Matheus V. Costa - 2216973
Rafael Roseira Machado - 2399091
Christopher Zai - 2134470

Implementação de Sistema de Arquivos EXT2

Relatório técnico do projeto solicitado pelo professor Rodrigo Campiolo na disciplina de Sistemas Operacionais do Bacharelado em Ciência da Computação da Universidade Tecnológica Federal do Paraná.

Universidade Tecnológica Federal do Paraná – UTFPR
Departamento Acadêmico de Computação – DACOM
Bacharelado em Ciência da Computação – BCC

Campo Mourão
Junho / 2025

Resumo

O projeto EXT2 foi desenvolvido com o objetivo de manipular imagens do sistema de arquivos EXT2, permitindo operações como listagem de diretórios (ls), leitura de arquivos, navegação (cd) e inspeção de metadados. A ferramenta utiliza estruturas de dados específicas para interpretar blocos, inodes, superbloco e tabelas de blocos do EXT2.

Palavras-chave: EXT2. Sistema de Arquivos. Manipulação de Imagens. C/C++. Linux.

Sumário

1 Introdução

O sistema de arquivos EXT2 (Second Extended File System) foi desenvolvido para sistemas operacionais baseados em Linux, com foco em estabilidade, eficiência e suporte a grandes volumes de dados. Diferente de sistemas baseados em tabelas de alocação, o EXT2 utiliza inodes para armazenar metadados sobre arquivos e diretórios, incluindo permissões, timestamps e endereços de blocos. Suporta arquivos de até 2 TB e permite a criação de milhares de subdiretórios em uma única pasta. Este projeto busca explorar a estrutura interna do EXT2 por meio da leitura e interpretação direta de imagens binárias, evidenciando seu funcionamento através de operações como listagem de diretórios, navegação e análise de metadados.

2 Objetivos

Este projeto teve como objetivo desenvolver um programa modular, capaz de interpretar e manipular imagens de sistemas de arquivos EXT2, possibilitando a exploração de dados de forma semelhante a um shell Unix. Para isso, foi necessário compreender a organização interna do EXT2, incluindo seus metadados, gerenciamento de blocos e o uso de inodes, e implementar comandos fundamentais para a navegação e análise do sistema de arquivos.

O projeto EXT2 foi desenvolvido com foco nas seguintes funcionalidades:

- Leitura e interpretação do Superbloco e estruturas auxiliares, extraindo informações essenciais sobre a configuração do sistema de arquivos, como número de inodes, blocos, grupos e estado geral da partição.
- Navegação na hierarquia de diretórios, com suporte à listagem de entradas *(ls)* e mudança de diretório *(cd)*, baseando-se na resolução de caminhos via análise das entradas de diretório e dos ponteiros armazenados nos inodes.
- Leitura de arquivos e inspeção de metadados, permitindo visualizar o conteúdo armazenado nos blocos referenciados pelos inodes e acessar informações como permissões, tamanhos e timestamps.
- Validação em ambiente Linux, garantindo a consistência e confiabilidade das operações sobre a imagem EXT2 e assegurando que a ferramenta respeite as especificações do sistema de arquivos.

3 Fundamentação

O sistema de arquivos **EXT2 (Second Extended File System)** foi introduzido no início da década de 1990 como parte do ecossistema Linux, com o propósito de superar as limitações do sistema EXT original. Sua arquitetura modular e eficiente proporcionou maior escalabilidade, desempenho e confiabilidade, consolidando-o como uma das soluções de armazenamento mais utilizadas em sistemas Unix. Apesar do surgimento de versões mais recentes, como o EXT3 e EXT4, o EXT2 continua relevante, especialmente em contextos onde a ausência de journaling é desejável, como em sistemas embarcados e dispositivos de armazenamento com recursos limitados.

