ST0257 – Sistemas operativos
Proyecto 2: Librería de gestión de versionamiento de archivos basada en Copy-on-write (COW)

MBA, I.S. José Luis Montoya Pareja
Departamento de Informática y Sistemas
Universidad EAFIT
Medellín, Colombia, Suramérica

RESUMEN
Este proyecto implica el desarrollo de una librería que implementa mecanismos de Copy-on-Write (COW) para gestionar las escrituras de archivos y el versionado. Esta librería proporcionará una alternativa a la escritura de archivos tradicional mediante la creación de nuevas versiones de archivos en cada operación de escritura, lo que garantiza la integridad de los datos y permite el acceso histórico. Las operaciones de lectura siempre accederán a la última versión disponible. La librería también incorporará técnicas de optimización de memoria para minimizar la sobrecarga de almacenamiento y garantizar una utilización eficiente de los recursos. Este proyecto permitirá a los estudiantes aplicar su comprensión de la virtualización de memoria y los conceptos del sistema de archivos para crear una herramienta de gestión de datos práctica y eficiente. Este proyecto también muestra la aplicación de la técnica COW para la duplicación y las instantáneas eficientes de archivos.

PALABRAS CLAVE
Copy-on-Write (COW), Versionado, Sistema de Archivos, Virtualización de Memoria, Integridad de Datos, Gestión de Datos, Desarrollo de Bibliotecas, Optimización de Memoria, Estructuras de Datos, Algoritmos, Eficiencia de Almacenamiento.

CONTEXTO

En diversas aplicaciones, mantener un historial de los cambios en los archivos es crucial para la recuperación de datos, la auditoría y la colaboración. Considere escenarios como:

- Edición de documentos: un editor de documentos colaborativo donde los usuarios necesitan volver a versiones anteriores o realizar un seguimiento de los cambios realizados por diferentes autores.
- Gestión de la configuración: un sistema que almacena archivos de configuración, donde los cambios deben auditarse y revertirse si es necesario.
- Registro de datos: una aplicación que registra datos, donde cada entrada de registro representa una nueva versión del archivo de registro.

- Instantáneas de bases de datos: creación de instantáneas livianas de páginas de bases de datos, para realizar copias de seguridad y recuperación.

Los métodos tradicionales de escritura de archivos sobrescriben los datos existentes, lo que dificulta el acceso a versiones anteriores. La técnica Copy-on-Write (COW) proporciona una solución al crear nuevas versiones de archivos en cada operación de escritura, lo que garantiza la integridad de los datos y permite el acceso histórico. Este proyecto tiene como objetivo crear una librería que simplifique la implementación de escrituras de archivos versionadas basadas en COW, ofreciendo una solución de gestión de datos sólida y eficiente.

OBJETIVOS

- Comprender los principios de Copy-on-Write (COW) y sus aplicaciones.
- Implementar una biblioteca que proporcione escrituras de archivos versionadas utilizando COW.
- Diseñar e implementar operaciones de lectura eficientes que accedan a la última versión del archivo.
- Desarrollar técnicas de optimización de memoria para minimizar el uso de almacenamiento.
- Adquirir experiencia práctica en la construcción de una biblioteca de software reutilizable.
- Aplicar el conocimiento de estructuras de datos y algoritmos para optimizar la gestión de archivos.

ACTIVIDADES

Se propone para el desarrollo de la práctica la siguientes lista de tareas:

- Diseñar la arquitectura de la biblioteca, incluida la API para operaciones de escritura y lectura.
- Definir las estructuras de datos para representar versiones de archivos, bloques de datos y metadatos.
- Implementar funciones básicas de asignación y desasignación de memoria.
- Crear la estructura base para la representación del archivo.
- Implementar la inicialización de la biblioteca.
- Implementar el mecanismo COW para escrituras de archivos, creando nuevas versiones en cada escritura.
- Desarrollar funciones para copiar bloques de datos y actualizar metadatos.
- Implementar la lógica para rastrear el historial de versiones.
- Crear pruebas para la funcionalidad de escritura.

- Crear el sistema de gestión de versiones.
- Implementar operaciones de lectura que siempre accedan a la última versión del archivo.
- Diseñar e implementar mecanismos de recuperación de datos eficientes.
- Optimizar el rendimiento de lectura minimizando la copia de datos.
- Crear pruebas para la funcionalidad de lectura.
- Implementar una función para listar todas las versiones de un archivo.
- Implementar técnicas de optimización de memoria para minimizar la sobrecarga de almacenamiento.
- Desarrollar un mecanismo de recolección de basura para liberar bloques de datos no utilizados.
- Analizar el uso de memoria e identificar posibles optimizaciones.
- Crear pruebas para la optimización de memoria.
- Crear una función que muestre el uso actual de memoria de la biblioteca.
- Realizar pruebas exhaustivas para garantizar la funcionalidad y estabilidad de la biblioteca.
- Escribir documentación detallada para la API y el uso de la biblioteca.
- Realizar análisis de rendimiento para evaluar la eficiencia de la implementación de COW.
- Escribir el informe que resuma los hallazgos y las conclusiones del proyecto.
- Crear un ejemplo de cómo utilizar la biblioteca creada.

