Linux Base - Capitolo n. 8

Edizioni ByteMan (05-11-2005)

revisione: 25/01/2008

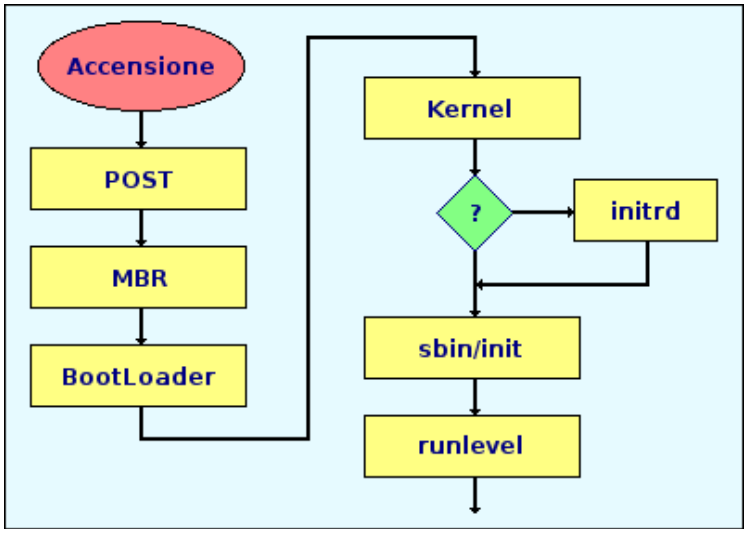
Le fasi di boot in Linux

Il processo di boot di una macchina Linux (su sistemi x86 Intel compatibili) comporta diverse fasi, la loro conoscenza ed interpretazione diventa fondamentale quando occorre risolvere malfunzionamenti durante l'avvio.

Questo processo, su un sistema Linux con processore **x386**, prevede i seguenti stadi (con altre CPU ci possono essere alcune differenze nelle fasi iniziali):

1. Ricerca del device da utilizzare per effettuare il boot da parte del BIOS in base ad una tabella preordinata .
2. Dal **boot sector** del device di boot parte il codice (o il salto di riferimento su dove trovarlo) del loader che esegue il **bootstrap** del sistema operativo. Nel caso di Linux i due più diffusi loader sono LILO ed il più evoluto GRUB.
3. Il loader lancia il caricamento del kernel di Linux, che copiandosi in memoria esegue i controlli ed il riconoscimento dell'hardware presente.
4. A fine caricamento il kernel esegue il processo **init**, padre di tutti i processi, che gestisce il caricamento di tutti gli altri programmi da eseguire per completare il boot.

La figura seguente mostra uno schema un po' più dettagliato della situazione che verrà illustrata subito dopo:



Accensione

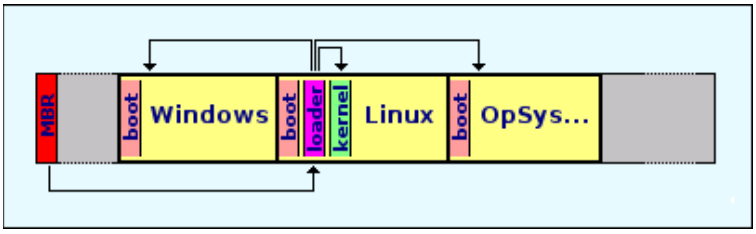
All'accensione, il controllo passa al codice presente nella ROM detto **BIOS** (Basic Input/Output System) che si occupa dell'inizializzazione della macchina. Come è noto ogni sistema Intel ha sulla motherboard questo chip di memoria con cui gestire l'hardware del sistema perchè all'avvio di un computer non c'è nulla di definito in RAM e nessun programma predefinito da caricare. Il BIOS controlla non solo la prima fase del processo di avvio, ma fornisce l'interfaccia di livello inferiore alle periferiche. Per questo motivo è scritto in una memoria permanente in sola lettura e può sempre essere utilizzato.

POST

Durante la fase di **POST** (Power On Self Test) vengono fatti i controlli di base ed individuato l'hardware di cui dispone il computer. Immediatamente dopo viene scelto, fra le impostazioni definibili dall'utente (secondo una sequenza di priorità modificabile), il dispositivo da usare per il boot. Nei BIOS più recenti è possibile effettuare il boot da: floppy, cdrom, hard disk, Zip, chiavette USB, schede di rete.

MBR

Spesso il primo disco fisso impostato per l'avvio è il disco C o il dispositivo IDE master del bus IDE primario. Gli hard disk, in particolare, hanno un settore di boot per ogni partizione ed inoltre un **MBR** (Master Boot Record) che è il primo settore di boot dell'intero hard disk. L'MBR ha dimensioni pari a soli 512 byte e contiene le istruzioni in codice macchina per l'avvio del computer oltre alla tabella delle partizioni. Quando si esegue il boot da un hard disk, è il codice contenuto nell'MBR che viene mandato in esecuzione. Contiene le informazioni per avviare il programma di boot secondario (**second stage boot loader**).
Nei casi più semplici MBR si limita a passare il controllo al codice presente nella cosiddetta partizione attiva (caso di MS-DOS).
Normalmente, però, i boot loader permettono un maggior controllo e la scelta di differenti sistemi operativi.



BootLoader

Esistono diversi loader che eseguono il bootstrap del sistema operativo per Linux, **LILO** e **GRUB** sono i più utilizzati con le distribuzioni GNU/Linux x86, ma ne esistono tanti altri: *GAG, xosl, LoadLin, SysLinux, BootLin*. Tutti di fatto eseguono la stessa funzione, alcuni (LoadLin e SysLinux) sono programmi DOS che eseguono il kernel caricandolo da una partizione DOS, altri sono adattamenti di LILO che riguardano sistemi non basati su processori Intel.
LILO e **GRUB** sono caratterizzati dall'avere una piccola porzione del codice macchina binario dell'MBR, il cui unico obiettivo è quello di rilevare il boot loader secondario e caricarlo in memoria.

- LILO (**LI**nux **LO**ader) è (storicamente) il primo boot loader per Linux. E' installabile sia nell'MBR che nel boot record della partizione Linux di avvio (**/** oppure **/boot**). Deve, però, essere reinstallato nuovamente quando si aggiungono nuove voci al suo file di configurazione (**/etc/lilo.conf**). Per reinstallarlo, basta digitare semplicemente uno dei due comandi:

```
lilo
lilo -v
```

il comando è presente nella cartella */sbin*, e come è ovvio bisogna essere utente root! Con il parametro -v si possono ottenere maggiori informazioni.
Ecco un esempio di file **/etc/lilo.conf**:

```
# Global Options
prompt
timeout=50
default=linux
boot=/dev/hda
map=/boot/map
install=/boot/boot.b
message=/boot/message
password=pippo
restricted
linear

label=linux
image=/boot/vmlinuz-2.6.5
initrd=/boot/initrd-2.6.5.img
read-only
root=/dev/hda3

label=linux.old
image=/boot/vmlinuz-2.6.4
read-only
```

```
root=/dev/hda3

other=/dev/hda1
optional
label=DOS
```

- GRUB (**GR**and **U**nified **B**ootloader) è il nuovo concorrente di LILO, è attualmente il boot loader di default per le distribuzioni RedHat/Fedora, ma si sta diffondendo rapidamente. A differenza di LILO, si presenta come una piccola shell di comandi. NON deve essere reinstallato nuovamente quando si aggiungono nuove voci all'apposito file di configurazione(**/boot/grub/menu.lst**).
  Ecco un esempio di file **/boot/grub/menu.lst**:

