Linux Extend FS e VFS

Estruturas de Suporte à Utilização dos Ficheiros

- Todos os sistemas de ficheiros definem um conjunto de estruturas em memória volátil para a gerir a informação persistente mantida em disco.

- Objetivos:
  - Criar e gerir os canais virtuais entre as aplicações e os ficheiros e diretórios
  - Aumentar o desempenho do sistema mantendo a informação em *caches*
  - Tolerar eventuais faltas
  - Isolar as aplicações da organização do sistema de ficheiros
  - Possibilitar a gestão de várias organizações de estruturas de ficheiros em simultâneo

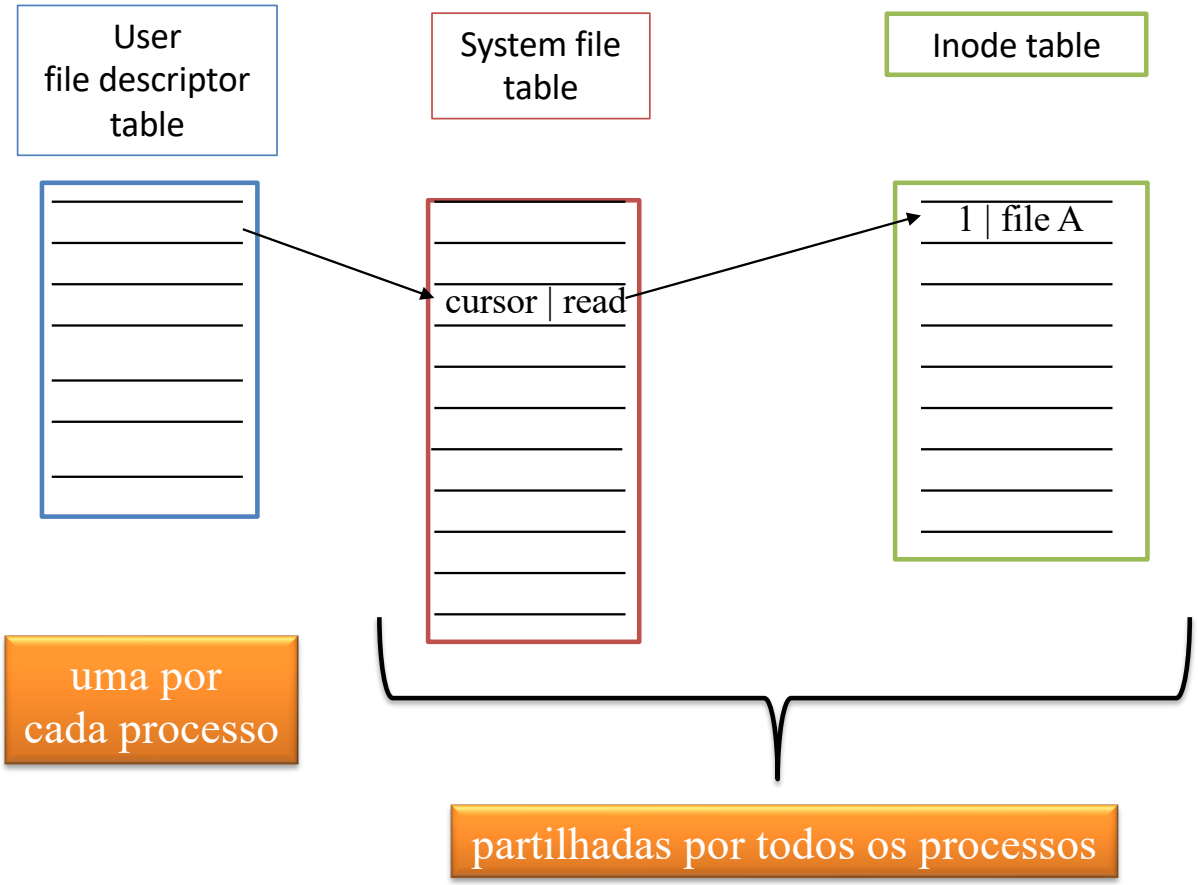
Tabelas de Ficheiros Abertos do Unix

- Existem duas tabelas para referenciar os ficheiros abertos, mantidas no espaço de memória protegido pelo que só podem ser acedidas pelo núcleo

- **Tabela de ficheiros abertos do processo**
  - Contém um descritor para cada um dos ficheiros abertos que referencia a tabela global de ficheiros abertos, o índice nessa tabela é o *file descriptor* que é devolvido no `open()`

- **Tabela de ficheiros abertos global/sistema**
  - Contém informação relativa a um ficheiro aberto: cursor com a posição atual de leitura/escrita, modo como o ficheiro foi aberto

Tabelas de Ficheiros



Exemplo na figura

```
fd1 = open ("fileA", O_RDONLY);
```

Estruturas de Suporte à Utilização dos Ficheiros

- A existência de duas tabelas justifica-se:
  - garantir o isolamento entre processos
  - permitir a partilha de ficheiros sempre que necessário (e.g. os cursores de escrita e leitura de um ficheiro entre dois ou mais processos)
  - Por exemplo um processo filho em Unix herda os ficheiros abertos do processo pai e os cursores

