UNIVERSIDADE DE FORTALEZA
CENTRO DE CIÊNCIAS TECNOLÓGICAS
CURSO: CIÊNCIA DA COMPUTAÇÃO

Simulador de Sistema de Arquivos

Dupla:

- *Luca Solon Fernandes da Cunha – 190486*
- *Marcos Antonio Felix de Oliveira Filho - 1819449*

Metodologia

O simulador foi desenvolvido em linguagem de programação Java, utilizando a API de manipulação de arquivos do Java NIO. O sistema simula as operações típicas de um sistema de arquivos, tais como copiar, remover, renomear, criar diretórios e listar arquivos. Para garantir a integridade e recuperação de operações em caso de falha, foi implementado um mecanismo de **Journaling**, que registrará todas as operações realizadas.

Cada operação é processada conforme o comando digitado pelo usuário, e o simulador exibirá o resultado na tela. Além disso, o **Journaling** permite o registro das ações em um arquivo de log, o que facilita a análise e recuperação de operações no sistema de arquivos.

Parte 1: Introdução ao Sistema de Arquivos com Journaling

Descrição do Sistema de Arquivos

Um **sistema de arquivos** é um componente essencial de qualquer sistema operacional (SO) que gerencia a organização, armazenamento e recuperação de dados em dispositivos de armazenamento. Ele fornece uma maneira estruturada para os dados serem organizados e acessados, e também garante que os dados possam ser lidos, gravados e manipulados de forma eficiente. O sistema de arquivos pode incluir recursos como permissão de acesso, organização hierárquica de arquivos e proteção contra falhas.

Journaling em Sistemas de Arquivos

O **Journaling** é uma técnica usada para garantir a integridade de um sistema de arquivos. Ele mantém um registro sequencial das operações realizadas, chamado de "journal". O principal objetivo do journaling é garantir que, em caso de falha (como uma queda de energia), o sistema possa ser recuperado sem perda de dados, ou pelo menos, com mínima perda. Existem diferentes tipos de journaling:

- **Write-ahead logging (WAL)**: As operações são registradas no journal antes de serem executadas.
- **Log-structured**: Toda a estrutura do sistema de arquivos é tratada como um log, sendo escrita sequencialmente em vez de em locais aleatórios.
- **Metadata Journaling**: Apenas a mudança nos metadados do sistema de arquivos é registrada no journal.

Parte 2: Arquitetura do Simulador

Estrutura de Dados

Para representar as operações de manipulação de arquivos e diretórios, foram utilizadas classes Java. A estrutura básica do simulador é composta por:

- **File**: Representa um arquivo no sistema de arquivos.
- **Directory**: Representa um diretório que contém arquivos ou outros diretórios.
- **FileSystemSimulator**: Controla as operações do sistema de arquivos, como copiar, mover, excluir e listar arquivos e diretórios.
- **Journal**: Gerencia o log de operações, armazenando todas as ações realizadas no sistema para fins de recuperação.

Journaling

O **Journaling** foi implementado de forma simples, utilizando a classe BufferedWriter para gravar cada operação no arquivo de log. Cada operação registrada no journal inclui o comando executado e seus parâmetros. O arquivo de journal foi armazenado no disco local, e todas as operações executadas pelo simulador (como copy, rm, mkdir, rmdir, rename, etc.) foram registradas.

A estrutura do log é simples, com cada linha representando uma operação, por exemplo:

copy arquivo1.txt arquivo2.txt

rm arquivo1.txt

mkdir /home/usuario

Parte 3: Implementação em Java

Classe FileSystemSimulator

A classe **FileSystemSimulator** é a classe principal do simulador e implementa as operações do sistema de arquivos. A seguir estão as implementações dos métodos principais:

- **copyArchive(String beginning, String ending)**: Copia um arquivo de beginning para ending. Registra a operação no journal.
- **deleteArchive(String pathway)**: Exclui um arquivo especificado por pathway e registra a operação no journal.
- **createDirectory(String pathway)**: Cria um diretório no caminho fornecido e registra a operação no journal.
- **deleteDirectory(String pathway)**: Exclui um diretório e todo o seu conteúdo, registrando a operação no journal.
- **rename(String pathway, String newPathway)**: Renomeia um arquivo ou diretório e registra a operação no journal.
- **listDirectory(String pathway)**: Lista os arquivos e diretórios dentro de um diretório especificado.

Classe File

A classe **File** representa um arquivo no sistema de arquivos. Ela armazena informações como o nome do arquivo e seu caminho. Métodos adicionais podem ser implementados para modificar o conteúdo do arquivo, mas, no contexto deste simulador, a classe apenas representa a entidade "arquivo".

Classe Directory

A classe **Directory** representa um diretório e mantém uma lista de arquivos e subdiretórios. A classe oferece métodos para adicionar, remover e listar arquivos e subdiretórios dentro do diretório.

Classe Journal

A classe **Journal** gerencia o log de operações. Ela mantém um arquivo no qual todas as operações executadas no sistema de arquivos são registradas. O método **Journaling(String operation)** é responsável por adicionar uma nova entrada no journal a cada operação realizada.

Resultados Esperados

Espera-se que o simulador forneça uma visão prática de como um sistema de arquivos funciona, incluindo a manipulação de arquivos e diretórios, a implementação do **Journaling** para garantir a integridade do sistema e a comparação entre operações realizadas de forma serial. O uso de operações de **Journaling** permite simular um comportamento resiliente do sistema, onde é possível verificar o estado do sistema em caso de falha e até mesmo recuperar operações.

O simulador, além de ajudar a entender o funcionamento de um sistema de arquivos, permite testar e analisar o desempenho das operações em uma implementação controlada. Ao executar o simulador, é possível observar o fluxo das operações e garantir que todas as alterações no sistema de arquivos sejam registradas adequadamente para fins de recuperação e análise.