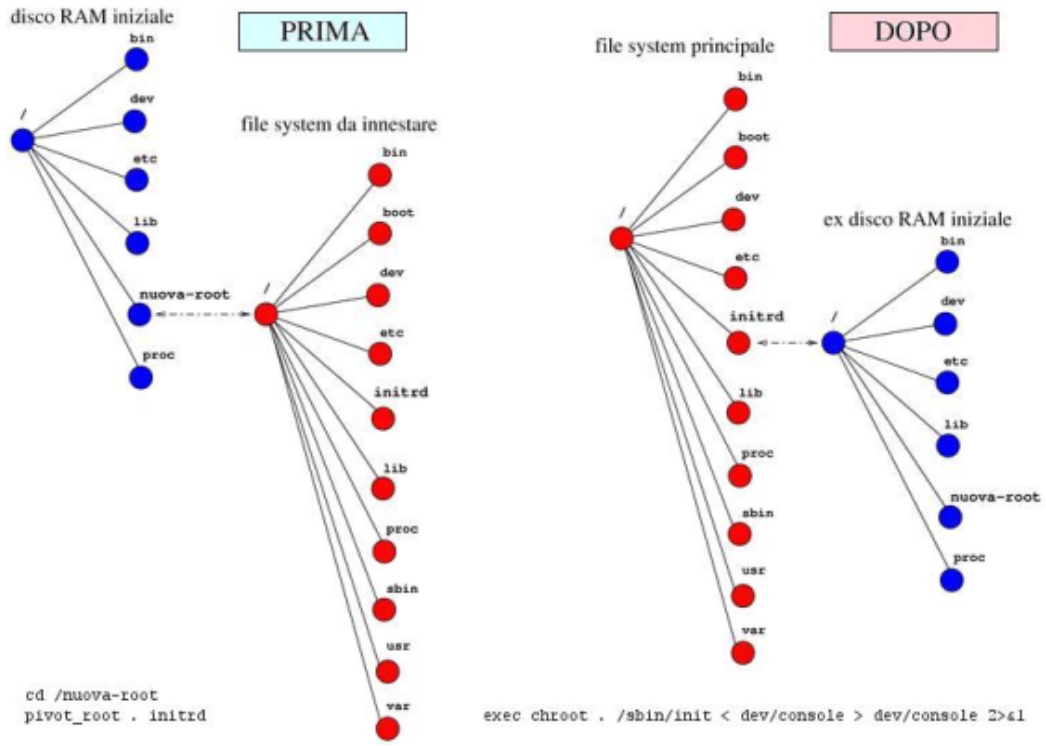
```
# Global Options
passwd ?md5
$1$6ðòüZßXÈ$bXTLL8IbDhnwmjyaNNcPG
timeout=50
default=0

title Debian GNU/Linux 2.6.5
kernel /boot/vmlinuz-2.6.5
initrd /boot/initrd-2.6.5.img
read-only
root (hd0,2)

title Debian GNU/Linux 2.6.4
kernel /boot/vmlinuz-2.6.4
root (hd0,2)

title DOS
rootnoverify (hd0,0)
chainloader +1
```

Quando il boot loader secondario (GRUB o LILO) è in memoria, viene visualizzata la schermata iniziale di Linux che mostra i diversi sistemi operativi o i kernel che sono stati configurati per l'avvio. Su questa schermata l'utente può usare i **tasti direzionali** per scegliere quale sistema operativo o kernel avviare e premere **Invio** per confermare. Se non viene premuto nessun tasto il boot loader carica la selezione di default dopo un periodo di tempo configurabile.

Nel caso in cui sia stato selezionato Linux, quando il boot loader di seconda fase ha determinato quale kernel avviare, rileva il kernel binario corrispondente nella directory **/boot/**. Il kernel binario è individuato da un nome che ha il formato che segue:
/boot/vmlinuz-<*versione-kernel*> per esempio: vmlinuz-2.6.0-xx

Oltre al kernel, viene caricata in memoria anche l'immagine RAM disk iniziale. Tale immagine si chiama **initrd** ed e' utilizzata dal kernel per caricare in memoria tutti i driver non compilati all'interno del kernel stesso che sono necessari per avviare il sistema.

initrd è un file system speciale che viene montato allo scopo di avviare un sistema minimo, con cui eseguire alcune operazioni preliminari. Al termine di queste operazioni, normalmente il sistema contenuto nel disco RAM iniziale monta il file system standard e passa il controllo al programma **init**.
La tecnica del disco RAM iniziale viene usata solitamente per caricare dei moduli prima di montare il file system definitivo, per esempio quando il kernel richiede un modulo speciale per accedere a tale file system. Si può vedere il disco RAM iniziale, come un file system contenente fondamentalmente un programma init, che convenzionalmente corrisponde a /linuxrc. La difficoltà sta nel ridurre al minimo il sistema di questo disco RAM; eventualmente il file /linuxrc potrebbe essere un programma realizzato appositamente, senza bisogno di altro.
Una volta che il programma o lo script /linuxrc ha compiuto il suo lavoro, questo deve innestare il file system che deve in seguito diventare quello principale, in una directory, quindi deve eseguire la funzione pivot_root() per scambiare i ruoli. Vedi l'immagine seguente.



Una volta caricati in memoria RAM il kernel e l'immagine RAM disk initrd, il controllo della macchina passa al kernel.

Avvio del Kernel

Dopo le prime fasi di boot giunge finalmente il momento dell'avvio del **kernel**, il cui inizio è spesso caratterizzato dallo scorrere di una lunga serie di messaggi sul video. Con riferimento, ancora, al primo grafico riportato nel paragrafo precedente ecco illustrate le fasi successive al **bootloader**.

Kernel

Quando il kernel viene caricato, inizializza e configura immediatamente la memoria del computer. Visualizza vari messaggi utili per capire e conoscere il proprio sistema. È possibile, in seguito, visualizzare questi messaggi, che intanto scorrono velocemente sul monitor, tramite il comando:

```
dmesg
```

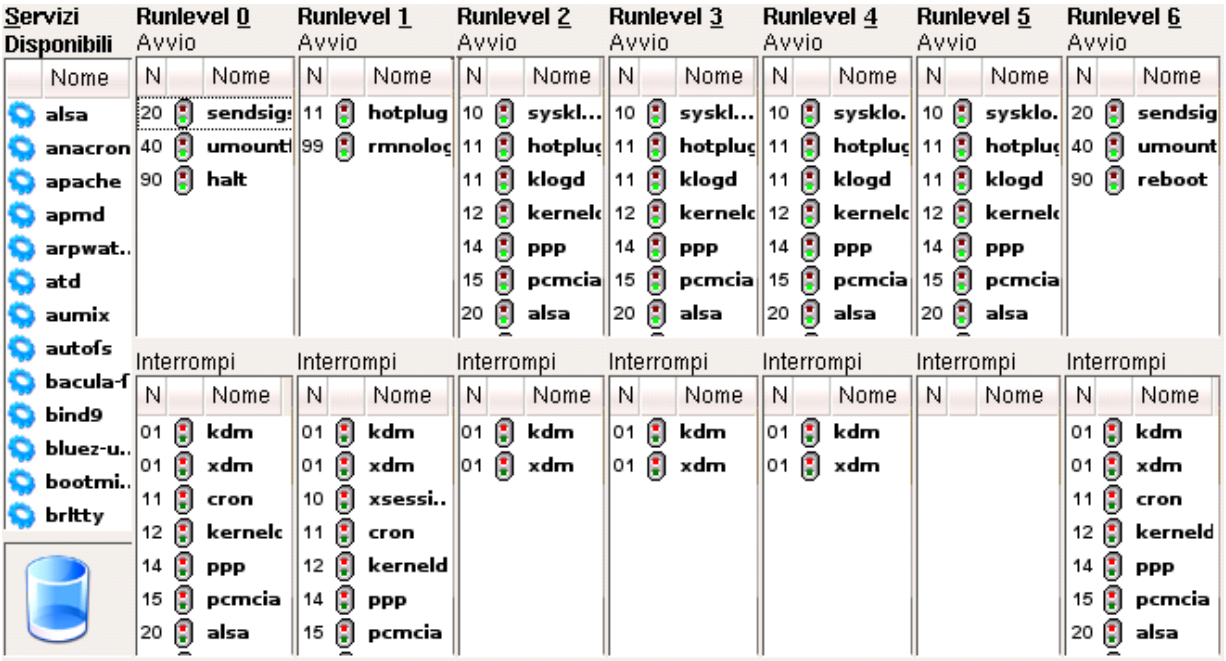
Il kernel configura i vari elementi hardware collegati al sistema, inclusi tutti i processori e i sottosistemi I/O, oltre a tutti i dispositivi di memorizzazione.

