# Eines bàsiques per a l'administrador

Josep Jorba Esteve

P07/M2003/02282

UOC
www.uoc.edu

# Índex

**Introducció** ..... 5

**1. Eines gràfiques i línies d'ordres** ..... 7

**2. Documents d'estàndards** ..... 9

**3. Documentació del sistema en línia** ..... 12

**4. Intèrpret d'ordres i *scripts*** ..... 14
4.1. Intèrprets d'ordres interactius ..... 15
4.2. Intèrprets d'ordres disponibles ..... 18
4.3. Variables del sistema ..... 21
4.4. Programació de *scripts* en Bash ..... 22
4.4.1. Variables en Bash ..... 22
4.4.2. Comparacions ..... 23
4.4.3. Estructures de control ..... 23

**5. Eines de gestió de paquets** ..... 26
5.1. Paquet TGZ ..... 27
5.2. Fedora / Red Hat: paquets RPM ..... 29
5.3. Debian: paquets DEB ..... 33

**6. Eines genèriques d'administració** ..... 37

**7. Altres eines** ..... 39

**Activitats** ..... 41

**Altres fonts de referència i informació** ..... 41

# Introducció

Cada dia, l'administrador de sistemes GNU/Linux s'ha d'enfrontar a una gran quantitat de tasques. En general, en la filosofia UNIX no hi sol haver una única eina per a cada tasca o una sola manera de fer les coses. El normal és que els sistemes UNIX proporcionin una gran quantitat d'eines més o menys simples per afrontar les diferents tasques.

Serà la combinació de les eines bàsiques, cada una amb una tasca molt definida, la que ens donarà la possibilitat de solucionar un problema o tasca d'administració.

**Nota**

El GNU/Linux té un conjunt molt ampli d'eines amb funcionalitats bàsiques, la potència de les quals rau en la seva combinació.

En aquesta unitat veurem diferents grups d'eines, identificarem algunes de les seves funcions bàsiques i veurem alguns exemples dels seus usos. Començarem per examinar alguns estàndards del món GNU/Linux, que ens permetran trobar algunes de les característiques bàsiques que esperem de qualsevol distribució de GNU/Linux. Aquests estàndards, com l'LSB (*Linux standard base*) [Linc] i l'FHS (*file system hierarchy standard*) [Linb], ens parlen d'eines que esperem trobar disponibles, d'una estructura comuna per al sistema de fitxers, i de diverses normes que s'han de complir-se perquè una distribució sigui considerada un sistema GNU/Linux i mantingui regles comunes per a la compatibilitat entre ells.

En l'automatització de tasques d'administració se solen utilitzar ordres agrupades en *shell scripts* (també anomenades *guions d'ordres*), mitjançant llenguatges interpretats per l'intèrpret d'ordres (*shell*) del sistema. En la programació d'aquests *shell scripts*, se'ns permet unir les ordres del sistema amb estructures de control de flux i així, disposar d'un entorn de prototip ràpid d'eines per a l'automatització de tasques.

Un altre esquema habitual és la utilització d'eines de compilació i depuració de llenguatges d'alt nivell (com, per exemple, C). En general, les utilitzarà l'administrador per generar nous desenvolupaments d'aplicacions o eines, o per incorporar al sistema aplicacions que vinguin com a codi font i s'hagin d'adaptar i compilar.

També analitzarem l'ús d'algunes eines gràfiques respecte a les habituals de la línia d'ordres. Aquestes eines solen facilitar les tasques a l'administrador, però el seu ús és limitat, ja que depenen fortament de la distribució del GNU/Linux o, fins i tot, de cada versió. Tot i així, hi ha algunes eines útils que són exportables entre distribucions.

Finalment, analitzarem un grup d'eines imprescindibles per a mantenir el sistema actualitzat, les eines de gestió de paquets. El programari servit en la distribució GNU/Linux, o incorporat posteriorment, se sol oferir en unitats anomenades *paquets*, que inclouen els fitxers d'un determinat programari, més passos diversos necessaris per a la preparació de la instal·lació, la configuració posterior o, si és el cas, l'actualització o desinstal·lació d'un programari determinat. Cada distribució sol aportar programari de gestió per mantenir les llistes de paquets instal·lats o per instal·lar, a més del control de les versions o possibilitats diverses d'actualització per mitjà de diferents fonts origen.

# 1. Eines gràfiques i línies d'ordres

Hi ha un gran nombre d'eines –de les quals examinem una petita porció en aquest mòdul i els següents– que són proporcionades com a eines d'administració per tercers de manera independent a la distribució o pel mateix distribuïdor del sistema GNU/Linux.

Aquestes eines poden cobrir més o menys aspectes de l'administració d'una tasca concreta, i presentar-se amb múltiples interfícies diferents: eines de línia d'ordres amb múltiples opcions o fitxers de configuració associats, o eines textuals amb algun tipus de menú elaborat; o bé eines gràfiques amb interfícies més adequades per al maneig d'informació, o assistents que automatitzin les tasques, o interfícies web d'administració.

Tot això ens ofereix un gran nombre de possibilitats amb vista a l'administració, però sempre hem de valorar-ne la facilitat d'ús juntament amb les prestacions i els coneixements que posseeixi l'administrador que es dedica a aquestes tasques.

Les tasques habituals de l'administrador GNU/Linux poden passar per treballar amb diferents distribucions (per exemple, les que comentarem, Fedora [Fed] o Debian [Debb] o qualsevol altra) o, fins i tot, per treballar amb variants comercials de d'altres UNIX. Això comporta que hàgim d'establir una certa manera de treballar que ens permeti fer les tasques uniformement en els diferents sistemes.

Per aquesta raó, al llarg dels diferents mòduls, intentarem destacar tots aquells aspectes més comuns, i la majoria de les tècniques d'administració es realitzaran a baix nivell, mitjançant una línia d'ordres o amb edició de fitxers de configuració associats.

Qualsevol de les distribucions del GNU/Linux sol aportar eines del tipus línia d'ordres, textual o –en particular– gràfiques, que complementen les anteriors i simplifiquen més o menys l'administració de les tasques [Sm02]. Però cal tenir en compte diverses coses:

**a)** Aquestes eines són una interfície més o menys elaborada de les eines bàsiques de línia d'ordres i els corresponents fitxers de configuració.

**b)** Normalment, no ofereixen totes les prestacions o configuracions que es poden realitzar a baix nivell.

**c)** És possible que els errors no es gestionin bé o, simplement, es proporcionin missatges del tipus "la tasca no s'ha pogut realitzar".

**d)** L'ús d'aquestes eines oculta, de vegades completament, el funcionament intern del servei o tasca. Comprendre bé el funcionament intern és bàsic per a l'administrador, i més si ha de desenvolupar tasques de correcció d'errors o optimització de serveis.

**e)** Un cop l'administrador té els coneixements adequats, aquestes eines són útils per a la millora de la producció, ja que li permeten automatitzar les tasques rutinàries i manejar-les amb més eficàcia.

**f)** O, també, pot passar el cas contrari, en què la tasca sigui tan complexa o necessitar tants paràmetres o generar tantes dades, que controlar-la manualment esdevingui impossible. En aquests casos, les eines d'alt nivell poden ser molt útils i tornar practicables algunes tasques que, d'una altra manera, serien difícils de controlar. Per exemple, dins d'aquesta categoria entrarien les eines de visualització, monitoratge, i resum d'activitats o serveis complexos.

**g)** En l'automatització de tasques, és possible que aquestes eines (de nivell més alt) no siguin les adequades: no hagin estat pensades per als passos que cal fer, o bé els facin d'una manera ineficaç. Un exemple concret pot ser la creació d'usuaris. En aquest cas, una eina visual pot ser molt atraient per la manera d'introduir les dades, però, què passa quan en lloc d'introduir un usuari o uns quants en volem introduir una llista de desenes o centenars?, l'eina, si no està preparada, es torna totalment ineficient.

**h)** Per acabar, els administradors solen voler personalitzar les seves tasques utilitzant les eines que consideren més còmodes i fàcils d'adaptar. Per a això, és habitual la utilització de les eines bàsiques de baix nivell i de *shell scripts* (en veurem els fonaments en aquesta unitat) per a combinar-les de manera que formin una tasca.

Hem de saber apreciar aquestes eines extra segons la vàlua que tinguin per a les nostres tasques.

Si tenim els coneixements suficients per tractar els errors que es puguin produir, podem donar a aquestes eines un ús casual (o quotidià), o bé utilitzar-les per facilitar algun procés per al qual hagi estat pensada l'eina, però sempre havent controlat les tasques que implementem i el coneixement tècnic subjacent.

## 2. Documents d'estàndards

Els estàndards –tant el genèrics del món UNIX com els particulars del GNU/Linux– ens permeten seguir uns criteris bàsics que ens guien en el moment d'aprendre o realitzar una tasca, i ens proporcionen informació bàsica per començar el nostre treball.

En el GNU/Linux, ens podem trobar amb estàndards com l'FHS (*file system hierarchy standard*) [Linb], que ens explica què ens podem trobar (o on buscar-ho) en l'estructura del sistema de fitxers del nostre sistema; o com l'LSB (*Linux standard base*), que ens comenta diferents components que solem trobar en els sistemes [Linc].

**Nota**

Vegeu FHS a: www.pathname.com/fhs

En l'estàndard **FHS** (*file system hierchachy standard*), es descriu l'estructura d'arbre del sistema de fitxers principal (/), en què s'especifica l'estructura dels directoris i els principals fitxers que contindran. Aquest estàndard també s'utilitza més o menys per als UNIX comercials, en els quals al principi hi va haver moltes diferències que van fer que cada fabricant canviés l'estructura al seu gust. L'estàndard pensat en origen per al GNU/Linux es va fer per a normalitzar aquesta situació i evitar canvis dràstics. Tot i així, l'estàndard se segueix amb diferents graus. La majoria de distribucions segueixen l'FHS en un alt percentatge, realitzant canvis menors o aportant fitxers o directoris que no hi havia en l'estàndard.

