Linux Man Page Howto

Jens Schweikhardt

howto@schweikhardt.net

$Id: Man-Page.sgml,v 1.2 2002/12/15 17:51:43 schweikh Exp schweikh $

[Traduzione di Fabio Teatini - teafab @ pluto.linux.it] Questo HOWTO descrive ciò che bisogna sapere per scrivere la documentazione in linea (cioè le cosiddette pagine man o pagine di manuale) e per renderla accessibile tramite il comando `man`*(1). Lungo tutto questo HOWTO useremo il termine "manuale" riferendoci semplicemente alle pagine man, senza indulgere in doppi sensi di carattere sessuale.*

Sommario

1. Qualche considerazione sulla documentazione

Perché scriviamo la documentazione? Domanda stupida. Perché vogliamo che le altre persone siano in grado di usare il nostro programma, la nostra libreria di funzioni o qualsiasi altra cosa avessimo scritto per metterla a disposizione. Ma scrivere documentazione non si riduce solo a questo:

- La documentazione deve essere accessibile; se venisse rintanata in qualche posizione non standard, dove gli strumenti per gestire la documentazione non possono trovarla, come potrebbe servire allo scopo?
- La documentazione deve essere affidabile e accurata; non c'è nulla di più irritante di un programma che si comporta diversamente da come è descritto. In casi del genere gli utenti possono semplicemente mandare l'autore a quel paese, ma possono anche arrivare a spedirgli email cariche d'odio; infine scaraventeranno qualunque suo lavoro alle ortiche, col fermo intento di non installare mai più qualunque cosa venga scritto da un tale balordo.

Il modo tradizionale e ben noto per accedere alla documentazione su UNIX è attraverso il comando man(1). Questo HOWTO descrive le procedure per scrivere una pagina di manuale che possa essere elaborata correttamente dagli strumenti per il trattamento della documentazione. Tra questi strumenti, i più importanti sono `man`(1), `xman`(1x), `apropos`(1), `makewhatis`(8) e `catman`(8). Il compito di garantire l'affidabilità e l'accuratezza delle informazioni spetta, ovviamente, tutto a voi; ma proseguendo nella lettura troverete comunque qualche idea che potrà aiutarvi a evitare alcuni errori tra quelli commessi più di frequente.

2. Come si accede alle pagine man?

È importante conoscere il meccanismo preciso per accedere alle pagine man, per poter dar loro il nome in modo corretto e installarle nel posto giusto. Ogni pagina man andrebbe classificata in una sezione specifica, identificata da un singolo carattere. Le sezioni più comuni utilizzate sotto Linux, insieme ai loro nomi di umana comprensione, sono:

```
Sezione Nome
   1    Comandi dell'utente avviabili da chiunque.
   2    Chiamate di sistema, cioè le funzioni fornite dal kernel.
   3    Subroutine, ovvero le librerie di funzioni.
   4    Dispositivi, ossia i nomi dei file speciali nella directory /dev.
   5    Descrizioni di formati dei file, ad esempio /etc/passwd.
   6    Giochi (si descrive da sé).
   7    Miscellanea, ad esempio i pacchetti di macro e le convenzioni.
   8    Strumenti per l'amministrazione di sistema che soltanto il root può eseguire.
   9    Un'altra sezione (specifica di Linux) per documentare le routine del kernel.
   n    (Deprecato) Nuova documentazione, da spostare in una sezione più adatta.
   o    (Deprecato) Documentazione obsoleta, da tenere qui solo per un breve periodo.
   l    (Deprecato) Documentazione locale, riguardante questo sistema in particolare.
```

Il nome del file sorgente della pagina man (quello di input per il sistema di formattazione) è quello del comando, della funzione o del file, seguito da un punto, seguito dal carattere identificativo della sezione. Se state scrivendo la documentazione riguardante il formato del file 'passwd', il file sorgente dovrà chiamarsi 'passwd.5'. Questo è anche un caso esemplare in cui il un nome di un file risulta identico al nome di un comando. Potrebbe esistere anche una libreria di subroutine chiamata passwd. La classificazione in sezioni è il modo consueto di risolvere queste ambiguità: la descrizione del comando è trattata nel file 'passwd.1', mentre l'ipotetica libreria è descritta in 'passwd.3'.