- Cerca quindi, se esiste (vedi grafico iniziale), l'immagine **initrd** compressa in un percorso predeterminato della memoria, la decomprime, la monta e carica tutti i driver necessari. Successivamente inizializza i dispositivi virtuali del sistema prima di smontare l'immagine disco initrd e liberare tutta la memoria. Più in dettaglio ecco il lavoro di **initrd**:
  - abilita un ramdisk in /dev/ram0
  - decomprime il contenuto dell'initrd (accessibile tramite /dev/initrd), e lo copia in /dev/ram0
  - monta /dev/ram0 in modalita' rw come filesystem di root iniziale.
    Se questo corrisponde al filesystem di root normale, si procede con la normale sequenza
  - se è presente l'eseguibile /linuxrc, esso viene eseguito.

Dopo l'inizializzazione di tutti i dispositivi del sistema da parte del kernel, viene creato un dispositivo root, montata la partizione root di sola lettura e liberata la memoria non utilizzata.
Il kernel risulta così caricato in memoria e operativo. Tuttavia, senza alcuna applicazione che consenta all'utente di fornire input significativo al sistema, il kernel non è molto utile.
Per configurare l'ambiente utente, il kernel esegue il comando /sbin/init.

/sbin/init

Il processo init, il cui file si trova in **/sbin/init/**, è definito il padre di tutti i processi, tramite il suo file di configurazione **/etc/inittab**, provvede a lanciare tutti i programmi che completano il processo di caricamento e configura l'ambiente per l'utente.
Innanzitutto esegue lo script /etc/rc.d/rc.sysinit che imposta il percorso, attiva lo swapping, controlla i filesystem e così via. In sostanza, si occupa di tutti i processi che vanno eseguiti all'inizializzazione del sistema. Per esempio, la maggior parte dei sistemi utilizza un orologio e rc.sysinit usa il file di configurazione /etc/sysconfig/clock per inizializzare l'orologio. Un'altro esempio è se dovete inizializzare processi speciali per le porte seriali, rc.sysinit può eseguire anche il file /etc/rc.serial.
In seguito il comando init esegue lo script **/etc/inittab**, che descrive il modo in cui il sistema va configurato per ogni runlevel SysV. Questo file specifica, tra le altre cose, che /etc/inittab imposta il runlevel predefinito e che /sbin/update va eseguito a ogni avvio dei runlevel. [4].
Successivamente il comando init configura la libreria delle funzioni sorgenti /etc/rc.d/init.d/functions per il sistema, che stabilisce come avviare o terminare un programma e come trovare il PID di un programma.
A questo punto il programma init avvia tutti i processi di background cercando nella relativa directory rc il runlevel specificato come predefinito in /etc/inittab. Le directory rc sono numerate per corrispondere ai runlevel che rappresentano. Per esempio /etc/rc.d/rc5.d/ è la directory per il runlevel cinque.



RunLevels

Un runlevel è una configurazione software del sistema che permette l'esistenza solo di un gruppo selezionato di processi. I processi avviati da **init** per ognuno di questi runlevel sono definiti nel file **/etc/inittab**. Init può essere in uno degli otto runlevel, [0-6, S/s].
Il runlevel viene cambiato da un utente privilegiato lanciando **/sbin/telinit**, il quale invia un segnale appropriato a init, indicandogli a quale runlevel passare.

I runlevel [0,1,6] sono riservati:

- **0** è usato per fermare il sistema, portarlo in **halt**
- **1** è usato per portare il sistema in **single user** mode
- **6** è usato per riavviare il sistema, portarlo in **reboot**

I runlevel [2-5] definiscono le normali configurazioni operative, secondo il seguente schema:

- **2** avvio del sistema in multi user mode senza rete
- **3** modalità multiutente completa, con rete
- **4** non utilizzato (definito dall'utente)
- **5** modalità multiutente completa, con rete, (con schermata di login basata sul server grafico X)

In generale, gli utenti utilizzano Linux ad un runlevel 3 o runlevel 5 (entrambe modalità multiutente). Solo alcuni utenti, a volte, personalizzano i runlevel 2 e 4 per soddisfare delle esigenze specifiche

Il runlevel **S** o **s** non è in realtà pensato per essere usato direttamente, ma più che altro per gli script che sono eseguiti quando si entra nel runlevel 1. Per maggiori informazioni su questo, si vedano le pagine di manuale di shutdown(1) e inittab(5).

Sono validi anche i runlevel 7-9, sebbene non realmente documentati. Ciò perché le varianti di Unix "tradizionali" non li usano.

Nei sistemi che utilizzano l'avvio stile SystemV, e' presente il file **/etc/inittab**. Normalmente questo file:

- contiene le informazioni necessarie per attivare le console virtuali
- specifica il runlevel di partenza
- Definisce anche altri parametri importanti, come ad esempio: la reazione del sistema alla combinazione di tasti CTRL-ALT-CANC, oppure la reazione ad un segnale *POWER DOWN* proveniente da un UPS

Applicativi (parte 2)

In questa seconda parte dedicata agli applicativi diventa d'obbligo occuparsi degli strumenti di produttività per l'ufficio. Anche perchè, usando sistemi proprietari, la suite da ufficio, dopo il sistema operativo, è quella che comporta una ulteriore spesa per la licenza, se non si vuole correre il rischio di incappare nei rigori legislativi a causa dell'utilizzo di una copia non registrata.

Open Office

Si tratta di una suite completa, multipiattaforma, a codice aperto ed ovviamente gratuita, con tutto quel che occorre per i lavori d'ufficio.
OpenOffice.org, in sigla **OOo**, è un progetto aperto che va avanti da molti anni e che è partito dalle *ceneri* di StarOffice, una suite per ufficio proprietaria acquisita, in seguito, da Sun Microsystem. Nel 2000 il codice sorgente venne rilasciato ed è nata la comunità **OpenOffice.org**, che si occupa dello sviluppo e della manutenzione della suite grazie al supporto di tantissimi sviluppatori e traduttori indipendenti e grazie all'aiuto di aziende come Novell, Red Hat, Debian, Intel e tantissime altre.

Il risultato è un gruppo di programmi gratuito e a codice aperto, disponibile in circa quaranta lingue e multipiattaforma (Windows, Linux, OsX, Solaris, etc) ed oggi anche con il pieno supporto al formato PDF ed ai formati **OpenDocument** che vengono sempre più spesso preferiti ai formati proprietari dagli enti governativi e dalle istituzioni pubbliche di moltissimi stati.