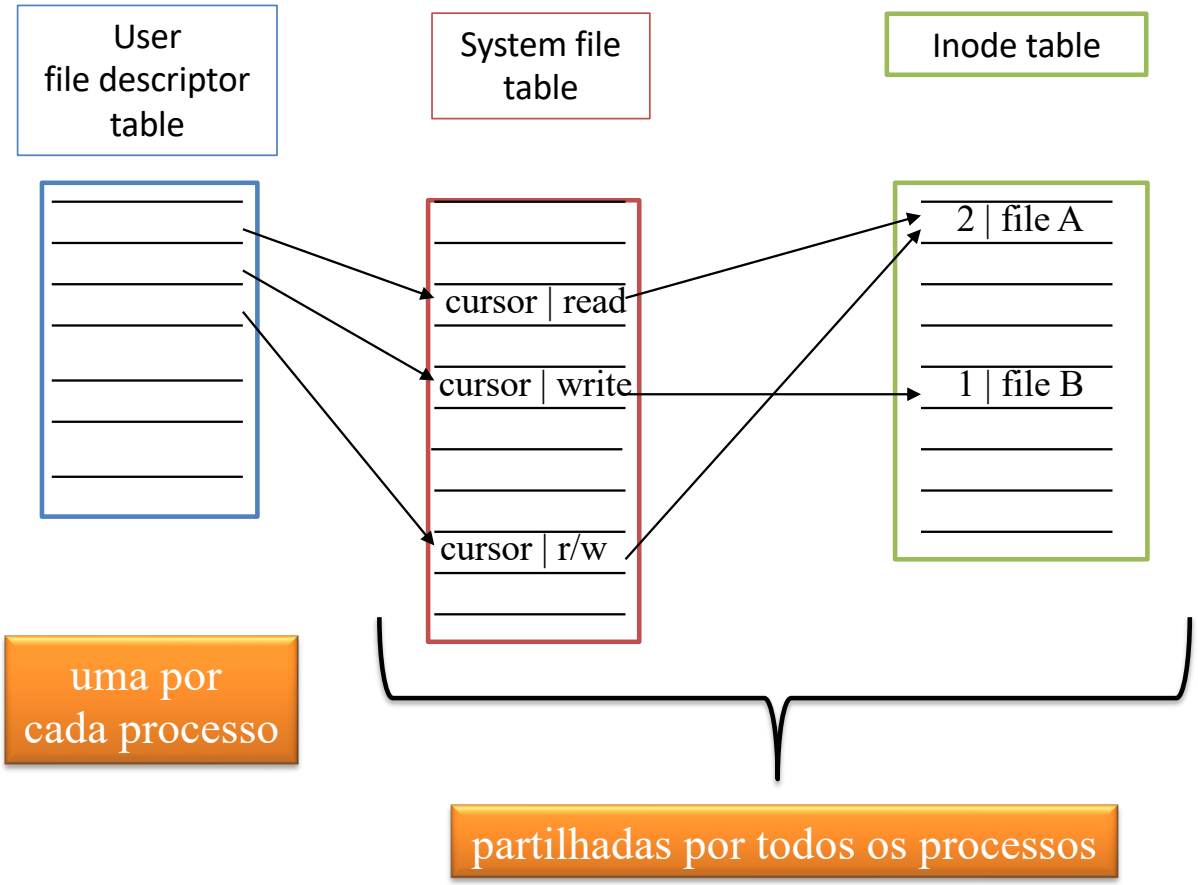
Objeto ficheiro

- Quando um processo chama `open()` é criado um objeto ficheiro e colocado na **primeira posição livre** da **tabela de descritores do processo** um ponteiro para esse objeto, sendo devolvido ao utilizador o índice dessa entrada na tabela (*file descriptor*)

- **Pode existir mais do que um objeto ficheiro para o mesmo ficheiro**. Se dois processos abrirem o mesmo ficheiro cada um deles fica com um ponteiro para um objeto ficheiro diferente, pois de outro modo partilhariam o mesmo cursor

- Acontece exatamente o mesmo **se um processo abrir um ficheiro duas vezes**. Por exemplo, se um processo abrir um ficheiro para leitura e abri-lo de novo para escrita fica com dois objetos ficheiro diferentes com dois cursores distintos, um para leitura e outro para escrita,

Tabelas de Ficheiros



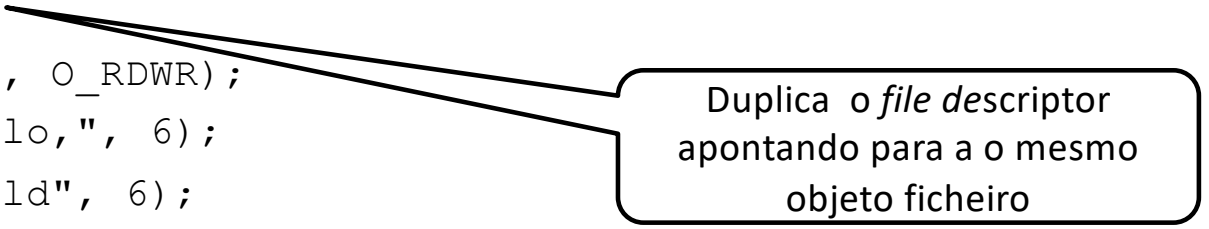
Exemplo na figura

```
fd1 = open ("fileA", O_RDONLY);
fd2 = open ("fileB", O_WRONLY);
fd3 = open ("fileA", O_RDWR);
```