**Nota**

L'estàndard FHS és una eina bàsica per al coneixement d'una distribució, que ens permet conèixer l'estructura i funcionalitat del sistema de fitxers principal del sistema.

Un esquema bàsic de directoris podria ser el següent:

- */bin*. Utilitats de base del sistema, normalment programes emprats pels usuaris, que inclouen des de les ordres bàsiques del sistema (com */bin/ls*, *llisteu directori*), passant pels intèrprets d'ordres *(/bin/bash*)...

- */boot*. Fitxers necessaris durant l'arrencada del sistema, per exemple la imatge del nucli Linux, en */boot/vmlinuz*.

- */dev*. Aquí trobem fitxers especials que representen els dispositius possibles en el sistema. L'accés als perifèrics en sistemes UNIX es fa com si fossin fitxers. Podem trobar fitxers com */dev/console*, */dev/modem*,*/dev/mouse*, */dev/cdrom*, */dev/floppy*... que solen ser enllaços a dispositius més específics del tipus de controlador o interfície que utilitzen els dispositius: */dev/mouse*, enllaçat a */dev/psaux*, representat per un ratolí de tipus PS2; o */dev/cdrom* a */dev/hdc*, un CD-ROM que és un dispositiu del segon connector IDE i màster. Aquí trobem els dispositius IDE com */dev/hdx*, els *scsi /dev/sdx*... amb una *x* que varia segons el número de dispositiu. Aquí, es pot comentar que al principi aquest directori era estàtic, amb els fitxers predefinits, o configurats en deter-

minats moments. Avui en dia, s'utilitzen tècniques i tecnologies dinàmiques (com *hotplug* o *udev*), que permeten detectar dispositius i crear els fitxers */dev* dinàmicament a l'inici del sistema o durant l'execució, amb la inserció de dispositius amovibles.

- */etc*. Fitxers de configuració. La majoria de tasques d'administració necessitaran examinar o modificar els fitxers continguts en aquest directori. Per exemple, */etc./passwd* conté part de la informació dels comptes dels usuaris del sistema.

- */home*. Conté els comptes dels usuaris, és a dir, els directoris personals de cada usuari.

- */lib*. Conté les biblioteques del sistema, compartides pels programes d'usuari, tant estàtiques (extensió *.a*) com dinàmiques (extensió *.so*). Per exemple, la biblioteca C estàndard, en fitxers *libc.so* o *libc.a*. També, en particular, se solen trobar els mòduls dinàmics del nucli Linux en */lib/modules*.

- */mnt*. Punt per a muntar (ordre *mount*) sistemes de fitxers de manera temporal, per exemple, */mnt/cdrom*, per a muntar un disc en el lector de CD-ROM momentàniament.

- */mitja*. Per a punt de muntatge habitual de dispositius extraïbles.

- */opt*. Se sol col·locar el programari afegit al sistema posterior a la instal·lació. Una altra instal·lació vàlida és en */usr/local*.

- */sbin*. Són utilitats de base del sistema. Solen ser ordres reservades a l'administrador (*root*); per exemple, */sbin/fsck* per a verificar l'estat dels sistemes de fitxers.

- */tmp*. Inclou fitxers temporals de les aplicacions o del propi sistema. Encara que són per a l'execució temporal, entre dues execucions l'aplicació/servei no pot assumir que trobarà els fitxers anteriors.

- */usr*. Inclou diferents elements instal·lats en el sistema. Algun programari de sistema més complet s'instal·la aquí, a més de complements multimèdia (icones, imatges, sons, per exemple en */usr/share*) i la documentació del sistema *(/usr/share/doc*). També en */usr/local* se sol utilitzar per a instal·lar programari.

- */var*. Són fitxers de registre de sessió o d'estat (fitxers de tipus *log*) o errors del propi sistema i de diversos serveis, tant locals com de xarxa. Per exemple, fitxers de sessió en */var/log*, contingut dels correus en /var/spool/mail, o treballs d'impressió en */var/spool/lpd*.

Aquests són alguns dels directoris definits en l'FHS per al sistema arrel, després s'especifiquen per exemple algunes subdivisions, com el contingut dels */usr* i */var*, i els fitxers de dades o executables típics que s'esperen trobar com a mínim en els directoris (vegeu les referències en els documents FHS).

Respecte a les distribucions, Fedora / Red Hat segueix l'estàndard FHS molt de prop. Només presenta alguns canvis en els fitxers presents en */usr*, */var*. En hi sol haver un directori per component configurable, i en */opt*, */usr/local* no hi acostuma a haver programari instal·lat, tret que l'usuari l'instal·li. Debian, per la seva part, segueix l'estàndard, encara que afegeix alguns directoris de configuració especials en */etc*.

**Nota**

http://www.linuxbase.org/spec/refspecs/LSB_1.3.0/gLSB/gLSB/rstandards.html

Un altre estàndard en procés és l'LSB (*Linux standard base*) [Linc]. La idea d'aquest és definir uns nivells de compatibilitat entre les aplicacions, biblioteques i utilitats, de manera que sigui possible la portabilitat de les aplicacions entre distribucions sense gaires problemes. A més de l'estàndard, proporcionen conjunts de prova (tests) per verificar el nivell de compatibilitat. L'LSB en si mateix és una recopilació de diversos estàndards aplicats al GNU/Linux.

**Nota**

www.linuxbase.org/spec/

# 3. Documentació del sistema en línia

Un dels aspectes més importants per a les nostres tasques d'administració serà disposar de la documentació correcta per al nostre sistema i el programari instal·lat. Hi ha moltes fonts d'informació, però destacarem les següents:

**a)** *man* és l'ajuda per excel·lència. Ens permet consultar el manual del GNU/Linux, que està agrupat en diverses seccions, corresponents a ordres d'administració, formats de fitxers, ordres d'usuari, crides de llenguatge C, etc. Normalment, per a obtenir l'ajuda associada, n'hi haurà prou amb el següent:

```
man ordre
```

Cada pàgina descriuria l'ordre juntament amb les seves opcions i, normalment, alguns exemples d'utilització. De vegades, hi pot haver més d'una entrada en el manual. Per exemple, pot ser que hi hagi una crida C amb el mateix nom que una ordre; en aquest cas, cal especificar la secció que es vol mirar:

```
man n ordre
```

sent *n* el número de secció.

També hi ha unes quantes eines d'exploració dels manuals, per exemple *xman* i *tkman* que, mitjançant interfície gràfica, faciliten l'examen de les diferents seccions i índexs de les ordres. Una altra ordre interessant és *apropos* paraula, que ens permetrà localitzar pàgines *man* que parlin d'un tema determinat (associat amb la paraula buscada).

**b)** *info* és un altre sistema d'ajuda habitual. És un programa desenvolupat pel GNU per a la documentació de moltes de les seves eines. Bàsicament, és una eina textual en què els capítols i pàgines es poden recórrer per mitjà d'un sistema de navegació simple (basat en el teclat).

**c)** A més de certes pàgines *man*, és habitual incloure en les aplicacions documentació extra, en forma de guies d'usuari manuals o tutorials, o simples. Normalment, aquests components de documentació s'instal·len en el directori */usr/share/doc* (o */usr/doc* depenent de la distribució), en què normalment es crea un directori per paquet d'aplicació (normalment, l'aplicació pot disposar de paquet de documentació separadament).

**d)** Les distribucions tenen sistemes propis. Red Hat sol venir amb uns CD de manuals de consulta que són instal·lables en el sistema i tenen formats HTML o PDF. Fedora disposa d'un projecte de documentació en el seu web.

Debian porta els manuals com un paquet de programari més i solen instal·lar-se en */usr/doc*. D'altra banda, disposa d'eines que classifiquen la documentació present en el sistema, i l'organitzen per menús per a la seva visualització, com *dwww* o *dhelp*, les quals presenten interfícies web per a examinar la documentació del sistema.

**e)** Finalment, els escriptoris X, com Gnome i KDE, també acostumen a portar sistemes de documentació propis amb la seva documentació i manuals, i informació per a desenvolupadors, en forma d'ajuts gràfics en les seves aplicacions, o en aplicacions pròpies que recopilen els ajuts (per exemple *devhelp* a Gnome).

# 4. Intèrpret d'ordres i *scripts*

El terme genèric *shell* (intèrpret d'ordres) s'utilitza per denominar un programa que serveix d'interfície entre l'usuari i el nucli (*kernel*) del sistema GNU/Linux. En aquest apartat, ens centrarem en els intèrprets d'ordres interactius de text, que seran els que ens trobarem com a usuaris una vegada estiguem validats i dins del sistema.

L'intèrpret d'ordres, com a programa, és una utilitat de sistema, que permet als usuaris interactuar amb el nucli per interpretació d'ordres que el mateix usuari introdueix en la línia d'ordres o en els fitxers de tipus *shell script*.

L'intèrpret d'ordres és el que els usuaris veuen del sistema. La resta del sistema operatiu roman essencialment ocult per als seus ulls. L'intèrpret d'ordres està escrit de la mateixa manera que un procés (programa) d'usuari; no està integrat en el nucli, sinó que s'executa com un programa més de l'usuari.

Quan el nostre sistema GNU/Linux arrenca, sol presentar als usuaris una interfície de cara determinada; aquesta interfície tant pot ser de text com gràfica. Això depèn dels modes (o nivells) d'arrencada del sistema, amb els diferents modes de consola de text o amb modes en què, directament, tinguem arrencada gràfica en X Window.

En els modes d'arrencada gràfica, la interfície està formada per algun administrador d'accés que gestiona el procés de *login* de l'usuari des d'una "caràtula" gràfica en què se li demana la informació d'entrada corresponent: el seu identificador com a usuari i el seu mot d'accés (o *password*). En el GNU/Linux són habituals els gestors d'accés: *xdm* (propi de l'X Window), *gdm* (Gnome) i *kdm* (KDE), i algun altre associat a diferents gestors de finestres *(window managers*). Una vegada validat el nostre accés, ens trobarem dins de la interfície gràfica de l'X Window amb algun gestor de finestres, com Gnome o KDE. Per a interactuar des d'un intèrpret d'ordres interactiu, només haurem d'obrir algun dels programes d'emulació de terminal disponibles.