A estrutura do EXT2 é organizada de forma hierárquica, segmentando dados e metadados em componentes distintos — incluindo o **Superbloco**, **descritores de grupo**, **bitmaps**, **tabelas de inodes** e **blocos de dados**. Essa separação visa otimizar o desempenho em operações de leitura e escrita, reduzir a fragmentação interna e facilitar mecanismos de recuperação em caso de falhas. A figura a seguir ilustra a estrutura do EXT2.

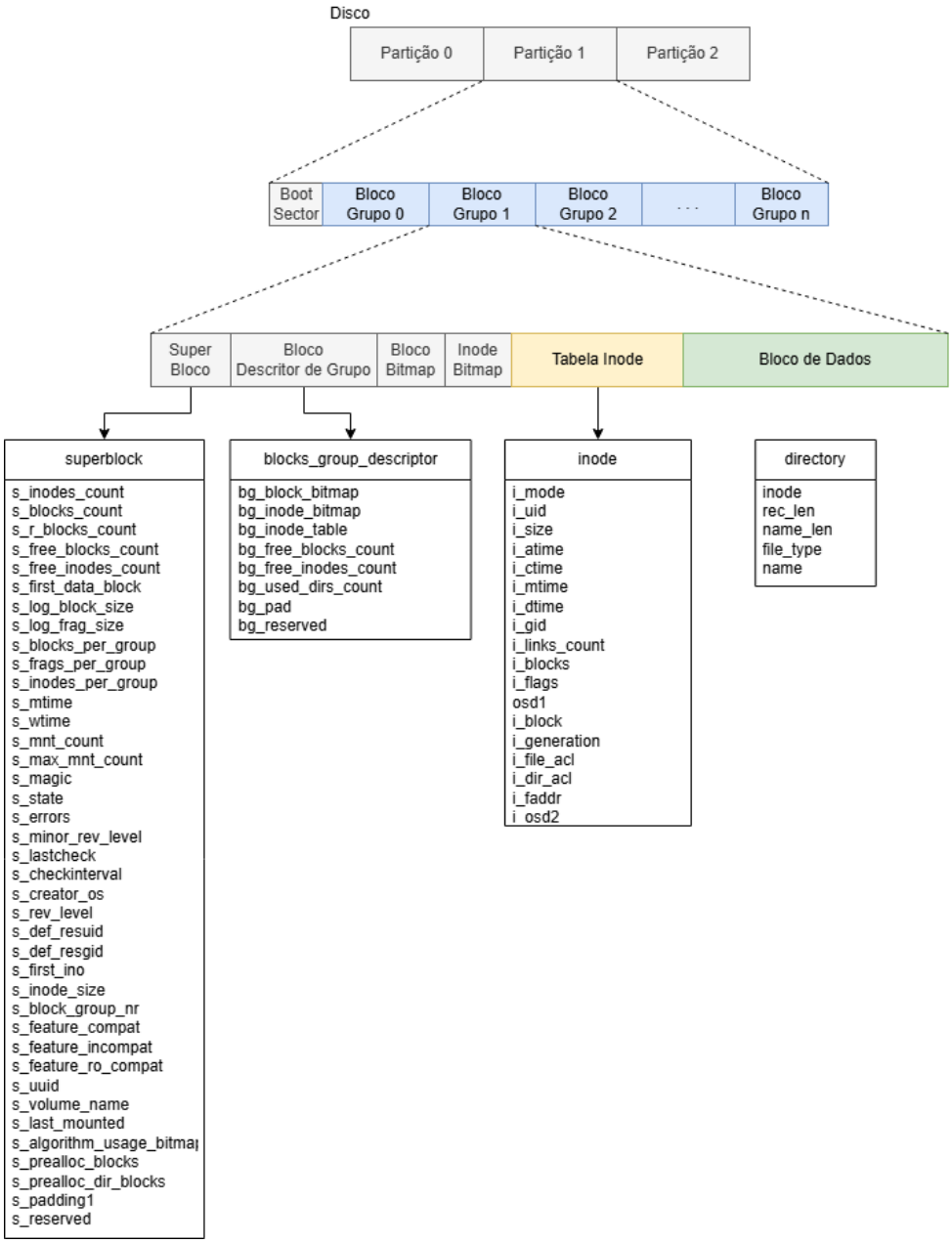

Figura 1 – Estrutura EXT2

Neste projeto, essas estruturas foram analisadas e manipuladas por meio do desenvolvimento de uma ferramenta capaz de operar diretamente sobre imagens binárias EXT2. A aplicação implementa comandos semelhantes aos de um shell Unix, como *ls*, *cd* e leitura de arquivos, possibilitando uma exploração prática das relações entre inodes, diretórios e blocos de dados.

Essa abordagem fornece uma base técnica sólida para o estudo de sistemas de arquivos em baixo nível, com aplicações em recuperação de dados, análise forense e desenvolvimento de ferramentas para ambientes embarcados, nos quais o controle direto sobre o armazenamento é essencial.

4 Materiais

- **Ambiente de Desenvolvimento**
  - **Sistemas Operacionais Testados:**
    - Debian 12.7.0
    - Ubuntu 24.04.01
  - **Linguagem:** C
  - **Ferramentas:**
    - GCC 14.2
    - VirtualBox 7.1.2
    - Git 2.48.1

5 Estrutura do Sistema de Arquivos EXT2

5.1 Superbloco

O Superbloco contém os metadados globais da partição, incluindo o total de blocos e inodes disponíveis, contadores de blocos livres e alocados, timestamps de montagem e modificação, versão do sistema de arquivos, bem como o identificador do sistema operacional que o formatou. Para manter a integridade do sistema, cópias redundantes do Superbloco são distribuídas em grupos de blocos ao longo da partição, garantindo tolerância a falhas em caso de corrupção de dados. A seguir veremos a estrutura in-code do superbloco.

```
typedef struct superblock
{
  uint32_t s_inodes_count;          /* Inodes count */