Non si commetta però l'errore frequente di considerare OpenOffice un clone di MSOffice, si tratta di 2 prodotti diversi sia nelle funzionalità sia nel tipo di pubblico a cui sono destinati, altrimenti si rischia anche di cadere in una sorta di *guerra di religione* che non avvantaggia nessuno.
Questa è una grande e definitiva scelta tra un sistema e dei formati aperti, da una parte, ed un sistema e dei formati di cui conoscono i codici e le sorti solo i proprietari, dall'altra. È la scelta di creare documenti, d'ufficio e non, che siano o meno dipendenti e quindi legati al prodotto che li ha generati.

OpenOffice è liberamente distribuibile ed utilizzabile in qualsiasi ambito: casalingo, professionale, commerciale, governativo, educativo; può essere installato su tutte le postazioni che si vuole. Farne CD distribuirli ad amici e parenti non costituisce un reato ed è anzi consigliato e desiderato da chi produce e promuove questo software.

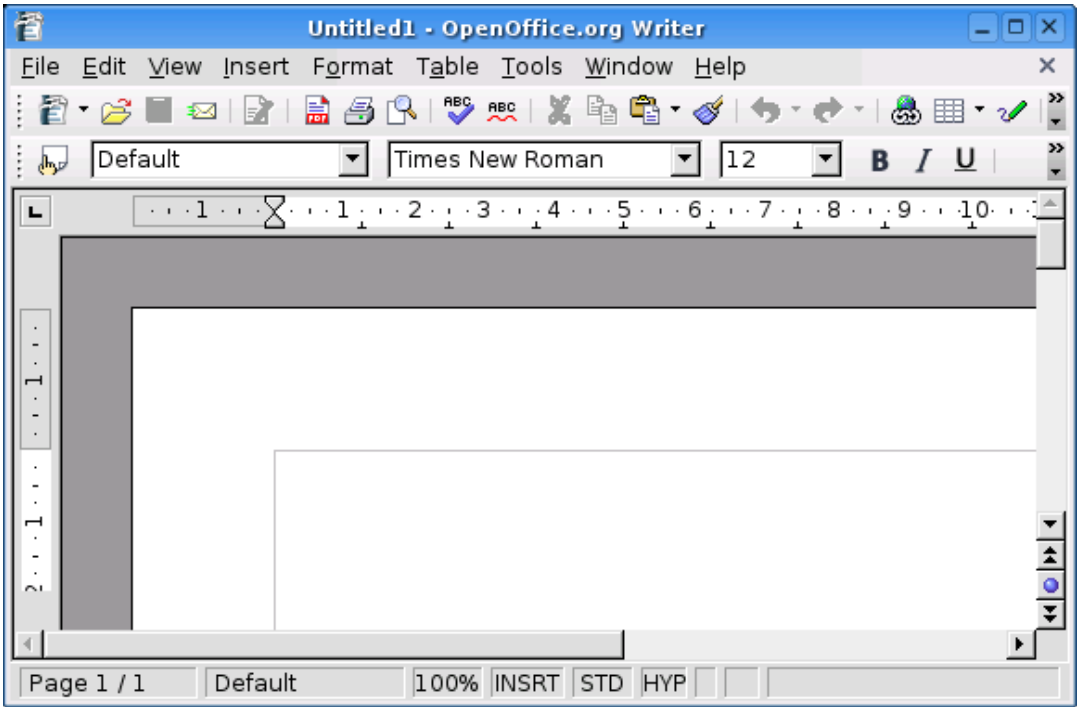
L'ultima versione, Open Office 2.0, è composto da cinque programmi principali: **Writer**, **Calc**, **Impress**, **Draw**, **Base**. Oltre a questi ne esiste anche un sesto, **Math**, che permette di creare equazioni e formule matematiche da utilizzare all'interno di presentazioni, fogli elettronici e documenti di testo. Per alcune funzioni, inoltre, **OOo** richiede la presenza di Java.

> Si rifletta sul fatto che **l'intera suite è multipiattaforma**, quindi i documenti prodotti con essa sono facilmente distribuibili **intatti** anche ad utenti che usano sistemi operativi diversi: Windows, Linux, OsX, Solaris.

Segue una brevissima rassegna dei 6 moduli di **OOo** che ha il solo scopo di presentarli molto sinteticamente, la loro conoscenza richiede infatti soprattutto un po' di impegno applicativo.

Writer

Il programma per la videoscrittura destinato alla redazione di documenti, lettere, buste e via dicendo. Sono disponibili tutte le normali funzionalità che ci si aspetta di trovare in un moderno word processor: impostazione dei margini mediante i righelli, correttore ortografico, correzione automatica, stili, colori, elenchi puntati, strumenti di ricerca e sostituzione, impostazione di instestazioni e piè pagina. Sono presenti anche stampa unione e gestione avanzata delle tabelle (anche annidate), e non mancano neanche tool appositi per creare modulistica e form.

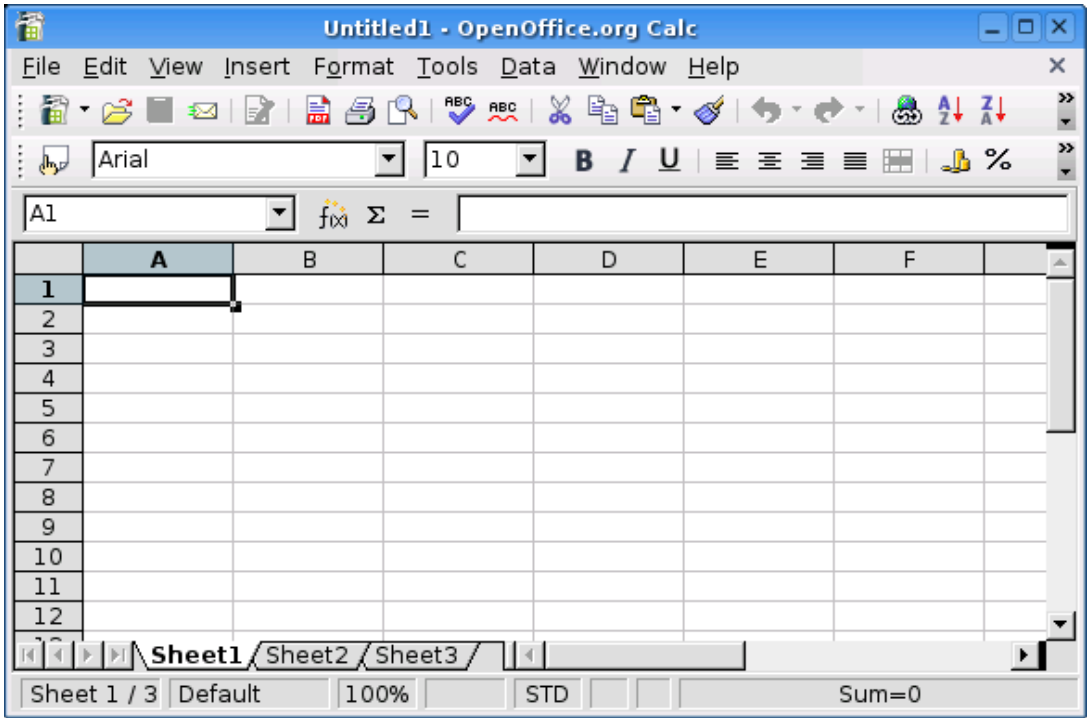


Ottima anche la gestione delle immagini, che possono essere allineate in molti modi: a filigrana, ridimensionate al volo, rese trasparenti e via dicendo. Il risultato è spesso la realizzazione di relazioni impaginate in modo impeccabile.

È possibile aprire e salvare in formato Microsoft Office con un altissima compatibilità fino alle ultime versioni, ma ricordiamo che Writer, da molti anni, permette di salvare anche in PDF. OpenOffice2 supporta e gestisce, inoltre, anche il formato **OpenDocument** apprezzato e consigliato persino dall'Unione Europea.
HTML, RTF, TXT e tantissimi altri formati arricchiscono la compatibilità di questo programma di videoscrittura gratuito. Dispone, infine, di una guida in italiano e del supporto alla firma digitale.