Exercício

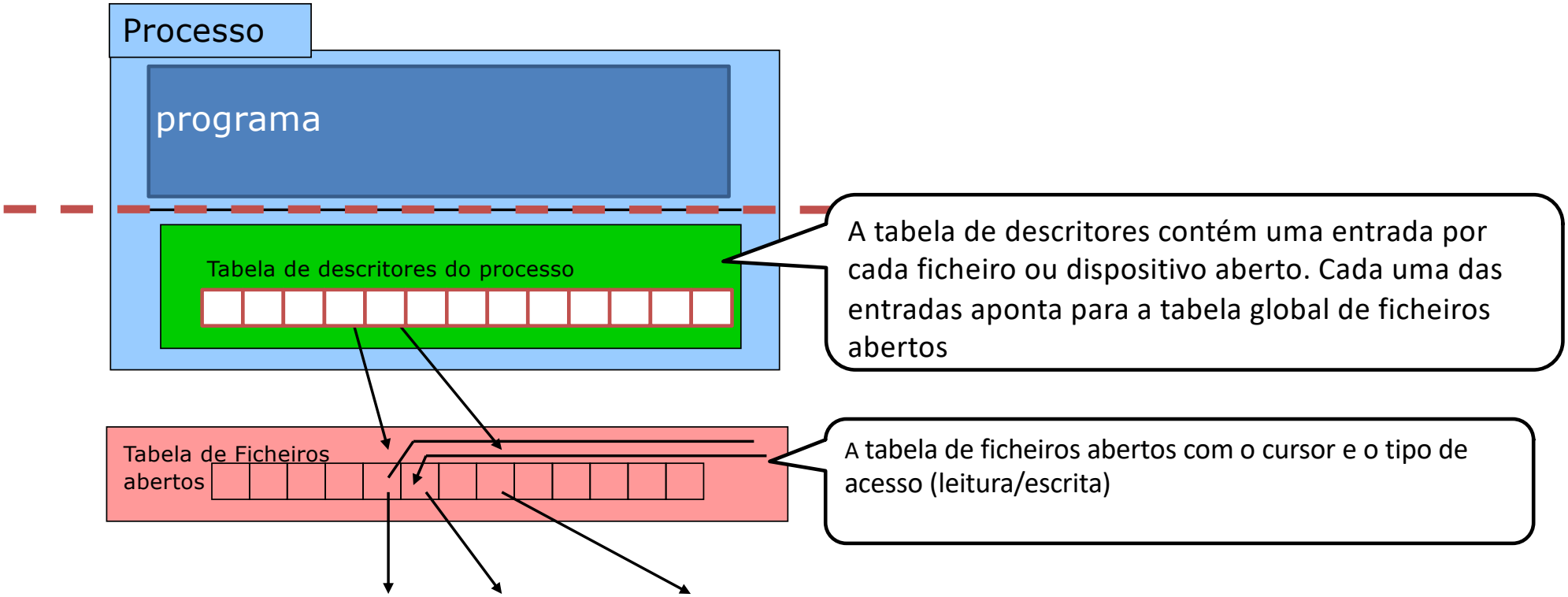
After each of the calls to write() in the following code, explain what the content of the output file would be, and why:

```
fd1 = open(file, O_RDWR | O_CREAT | O_TRUNC, S_IRUSR | S_IWUSR);
fd2 = dup(fd1);
fd3 = open(file, O_RDWR);
write(fd1, "Hello,", 6);
write(fd2, "world", 6);
lseek(fd2, 0, SEEK_SET);
write(fd1, "HELLO,", 6);
write(fd3, "Gidday", 6);
```

