Introducción En el presente informe se expone sobre el paradigma orientado a objetos aplicado a un simulador de sistema de archivos de un sistema operativo, todo esto bajo el lenguaje de programación de alto nivel "Java". Este escrito está compuesto por la presentación del problema en el que se comentan los procesos llevados a cabo para dar con una solución a este dicho problema, además de dar algunas instrucciones y sus respectivos ejemplos para terminar finalmente con sus resultados. Finalmente, se da una conclusión de este presente trabajo y se entregan los resultados.

Descripción del problema En el enunciado del problema se solicita diseñar un sistema de archivos que comprenda procesos, métodos y reglas que emplee un sistema operativo para la organización y almacenamiento de datos en la memoria no volátil del computador. Dentro de estos archivos se identifica un sistema de archivos que es nuestro primer Objeto de Datos (ODD). Luego, este sistema de archivos está compuesto de una unidad de almacenamiento (ODD Drive), y dentro de este se almacenan Archivos (ODD File) y carpetas (ODD Folder). Todo esto se requiere implementar bajo el paradigma orientado a objetos en conjunto con representar los mencionados ODD anteriormente.

Descripción del Paradigma El paradigma orientado a objetos se basa en la creación de objetos que representan entidades del mundo real o conceptos abstractos. Estos objetos tienen atributos (variables) que representan su estado y métodos (funciones) que definen su comportamiento. Los objetos interactúan entre sí mediante mensajes, lo que permite modelar las relaciones y el flujo de información entre ellos.

En el paradigma orientado a objetos, se busca encapsular los datos y los métodos relacionados en clases. Las clases son plantillas que definen la estructura y el comportamiento de los objetos. Los objetos se crean a partir de estas clases mediante un proceso llamado instanciación.

Análisis del Problema El problema requiere el uso de clases y objetos para resolverlo. Se busca consultar archivos e información del sistema de archivos lo más rápido posible, por lo que todos los archivos y carpetas estarán organizados de manera jerárquica en un árbol. Se deben implementar las siguientes operaciones bajo el uso estricto de los ODD:

