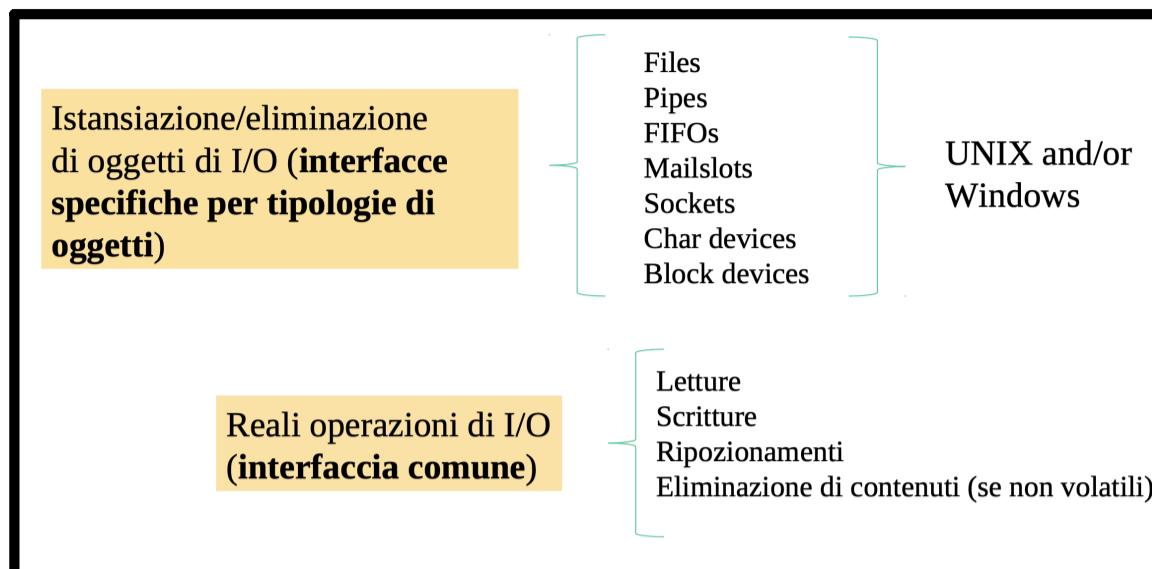


Overview

Andiamo a cercare di capire quali sono questi oggetti di I/O, quindi quelli più classici che abbiamo all'interno di un sistema operativo moderno e poi cerchiamo di capire oltre agli oggetti elencati sotto, quali sono le operazioni che possiamo eseguire su di essi.



Per quanto riguarda gli oggetti, tipicamente noi abbiamo i "Files", sono degli oggetti di I/O tali per cui quando noi andiamo ad inserire dei dati all'interno di uno di questi oggetti, permettiamo a questi dati di "SOPRAVVIVERE" allo shut-down del sistema e poi di ritrovarli direttamente allo start-up.