Si el nostre accés és per mode consola (en text), una vegada validats obtindrem l'accés directe a l'intèrpret d'ordres interactiu.

Un altre cas d'obtenció d'un intèrpret d'ordres interactiu és l'accés remot a la màquina, tant per qualsevol de les possibilitats de text com Telnet, *rlogin*, *ssh*, com per gràfiques com els emuladors X Window.

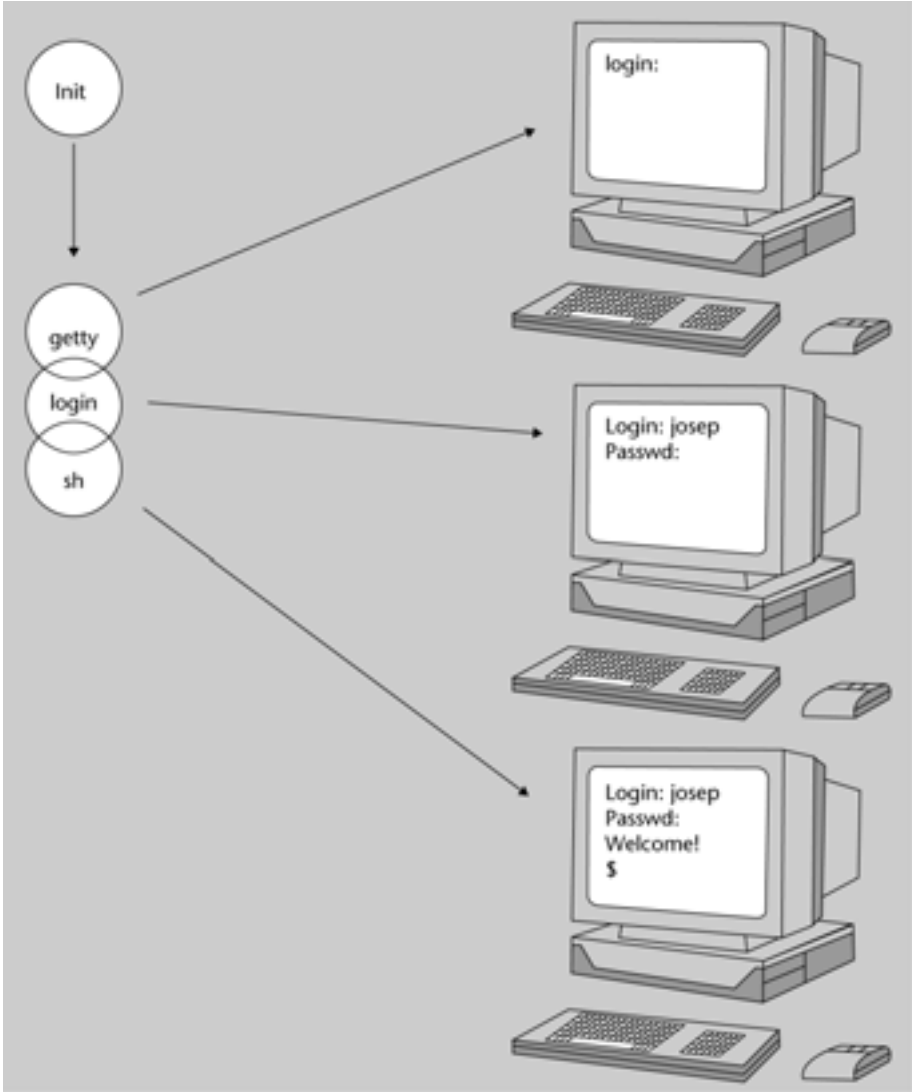


Figura 1. Exemple d'arrencada *shell* textual i processos de sistema involucrats [Oke]

### 4.1. Intèrprets d'ordres interactius

Una vegada iniciat l'intèrpret d'ordres interactiu [Qui01], es mostra un *prompt* de cara a l'usuari, que li indica que pot introduir una línia d'ordre. Després de la introducció, el *shell* assumeix la responsabilitat de validar-la i executar els processos necessaris mitjançant una sèrie de fases:

- Llegir i interpretar la línia d'ordres.
- Avaluar els caràcters comodí com $ * ? o d'altres.
- Gestionar les redireccions d'E/S necessàries, els *pipes* i els processos en segon pla (*background*) necessaris (&).
- Manejar senyals.
- Preparar l'execució dels programes.

Normalment, les línies d'ordres podran ser execucions d'ordres del sistema, ordres pròpies de l'intèrpret d'ordres interactiu, posada en marxa d'aplicacions o *shell scripts*.

Els *shell scripts* són fitxers de text que contenen seqüències d'ordres de sistema, més una sèrie d'ordres pròpies internes de l'intèrpret d'ordres interactiu, més les estructures de control necessàries per a processar el flux del programa (tipus *while, for*, etc.).

Els fitxers *script* són directament executables pel sistema amb el nom que s'hagi donat al fitxer. Per executar-los, es crida l'intèrpret d'ordres juntament amb el nom del fitxer, o bé es donen permisos d'execució al *shell script*.

En certa manera, podem veure el *shell script* com un codi d'un llenguatge interpretat que s'executa sobre l'intèrpret d'ordres interactiu corresponent. Per a l'administrador, els *shell scripts* són molt importants bàsicament per dues raons:

**1)** La configuració del sistema i de la majoria dels serveis proporcionats es realitzen mitjançant eines proporcionades en forma de *shell script*s.

**2)** La manera principal d'automatitzar processos d'administració és mitjançant la creació de *shell script*s per part de l'administrador.

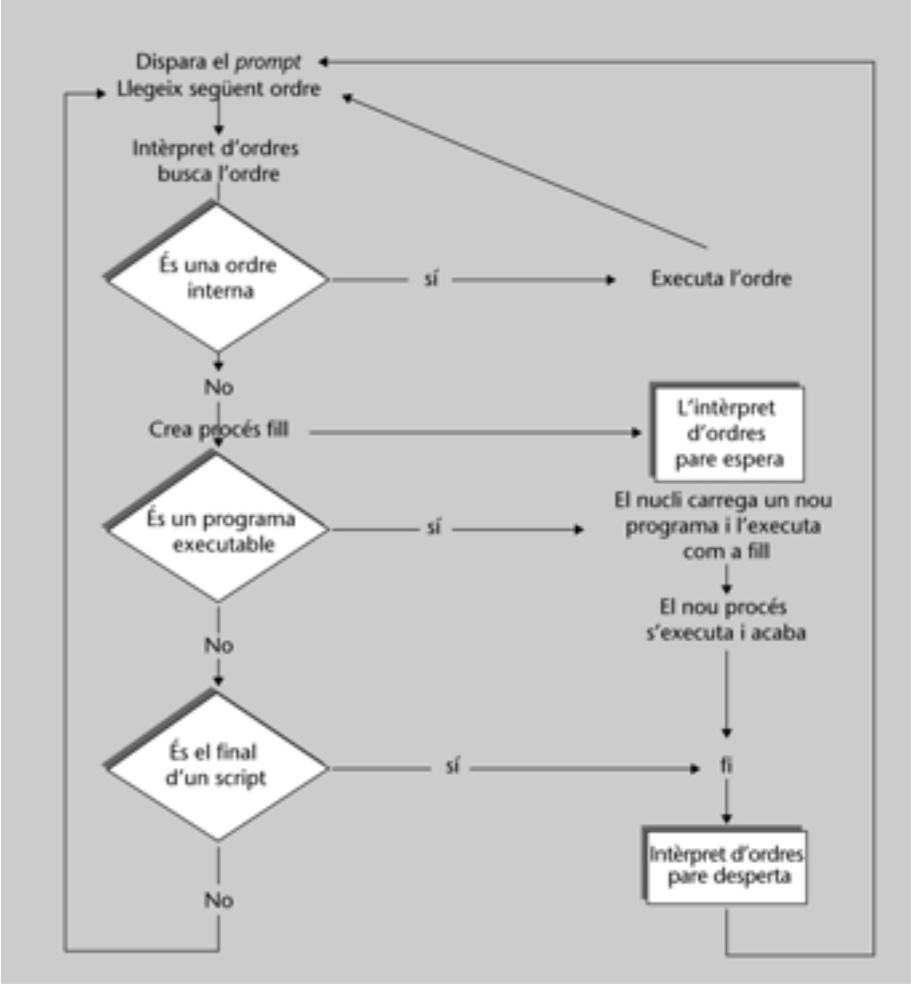


Figura 2. Flux de control bàsic d'un intèrpret d'ordres

Tots els programes cridats mitjançant un intèrpret d'ordres inclouen tres fitxers predefinits, especificats pels descriptors de fitxers (*file handles*) corresponents. Per defecte, aquests fitxers són els següents:

**1)** Entrada estàndard (*standard input*). Normalment, està assignada al teclat del terminal (consola). Utilitza el descriptor número 0 (en el UNIX els fitxers utilitzen *descriptors* sencers).

**2)** Sortida estàndard (*standard output*). Normalment, està assignada a la pantalla del terminal. Utilitza el descriptor 1.

**3)** Sortida estàndard d'errors (*standard error*). Normalment, està assignada a la pantalla del terminal. Utilitza el descriptor 2.

Això ens indica que, per defecte, qualsevol programa executat des de l'intèrpret d'ordres tindrà l'entrada associada al teclat del terminal, la seva sortida cap a la pantalla i, si es produeixen errors, també els enviarà a la pantalla.

A més, els intèrprets d'ordres solen proporcionar els tres mecanismes següents:

**1) Redirecció.** Atès que els dispositius d'E/S i els fitxers es tracten de la mateixa manera en el UNIX, l'intèrpret d'ordres els tracta a tots simplement com a fitxers. Des del punt de vista de l'usuari, es poden reassignar els descriptors dels fitxers perquè els fluxos de dades d'un descriptor vagin a qualsevol altre descriptor; això s'anomena *redirecció* Per exemple, ens referirem a la redirecció dels descriptors 0 o 1 com a la redirecció de l'E/S estàndard.