  uint32_t s_blocks_count; /* Blocks count */
  uint32_t s_r_blocks_count; /* Reserved blocks count */
  uint32_t s_free_blocks_count; /* Free blocks count */
  uint32_t s_free_inodes_count; /* Free inodes count */
  uint32_t s_first_data_block; /* First Data Block */
  uint32_t s_log_block_size; /* Block size */
  int32_t s_log_frag_size; /* Fragment size */       /*** __s32 ***/
  uint32_t s_blocks_per_group; /* # Blocks per group */
  uint32_t s_frags_per_group; /* # Fragments per group */
  uint32_t s_inodes_per_group; /* # Inodes per group */
  uint32_t s_mtime; /* Mount time */
  uint32_t s_wtime; /* Write time */
  uint16_t s_mnt_count; /* Mount count */
  int16_t   s_max_mnt_count; /* Maximal mount count */ /*** __s16 ***/
  uint16_t s_magic; /* Magic signature */
  uint16_t s_state; /* File system state */
  uint16_t s_errors; /* Behaviour when detecting errors */
  uint16_t s_minor_rev_level; /* minor revision level */
  uint32_t s_lastcheck; /* time of last check */
  uint32_t s_checkinterval; /* max. time between checks */
  uint32_t s_creator_os; /* OS */
  uint32_t s_rev_level; /* Revision level */
  uint16_t s_def_resuid; /* Default uid for reserved blocks */
  uint16_t s_def_resgid; /* Default gid for reserved blocks */
  uint32_t s_first_ino; /* First non-reserved inode */
  uint16_t s_inode_size; /* size of inode structure */
  uint16_t s_block_group_nr; /* block group # of this superblock */
  uint32_t s_feature_compat; /* compatible feature set */
  uint32_t s_feature_incompat; /* incompatible feature set */
  uint32_t s_feature_ro_compat; /* readonly-compatible feature set */
  uint8_t s_uuid[16]; /* 128-bit uuid for volume */
  char s_volume_name[16]; /* volume name */
  char s_last_mounted[64]; /* directory where last mounted */
  uint32_t s_algorithm_usage_bitmap; /* For compression */
  uint8_t s_prealloc_blocks; /* Nr of blocks to try to preallocate*/
  uint8_t s_prealloc_dir_blocks; /* Nr to preallocate for dirs */
  uint16_t s_padding1;
  uint32_t s_reserved[204]; /* Padding to the end of the block */
} Superblock;
```