A volte capita di incontrare nomi di file con caratteri terminali aggiuntivi, come in 'xterm.1x' o 'wish.1tk'; in questo modo si vuol indicare che si tratta di documentazione per un programma di X Window o di un'applicazione Tk, rispettivamente. Qualche visualizzatore delle pagine man può far uso di quest'informazione aggiuntiva. Per esempio, nella lista della documentazione messa a disposizione da xman compariranno le voci 'xterm(x)' e 'wish(tk)'.

Si prega di non usare le sezioni n, o e l; queste sezioni, conformemente al File System Standard, sono deprecate. Limitatevi alle sezioni numeriche. Si faccia attenzione a non usare nomi di programmi, funzioni o file già esistenti. Se doveste sviluppare un nuovo editor, sarebbe di sicuro una pessima idea quella di chiamarlo ed, sed (volendo intendere "super ed") o red (intendendo "il ritorno di ed"). Assicurandovi l'unicità del nome, eviterete all'utente di eseguire il vostro programma facendogli leggere la pagina man di un altro programma, e viceversa.

Ora che sappiamo il nome da dare al nostro file, la prossima decisione da prendere riguarda la directory in cui installarlo (ad esempio facendo eseguire all'utente il '`make install`' del vostro pacchetto). Su Linux, tutte le pagine di manuale si trovano all'interno delle directory elencate nella variabile d'ambiente MANPATH. Gli strumenti adibiti al trattamento della documentazione utilizzano MANPATH nello stesso modo in cui una shell utilizza PATH per localizzare gli eseguibili. In effetti, MANPATH ha lo stesso formato di PATH; ognuna di essa contiene un elenco di directory separate da punti e virgole (se si eccettua il fatto che MANPATH non ammette campi vuoti e nomi di percorso relativi, utilizzando solo nomi assoluti). Se MANPATH non è stato impostato o esportato, verrà utilizzato un valore predefinito che conterrà almeno la directory /usr/man. Per velocizzare la ricerca e mantenere snello l'elenco delle directory, le directory indicate in MANPATH (chiamate "directory base") contengono una manciata di sottodirectory chiamate 'man<s>' dove <s> sta per il carattere che descrive la sezione introdotta nella tabella precedente. Non tutte le sezioni saranno rappresentate da una sottodirectory semplicemente perché non c'è alcuna ragione di tenere delle sottodirectory vuote. Tuttavia, possono essere presenti alcune sottodirectory di nome 'cat<s>', 'dvi<s>' e 'ps<s>' che contengono documentazione pronta da visualizzare o stampare; di questo parleremo più tardi. L'unico altro file che deve essere presente in qualsiasi directory base è quello chiamato 'whatis', di cui tratteremo scopi e preparazione all'uso nel paragrafo 12. Affinché una pagina man della sezione <s> sia collocata con certezza nel posto giusto, basterà copiarla nella directory /usr/man/man<s>. Un buon `Makefile`, però, permetterà all'utente di scegliersi la propria directory base, per mezzo di una variabile di `make`, ad esempio MANDIR. La maggior parte dei pacchetti GNU possono essere configurati con l'opzione `--prefix=/qualsiasi/dir`. I manuali verranno perciò installati all'interno della directory base /qualsiasi/dir. Il mio suggerimento è che venga fornito anche un modo per fare qualcosa di simile.