**2) Canonades** (*pipes*). La sortida estàndard d'un programa es pot utilitzar com a entrada estàndard d'un altre per mitjà de canonades. Es poden interconnectar diversos programes mitjançant canonades per a formar el que es denomina un *pipeline*.

**3) Concurrència** de programes d'usuari. Els usuaris poden executar diversos programes simultàniament, indicant que la seva execució es produirà en segon pla (*background*), en termes oposats a primer pla (*foreground*), en què es té un control exclusiu de pantalla. Una altra utilització consisteix a permetre treballs llargs en segon pla quan interactuem amb l'intèrpret d'ordres amb altres programes en primer pla.

En els intèrprets d'ordres UNIX/Linux, en la pràctica, aquests tres aspectes suposen el següent:

- **Redirecció.** Una ordre pot rebre la seva entrada o sortida des d'altres fitxers o dispositius.

  **Exemple**

  Vegem

  ```
  ordre op fitxer
  ```

en què `op pot` ser el següent:

- < : rebre entrada del fitxer.
- > : enviar sortida al fitxer.
- >> : indica que s'afegeixi la sortida (per defecte, amb > es torna a crear el fitxer).

- *Pipes*: Hi ha un encadenament de diverses ordres, amb transmissió de les seves dades:

  ```
  ordre1 | ordre2 | ordre3
  ```

  Aquesta instrucció ens indica que ordre1 rebrà entrada possiblement de teclat, enviarà la seva sortida a ordre2, que la rebrà com a entrada, i aquesta produirà sortida cap a ordre3, que la rep i produeix la seva sortida cap a una sortida estàndard (la pantalla, per defecte).

- **Concurrència en segon pla.** Qualsevol ordre executada amb el signe & al final de línia s'executa en segon pla i el *prompt* de l'intèrpret d'ordres es torna immediatament mentre en continua l'execució. Podem seguir l'execució de les ordres amb l'ordre *ps* i les seves opcions, que ens permet veure l'estat dels processos en el sistema. I també disposem de l'ordre *kill*, que ens permet eliminar processos que encara s'estiguin executant o que hagin entrat en alguna condició d'error: *kill 9 pid* permet "matar" el procés número pid. pid és l'identificador associat al procés, un nombre enter que el sistema li assigna i que es pot obtenir amb l'ordre *ps*.

### 4.2. Intèrprets d'ordres disponibles

La independència de l'intèrpret d'ordres respecte al nucli de l'operatiu (l'intèrpret d'ordres només és una capa d'interfície) ens permet disposar d'uns quants en el sistema [Qui01]. Alguns dels més comuns són els següents:

**a)** L'intèrpret d'ordres Bash (*bash*). L'intèrpret d'ordres GNU/Linux per defecte.

**Nota**

El *shell* és un programa més d'usuari. Hi ha diferents possibilitats d'elecció, amb funcionalitats i prestacions diferents.

**b)** L'intèrpret d'ordres Bourne (*sh*). Aquest ha estat l'estàndard *shell* UNIX des de sempre, i el que tots els UNIX inclouen en alguna versió. Normalment, és l'intèrpret d'ordres de l'administrador (*root*) per defecte. En el GNU/Linux sol ser l'anterior Bash, una versió millorada del Bourne. L'*sh* va ser creat per Stephen Bourne en AT&T al final dels setanta. L'indicador (*prompt*) per defecte sol ser un $ (en *root* un #).

**c)** L'intèrpret d'ordres Korn (*ksh*). És un superconjunt del Bourne (es manté certa compatibilitat), escrit en AT&T per David Korn (a mitjan anys vuitanta), en el qual es va fer certa barreja de funcionalitats del Bourne i del C, més algun afegit. El *prompt* per defecte és el $.

**d)** L'intèrpret d'ordres C (*csh*). Va ser desenvolupat a la Universitat de Berkeley per Bill Joy al final dels setanta i té uns quants afegits interessants en el Bourne, com un històric d'ordres, àlies, aritmètica des de la línia d'ordres, i completa noms de fitxers i controla treballs en segon pla. El *prompt* per defecte per als usuaris és % . Els usuaris UNIX solen preferir aquest intèrpret d'ordres com a interactiu, però els administradors UNIX prefereixen utilitzar el Bourne, perquè els *scripts* solen quedar més compactes, i l'execució sol ser més ràpida. D'altra banda, un avantatge dels *scripts* en C*shell* és que, com el seu nom indica, tenen una sintaxi està basada en el llenguatge C (encara que no igual).

**e)** Altres, com versions restringides o especialitzades dels anteriors.

L'intèrpret d'ordres Bash (*Bourne again shell*) [Bas] [Coo] ha adquirit importància des de la seva inclusió en els sistemes GNU/Linux com a intèrpret d'ordres per defecte. Aquest intèrpret d'ordres forma part del programari del projecte GNU. És un intent de combinar els tres intèrprets d'ordres anteriors (Bourne, C i Korn), mantenint la sintaxi de l'intèrpret d'ordres Bourne original. Ens fixarem en aquest per desenvolupar exemples posteriors.

Una manera ràpida de saber quin intèrpret d'ordres tenim com a usuaris és mitjançant la variable $*SHELL*, des d'una línia d'ordres amb la instrucció següent:

```
echo $SHELL
```

Vegem algunes característiques comunes a tots els intèrprets d'ordres:

- Tots permeten l'escriptura de *shell scripts*, que després s'interpreten i s'executen bé pel nom (si el fitxer té permís d'execució) o bé passant-lo com a paràmetre a l'ordre de l'intèrpret d'ordres.

- Els usuaris del sistema tenen un intèrpret d'ordres per defecte associat a ells. Aquesta informació es proporciona en crear els comptes dels usuaris. L'administrador assigna un intèrpret d'ordres a cada usuari o, si no, s'assigna l'intèrpret d'ordres per defecte (*bash* en el GNU/Linux). Aquesta informació es desa en fitxer de contrasenyes en */etc/passwd,* i es pot canviar amb l'ordre *chsh,* aquesta mateixa ordre amb opció -l ens llista els intèrprets d'ordres disponibles en el sistema (vegeu també */etc/shells*).

- En realitat, cada intèrpret d'ordres és una ordre executable, normalment present en els directoris */bin* en el GNU/Linux (o */usr/bin*).

- Es poden escriure *shell scripts* en qualsevol d'ells, però ajustant-se a la sintaxi de cada un, que és normalment diferent (de vegades, només hi ha diferències petites). La sintaxi de les construccions i les ordres internes estan documentades en la pàgina *man* de cada intèrpret d'ordres (man *bash*, per exemple).

- Cada intèrpret d'ordres té alguns fitxers d'arrencada associats (fitxers d'inicialització), que cada usuari pot adaptar a les seves necessitats, incloent-hi codi, variables, camins (*path*)...

- La potència en la programació s'adquireix en combinar la sintaxi de cada intèrpret d'ordres (de les seves construccions) amb les ordres internes de cada intèrpret d'ordres i una sèrie d'ordres UNIX molt utilitzades en els *scripts* com, per exemple, els *cut*, *sort*, *cat*, *more*, *echo*, *grep*, *wc*, awk, *sed*, *mv*, *ls*, *cp*...

**Nota**

Per a la programació *shell* és recomanable tenir un bon coneixement d'aquestes ordres UNIX i les seves diferents opcions.

- Si com a usuaris utilitzem un intèrpret d'ordres determinat, res no impedeix arrencar una còpia nova d'intèrpret d'ordres (l'anomenem *subshell*), que pot ser el mateix o un altre de diferent. Senzillament, el cridem pel nom de l'executable, que pot ser l'*sh*, *bash*, *csh* o *ksh*. També, quan executem un intèrpret d'ordres *script*, es llança un *subshell* amb l'intèrpret d'ordres que correspongui per a executar l'*script* demanat.

Algunes diferències bàsiques entre ells [Qui01]:

**a)** Bash és l'intèrpret d'ordres per defecte en el GNU/Linux (si no s'especifica el contrari en crear el compte de l'usuari). En altres sistemes UNIX sol ser l'intèrpret d'ordres Bourne (*sh*). Bash és compatible amb *sh* i, a més, incorpora algunes característiques dels altres intèrprets d'ordres, *csh* i *ksh*.

**b)** Els fitxers d'arrencada *sh* i *ksh* tenen *.profile* (en el compte de l'usuari, i s'executa en el *login* de l'usuari) i, també, *ksh* sol tenir un *.kshrc* que s'executa a continuació; *csh* utilitza *.login* (s'executa en iniciar-se el *login* de l'usuari una sola vegada), *.logout* (abans de la sortida de la sessió de l'usuari) i *.cshrc* (semblant al *.profile*, en cada sub*shell* C que s'inicia). I Bash utilitza el *.bashrc* i el *.bash_profile*. A més, l'administrador pot col·locar variables i camins comuns en el fitxer*/etc/profile* que s'executarà abans que els fitxers que tingui cada usuari. Els fitxers d'inicialització dels intèrprets d'ordres es posen en el compte de l'usuari en crear-lo (normalment es copien del directori */etc./skel*), en què l'administrador pot deixar uns esquelets dels fitxers preparats.

**c)** Els *scripts* de configuració del sistema o de serveis solen estar escrits en l'intèrpret d'ordres Bourne (*sh*), ja que la majoria dels UNIX els utilitzaven així. En el GNU/Linux també ens podem trobar amb alguns en el Bash, i també en altres llenguatges de *script* no associats als intèrprets d'ordres com *perl* o *python*.

**d)** Podem identificar en quin *shell* s'executa l'*script* mitjançant l'ordre *file* (per exemple, *file <nomscript>*), o bé examinant la primera línia de l'*script* (que sol ser *#!/bin/nom*, en què nom és *bash, sh, csh, ksh*és...). En el moment d'executar l'*script*, aquesta línia li diu quin intèrpret d'ordres cal utilitzar a l'hora d'interpretar-lo (és a dir, quin *subshell* cal llançar per executar-lo). És important que tots els *scripts* la continguin, ja que, si no, s'intentaran executar en l'intèrpret d'ordres per defecte (Bash, en el nostre cas), i la sintaxi pot no correspondre i causar molts errors sintàctics en l'execució.