Listing 1 – Estrutura do superbloco

5.2 Grupos de Blocos

Para melhorar a escalabilidade e a eficiência no acesso aos dados, o EXT2 organiza os blocos em unidades chamadas grupos de blocos (block groups). Essa abordagem visa manter metadados e dados próximos fisicamente no disco, reduzindo o tempo de acesso e a fragmentação. Cada grupo contém áreas reservadas para o bitmap de blocos, bitmap de inodes e a tabela de inodes, além dos próprios blocos de dados. O gerenciamento de cada grupo é realizado por uma estrutura de descritor de grupo (group descriptor), armazenada em uma tabela acessível via o Superbloco, que aponta para as localizações das estruturas internas de cada grupo.A seguir veremos a estrutura in-code do superbloco.

```
typedef struct blocks_group_descriptor {
    uint32_t    bg_block_bitmap;        /* Blocks bitmap block */
    uint32_t    bg_inode_bitmap;        /* Inodes bitmap block */
    uint32_t    bg_inode_table;         /* Inodes table block */
    uint16_t    bg_free_blocks_count;   /* Free blocks count */
    uint16_t    bg_free_inodes_count;   /* Free inodes count */
    uint16_t    bg_used_dirs_count;     /* Directories count */
    uint16_t    bg_pad;
    uint32_t    bg_reserved[3];
} BlocksGroupDescriptor;
```

Listing 2 – Estrutura do blocks group descriptor

5.3 Inode

O inode (índice de nó) é uma estrutura fundamental no EXT2, responsável por representar arquivos e diretórios no nível do sistema de arquivos. Cada inode armazena atributos como permissões de acesso, identificadores de usuário e grupo, tamanho do arquivo, timestamps (criação, modificação, acesso), além dos ponteiros que indicam onde os dados do arquivo estão fisicamente armazenados. O campo *i_block*, composto por um array de 15 ponteiros de 32 bits, implementa um esquema hierárquico de endereçamento: os 12 primeiros apontam diretamente para blocos de dados, o 13º é um ponteiro indireto simples, o 14º é um ponteiro indireto duplo, e o 15º é um ponteiro indireto triplo. Esse mecanismo permite ao EXT2 suportar arquivos de tamanho variável com alta eficiência.

```
typedef struct inode
{
  uint16_t i_mode;        /* File mode */
  uint16_t i_uid;         /* Low 16 bits of Owner Uid */
  uint32_t i_size;        /* Size in bytes */
  uint32_t i_atime;       /* Access time */
  uint32_t i_ctime;       /* Creation time */
  uint32_t i_mtime;       /* Modification time */
  uint32_t i_dtime;       /* Deletion Time */
  uint16_t i_gid;         /* Low 16 bits of Group Id */
  uint16_t i_links_count; /* Links count */
  uint32_t i_blocks;      /* Blocks count IN DISK SECTORS*/
  uint32_t i_flags;       /* File flags */
  uint32_t osd1;          /* OS dependent 1 */
  uint32_t i_block[15];   /* Pointers to blocks */
  uint32_t i_generation;  /* File version (for NFS) */
  uint32_t i_file_acl;    /* File ACL */
  uint32_t i_dir_acl;     /* Directory ACL */
  uint32_t i_faddr;       /* Fragment address */
  uint32_t i_osd2[3];
} Inode;
```

Listing 3 – Inode

5.4 Bitmap

Os bitmaps são estruturas compactas utilizadas para o gerenciamento de alocação de recursos dentro dos grupos de blocos. No contexto do EXT2, existem dois tipos principais: o bitmap de blocos e o bitmap de inodes. Cada bit representa o estado de um recurso — se está livre (0) ou alocado (1). Essas estruturas permitem ao sistema localizar rapidamente blocos ou inodes disponíveis durante operações de escrita e criação, otimizando a alocação de espaço de forma eficiente e minimizando a sobrecarga de busca.