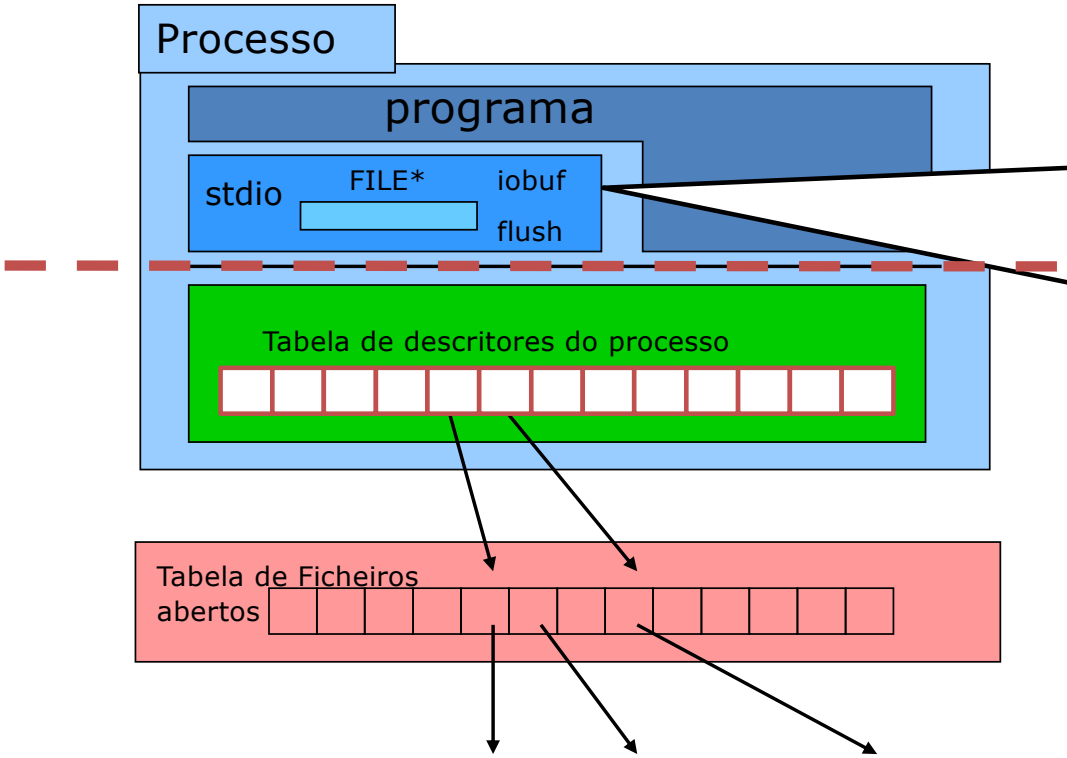


Linux programming interface

Visão Global



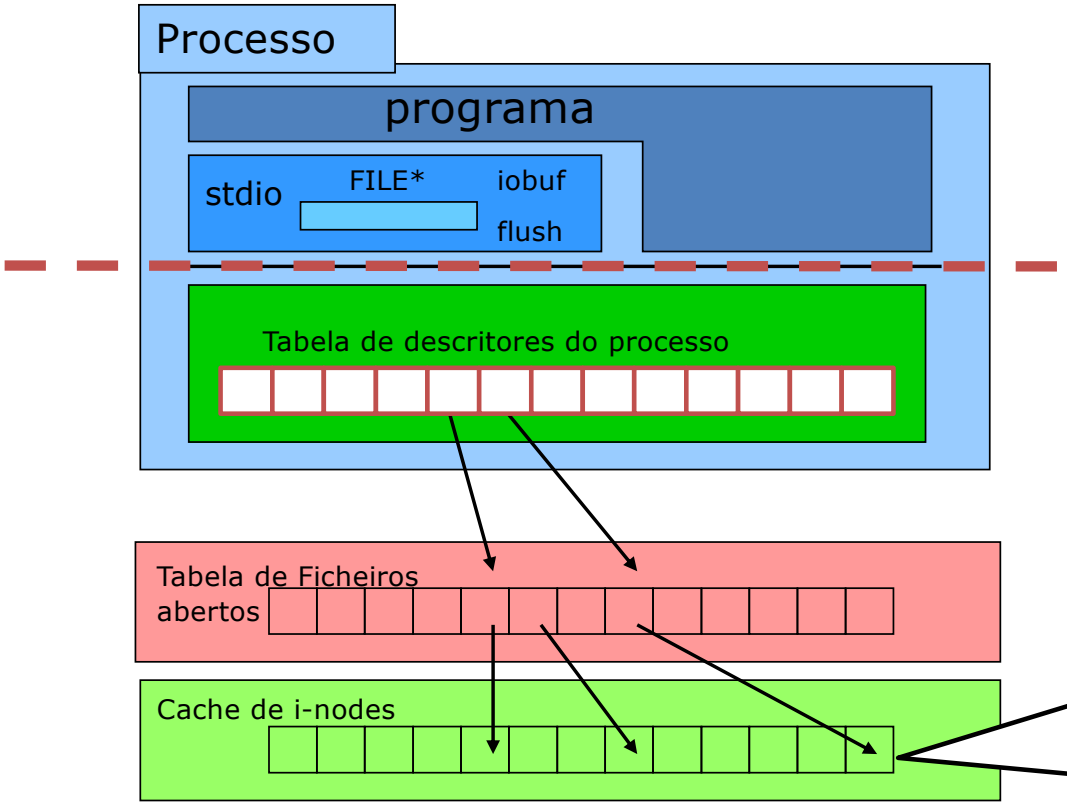
Visão Global – com stdio



O programa tem um ponteiro para uma estrutura do tipo `FILE`

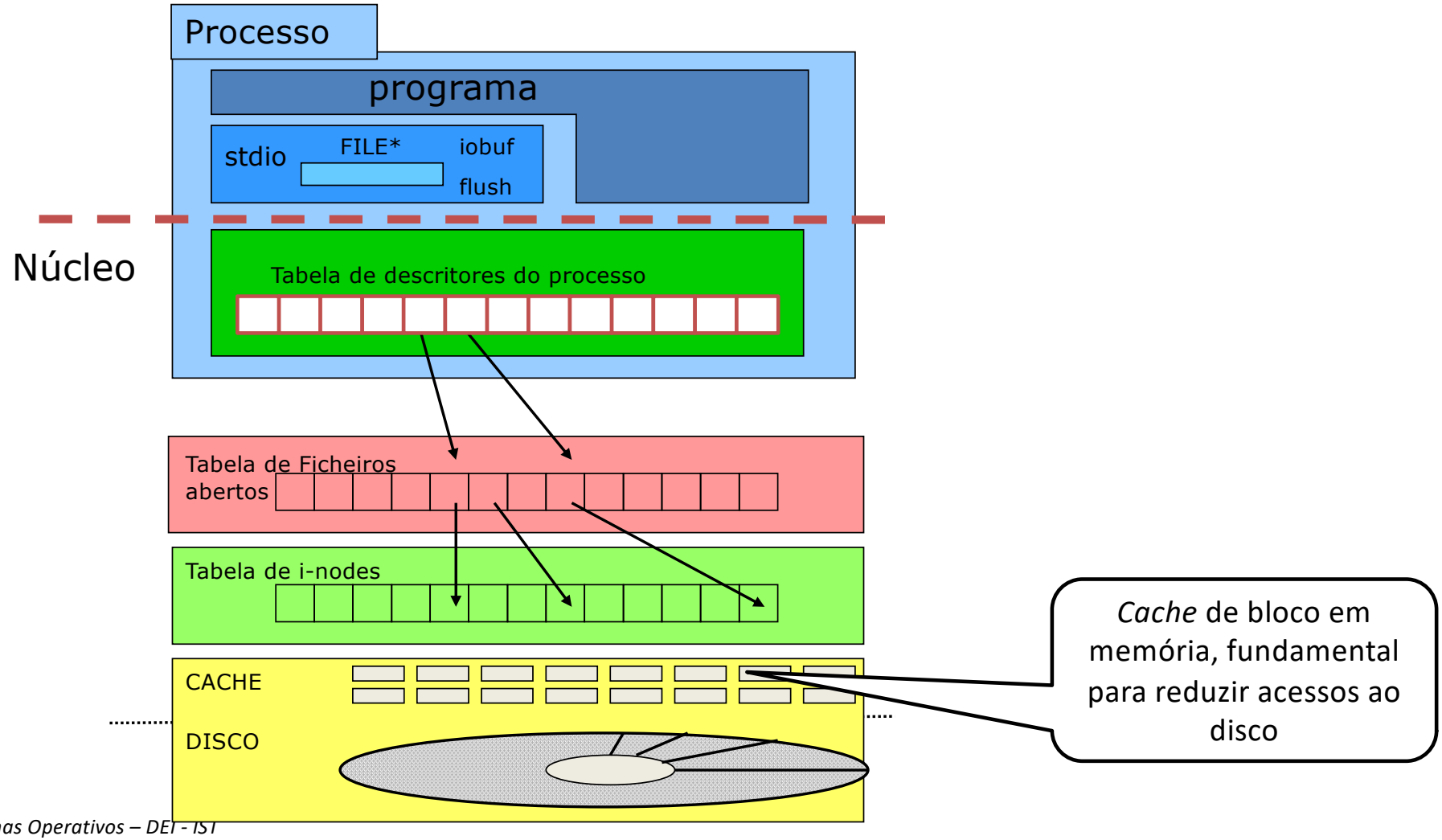
A `FILE` é mantida pela biblioteca `stdio` e contém um *buffer* e o *file descriptor* do ficheiro aberto. O *buffer* serve para otimizar as escritas e leituras.

Visão Global



A tabela de *inodes* é uma cache dos *inodes* em disco.
Os índices no inode ou em blocos auxiliares referenciam os blocos do disco com a informação do ficheiro.