### 4.3. Variables del sistema

Algunes variables de sistema útils (les podem veure, per exemple, amb l'ordre *echo*) que es poden consultar en la línia d'ordres o en la programació de *shell scripts* són:

| Variable | Valor Exemple | Descripció |
|---|---|---|
| HOME | /home/juan | Directori arrel de l'usuari |
| LOGNAME | Juan | *Id* de l'usuari en el *login* |
| PATH | /bin:/usr/local/bin:/usr/X11/bin | Camins |
| SHELL | /bin/bash | Intèrpret d'ordres de l'usuari |
| PS1 | $ | *Prompt* de l'intèrpret d'ordres, l'usuari el pot canviar |
| MAIL | /var/mail/juan | Directori de la bústia de correu |
| TERM | xterm | Tipus de terminal que l'usuari utilitza |
| PWD | /home/juan | Directori actual de l'usuari |

Les diferents variables de l'entorn es poden veure amb l'ordre *env*. Per exemple:

```
$ env
SSH_AGENT_PID = 598
MM_CHARSET = ISO-8859-15
TERM = xterm
DESKTOP_STARTUP_ID =
SHELL = /bin/bash
WINDOWID = 20975847
LC_ALL = es_ES@euro
USER = juan
LS_COLORS = no = 00:fi = 00:di = 01;34:ln = 01;
SSH_AUTH_SOCK = /tmp/ssh-wJzVY570/agent.570
SESSION_MANAGER = local/aopcjj:/tmp/.ICE-unix/570
USERNAME = juan
PATH=/soft/jdk/bin:/usr/local/bin:/usr/bin:/bin:/usr/bin/
X11:/usr/games
MAIL = /var/mail/juan
PWD = /etc/skel
JAVA_HOME = /soft/jdk
LANG = es_ES@euro
GDMSESSION = Gnome
JDK_HOME = /soft/jdk
SHLVL = 1
HOME = /home/juan
GNOME_DESKTOP_SESSION_ID = Default
LOGNAME = juan
DISPLAY = :0.0
COLORTERM = gnome-terminal
XAUTHORITY = /home/juan/.Xauthority
_ = /usr/bin/env
OLDPWD = /etc
```

## 4.4. Programació de *scripts* en Bash

Aquí veurem alguns conceptes bàsics dels *shell scripts* en Bash. Recomanem ampliar en [Bas] [Coo].

Tots els *scripts* Bash han de començar amb la línia següent:

```
#!/bin/bash
```

Aquesta línia indica, a l'intèrpret d'ordres que utilitza l'usuari que tinguem actiu en aquest moment, amb quin intèrpret d'ordres s'ha d'executar l'*script* que aparegui a continuació.

L'*script* es pot executar de dues maneres diferents:

**1)** Per execució directa des de la línia d'ordres, sempre que tingui permís d'execució. Si no és així, posem el permís amb *chmod +x script*.

**2)** Per execució mitjançant l'intèrpret d'ordres, cridem l'intèrpret d'ordres explícitament: */bin/bash script*.

Cal tenir en compte que, independentment del mètode d'execució, sempre creem un *subshell* en què s'executarà el nostre *script*.

### 4.4.1. Variables en Bash

L'assignació de variables es realitza de la manera següent:

```
variable = valor
```

El valor de la variable es pot veure amb el següent:

```
echo $variable
```

en què $ ens fa referència al valor de la variable.

La variable per defecte només és visible en l'*script* (o en l'intèrpret d'ordres). Si la variable ha de ser visible fora de l'*script*, al nivell de l'intèrpret d'ordres o de qualsevol intèrpret d'ordres fill (o *subshell*) que es generi *a posteriori*, serà necessari "exportar-la" a més d'assignar-la. Podem fer dues coses:

- Assignar i exportar després:

  ```
  var = valor

  export var
  ```

- Exportar en l'assignació:

  ```
  export var = valor
  ```

En els *scripts* Bash, tenim algunes variables predeterminades accessibles:

- $1-$N : Desa els arguments passats com a paràmetres en l'*script* des de la línia d'ordres.
- $0 : Desa el nom de l'*script*. Seria el paràmetre 0 de la línia d'ordres.
- $* : Desa tots els paràmetres de l'1 a la N en aquesta variable.
- $ : Desa tots els paràmetres, però amb cometes dobles (" ") en cada un.
- $? : "Status". Desa el valor tornat per l'última ordre executada. És útil per a verificar condicions d'error, ja que el UNIX sol tornar 0 si l'execució ha estat correcta, i un valor diferent com a codi d'error.

Una altra qüestió important en les assignacions és l'ús de les cometes:

- Les dobles (" ") permeten que sigui considerat tot com una unitat.
- Les simples (' ') són semblants, però s'ignoren els caràcters especials que hi hagi a dins.
- Les inclinades cap a l'esquerra ('ordre') s'utilitzen per a avaluar l'interior, si hi ha alguna execució o substitució per fer. Primer s'executa el contingut, i se substitueix el que hi havia pel resultat de l'execució. Per exemple: var ?=`ls` desa la llista del directori en *$var*.

### 4.4.2. Comparacions

Per a les condicions se sol utilitzar l'ordre *test expressió* o directament *[expressió]*. Podem agrupar les condicions disponibles de les maneres següents:

- Comparació numèrica. *-eq -ge, -gt -le, -lt, -ne,* que corresponen a *igual que*, *més gran o igual que* (*ge*), *més gran que*, *menor o igual que* (*le*), *menor que*, *diferent de*.
- Comparació de cadenes. := , ! = , *-n, -z*, que corresponen a cadenes de caràcters *iguals*, *diferents*, *amb longitud major que 0*, *longitud igual a zero o buit*.
- Comparació de fitxers. *-d*, *-f*, *-r*, *-s*, *-w*, *-x*. El fitxer és *un directori*, *un fitxer ordinari*, *és llegible*, *és no buit*, *és escribible*, *és executable*.
- Booleans entre expressions. !, -a, -o, condicions de *not*, *and* i *or*.

### 4.4.3. Estructures de control

Respecte a la programació interna de l'*script*, hem de pensar que, bàsicament, ens trobarem amb el següent:

- Ordres pròpies de l'operatiu.
- Ordres pròpies internes de l'intèrpret d'ordres Bash (vegeu *man* bash).
- Les estructures de control pròpies de programació (*for*, *while*...), amb la sintaxi pròpia de Bash.

La sintaxi bàsica de les estructures de control és la següent:

**a)** Estructura *if...then*. S'avalua l'expressió i si s'obté un valor cert, llavors s'executen els *commands*.

```
if [ expresion ]
  then
    commands
fi
```

**b)** Estructura *if..then...else*. S'avalua l'expressió i, si s'obté un valor de cert, s'executen els commands1. En cas contrari, s'executen els comands2.

```
if [ expresion ]
  then
    commands1
  else
    commands2
fi
```

**c)** Estructura *if..then...else if...else*. Sutilitza igual que l'anterior, amb implantacions d'estructures *if*.

```
if [ expresion ]
  then
    commands
  elif [ expresion2 ]
  then
    commands
  else
    commands
fi
```

**d)** Estructura*case select*. És una estructura de selecció múltiple segons el valor de selecció (en *case*).

```
case string1 in
  str1)
    commands;;
  str2)
    commands;;
  *)
    commands;;
esac
```

**Nota**

Els intèrprets d'ordres com Bash ofereixen un conjunt ampli d'estructures de control que els fa comparables a qualsevol altre llenguatge.

**e)** Bucle, *for*. Substitueix la variable per cada element de la llista.

```
for var1 in list
do
   commands
done
```

**f)** Bucle, while. Mentre es compleixi l'expressió

```
while [ expresion ]
do
   commands
done
```

**g)** Bucle, *until*. Fins que es compleixi l'expressió

```
until [ expresion ]
do
   commands
done
```

**h)** Declaració de funcions.

```
fname() {
   commands
}
```

o bé amb crida acompanyada de paràmetres

```
fname2(arg1,arg2...argN) {
   commands
}
```

i la crida de la funció amb *fname* o *fname2 p1 p2 p3... pN*.

# 5. Eines de gestió de paquets

En qualsevol distribució, els paquets són l'element bàsic per a les tasques d'instal·lació de nou programari, actualització o eliminació del que no s'utilitza.

Bàsicament, un paquet és un conjunt de fitxers que formen una aplicació o una unió de diverses aplicacions relacionades, i normalment formen un únic fitxer (anomenat *paquet*), amb un format propi i normalment comprimit, que és el que es distribueix, tant amb CD com amb disquet o mitjançant accés a serveis d'*ftp* o web.

L'ús de paquets facilita afegir o treure programari en considerar-lo una unitat i no haver de treballar amb els fitxers individuals.

En el contingut de la distribució (els seus CD), els paquets solen estar agrupats per categories com les següents: a) base (paquets indispensables per al funcionament del sistema –eines, programes d'inici, biblioteques de sistema–); b) sistema (eines d'administració, ordres d'utilitat); c) desenvolupament o *developement* (eines de programació, com editors, compiladors, depuradors...) d) gràfics (controladors i interfícies gràfiques, escriptoris, gestors de finestres...) i e) altres categories.

Normalment, per a instal·lar un paquet s'hauran d'efectuar una sèrie de passos:

**1)** Previ (preinstal·lació). Comprovar que hi ha el programari necessari (i amb les versions correctes) per al seu funcionament (dependències), tant si són biblioteques de sistema com altres aplicacions que utilitzi el programari.

**2)** Descompressió i còpia. Descomprimir el contingut del paquet i copiar els fitxers en les seves localitzacions definitives, tant si són absolutes (tindran una posició fixa) com si es permet resituar-les en altres directoris.

**3)** Postinstal·lació. Retocar els fitxers necessaris, configurar possibles paràmetres del programari, adequar-lo al sistema...

Depenent dels tipus de paquets, aquests passos poden ser principalment automàtics (així és en el cas de RPM [Bai03] i DEB [Deb02]) o s'hauran de fer a mà (cas *.tgz*) depenent de les eines que proporcioni la distribució.

