Enterprise Application Architecture*. Addison-Wesley.
3. Freeman, E., & Robson, E. (2020). *Head First Design Patterns* (2nd ed.). O'Reilly Media.
4. Oracle Corporation. (2024). *Code Conventions for the Java Programming Language*.
5. Oracle Corporation. (2024). *Java Platform, Standard Edition 17 API Specification*.
6. Silberschatz, A., Galvin, P. B., & Gagne, G. (2018). *Operating System Concepts* (10th ed.). John Wiley & Sons.
7. Stallings, W. (2017). *Operating Systems: Internals and Design Principles* (8th ed.). Pearson.
8. Tanenbaum, A. S., & Bos, H. (2014). *Modern Operating Systems* (4th ed.). Pearson.
9. Universidad Veracruzana. (2025). *Material del Curso de Sistemas Operativos*.