6 Implementação

Com base nos conceitos apresentados sobre o sistema de arquivos EXT2, sua estrutura pode ser representada conforme ilustrado na Figura 1. Nela, observa-se a organização típica de cada grupo de blocos, no qual estão dispostas as principais estruturas do sistema: o Superbloco, a tabela de descritores de grupo, os bitmaps de blocos e de inodes, a tabela de inodes e, por fim, os próprios blocos de dados, responsáveis pelo armazenamento das informações do sistema de arquivos.

O Projeto está modularizado em pastas da seguinte maneira:

```
EXT2_FileManager/
├── imagem/ ..................................... Contém a imagem EXT2 para testes
│   └── EXT2_TESTE.img ..................... Imagem de sistema de arquivos EXT2
├── src/ ..................................... Código-fonte principal do programa
│   ├── blocks-group-descriptor/ .. Manipulação e estrutura dos grupos de blocos
│   │   ├── blocks-group-descriptor.cpp
│   │   └── blocks-group-descriptor.hpp
│   ├── directory/ ......................... Manipulação e estrutura de diretórios
│   │   ├── directory.cpp
│   │   └── directory.hpp
│   ├── error/ ............................ Exibição e controle de mensagens de erro
│   │   └── error.hpp
│   ├── utils/ ..................................... Operações básicas utilitárias.
│   │   ├── utils.cpp
│   │   └── utils.hpp
│   ├── inode/ ................................ Manipulação e estrutura dos inodes
│   │   ├── inode.cpp
│   │   └── inode.hpp
│   ├── main/ ................................ Shell e função principal do programa
│   │   ├── main.cpp
│   │   ├── shell.cpp
│   │   └── shell.hpp
│   ├── manager/ ................... Controlador que gerencia o acesso às estruturas
│   │   ├── file-system-manager.cpp
│   │   └── file-system-manager.hpp
│   └── superblock/ ....................... Manipulação e estrutura do superbloco
│       ├── superblock.cpp
│       └── superblock.hpp
├── .gitignore ....................... Define arquivos a serem ignorados pelo Git
├── Makefile ..................................... Script de compilação do projeto
└── README.md .................. Arquivo com instruções e informações do projeto
```

Para fins de demonstração das funcionalidades implementadas, será utilizada a imagem de sistema de arquivos nomeada ***EXT2_TESTE.img***. A Figura 2 ilustra a disposição interna dos arquivos contidos nessa imagem, permitindo um melhor acompanhamento da estrutura e das operações realizadas durante a execução da ferramenta.