A continuació, veurem els tres paquets possiblement més clàssics de la majoria de les distribucions. Cada distribució en té un per estàndard i en suporta algun dels altres.

### 5.1. Paquet TGZ

Els paquets TGZ són potser els d'utilització més antiga. Les primeres distribucions del GNU/Linux els utilitzaven per instal·lar el programari, i encara els utilitzen diverses distribucions (per exemple, Slackware) i alguns UNIX comercials. Són una combinació de fitxers units per l'ordre *tar* en un únic fitxer *.tar*, que després ha estat comprimit per la utilitat *gzip*, i sol aparèixer amb l'extensió *.tgz* o bé .tar.gz. D'altra banda, avui en dia és comú trobar els *tar.bz2* que utilitzen, en lloc de *gzip*, una altra utilitat anomenada *bzip2*, que en alguns casos aconsegueix comprimir més el fitxer.

**Nota**

Els paquets TGZ són una eina bàsica a l'hora d'instal·lar programari no organitzat. A més, són una eina útil per a realitzar processos de còpia de seguretat i restauració de fitxers.

Aquest tipus de paquet no conté cap informació de dependències, i pot presentar tant contingut d'aplicacions en format binari com en codi font. El podem considerar com una mena de col·lecció de fitxers comprimida.

En contra del que pugui semblar, és un format molt utilitzat, sobretot per creadors o distribuïdors de programari extern a la distribució. Molts creadors de programari que treballen per a plataformes diverses, com diversos UNIX comercials, i diferents distribucions de GNU/Linux el prefereixen com a sistema més senzill i portable.

Un exemple d'aquest cas és el projecte GNU, que distribueix el seu programari en aquest format (en forma de codi font), ja que es pot utilitzar en qualsevol UNIX, tant en un sistema propietari, com una variant BSD o una distribució GNU/Linux.

Si es tracta de format binari, haurà de ser adequat per al nostre sistema. Per exemple, és freqüent alguna denominació com la que segueix (en aquest cas, la versió 1.4 del navegador web Mozilla):

```
mozilla-i686-pc-linux-gnu-1.4-installer.tar.gz
```

en què tenim el nom del paquet, Mozilla, arquitectura a la qual ha destinat i686 (Pentium II o superiors o compatibles), i que podria ser i386, i586, i686, k6 (*amd* k6), k7 (*amd athlon*), amd64 o x86_64 (per a AMD64 i alguns Intel de 64bits amb em64t) o ia64 (Intel Itaniums) altres per a arquitectures d'altres màquines com *sparc*, *powerpc*, *mips*, *hppa*, *alpha*... Després ens indica què és per al Linux, en una màquina PC, la versió del programari 1.4.

Si és en format font, sol aparèixer de la manera següent:

```
mozilla-source-1.4.tar.gz
```

en què se'ns indica la paraula *source*. En aquest cas, no esmenta la versió d'arquitectura de màquina, la qual cosa ens indica que està preparat per compilar-se en diferents arquitectures.

Altrament, hi hauria diferents codis per a cada sistema operatiu o font: GNU/Linux, Solaris, Irix, *bsd*...

El procés bàsic amb aquests paquets consisteix en el següent:

**1)** Descomprimir el paquet (no solen utilitzar *path* absolut, amb la qual cosa es poden descomprimir en qualsevol part):

```
tar -zxvf fitxer.tar.gz (o fitxer.tgz)
```

Amb l'ordre *tar* posem opcions de *z*: *descomprimir*, *x*: *extreure fitxers*, *v*: *veure procés*, *f*: *donar nom del fitxer a tractar*.

També es pot fer separadament (sense la *z* del *tar*):

```
gunzip fitxer.tar.gz
```

(ens deixa un fitxer *tar*)

```
tar -xvf fitxer.tar
```

**2)** Una vegada descomprimit el *tgz*, tindrem els fitxers que contenia. Normalment, el programari ha d'incloure algun fitxer de tipus *readme* o *install*, en què s'especifiquen les opcions d'instal·lació pas a pas i, també, possibles dependències del programari.

En primer lloc, caldrà verificar les dependències per si disposem del programari adequat i, si no, buscar-lo i instal·lar-lo.

Si es tracta d'un paquet binari, la instal·lació sol ser bastant fàcil, ja que o ja serà directament executable on l'hàgim deixat o portarà algun instal·lador propi. Una altra possibilitat és que ho haguem de fer manualment; llavors, n'hi haurà prou amb copiar (*cp -r*, còpia recursiva) o moure (ordre *mv*) el directori a la posició desitjada.

Un altre cas és el del format de codi font. Llavors, abans d'instal·lar el programari haurem de passar per una compilació. Per a això caldrà llegir amb deteniment les instruccions del programa. Però la majoria de desenvolupadors usen un sistema del GNU anomenat *autoconf* (d'autoconfiguració) en el qual, habitualment, si no apareixen errors s'usen els passos següents:

- *./configure*. És un *script* que configura el codi perquè es pugui compilar en la nostra màquina, i verifica que hi hagi les eines adequades. L'opció *--prefix = directori* permet especificar on s'instal·larà el programari.

- *make*. És la compilació pròpiament dita.

- *make install*. És la instal·lació del programari en un lloc adequat, normalment especificada com a opció al *configure* o assumida per defecte.

Aquest és un procés general, que el programari no sempre segueix. Hi ha casos en què tot el procés s'ha de realitzar a mà, retocant fitxers de configuració o el mateix *makefile*, o compilant un a un els fitxers; però, per sort, això és cada vegada menys habitual.

En cas de voler esborrar el programari instal·lat, caldrà utilitzar el desinstal·lador, si ens el proporcionen, o si no, esborrar directament el directori o fitxers que es van instal·lar, anant amb compte amb possibles dependències.

Els paquets *tgz* són força habituals com a mecanisme de *còpia de seguretat* en tasques d'administració, per exemple, per a desar còpies de dades importants, fer còpies de seguretat de comptes d'usuari, o desar còpies antigues de dades que no sabem si tornarem a necessitar. Se sol utilitzar el procés següent: suposem que volem desar una còpia del directori *dir* com a *tar -cvf dir.tar dir* (*c*: compactar *dir* al fitxer *dir.tar*) *gzip dir.tar* (empaquetar) o bé en una sola instrucció:

```
tar -zcvf dir.tgz dir
```

El resultat serà un fitxer *dir.tgz*. Hem d'anar amb compte si ens interessa conservar els atributs dels fitxers, permisos d'usuari i fitxers d'enllaç (*links*) que hi poden haver (haurem d'examinar les opcions de *ta* perquè s'ajusti a les opcions de còpia de seguretat que volem).

### 5.2. Fedora / Red Hat: paquets RPM

El sistema de paquets RPM [Bai03] creat per Red Hat suposa un pas endavant, ja que inclou la gestió de dependències i tasques de configuració del programari. A més, el sistema guarda una petita base de dades amb els paquets ja instal·lats, que es pot consultar i que s'actualitza amb les noves instal·lacions.

Per convenció, els paquets RPM acostumen a utilitzar un nom com el següent:

```
paquet-version-rev.arq.rpm
```

en què *paquet* és el nom del programari, *version* és la numeració de versió del programari, *rev* sol ser la revisió del paquet RPM, que indica les vegades que s'ha construït, i *arq*, l'arquitectura a la qual va destinat el paquet, que pot ser Intel/AMD (i386, i586, i686, x86_64, em64t, ia64) o altres com *alpha*, *aparc*, *ppc*... L'"arquitectura" Noarch se sol utilitzar quan és independent, per exemple, un conjunt de *scripts*, i *src* quan són paquets de codi font. L'execució típica

**Nota**

El paquet apache-1.3.19-23.i686.rpm indicaria que es tracta del programari Apache (el servidor web), en la seva versió 1.3.19, revisió del paquet RPM 23, per a arquitectures Pentium II o superiors.

inclou l'execució d'*rpm*, les opcions de l'operació a realitzar, juntament amb un o més noms de paquets per processar junts.

Les operacions típiques amb els paquets RPM inclouen el següent:

- **Informació del paquet.** En el paquet es consulta una informació determinada, s'utilitza l'opció *-q* acompanyada del nom del paquet (amb *-p* si es fa sobre un fitxer *rpm*). Si el paquet encara no s'ha instal·lat, l'opció serà *-q* acompanyada de l'opció d'informació que es vulgui demanar, i si es vol preguntar a tots els paquets instal·lats alhora, l'opció serà *-qa*. Per exemple, preguntes a un paquet instal·lat:

| Consulta | Opcions RPM | Resultats |
|---|---|---|
| Fitxers | `rpm -ql` | Llista dels fitxers que conté. |
| Informació | `rpm -qi` | Descripció del paquet. |
| Requisits | `rpm -qR` | Requisits previs, biblioteques o programari. |

- **Instal·lació.** Simplement *rpm -i paquet.rpm*, o bé amb URL on trobar el paquet. Per a baixar-lo des de servidors FTP o web, només cal utilitzar la sintaxi *ftp://* o *http://* per a donar la localització del paquet.

  La instal·lació es podrà realitzar sempre que es compleixin les dependències del paquet, tant si són programari previ com biblioteques que haurien d'estar instal·lades. Si no es compleixen, ens sortirà escrit el programari que falta i el nom del paquet que el proporciona. La instal·lació es pot forçar (a risc que no funcioni) amb les opcions `--force` o `--nodeps`, o simplement ignorant la informació de les dependències.

  La tasca d'instal·lació (realitzada per *rpm*) d'un paquet comporta diferents subtasques: a) verificar les possibles dependències; b) examinar per conflictes amb altres paquets prèviament instal·lats; c) efectuar tasques prèvies a la instal·lació; c) decidir què fer amb els fitxers de configuració associats al paquet si prèviament existien; d) desempaquetar els fitxers i col·locar-los al lloc correcte; e) dur a terme tasques de postinstal·lació; finalment, f) emmagatzemar registre de les tasques efectuades en la base de dades de RPM.