Visão Global



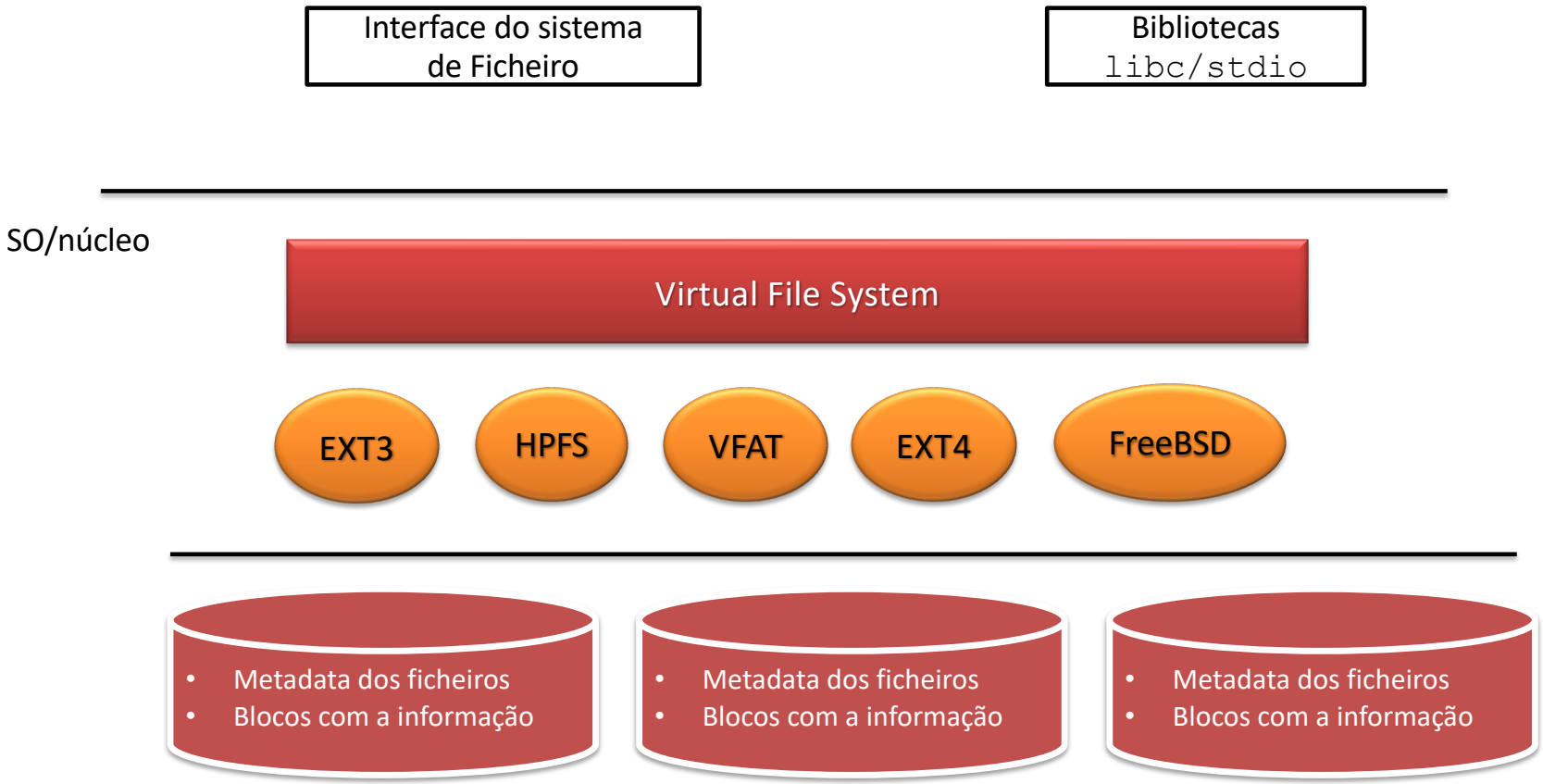
Virtual File System em Linux

Estruturas em Memória - VFS

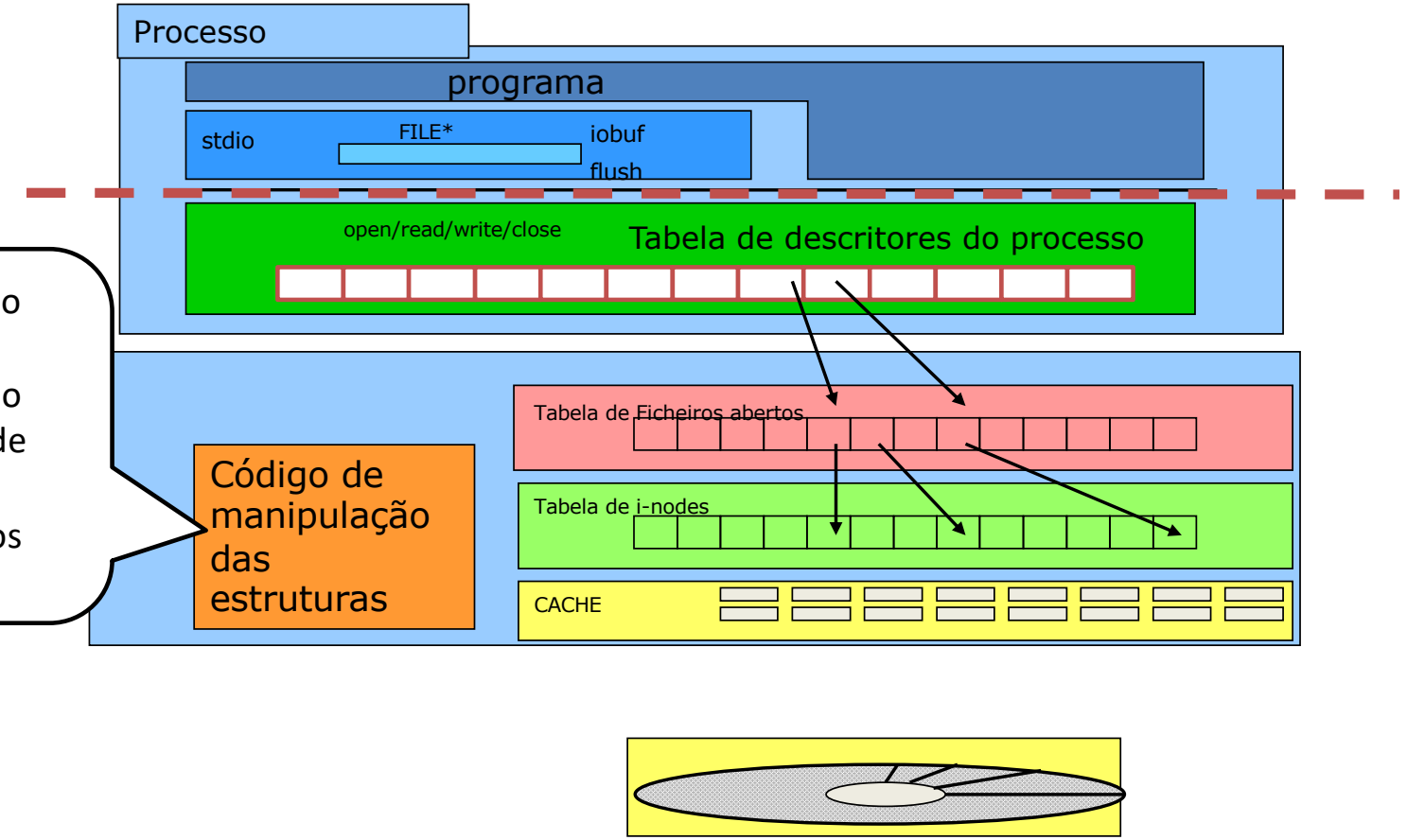
- O objectivo do VFS :
  - a criação de um sistema de ficheiros comum, virtual, que suporta vários sistemas de ficheiros nativos, em simultâneo.