Calc

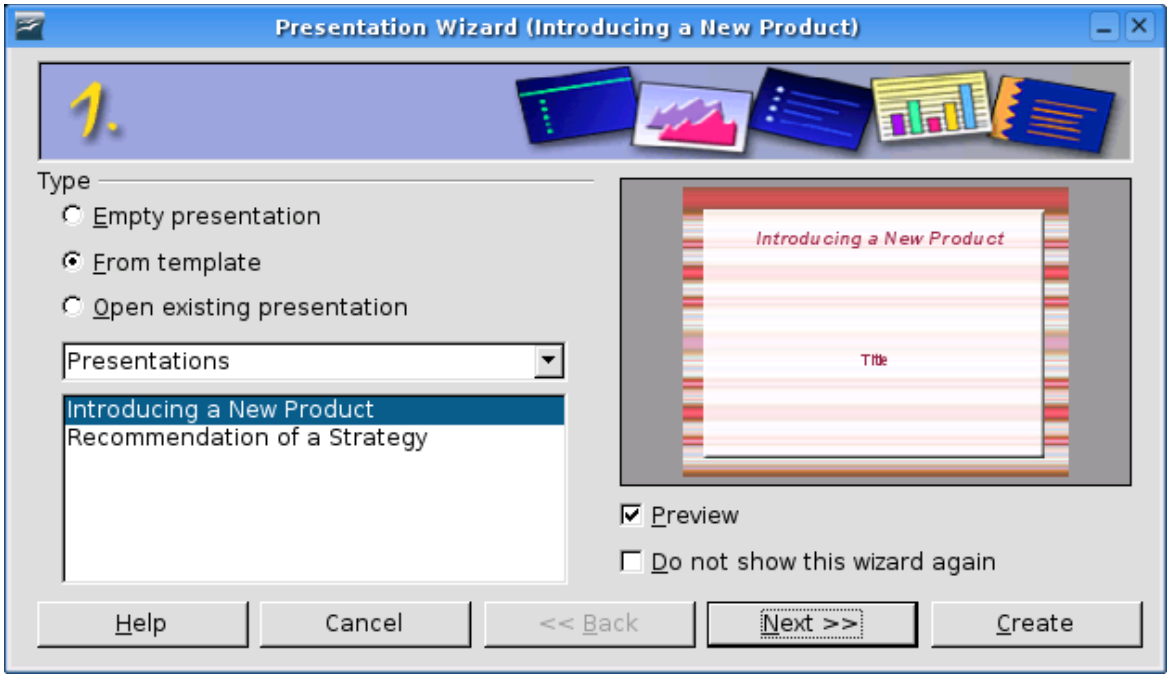
Si tratta del classico foglio di calcolo completo e flessibile. Importa alla perfezione gli spreadsheet creati con Excel, anche nel caso in cui includano grafica e/o grafici complessi. Ogni funzionalità è proprio dove ci aspetterebbe di trovarla: formato delle celle e delle formule, funzioni matematiche e strumenti per la creazione dei grafici, ed in caso di problemi l'help in linea si dimostra eccellente.



Se è vero che l'apertura di file generati da Excel è sempre ottima, anche il viceversa è possibile, ovvero creare un file xls da Calc, ed aprirlo quindi in Excel. Nonostante OpenOffice.org avvisi che il formato non è pienamente compatibile con i dati del documento e consigli di salvare invece in OpenDocument, il risultato finale è buono anche se con qualche differenza nei grafici.

Impress

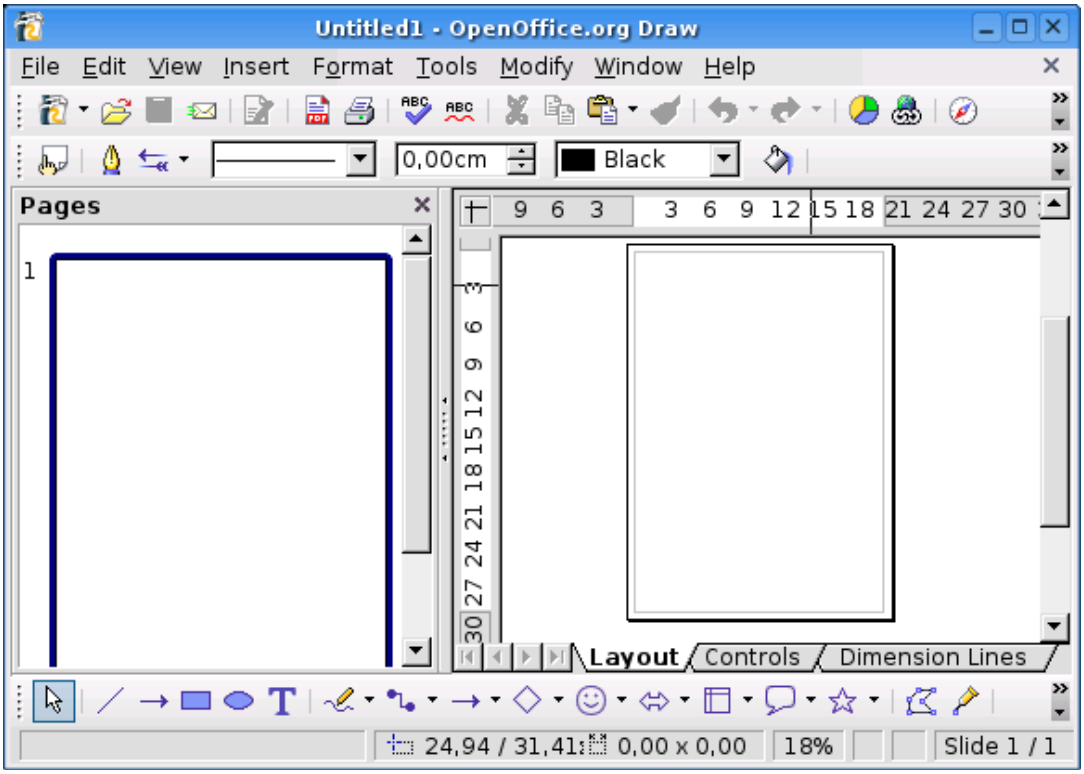
Uno strumento appositamente studiato per presentazioni multimediali con animazioni, effetti e dissolvenze. La compatibilità con Power Point lascia un poco a desiderare, ma le applicazioni interamente sviluppate con Impress assolvono abbastanza bene il loro compito.



E' anche vero che mancano i modelli, ma sono comunque presenti numerosi layout, che consentono di semplificare non di poco l'immissione di dati e testo. Come si può però constatare, lavorandoci di persona, questo componente è ancora piuttosto grezzo, e non è attualmente in grado di competere ad armi pari con PowerPoint. Non si può avere tutto, speriamo nelle prossime versioni.