- **Actualització.** Equival a la instal·lació però comprovant primer que el programari ja té *rpm -U paquet.rpm*. S'encarregarà d'esborrar la instal·lació prèvia.

- **Verificació.** Durant el funcionament normal del sistema, molts fitxers instal·lats canvien. En aquest sentit, RPM permet verificar els fitxers per detectar les modificacions, bé per procés normal, bé per algun error que podria indicar dades corrompudes. Amb *rpm -V paquet* verifiquem un paquet concret, i amb *rpm -Va* els verifiquem tots.

- **Eliminació.** Consisteix a esborrar el paquet del sistema RPM (*-e* o *--erase*). Si hi ha dependències, pot ser necessari eliminar-ne d'altres prèviament.

  Per a un cas remot:

  ```
  rpm -i ftp://lloc/directori/paquet.rpm
  ```

  ens permetria baixar el paquet des del lloc *ftp* o web proporcionat, amb la seva localització de directoris, i en aquest cas instal·lar el paquet.

S'ha de vigilar la procedència dels paquets i només utilitzar fonts de paquets conegudes i fiables, que poden ser del propi fabricant de la distribució o llocs en què confiem. Normalment, els paquets porten alguna firma digital perquè en puguem comprovar l'autenticitat. S'acostumen a utilitzar les sumes *md5* per a comprovar que el paquet no s'ha alterat, i altres sistemes com GPG (versió Gnu de PGP) per a comprovar l'autenticitat de l'emissor del paquet. A més a més, a Internet, podem trobar diferents magatzems de paquets RPM, en què estan disponibles per a diferents distribucions que utilitzin o permetin el format RPM.

**Nota**

Veure el lloc: www.rpmfind.net

Per a un ús segur de paquets, actualment, els reposadors (oficials, i alguns de tercers) firmen electrònicament els paquets, per exemple, amb el GPG esmentat; això ens permet assegurar (si disposem de les firmes) que els paquets procedeixen de la font fiable. Normalment, cada proveïdor (el reposador) inclou uns fitxers de firma PGP amb la clau per al seu lloc. Dels reposadors oficials, normalment ja vénen instal·lades; si procedeixen de tercers, haurem d'obtenir el fitxer de clau, i incloure-la en RPM, típicament:

```
$ rpm -import GPG-KEY-FILE
```

En què GPP-KEY-FILE és el fitxer clau GPG o l'URL del fitxer esmentat. Normalment, aquest fitxer també tindrà suma *md5* per a comprovar la seva integritat. I podem conèixer les claus existents en el sistema amb el següent:

```
$ rpm -qa | grep ^gpg-pubkey
```

podem veure més detalls a partir de la clau obtinguda:

```
$ rpm -qi gpg-key-xxxxx-yyyyy
```

Per a un paquet *rpm* concret podrem comprovar si disposa de firma i quin s'ha utilitzat amb el següent:

```
$ rpm -checksig -v <paquet>.rpm
```

I per verificar que un paquet és correcte segons les firmes disponibles:

```
$ rpm -K <paquet.rpm>
```

Hem d'anar amb compte i importar només les claus dels llocs en què confiem. Quan RPM trobi paquets amb firma que no tenim en el nostre sistema, o el paquet no estigui firmat, ens avisarà, i l'acció ja dependrà de la nostra actuació.

Quant al suport RPM en les distribucions, a Fedora (Red Hat i, també, en les seves derivades), RPM és el format per defecte de paquets i el que utilitza àmpliament la distribució per a les actualitzacions i la instal·lació de programari. A Debian s'utilitza el format anomenat DEB (com veurem) i hi ha suport per a RPM (hi ha l'ordre *rpm*), però només per a consulta o informació de paquets. Si és imprescindible instal·lar un paquet *rpm* a Debian, es recomana utilitzar la utilitat *alien*, que permet convertir formats de paquets, en aquest cas d'RPM a DEB, i instal·lar el paquet convertit.

A més del sistema base d'empaquetat de la distribució, avui en dia, cada una sol suportar un sistema de gestió de programari intermedi de nivell més alt, que afegeix una capa superior al sistema base, la qual cosa facilita les tasques de gestió del programari i afegeix una sèrie d'utilitats per controlar millor el procés.

En el cas de Fedora (Red Hat i derivats) s'utilitza el sistema *yum*, que permet –com a eina de nivell més alt– la instal·lació i gestió de paquets en sistemes *rpm*, a més de la gestió automàtica de dependències entre els paquets. Permet accedir a diferents reposadors, centralitza la seva configuració en un fitxer (*/etc/yum.conf*, habitualment) i té una interfície d'ordres simple.

**Nota**

YUM a:
http://linux.duke.edu/projects/yum

La configuració de *yum* es basa en el següent:

*/etc/yum.config* (fitxer d'opcions)

*/etc/yum* (directori per a algunes utilitats associades)

*/etc/yum.repos.d* (directori d'especificació de reposadors, un fitxer per a cada un, s'inclou informació de l'accés, i localització de les firmes *gpg*).

Per a les operacions típiques de *yum* un resum seria el següent:

| Ordre | Descripció |
|---|---|
| yum install <Nom> | instal·lació del paquet amb el nom |
| yum update <Nom> | actualització d'un paquet |
| yum remove <Nom> | eliminació del paquet |
| yum list <Nom> | cerca del paquet per nom (només nom) |
| yum search <Nom> | cerca més àmplia |
| yum provices <File> | cerca de paquets que proporcionin el fitxer |
| yum update | actualització de tot el sistema |
| yum upgrade | ídem a l'anterior incloent-hi paquets addicionals |

Per acabar, Fedora també ofereix un parell d'utilitats gràfiques per a YUM, *pup* per controlar les actualitzacions recents disponibles, i *pirut* com a paquet de gestió de programari. També n'hi ha algunes altres com *yumex,* amb més control de la configuració interna de *yum*.

### 5.3. Debian: paquets DEB

Debian té eines interactives com *tasksel*, que permet escollir uns subconjunts de paquets agrupats per tipus de tasques: paquets per a X, per a desenvolupament, per a documentació, etc., o com *dselect*, que ens permet navegar per tota la llista de paquets disponible (n'hi ha milers) i escollir els que vulguem instal·lar o desinstal·lar. De fet, aquestes només són un *front-end* del gestor de programari de nivell intermedi APT.

En el nivell de línia d'ordres disposa de *dpkg,* que és l'ordre de nivell més baix (*base* seria l'equivalent a *rpm*), per a gestionar directament els paquets DEB de programari [Deb02], típicament, *dpkg -i paquet.deb* per realitzar la instal·lació. Es poden realitzar tot tipus de tasques, d'informació, instal·lació, esborrament o canvis interns en els paquets de programari.

El nivell intermedi (com el cas de Yum a Fedora) el presenten les eines APT (la majoria són ordres *apt-xxx*). APT permet gestionar els paquets amb una llista de paquets actuals i disponibles a partir de diverses fonts de programari, tant des dels propis CD de la instal·lació com llocs *ftp* o web (HTTP). Aquesta gestió es fa transparentment, de manera que el sistema és independent de les fonts de programari.

La configuració del sistema APT s'efectua des dels fitxers disponibles en */etc/apt*, en què */etc/apt/sources.list* és la llista de fonts disponibles. Vegem-ne un exemple:

```
deb http://http.us.debian.org/debian stable main contrib non-free
debsrc http://http.us.debian.org/debian stable main contrib non-free
deb http://security.debian.org stable/updates main contrib non-free

#Sources Oficials de Debian STABLE "etch"
deb http://ftp.debian.org/debian/ etch main non-free contrib
debsrc http://ftp.debian.org/debian/ etch main non-free contrib
```

En què hi ha recopilades diverses fonts "oficials" per a una Debian (*etch* en aquest cas, suposant aquesta com *stable*), des d'on es poden obtenir els paquets de programari, i les actualitzacions que estiguin disponibles. Bàsicament, s'especifica el tipus de font (*web/ftp* en aquest cas), el lloc, la versió de la distribució (*stable* o *etch* en aquest exemple) i les categories del programari que es buscaran (lliure o contribucions de tercers o de llicència no lliure o comercial).

Els paquets de programari estan disponibles per a les diferents versions de la distribució Debian, hi ha paquets per a les versions *stable*, *testing*, i *unstable*. L'ús dels uns o dels altres determina el tipus de distribució (previ canvi de les fonts de reposadors en *sources.list*). Es Poden tenir fonts de paquets barrejades, però no és gaire recomanable, ja que podrien provocar conflictes entre les versions de les diferents distribucions.

**Nota**

Els paquets DEB de Debian és potser el sistema d'instal·lació més potent del GNU/Linux. Una prestació destacable és la independència del sistema de les fonts dels paquets (mitjançant APT).

Una vegada tenim les fonts de programari configurades, la principal eina per a manejar-les en el nostre sistema és *apt-get*, que ens permet instal·lar, actualitzar o esborrar el paquet individual i actualitzar la distribució sencera. També hi ha un *front-end* a *apt-get*, anomenat *aptitude,* amb una interfície d'opcions pràcticament igual (de fet, es podria qualificar d'emulador d'*apt-get*, ja que la interfície és equivalent). Com a avantatge aporta una gestió de dependències dels paquets millor i permet una interfície interactiva. De fet, s'espera que *aptitude* sigui la interfície per defecte en línia d'ordres per a la gestió de paquets.

Algunes funcions bàsiques d'*apt-get*:

- Instal·lació d'un paquet particular.

```
apt-get install paquet
```

- Esborrament d'un paquet.

```
apt-get remove paquet
```

- Actualització de la llista de paquets disponibles.

```
apt-get update
```

- Actualització de la distribució. Podríem efectuar els passos combinats següents:

```
apt-get update
apt-get upgrade
apt-get dist-upgrade
```

Mitjançant aquest últim procés, podem mantenir la nostra distribució actualitzada permanentment, actualitzant els paquets instal·lats i verificant les dependències amb els nous. Unes eines útils per construir una llista és *apt-spy,* que busca els llocs oficials més ràpids, o *netselect*, que ens permet provar una llista de llocs. D'altra banda, podem buscar les fonts oficials (les podem configurar amb *apt-setup*) o copiar algun fitxer de fonts disponible. Un programari addicional (de tercers) pot necessitar afegir més fonts (en *sources.list*). Es poden obtenir llistes de llocs de fonts disponibles (per exemple, en l'adreça d'Internet: http://www.apt-get.org).