- Neste modelo:
  - cada ficheiro é manipulado por um conjunto de operações (leitura, escrita, abertura, etc.) diferente,
  - dependendo do sistema de ficheiros nativo em que estão armazenados

Organização VFS



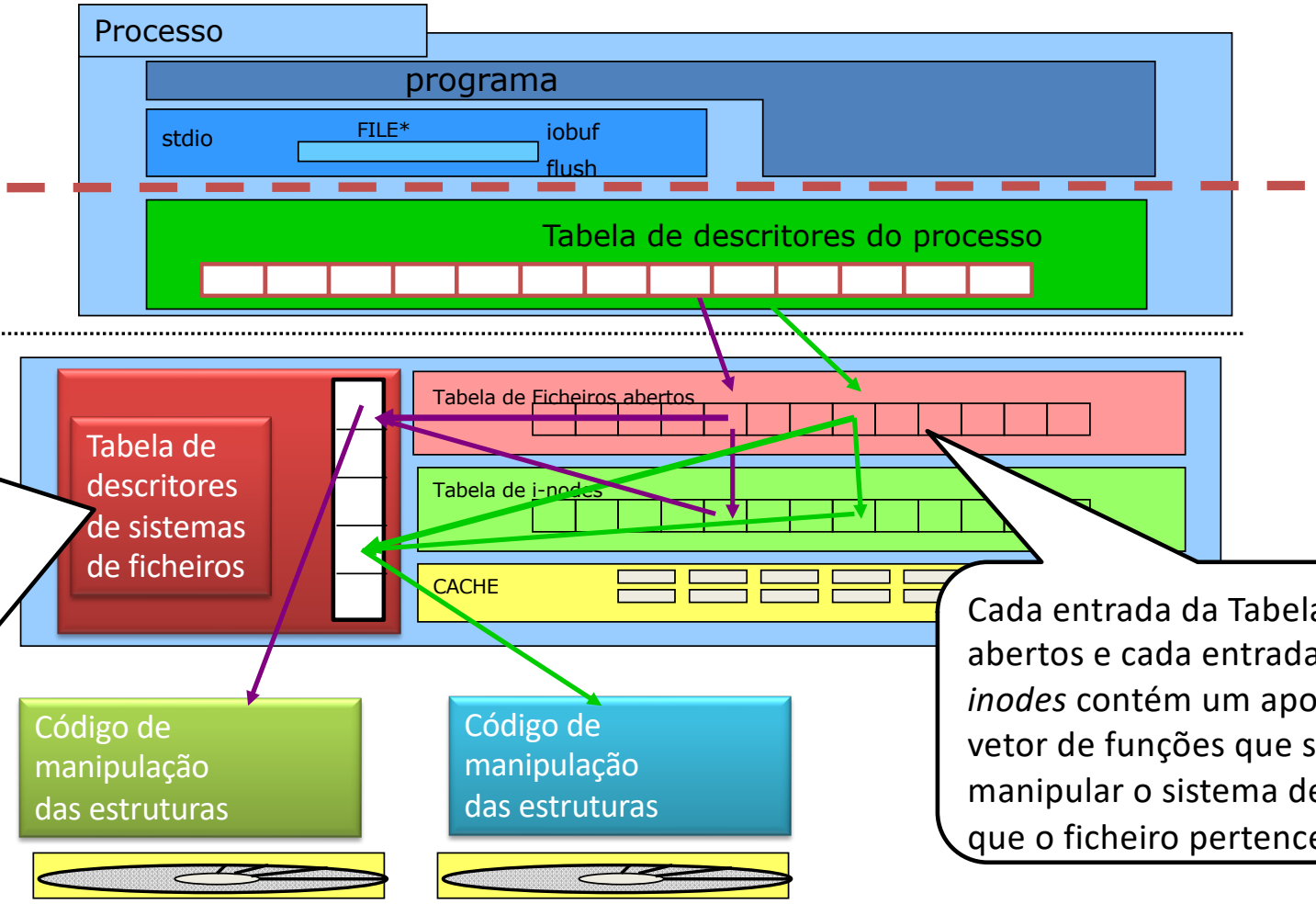
Um único Sistema de Ficheiros



- Leitura de blocos do disco para a cache.
- Leitura de *inodes* do disco para a tabela de *inodes*
- Leitura de diretórios