CONSIDERACIONES GENERALES

1. El desarrollo de la práctica puede ser individual o en equipos de máximo tres personas.

2. La biblioteca puede ser creada en cualquiera de los siguientes escenarios:
    a. En C++ para que pueda ser usada en C, C++, Java o Python.
    b. En Java para ser usada en otros programas de Java.
    c. En Python para ser usada en otros programas Python.

3. La biblioteca debe tener las siguientes funciones:
    a. create: crear el archivo si no existe.
    b. open: Abrir el archivo si existe.
    c. read: Leer desde la última versión del archivo.
    d. write: Escribir en el archivo que está abierto.
    e. close: Cerrar el archivo.

    Pueden agregar funciones que consideren adicionales para mejorar el funcionamiento de la biblioteca.

4. La entrega de la práctica se realizará entregando los fuentes en archivos (no comprimidos) y el informe por el buzón recepción de trabajos de Eafit Interactiva (cualquier otro medio no será admitido).

5. Se debe informar al profesor a más tardar el 21 de marzo a las 9:00 p.m. los integrantes del equipo vía Teams.

6. El informe final es una presentación que deberá contener una breve descripción de cómo funciona el programa, tablas o gráficos donde se muestre la ejecución de su programa en diferentes máquinas describiendo de cada una el tipo de procesador, cantidad de memoria RAM y sistema operativo que tienen instalado (puede ser en varios) y unas conclusiones que ustedes hagan sobre los datos obtenidos.

7. Cada semana se sacará un espacio de 10 a 20 minutos al inicio de la clase para hablar de la práctica y resolver dudas.

8. Criterios de evaluación (ver Anexo 1)

FECHA DE ENTREGA

Viernes 11 de abril en clase a través de Eafit Interactiva.

SUSTENTACIÓN

Viernes 11 de abril en clase. El mecanismo de sustentación es el siguiente:

1. Al inicio de la clase, se realiza la evaluación de 15 minutos.
2. Cada equipo muestra su desarrollo ejecutando en vivo, dejando ver que el programa se ejecuta con los tres modos solicitados.
3. Debe mostrar cómo solucionó cada uno de los retos y cómo lo relacionaron con conceptos vistos en clase.
4. Mostrar los resultados y explicar las conclusiones.
5. Se realizarán preguntas por parte del docente para validar el entendimiento individual de los conceptos aplicados. Esto significa que, aunque el trabajo es en equipo, la nota del trabajo puede ser diferente para cada uno de acuerdo con la calidad de las respuestas.

Nombre de la asignatura: Sistemas Operativos

Competencia a la que aporta la asignatura: Conocer el sistema operativo del computador para un mejor desarrollo, diseño y ejecución de las aplicaciones y aplicar nuevas soluciones.

Resultado de asignatura evaluado: Biblioteca de gestión de archivos con COW. Evento evaluativo: Proyecto 2 % del evento evaluativo: 25%

Criterios (que tributen al RA de asignatura)	**Cumple con altos estándares (4.5 -5)**	**Cumple a satisfacción (4 -4.4)**	**Cumple parcialmente (3.5-3.9)**	**Incumple parcialmente (2.5-3.4)**	**Incumple totalmente (0 -2.4)**	**Peso asignado al criterio sobre la calificación.**
Análisis de fundamentación para la creación de la biblioteca. Claridad en el concepto	Entiende completamente la necesidad y plantea varias opciones de solución. Utiliza conceptos adecuadamente para la solución.	Entiende completamente la necesidad y plantea una opción de solución.	Omitió un elemento clave para el entendimiento de la necesidad	Omitió varios elementos para el entendimiento de la necesidad.	Demuestra poco o nulo entendimiento del problema.	40%
Diseño de la solución Solución óptima	El diseño tiene en cuenta los conceptos vistos en clase y argumenta la elección de su solución. La solución elegida es la óptima para el problema.	El diseño tiene en cuenta los conceptos vistos en clase y argumenta la elección de su solución.	Aunque se tuvieron en cuenta conceptos vistos en clase, no hubo argumentación correcta en la elección de la solución.	No se tuvieron en cuenta los conceptos vistos en clase.	Demuestra poco o nulo entendimiento de patrones de paralelismo al momento de explicar la solución.	40%

Funcionalidad Calidad de la solución frente a necesidad planteada	El programa funciona correctamente y se entrega de la forma descrita en el documento.	La solución elegida no es la óptima pero el programa funciona correctamente y se entrega de la forma descrita en el documento.	El programa con los conceptos vistos en clase no funciona correctamente o no se entrega de la forma descrita en el documento.	El programa no tiene en cuenta ningún concepto visto en clase y la solución funciona correcta o parcialmente o no se entrega correctamente.	Se entrega la solución parcialmente o no se entrega ninguna solución o no se entrega correctamente.	20%