L'actualització del sistema en particular genera una baixada d'un gran nombre de paquets (en especial en *unstable*), la qual cosa fa recomanable buidar la *caché* i el reposador local, amb els paquets baixats (es mantenen en */var/cache/apt/arxivi*) que ja no s'hagin d'utilitzar. Farem servir *apt-get clean* per eliminar-los tots, o bé *apt-get autoclean* per eliminar els paquets que no siguin necessaris perquè ja hi ha noves versions (en principi). Cal tenir en compte si tornem a necessitar aquests paquets per raons de reinstal·lació, ja que, si és així, els haurem de tornar a baixar.

El sistema APT també permet allò que es diu SecureAPT, que és la gestió segura de paquets mitjançant verificació de sumes (*md5*) i la firma de fonts de paquets (de tipus GPG). Si durant la baixada les firmes no estan disponibles, apt-get n'informa, genera una llista amb els paquets no firmats i demana si es deixen instal·lar o, però no deixa decisió a l'administrador. La llista de fonts confiables actuals s'obté amb el següent:

```
# apt-key list
```

Les claus *gpg* dels llocs oficials de Debian es distribueixen mitjançant un paquet. Les instal·lem d'aquesta manera:

```
apt-get install debian-archive-keyring
```

evidentment, si considerem que tenim *sources.list* amb els llocs oficials. Es espera que (depenent de la versió Debian) aquestes claus ja s'instal·lin per defecte en iniciar el sistema. Per a altres llocs no oficials (que no proporcionin la clau en paquet), però que considerem confiables, podem importar la clau, que obtindrem des del reposador (com que no hi ha un estàndard definit, haurem de consultar on tenen la clau disponible, encara que sol ser en la pàgina web inicial del reposador). Per afegir la clau podem utilitzar *apt-key add* amb el fitxer o, també, el següent:

```
# gpg -import fitxer.key
# gpg -export -armor XXXXXXXX | apt-key add -
```

en què X és un hexadecimal relacionat amb la clau (vegeu les instruccions del reposador per comprovar la manera recomanada d'importar la clau i les dades necessàries).

Una altra funcionalitat important del sistema AP són les funcions de consulta d'informació dels paquets, amb l'eina *apt-cache*, que ens permet interactuar amb les llistes de paquets de programari Debian.

**Exemple**

L'eina *apt-cache* disposa d'ordres que ens permeten buscar informació sobre els paquets, com per exemple:

- Buscar paquets sobre la base d'un nom incomplet.

```
apt-cache search nom
```

- Mostrar la descripció del paquet.

```
apt-cache show paquet
```

- De quins paquets depèn.

```
apt-cache depends paquet
```

Altres eines o funcionalitats d'*apt* interessants:

- *apt-show-versions*. Ens especifica quins paquets poden ser actualitzats (i per quines versions, vegeu opció *-u*).

Altres tasques més específiques necessitaran realitzar-se amb l'eina de nivell més baix, com *dpkg*. Per exemple, obtenir la llista de fitxers d'un paquet determinat ja instal·lat:

```
dpkg -L paquet
```

La llista de paquets sencera amb

```
dpkg -l
```

O buscar de quin paquet prové un element (fitxer per exemple):

```
dpkg -S fitxer
```

Aquest, en particular, funciona per a paquets instal·lats; `apt-file` també permet buscar per a paquets encara no instal·lats.

Finalment, també es poden esmentar algunes eines gràfiques per a Apt com synaptic, gnome-apt per a gnome, i kpackage o adeptpar per a KDE; o les textuals ja esmentades com *aptitude* o *dselect*.

Com a conclusió, es pot destacar que el sistema de gestió APT (en combinació amb el base *dpkg*) és molt flexible i potent a l'hora de gestionar les actualitzacions, i és el sistema de gestió de paquets que es fa servir a Debian i les seves distribucions derivades com Ubuntu, Kubuntu, Knoppix, Linex, etc.

# 6. Eines genèriques d'administració

En el camp de l'administració, també podríem considerar algunes eines, com les pensades de manera genèrica per a l'administració. Encara que es pot destacar que per a aquestes eines és difícil mantenir-se al dia, a causa dels plans de versions actuals de les distribucions, que tenen una evolució molt ràpida. Deixem alguns exemples (però, en un moment determinat, poden no ser completament funcionals):

**a) Linuxconf**. És una eina genèrica d'administració que agrupa diferents aspectes en una interfície de menús textual, que en les ultimes versions va evolucionar a suport web. Es pot utilitzar en gairebé qualsevol distribució GNU/Linux, i suporta diversos detalls propis de cada una (per desgràcia, fa temps que no s'actualitza).

**Nota**

Les podem trobar en Linuxconf:

http://www.solucorp.qc.ca/linuxconfWebmin

**b) Webmin.** És una altra eina d'administració pensada des d'una interfície web. Funciona amb una sèrie de *plugins* que es poden afegir per a cada servei que cal administrar. Normalment, compta amb formularis en què s'especifiquen els paràmetres de configuració dels serveis. A més, ofereix la possibilitat (si s'activa) de permetre administració remota des de qualsevol màquina amb navegador.

**c)** Altres en desenvolupament com cPanel, ISPConfig.

D'altra banda, en els entorns d'escriptori de Gnome i KDE, solen disposar del concepte de plafó de control, que permet gestionar l'aspecte visual de les interfícies gràfiques i tractar alguns paràmetres dels dispositius del sistema.

Quant a les eines gràfiques individuals d'administració, la pròpia distribució de GNU/Linux n'ofereix algunes directament (eines que acompanyen tant a Gnome com KDE), eines dedicades a gestionar un dispositiu (impressores, so, targeta de xarxa, etc.), i altres, a l'execució de tasques concretes (connexió a Internet, configurar arrencada de serveis del sistema, configurar X Window, visualitzar *logs*...). Moltes són simples caràtules (*frontends*) en les eines bàsiques de sistema, o bé estan adaptades a particularitats de la distribució.

Es pot destacar, en especial en aquest apartat, la distribució Fedora (Red Hat i derivats), que intenta disposar de diverses utilitats (més o menys minimalistes) per a diferents funcions d'administració. Les podem trobar en l'escriptori (en el menú d'administració) o en ordres com *system-config-xxxxx* per a diferents funcionalitats, com gestió de pantalla, impressora, xarxa, seguretat, usuaris, paquets, etc. En la figura, en podem veure algunes:

Figura 3. Diverses utilitats gràfiques d'administració a Fedora

# 7. Altres eines

En l'espai limitat d'aquesta unitat no es poden comentar totes les eines que ens poden aportar beneficis per a l'administració. Esmentarem algunes de les eines que considerem bàsiques:

- Les múltiples ordres UNIX bàsiques, com *grep*, *awk*, *sed*, *find*, *diff*, *gzip*, *bzip2*, *cut*, *sort*, *df*, *du*, *cat*, *more*, *file*, *which*...

- Els editors, imprescindibles per a qualsevol tasca d'edició, com Vi, Emacs i molts altres. El Vim s'utilitza molt en tasques d'administració per la rapidesa d'efectuar petits canvis en els fitxers. El Vim és l'editor compatible Vi, que sol portar el GNU/Linux, i permet una sintaxi acolorida en diversos llenguatges. L'Emacs és un editor molt complet, adaptat a diferents llenguatges de programació (sintaxi i modes d'edició), disposa d'un entorn molt complet i d'una versió X denominada Xemacs. Joe, i és compatible amb Wordstar.

**Nota**

Vegeu el material associat al curs d'introducció en el GNU/ Linux, o en les pàgines man de les ordres, o una referència d'eines com [Stu01].

- Els llenguatges de tipus *script*, útils per a l'administració, com Perl, PHP i Python. El Perl és molt útil per al tractament d'expressions regulars, i anàlisi de fitxers (filtrat, ordenació, etc.). El PHP s'utilitza molt en entorns web. Python, un altre llenguatge que permet fer prototips ràpids d'aplicacions...

- Les eines de compilació i depuració de llenguatges d'alt nivell, com GNU `gcc` (compilador de C i C++), `gdb` (depurador), `xxgdb` (interfície X per a `gdb`) i `ddd` (depurador per a diversos llenguatges).

## Activitats

**1.** Feu una lectura ràpida de l'estàndard FHS, que ens servirà per tenir una bona guia a l'hora de buscar fitxers per la nostra distribució.

**2.** Per a repassar i ampliar conceptes i programació de *shell scripts* en *bash*, vegeu [Bas] [Coo].

**3.** Per als paquets RPM, com resoldríeu algunes de les qüestions següents?:
- Saber quin paquet va instal·lar una ordre determinada.
- Obtenir la descripció del paquet que va instal·lar una ordre.
- Esborrar un paquet sense saber-ne el nom complet.
- Mostrar tots els fitxers que eren en el mateix paquet que un fitxer determinat.

**4.** Feu el mateix que en l'activitat anterior, però per a paquets Debian, utilitzant eines APT.

**5.** Actualitzeu una distribució Debian (o Fedora).

**6.** Instal·leu en la nostra distribució alguna eina genèrica d'administració, per exemple, Linuxconf o Webadmin. Què us ofereixen? Enteneu les tasques executades i els efectes que provoquen?

## Altres fonts de referència i informació

[Bas][Coo] ofereixen una àmplia introducció (i conceptes avançats) a la programació de *shell scripts* en *bash*, i nombrosos exemples. [Qui01] comenta els diferents intèrprets d'ordres de programació en el GNU/Linux, i les seves semblances i diferències.

[Deb02][Bai03] ofereixen una àmplia visió dels sistemes de paquets de programari de les distribucions Debian i Fedora / Red Hat.

[Stu] és una àmplia introducció a les eines disponibles en el GNU/Linux.