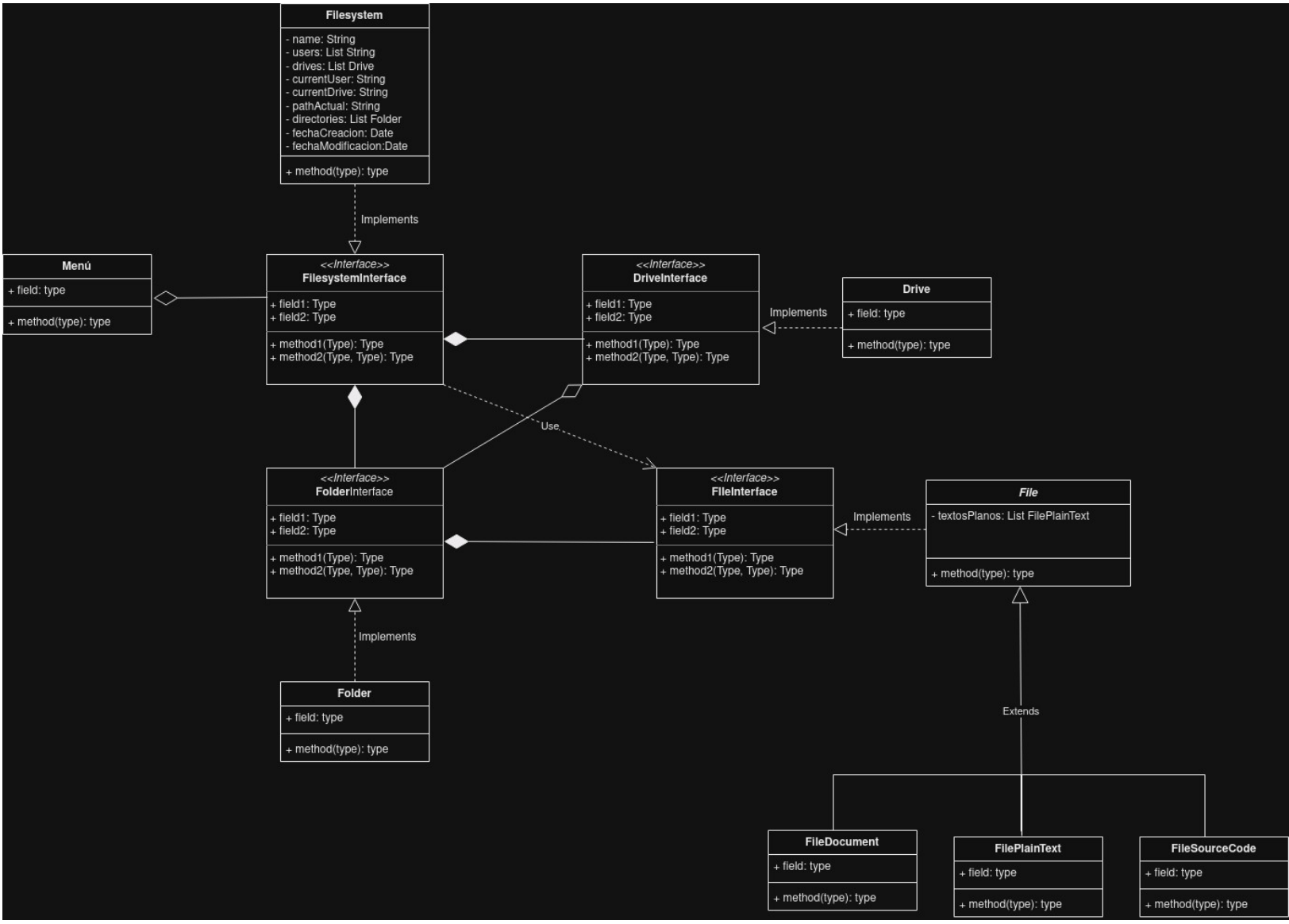
1. Soporte para múltiples unidades físicas y lógicas.
2. Soporte para múltiples usuarios.
3. Permisos por archivo, carpeta y por usuario.
4. Operaciones del tipo:
    - Compartir (share)
    - Copiar (copy)
    - Mover (move)
    - Eliminar archivos (delete)
    - Añadir archivos (addFile)
    - Crear carpetas y subcarpetas, también llamados directorios (createDirectory)
    - Eliminar carpetas (deleteDirectory)
    - Cambiar de directorio (changeDirectory)
    - Renombrar (rename)
    - Buscar dentro del contenido de los archivos (search)
    - Listar archivos con parámetros tipo /s /a (list)

    - Formatear una unidad (format)
    - Encriptar (encrypt)
    - Desencriptar (decrypt)
5. Registro de fechas de creación y última modificación de carpetas y archivos.
6. Contará con una papelera donde los elementos eliminados quedan alojados de forma temporal.
7. Vaciar papelera de reciclaje.
8. Restaurar elementos de papelera de reciclaje.

Diseño de la solución Para el desarrollo de esta solución se hará uso de las siguientes clases:

1. Clase FileSystem
    - Atributos: name, users, drives, currentUser, currentDrive, path, trash, creationDateSystem.
    - Métodos: getters y setters para los atributos.
2. Clase Drive
    - Atributos: name, letter, folders, capacity.
    - Métodos: getters y setters para los atributos.
3. Clase File
    - Atributos: name, content, path.
    - Métodos: getters y setters para los atributos.
4. Clase Folder
    - Atributos: name, password, decryptFn, userCreator, path.

- Métodos: getters y setters para los atributos.



Resultados y Autoevaluación Los resultados fueron exitosos y se lograron implementar los requerimientos funcionales según lo especificado en el informe.

Conclusiones Al utilizar el paradigma orientado a objetos en comparación con otros paradigmas, como el paradigma lógico, se logra una mayor estructura y organización del código. El uso de clases y objetos permite modelar de manera más clara y concisa los conceptos y entidades del sistema de archivos. Además, se facilita la reutilización de código y se promueve el encapsulamiento y la modularidad. Sin embargo, la programación orientada a objetos puede ser más compleja de entender y requerir un mayor esfuerzo en su implementación en comparación con otros paradigmas.