Draw

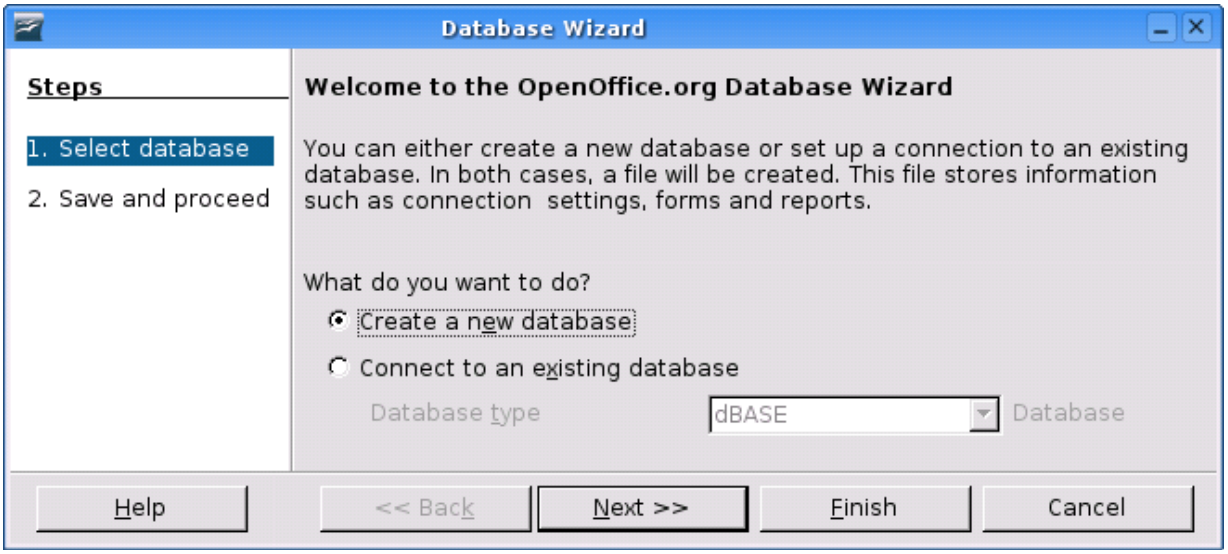
Semplice ma completo programma grafico vettoriale, con molti strumenti per creare diagrammi di flusso, grafici e *illustrazioni* aziendali. E' di fatto progettato per essere il rivale di Visio, con i cui file però non è compatibile: non è quindi possibile salvare o aprire documenti creati con il programma Microsoft all'interno di Draw.



La ricca dotazione di forme e connettori e la possibilità di importare grafica in diversi formati (gif, jpeg, bmp etc...), rendono questo programma una alternativa concreta al costoso Visio. Il mancato supporto verso il formato **vsd** dovrebbe essere valutato attentamente prima di progettare una migrazione in ambiente aziendale in cui fosse necessario aprire documenti Visio preesistenti; in tutte le alte realtà, invece, Draw si rivela un eccellente strumento di produzione.

Base

Completamente nuovo, questo programma è dedicato alla gestione ed interazione con database Adabas D, Ado, MySQL e compatibile con lo standard ODBC e JDBC. Ma Base è dedicato anche a chi vuole creare da zero un archivio di qualsiasi tipo, sfruttando il *motore* HSQLDB contenuto dentro la suite OpenOffice2, con l'utilizzo di procedure guidate, rapporti e tabelle.
Tutti i comandi sono esattamente dove ci aspetteremo di trovarli in Access, e alcune procedure guidate (i cosiddetti *wizard*) sono addirittura più semplici da usare dello stesso prodotto Microsoft.

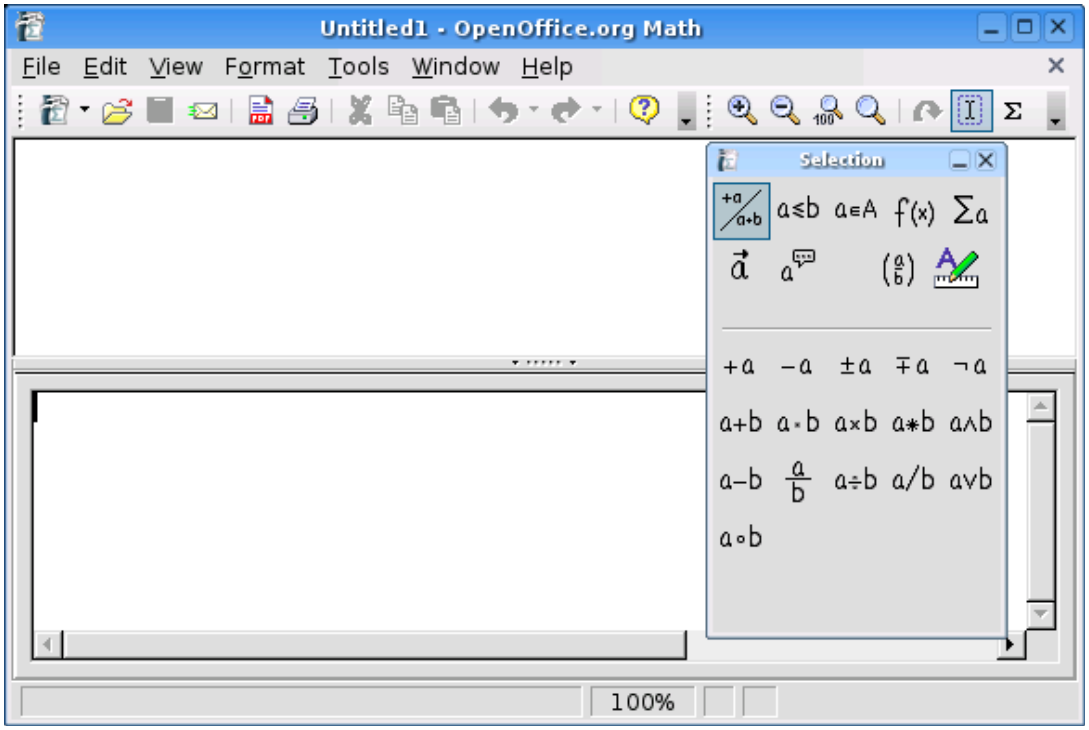


Il sistema è composto da due elementi: il motore **HSQLDB** (sviluppato tutto all'interno di Sourceforge e poi scelto da OpenOffice.org) e il **DBMS** vero e proprio. Se Base mira a fronteggiare Access come fascia di utilizzatori, le prestazioni in termini prettamente velocistici di HSQLDB *sembrano* essere di gran lunga migliori, tanto da arrivare a poter rivaleggiare con i punti di riferimento del segmento server come MySQL.

Il DBMS può connettersi, oltre che al già citato MySQL, più in generale a qualsiasi fonte dati sia dotata di un opportuno driver ODBC. Con un po' di smanettamento quindi è possibile visualizzare e modificare basi di dati in formato Access MDB anche senza aver installato il programma Microsoft, anche se l'operazione non è direttamente supportata dal comando File->Apri.
Il formato proposto per il salvataggio è OpenDocument Database, che in un solo file salva sia i dati che lo schema del database: un ottima scelta che consente di avere in un unico file l'intera base dati, facilitando la condivisione, la copia e lo spostamento fra sistemi diversi.

Math

Lo strumento ideale per inserire una formule matematiche in un documento. Math è comodo per la rappresentazione di formule e funzioni: di fatto un piccolo LaTex, per chi lo conosce.



L'applicazione si distingue per l'interfaccia utente molto intuiva ed una buona facilità d'uso. Si tratta per lo più di un programma destinato ad essere usato di rado nella realtà di tutti i giorni, se non, naturalmente, in ambito accademico e di ricerca.