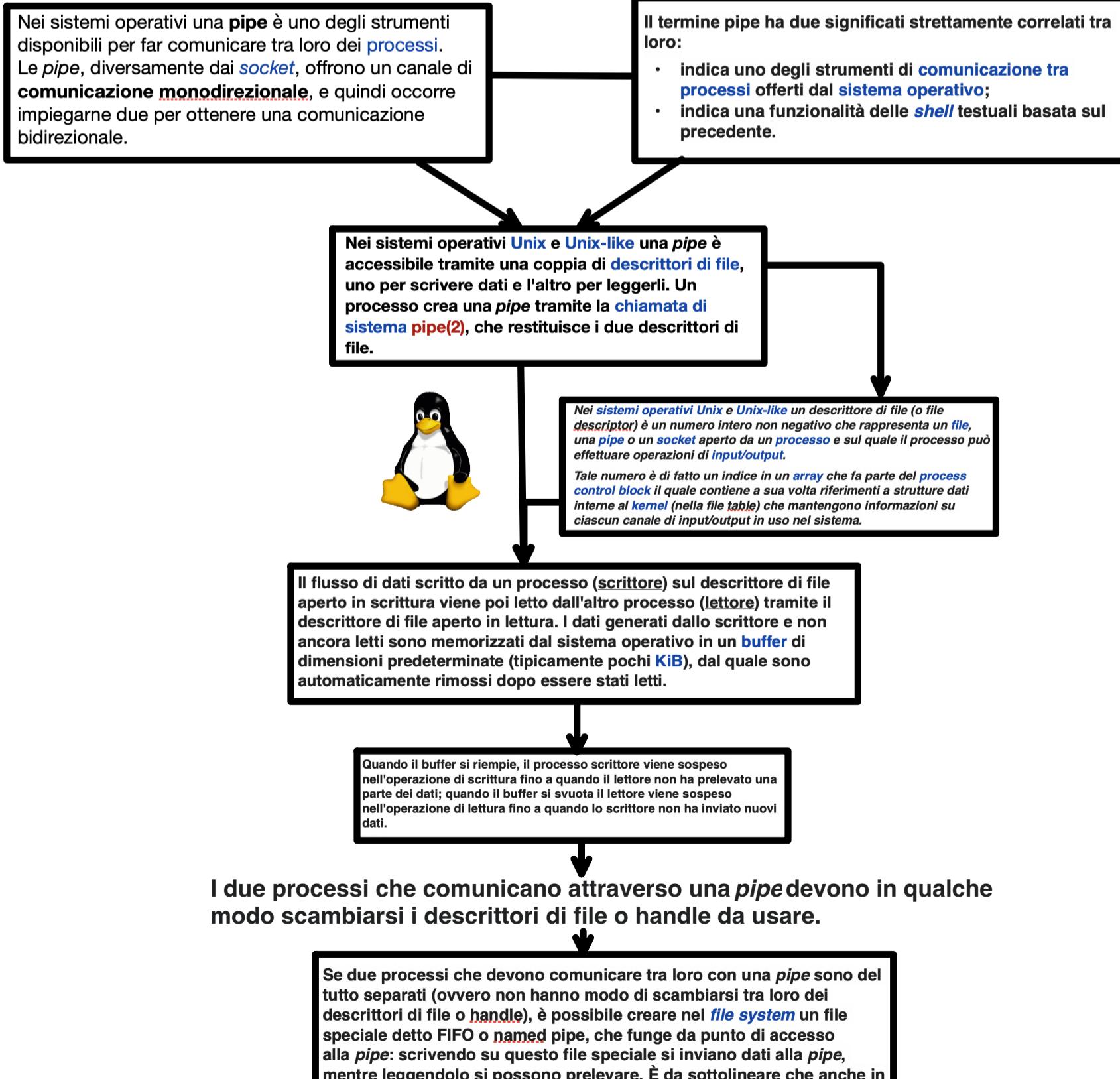
Le Pipes e le Fifo sono oggetti che in ci permettono di andare ad inserire dati/scrivere dati sull'oggetto di I/O che possono essere letti, ma queste le utilizziamo per permettere a processi attivi di parlare, come se stessero scrivendo o leggendo su dei file, ma in realtà lo fanno utilizzando le pipes.

Típicamente abbiamo utilizzato comando quando abbiamo digitato comandi del tipo:

`>> a | b`

E lo avevamo fatto quando lavoravamo sugli esempi del buffer overflow, e utilizzavamo la "Pipe" per indicare che questi due programmi a e b dovevano utilizzare come dispositivo di I/O esattamente una pipe.

Quindí permette alle applicazioni di poter parlare.



questo caso i dati scambiati non sono memorizzati temporaneamente nel *file system*, ma transitano da un processo all'altro tramite un buffer.

In questo modo, un processo può offrire un servizio ad altri processi aprendo una *named pipe* in una posizione nota del *file system*.

Nei sistemi operativi **Unix** e **Unix-like** si può creare una *named pipe* tramite il comando **mknod**, o tramite l'omonima chiamata di sistema **mknod**, o ancora tramite il comando **mkfifo**; i processi che comunicano tramite essa debbono essere in funzione sullo stesso *host* (ad esempio non è possibile usare una *named pipe* creata in un *file system* condiviso in rete come **NFS** per far comunicare processi residenti su *host* diversi).

Il mailslots implementa delle operazioni di I/O offerte in particolare dai sistemi Windows.

Abbiamo poi le sockets che sono interessantissime perché tramite queste in realtà quando noi andiamo a lavorare in I/O a livello "sistema" quello che stiamo facendo è utilizzare dei protocolli che ci permettono di avere la diffusione delle informazioni che stiamo emettendo in output in rete: permetteremo la costruzione di applicazioni Client/Server e che sono distribuite.

I char devices e i block devices sono dispositivi arbitrari.