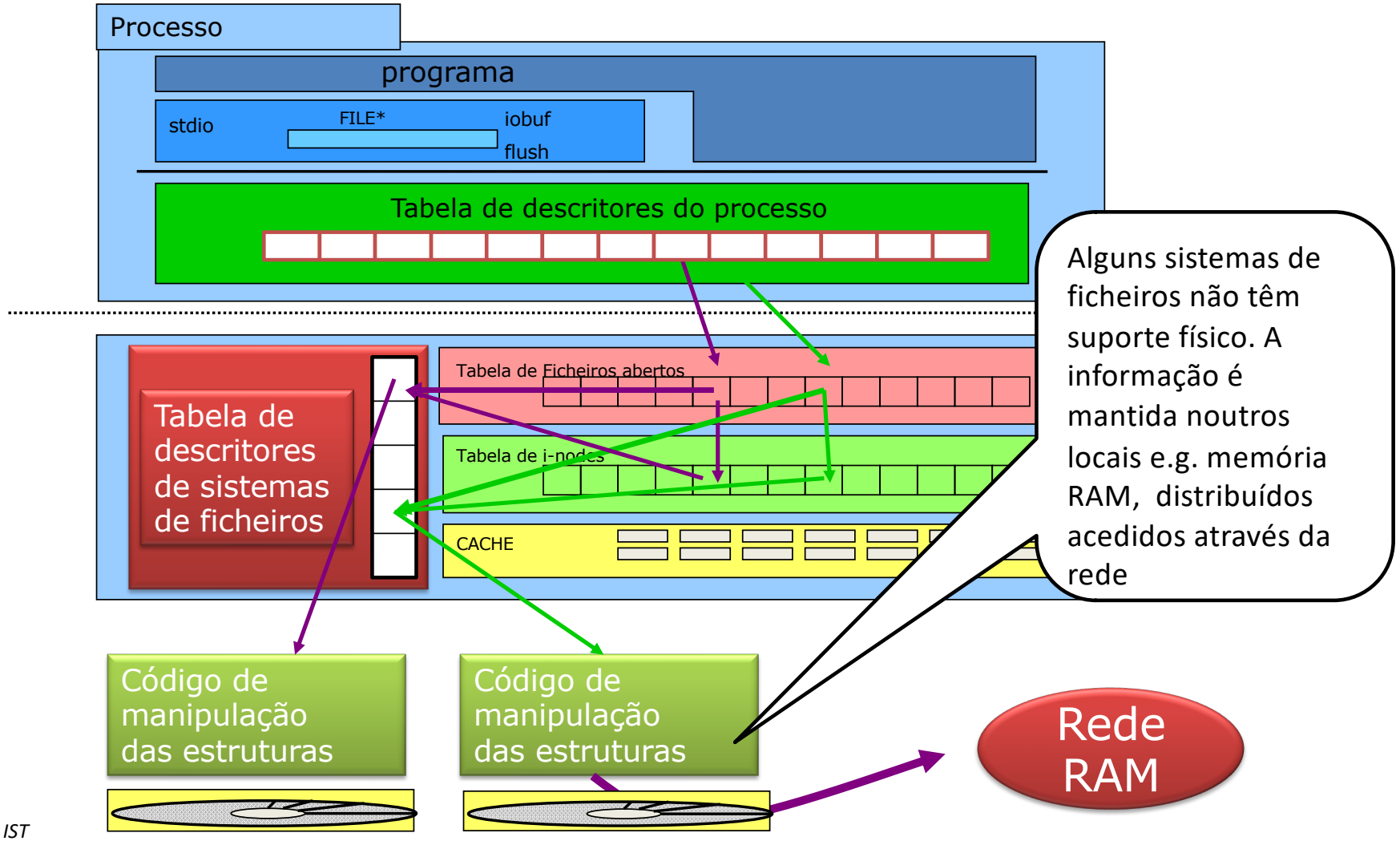
Mais do que um Sistema de ficheiros

Cada entrada:
- é designada por VFS *superblock*.
- contém informações sobre cada sistema de ficheiros
- contém um vetor de ponteiros para funções que sabem obter a informação do sistema de ficheiros.



Cada entrada da Tabela de ficheiros abertos e cada entrada da tabela de *inodes* contém um apontador para o vetor de funções que sabe manipular o sistema de ficheiros a que o ficheiro pertence

Sistemas de Ficheiros virtuais



Conclusões

- As estruturas de dados em memória RAM são fundamentais para o funcionamento do sistema de ficheiros
- Em Linux, existem duas tabelas para os ficheiros abertos, uma pertence a cada processo e outra é global ao sistema, contendo esta o cursor e o modo de abertura
- Na tabela de ficheiros abertos globais a entrada (objeto ficheiro) aponta para a cache de *inodes* para poder aceder à metadata em particular os índices de blocos
- No Linux, é possível no VFS montar diferentes sistemas de ficheiros. O objeto ficheiro tem uma tabela que redireciona a execução para as funções do SF a que o ficheiro pertence