Windows *Nella versione per Windows, il pacchetto completo **Open Office** sostituisce praticamente Microsoft Office del costo di oltre 500 Euro. Si tenga comunque conto che si tratta di **due prodotti diversi** anche se moltissime delle funzionalità d'ufficio lavorano allo stesso modo. La versione più recente attualmente è la 2.0.1 ed il sito di riferimento è **it.openoffice.org**.*

*Si scarica dal sito Internet la versione **.exe** relativa alla lingua italiana e si procede immediatamente con l'installazione utilizzando come cartella temporanea lo stesso Desktop. Per utilizzare tutte le funzionalità di OpenOffice.org occorre un Java Runtime Environment, che si può scaricare gratuitamente, ad esempio, da **java.com**. Java non è comunque necessario, quindi si può scegliere di non installarlo se si usano solo le funzioni più comuni.*

*Verrà chiesto se usare Openoffice come applicazione predefinita per aprire i file di Word, Excel e Powerpoint; si può rispondere di sì se sul computer non sono già installati Word, Excel e Powerpoint; comunque le due suite possono coesistere tranquillamente. Si sconsiglia, invece, di impostate OpenOffice come editor Html predefinito. Una funzione sicuramente molto apprezzata è l'esportazione dei file in **PDF**.*

*OpenOffice propone sempre di salvare nel suo formato **ODT** (o nei corrispondenti ODC e ODP); è un formato evoluto (che effettua tra l'altro una compressione automatica dei documenti) ma non può essere letto da Word (Excel/Powerpoint). Volendo si può impostare (in Strumenti/Opzioni) il formato Word come predefinito, per potere scambiare più facilmente i documenti con chi usa Word, e analogamente per Excel e Powerpoit. Tanto si può sempre utilizzare OpenOffice per leggere, aprire e modificare i file di altri formati.*

*Per quanto riguarda Access il nuovo modulo di database può sostituirlo per le applicazioni più semplici. Non si dimentichi mai che Open Office **non è un clone** di Office: gli insiemi delle funzionalità di ciascuno*

coincidono in molte parti, ma differiscono in altre.

Comandi Base (parte 7)

Passeremo in rassegna alcuni dei principali comandi attinenti i processi. Ricordiamo sinteticamente che un processo è un qualsiasi programma in esecuzione. Ad ogni processo il sistema associa un numero univoco, chiamato **pid** (process identification). Per visualizzare l'elenco dei processi attivi nel sistema basta digitare:

```
ps aux
```

che farà apparire qualcosa di simile:

```
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.0    152    72 ?        S    13:58   0:01 init [5]
root         2  0.0  0.0      0     0 ?        SN   13:58   0:00 [ksoftirqd/0]
.................................
.................................
root      2031  0.0  0.1   1772   644 ?        Ss   13:59   0:00 /sbin/syslogd
root      2038  0.0  0.3   2828  1652 ?        Ss   13:59   0:00 /sbin/klogd
root      2079  0.0  0.1   1724   652 ?        Ss   13:59   0:00 /usr/sbin/acpid -c /etc/acpi/events -
root      2160  0.0  0.1   1736   692 ?        S    13:59   0:00 /usr/sbin/automount --pid-file=/var/r
root      2165  0.0  0.5   5952  2696 ?        Ss   13:59   0:00 /usr/sbin/cupsd -F
root      2667  0.0  0.1   2116   888 ?        Ss   13:59   0:00 /usr/sbin/cron
root      2736  0.0  0.1   2756   748 ?        Ss   13:59   0:00 /usr/bin/kdm
.................................
.................................
user01    2870  0.0  0.3   5328  1624 ?        Ss   13:59   0:00 /bin/sh /usr/bin/x-session-manager
user01    2912  0.0  0.1   3116   928 ?        Ss   13:59   0:00 /usr/bin/ssh-agent x-session-manager
user01    2935  0.0  1.8  24712  9696 ?        Ss   13:59   0:00 kdeinit Running...
user01    2938  0.0  1.7  24448  8936 ?        S    13:59   0:00 dcopserver [kdeinit] --nosid
user01    2940  0.0  1.9  25672 10020 ?        S    13:59   0:00 klauncher [kdeinit]
user01    2944  0.0  2.8  53756 14688 ?        S    13:59   0:00 kded [kdeinit]
user01    2954  0.0  2.5  26816 13156 ?        S    13:59   0:00 kxkb [kdeinit]
user01    2957  0.0  2.1  25468 11316 ?        S    13:59   0:00 kaccess [kdeinit]
user01    2961  0.0  0.0   1580   336 ?        S    13:59   0:00 kwrapper ksmserver
user01    2965  0.0  2.2  25512 11464 ?        S    13:59   0:00 ksmserver [kdeinit]
user01    2966  0.0  2.6  27828 13816 ?        S    13:59   0:00 kwin [kdeinit] -session 109e697631000
user01    2969  0.0  3.2  29560 16716 ?        S    13:59   0:00 kdesktop [kdeinit]
user01    2971  0.0  3.6  36416 18696 ?        S    13:59   0:01 kicker [kdeinit]
.................................
.................................
www-data  3862  0.0  1.0  13188  5220 ?        S    14:06   0:00 /usr/sbin/apache2 -k start -DSSL
www-data  3985  0.0  0.9  13072  4692 ?        S    14:08   0:00 /usr/sbin/apache2 -k start -DSSL
user01    5192  0.4 11.5 119316 59272 ?        S    14:27   0:02 /usr/bin/../lib/Adobe/Acrobat7.0/Read
user01    5289  1.1  7.4  57008 38452 ?        S    14:28   0:05 quanta
user01    5756  0.4  2.6  27296 13836 ?        S    14:36   0:00 kio_uiserver [kdeinit]
user01    5759  0.2  2.7  32568 14328 ?        S    14:36   0:00 knotify [kdeinit]
user01    5774  2.5  2.9  29848 15252 ?        R    14:36   0:00 konsole [kdeinit]
user01    5775  2.0  0.8   7920  4304 pts/1    Ss   14:36   0:00 /bin/bash
root      5797  1.8  0.5   6332  2976 pts/1    S    14:36   0:00 -su
root      5813  0.0  0.1   4836   920 pts/1    R+   14:36   0:00 ps aux
```

L'elenco evidenzia nell'ordine l'utente proprietario del processo, il numero di identificazione del processo (il suo pid), la percentuale di potenza del processore utilizzata, la percentuale di memoria usata, altre informazioni meno importanti fino all'ultima colonna che elenca il comando che ha avviato il processo.
Tutto ciò è molto utile se un programma si blocca e non si riesce a farlo sparire con i soliti metodi. In questa eventualità il comando precedente consente di conoscere il pid ad esso associato per poi inserire:

```
kill -9 <pid>
```

L'equivalenza tra segnale e numero intero (come sopra illustrato) è mostrata digitando:

```
kill -l
```

che visualizza l'elenco dei segnali che possiamo inviare:

```
 1) SIGHUP       2) SIGINT       3) SIGQUIT      4) SIGILL
 5) SIGTRAP      6) SIGABRT      7) SIGBUS       8) SIGFPE
 9) SIGKILL     10) SIGUSR1     11) SIGSEGV     12) SIGUSR2
13) SIGPIPE     14) SIGALRM     15) SIGTERM     17) SIGCHLD
18) SIGCONT     19) SIGSTOP     20) SIGTSTP     21) SIGTTIN
22) SIGTTOU     23) SIGURG      24) SIGXCPU     25) SIGXFSZ
```

```
26) SIGVTALRM     27) SIGPROF       28) SIGWINCH      29) SIGIO
30) SIGPWR        31) SIGSYS        33) SIGRTMIN      34) SIGRTMIN+1
35) SIGRTMIN+2    36) SIGRTMIN+3    37) SIGRTMIN+4    38) SIGRTMIN+5
39) SIGRTMIN+6    40) SIGRTMIN+7    41) SIGRTMIN+8    42) SIGRTMIN+9
43) SIGRTMIN+10   44) SIGRTMIN+11   45) SIGRTMIN+12   46) SIGRTMIN+13
47) SIGRTMIN+14   48) SIGRTMIN+15   49) SIGRTMAX-15   50) SIGRTMAX-14
51) SIGRTMAX-13   52) SIGRTMAX-12   53) SIGRTMAX-11   54) SIGRTMAX-10
55) SIGRTMAX-9    56) SIGRTMAX-8    57) SIGRTMAX-7    58) SIGRTMAX-6
59) SIGRTMAX-5    60) SIGRTMAX-4    61) SIGRTMAX-3    62) SIGRTMAX-2
63) SIGRTMAX-1    64) SIGRTMAX
```

ove si vede che il segnale kill (SIGKILL) corrisponde al numero nove. Se non si riesce neanche così ad arrestare il programma impazzito allora occorre riavviare la macchina con:

```
reboot
```