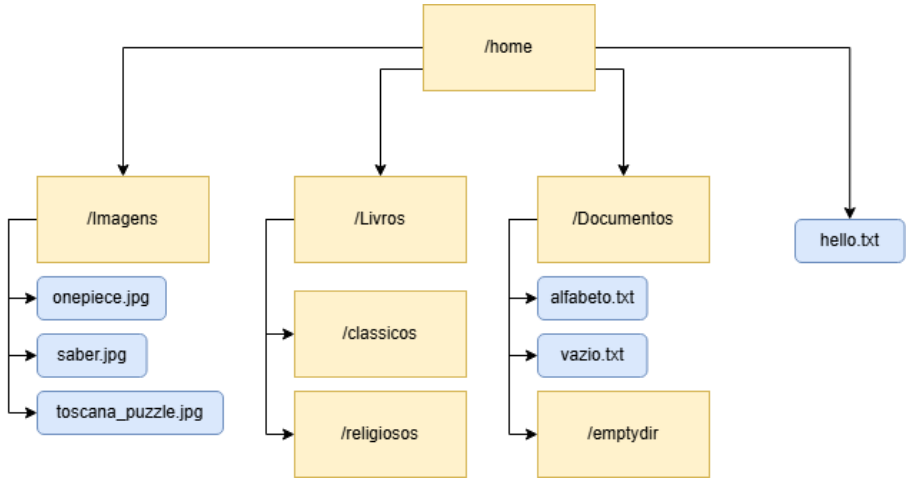

Figura 2 – Organização interna dos arquivos na imagem EXT2_TESTE.img

6.1 Comandos

A seguir, a tabela com os comandos implementados do sistema de arquivos EXT2:

Comando	Descrição
`info`	Exibe informações gerais da imagem EXT2, incluindo caminho atual, inode atual e identificador do grupo.
`ls`	Lista os arquivos e diretórios presentes no inode atual (diretório corrente).
`pwd`	Exibe o caminho absoluto do diretório atual.
`cd <path>`	Altera o diretório corrente para o caminho especificado, atualizando o inode de referência.
`attr <file | dir>`	Exibe os atributos de um arquivo ou diretório com base em seu inode, incluindo permissões, tamanho e número de links.
`cat <file>`	Lê e imprime o conteúdo de um arquivo utilizando os blocos apontados pelo inode.
`print superblock`	Exibe todos os campos do Superbloco, incluindo contagem de blocos/inodes, versão, sistema operacional e timestamps.
`print groups`	Exibe os descritores de todos os grupos de blocos, incluindo localização de bitmaps e contagens de blocos/inodes livres.
`print group <número>`	Exibe os dados do grupo de blocos especificado.
`print inode <número>`	Exibe os metadados associados ao inode especificado, como permissões, ponteiros, tamanho e timestamps.

Tabela 1 – **Tabela de Comandos EXT2**

Cada comando será demonstrado com exemplos de uso e uma breve explicação sobre seu funcionamento.

info

O comando `info` fornece um resumo do estado atual da navegação na imagem EXT2. Ele apresenta informações como o caminho absoluto do diretório corrente, o número do inode atualmente referenciado e o identificador do grupo de blocos ao qual esse inode pertence. Esses dados são obtidos por meio da análise do inode corrente, bem como de informações previamente extraídas do Superbloco e da tabela de descritores de grupo.

ls

O comando `ls` realiza a listagem dos arquivos e subdiretórios presentes no diretório atual. Para isso, ele percorre os blocos de dados associados ao inode corrente, interpretando as entradas de diretório (`ext2_dir_entry`) para extrair informações como o nome, o tamanho do nome e o tipo de cada item encontrado.

pwd

O comando `pwd` (*print working directory*) imprime o caminho absoluto do diretório corrente, desde a raiz do sistema de arquivos até o inode atualmente referenciado. Para reconstruir esse caminho, a ferramenta percorre a hierarquia de diretórios no sentido reverso, identificando os diretórios pai por meio da entrada `".."` e comparando os números de inode contidos nas suas respectivas entradas. A partir dessa lógica, os nomes dos diretórios são concatenados de forma ordenada, resultando em uma representação exata do caminho completo.

cd

O comando `cd <diretório>` permite alterar o diretório corrente dentro da hierarquia do sistema de arquivos EXT2. Para sua execução, a ferramenta percorre as entradas do diretório atual, armazenadas nos blocos de dados apontados pelo inode corrente, até localizar o nome especificado. Uma vez encontrado, o número do inode correspondente é recuperado e utilizado para atualizar o inode de referência, alterando assim o contexto de navegação. A ferramenta suporta tanto navegação para subdiretórios quanto retorno ao diretório pai por meio da entrada especial `".."`.

attr

O comando `attr <arquivo | diretório>` permite acessar e exibir os atributos de um determinado arquivo ou diretório localizado no diretório atual. A ferramenta localiza o item especificado entre as entradas do diretório, recupera seu inode e apresenta informações como o tamanho do arquivo, permissões de acesso, número de links, identificadores de usuário e grupo, além dos tempos de criação, modificação, acesso e exclusão.

