File System

Implementación de un File System

VSFS (Very Simple File System)

Este file system es una versión simplificada de un típico sistema de archivos unix-like.

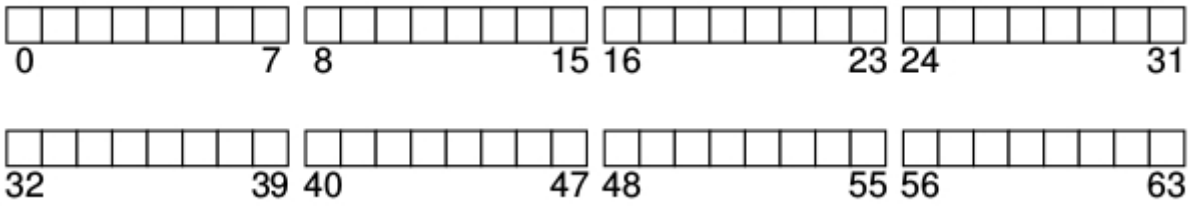
Dos conceptos fundamentales:

1. La **estructura de datos** de un sistema de archivos, como se guarda la información en el disco para organizar los datos y metadatos de los archivos. El sistema de archivos **vsfs** emplea una simple estructura, que parece un **arreglo de bloques**.
2. El método de acceso, como se machean las **llamadas hechas por los procesos**, como `open()`, `read()`, `write()`, etc. en la estructura del sistema de archivos.

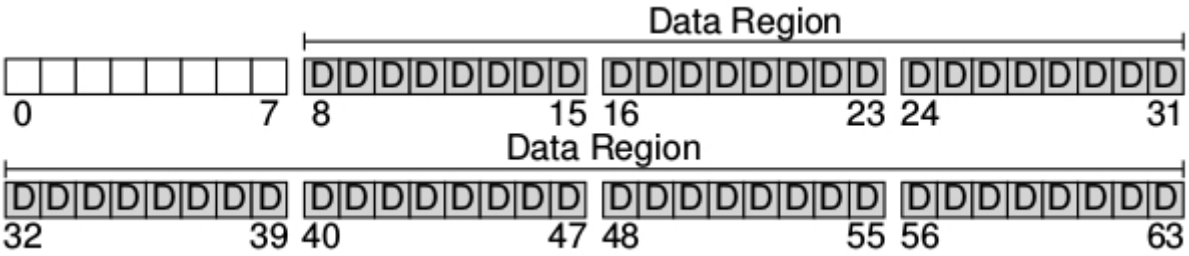
Organización general

Lo primero que se debe hacer es dividir al disco en **bloques**, los sistemas de archivos simples, como este suelen tener bloques de un solo tamaño. Los bloques tienen un tamaño de 4 kBytes.

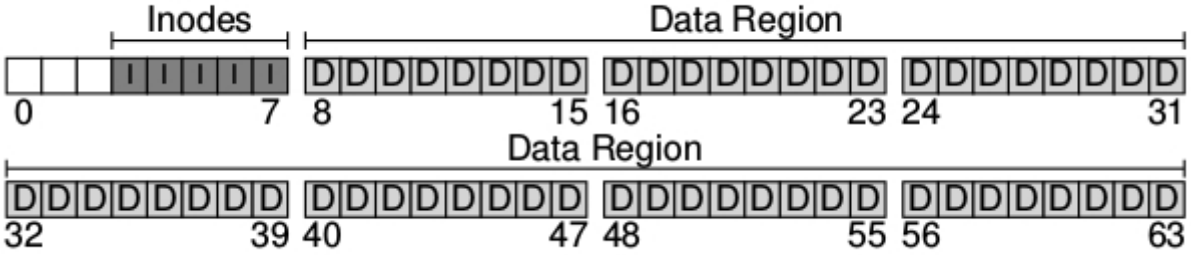
La visión del sistema de archivos debe ser la de una partición de N bloques (de 0 a N-1) de un tamaño de N * 4 KB bloques. Si suponemos en un disco muy pequeño, de unos 64 bloques, este podría verse así:



En un sistema de archivos es necesario almacenar **datos**. Esta región se llama **data region**. Nuestro pequeño disco es ocupado por ejemplo por 56 bloques de datos de los 64:



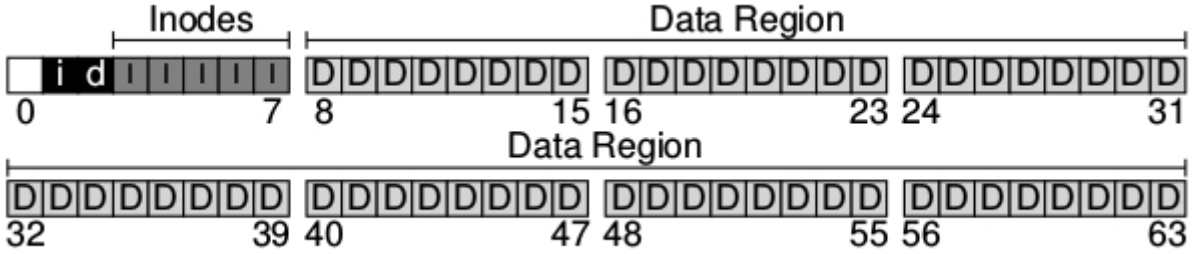
El sistema de archivos debe mantener información sobre cada uno de estos archivos. Esta información es la **metadata** y es de vital importancia ya que mantiene información sobre que bloque de datos pertenece a un determinado archivo, el tamaño del archivo, etc. Para guardar esta información, en los sistemas operativos unix-like, se almacena en una estructura llamada **inodo**. Los **inodos** también deben guardarse en el disco, para ello se los guarda en una tabla llamada **inode table** que es un array de inodos almacenados en el disco:



Los inodos no son estructuras muy grandes, normalmente ocupan unos 128 o 256 bytes. Suponiendo que los inodos ocupan 256 bytes, un bloque de 4KB puede guardar 16 inodos por ende nuestro sistema de archivo tendrá como máximo 80 inodos. Esto representa también la cantidad máxima de archivos que podrá contener nuestro sistema de archivos.

El sistema de archivo tiene los **datos (D)** y los **inodos (I)** pero una de las cosas que faltan es saber cuales inodos y cuales bloques están siendo utilizados o están libres. Esta **estructura de alocación** es fundamental en cualquier sistema de archivos. Existen muchos métodos para llevar este registro pero en este caso se utilizará una estructura muy popular llamada **bitmap**. Una para los datos **data bitmap** y otra para los inodos **inode bitmap**.

Un bitmap es una estructura bastante sencilla en la que se mapea 0 si un objeto está libre y 1 si el objeto está ocupado. En éste, cada i sería el bitmap de inodos y d seria el bitmap de datos:



Cada bitmap ocupa menos de 4KB, pero se utiliza un bloque por cada uno.

Queda un único bloque libre en todo el disco. Este bloque es llamado **Super Bloque** (S). Contiene la información de todo el file system, incluyendo:

- Cantidad de inodos.
- Cantidad de bloques.
- Donde comienza la tabla de inodos –> bloque 3.
- Donde comienzan los bitmaps.