Il programma killall invia, invece, un segnale a tutti i processi che eseguono i comandi specificati.

```
killall [opzioni] [-segnale] [comando...]
```

Si utilizza quindi killall per inviare un segnale a dei processi identificati per nome. Se non viene specificato il segnale da inviare, si utilizza SIGTERM. I segnali possono essere indicati per nome o per numero.
L'esempio seguente invia il segnale **SIGHUP** a tutti i processi avviati con il comando **mycmd**. I processi soggetti a questo sono solo quelli che appartengono all'utente che invia il segnale.

```
$ killall -HUP mycmd
```

Si potrebbe infine avere bisogno di sospendere temporaneamente un processo per poi riavviarlo in seguito. Debian fornisce una serie di script nella cartella /etc/init.d che permettono di avviare e fermare molti processi a comando, senza bisogno di riavviare la macchina. Esempio, per fermare il servizio web server Apache:

```
/etc/init.d/apache stop
```

Allo stesso modo per riavviare il servizio:

```
/etc/init.d/apache start
```

Invece, il comando:

```
pstree
```

ha un funzionamento molto simile a **ps**, ma rappresenta i processi attivi in un'alberatura particolare:

Il comando **top**, a differenza di **ps**, è dinamico (cioè mostra l'utilizzo delle risorse da parte dei processi a intervalli regolari) e offre funzionalità aggiuntive che possono essere di aiuto in caso di eventuali problemi. Lanciato da riga di comando:

```
top
```

produce un output sul monitor che presenta una intestazione con le informazioni generali sul sistema, ed una tabella sottostante che mostra i processi che usano più Cpu.

```
Tasks:  83 total,   2 running,  81 sleeping,   0 stopped,   0 zombie
Cpu(s):  0.3% us,  0.0% sy,  0.0% ni, 99.7% id,  0.0% wa,  0.0% hi,  0.0% si
Mem:   1034732k total,   481780k used,   552952k free,    57800k buffers
Swap:  2538228k total,        0k used,  2538228k free,   244688k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
    1 root      16   0   156   76   52 S  0.0  0.0   0:01.32 init
    2 root      34  19     0    0    0 S  0.0  0.0   0:00.04 ksoftirqd/0
    3 root      10  -5     0    0    0 S  0.0  0.0   0:00.08 events/0
    4 root      10  -5     0    0    0 S  0.0  0.0   0:00.00 khelper
    9 root      10  -5     0    0    0 S  0.0  0.0   0:00.00 kthread
   24 root      20  -5     0    0    0 S  0.0  0.0   0:00.00 kacpid
  114 root      10  -5     0    0    0 S  0.0  0.0   0:00.03 kblockd/0
  172 root      20   0     0    0    0 S  0.0  0.0   0:00.00 pdflush
  173 root      15   0     0    0    0 S  0.0  0.0   0:00.11 pdflush
  175 root      19  -5     0    0    0 S  0.0  0.0   0:00.00 aio/0
  174 root      25   0     0    0    0 S  0.0  0.0   0:00.00 kswapd0
  770 root      25   0     0    0    0 S  0.0  0.0   0:00.00 kseriod
  863 root      11  -5     0    0    0 S  0.0  0.0   0:00.00 ata/0
  870 root      16   0     0    0    0 S  0.0  0.0   0:00.00 scsi_eh_0
  871 root      17   0     0    0    0 S  0.0  0.0   0:00.00 scsi_eh_1
  872 root      17   0     0    0    0 S  0.0  0.0   0:00.00 scsi_eh_2
  895 root      11  -5     0    0    0 S  0.0  0.0   0:00.00 kcryptd/0
  896 root      11  -5     0    0    0 S  0.0  0.0   0:00.00 kmirrord/0
  937 root      10  -5     0    0    0 S  0.0  0.0   0:00.00 reiserfs/0
```

Nella prima vengono visualizzati l'ora, il tempo di attività, il numero di utenti loggati e il carico medio di sistema ogni minuto, ogni 5 minuti e ogni 15 minuti (se questi valori sono molto alti c'è qualcosa che non funziona correttamente). Inoltre sono presenti informazioni sui processi, sul processore, sulla memoria e lo swap.
Accertatevi che non ci siano processi zombie (che riducono le prestazioni della macchina) e che lo swap usato sia a 0 (zero) oppure molto basso (altrimenti abbiamo problemi di memoria).

Nella seconda ci sono le informazioni relative ai processi, ad esempio: il numero del processo (pid), il nome dell'utente proprietario del processo (user), la priorità del processo (pr), il valore nice (ni), l'utilizzo del processore (%cpu), l'utilizzo della memoria (%mem), lo stato del processo (s), il comando utilizzato per avviare il processo (command).
Il valore nice indica la priorità, da -20 (massima priorità) a 19 (minima priorità). Quindi un valore nice negativo migliora le prestazioni di esecuzione di un processo rispetto ad uno positivo.

top accetta una serie di comandi interattivi, lanciati premendo un determinato tasto della tastiera, ad esempio:

```
-k <pid>              termina il processo in questione
-r <pid>              modifica il valore nice di un processo
-f                    aggiunge/toglie alcuni campi nella tabella
-s                    cambia il tempo tra un aggiornamento e l'altro
q                     uscita da top
```

Il comando **lsof** permette di sapere quale programma abbia lanciato e controlli ogni singolo servizio. Il suo nome è una abbreviazione della descrizione del comando: **lis**t **o**pen **f**iles (elenca i files aperti).
Dato che le connessioni di rete sono rappresentate da veri e propri files, possiamo usare lsof per ottenere informazioni su di esse. Generalmente viene utilizzato un filtro come *grep* per selezionare dall'output solo le righe che contengono una certa stringa. Poniamo il caso di voler ottenere informazioni sui servizi www, il comando, dato dall'amministratore:

```
lsof -i |grep www
```

fornirà il seguente output.

```
apache2   7626        root    3u  IPv4  10276       TCP *:www (LISTEN)
apache2   7628    www-data    3u  IPv4  10276       TCP *:www (LISTEN)
```

```
apache2   7629    www-data    3u  IPv4  10276       TCP *:www (LISTEN)
```

Lo **scheduler**, una parte del kernel, organizza l'esecuzione dei processi in base ad un certo livello di priorità. Tale priorità viene calcolata dal sistema in base al tipo di processo, al suo comportamento e ad una variabile additiva che chiameremo convenzionalmente *vnice*. I valori correnti di priorità e di *vnice* per i processi possono essere visualizzati con il comando **ps -l**. Piú alto è il numero di priorità piú lentamente viene eseguito il processo. Inoltre la variabile *vnice* viene in qualche modo sommata nel calcolo della priorità, per cui priorità e *vnice* elevati implicano esecuzione piú lenta del processo. L'utente (normale o super-user) non può intervenire sulla priorità ma solo su *vnice*.

```
nice [priority] [command]
```

Nell'avviare un certo comando con **nice** si assegna anche un prefissato livello di priorità (*vnice*): -20 vuol dire massima priorità, 19 minima priorità.
Per cambiare *vnice* ad un processo già attivo si deve usare **renice**.