cat

O comando `cat <arquivo>` é utilizado para exibir o conteúdo de um arquivo diretamente no terminal. A ferramenta realiza a busca do nome especificado no diretório atual, acessa o inode correspondente e, a partir dos ponteiros contidos no campo `i_block`, realiza a leitura sequencial dos blocos de dados que compõem o conteúdo do arquivo. A implementação considera o uso de ponteiros diretos, e pode ser expandida para interpretar também os níveis de indireção (simples, dupla e tripla). O conteúdo dos blocos lidos é convertido e impresso como texto, possibilitando a visualização completa dos dados armazenados no arquivo.

Comandos de consulta `print`

A ferramenta também disponibiliza comandos de consulta com a palavra-chave `print`, voltados à inspeção direta das estruturas internas do sistema de arquivos EXT2. O comando `Print superblock` exibe todos os campos do Superbloco, incluindo informações sobre a contagem de inodes e blocos, identificador do sistema operacional, versão do sistema de arquivos, tempos de montagem e verificação, entre outros. Já o comando `Print groups` lista os descritores de todos os grupos de blocos existentes na imagem, com dados como localização de bitmaps e tabela de inodes, além de contagens de blocos e inodes livres. O comando `Print group <número>` permite visualizar os dados de um grupo de blocos específico, enquanto `Print inode <número>` fornece todos os metadados do inode indicado. Esses comandos não alteram o estado de navegação, sendo utilizados exclusivamente para inspeção técnica e validação da estrutura interna da imagem EXT2.

7 Bugs Conhecidos

Atualmente, não foram identificados bugs conhecidos no funcionamento da ferramenta.

8 Conclusão

O projeto teve como principal objetivo explorar e demonstrar, de forma prática, os conceitos fundamentais do sistema de arquivos EXT2, por meio da implementação de comandos capazes de manipular diretamente uma imagem. A execução da ferramenta exigiu a compreensão de estruturas internas como o Superbloco, a tabela de descritores de grupo, os bitmaps de blocos e inodes, a tabela de inodes e as entradas de diretório. Tais estruturas foram acessadas e manipuladas diretamente em baixo nível, utilizando ponteiros.

Durante o desenvolvimento, foram implementadas funcionalidades que simulam operações de um ambiente de shell, como navegação entre diretórios, leitura de arquivos, exibição de atributos e listagem de conteúdo, além de comandos de inspeção. Cada comando foi feito para reforçar o entendimento da relação entre os inodes, os blocos de dados e as estruturas de controle distribuídas ao longo dos grupos de blocos.

O projeto demonstrou a importância da abstração oferecida pelos sistemas de arquivos. A experiência obtida mostra a relevância da compreensão de sistemas internos relativos ao sistema operacional.

9 Referências

Design and Implementation of the Second Extended Filesystem. Disponível em: <http://web.mit.edu/tytso/www/linux/ext2intro.html>. Acesso em: 18 jun. 2025. Nenhuma citação no texto.

Ext2 - OSDev Wiki. Disponível em: <https://wiki.osdev.org/Ext2>. Acesso em: 18 jun. 2025. Nenhuma citação no texto.

The Ext2 Filesystem. Disponível em: <https://www.science.smith.edu/~nhowe/262/oldlabs/kernel/ext2_fs.h>. Acesso em: 25 jun. 2025. Nenhuma citação no texto.

The Second Extended Filesystem — The Linux Kernel documentation. Disponível em: <https://docs.kernel.org/filesystems/ext2.html>. Acesso em: 25 jun. 2025. Nenhuma citação no texto.

The Second Extended File System. Disponível em: <https://www.nongnu.org/ext2-doc/ext2.html>. Acesso em: 25 jun. 2025. Nenhuma citação no texto.