Con l'avvento del Linux File System Standard (FS-Stnd) le cose sono diventate più complicate. [Nota: Il FS-Stnd sembra esser stato sostituito dal Filesystem Hierarchy Standard (http://www.pathname.com/fhs/), FHS.] Il FS-Stnd 1.2 dichiara che

"La struttura di /usr/man va predisposta per ospitare le pagine di manuale scritte nei vari linguaggi (o con più linguaggi). "

Questo obiettivo può essere raggiunto introducendo un altro livello di directory che distingua tra i diversi linguaggi. Citando ancora dal FS-Stnd 1.2:

"Questa denominazione delle sottodirectory dei linguaggi sotto /usr/man è basata sull'Appendice E dello standard POSIX 1003.1, che descrive la stringa identificativa della localizzazione (si tratta del metodo più accettato per descrivere un ambiente multiculturale o poliglotta). La stringa di localizzazione o <locale> è: <linguaggio>[_<territorio>][.<set-di-caratteri>][,<versione>]"

(Si veda il FS-Stnd per trovare un elenco di stringhe <locale> d'uso frequente). Seguendo queste direttive, le nostre pagine man si troveranno in /usr/man/<locale>/man[1-9lno]. Ovviamente, le versioni formattate dovrebbero di conseguenza trovarsi in /usr/man/<locale>/cat[1-9lno], altrimenti potremmo fornirle soltanto per un'unica localizzazione. TUTTAVIA, per ora mi sento di sconsigliare il passaggio a questa struttura. Il FS-Stnd 1.2, inoltre, permette quanto segue:

"I sistemi che utilizzano un solo linguaggio ed un solo set di codici per tutte le pagine di manuale possono ottere la sottostringa <locale> e conservare tutte le pagine di manuale in <mandir>. Per esempio, i sistemi che hanno solo pagine di manuale in inglese codificate in ASCII possono conservarle direttamente in /usr/man, nelle sottodirectory `man[1-9]` (di fatto, questo è quel che avviene di solito)").

Io non farei il passaggio alla nuova struttura finché tutti gli strumenti (come xman, tkman, info e molte altre che leggono le pagine di manuale) non saranno in grado di farvi fronte.

3. Quale dovrebbe essere l'aspetto di una pagina man formattata?

Cominciamo con un esempio che verrà descritto nel dettaglio più avanti; se lo leggete come testo puro non mostrerà i caratteri in maniera differente a seconda della loro formattazione (*grassetto* e *corsivo*). Per avere ulteriori informazioni si faccia riferimento al paragrafo "Quali sono le convenzioni sui font?". Ma ora veniamo alla pagina di manuale relativa al programma (ipotetico) `foo`.

```
FOO(1)                   Manuale Utente                   FOO(1)


NOME
    foo - sentarbinisce la libreria zac

SINTASSI
    foo [-bar] [-c file-config ] file ...

DESCRIZIONE
     foo sentarbinisce la libreria zac sminestrando nelle tabelle
     interne dei simboli; per default parserizza tutti i segmenti
     baz e li risistema in  ordine inverso di tempo per mezzo del
     linker  xyzzy(1), per sottoporli a ricerca.  Successivamente
     viene  compresso l'elemento  symdef  utilizzando l'algoritmo
     WBG (Whiz-Bang-Gizmo).  Tutti i file vengono  elaborati nel-
     l'ordine indicato.

OPZIONI
     -b   In fase di elaborazione non scrive 'busy' sullo stdout.

     -c file-config
          Usa il file di configurazione  alternativo  file-config
  invece  di /etc/foo.conf.   Ciò comporta l'annullamento
  della variabile d'ambiente FOOCONF.

     -a   Oltre ai segmenti baz,  vengono  parserizzati anche gli
          header blurfl.

     -r   Modalità ricorsiva.  Opera alla velocità della luce, al
          costo di un megabyte di memoria virtuale.

FILE
     /etc/foo.conf
          Il file di configurazione generale.  Per altri dettagli
  si veda foo(5).
     ~/.foorc
          File di configurazione personale dell'utente. Per altri
```

```
          dettagli si veda foo(5).

AMBIENTE
     FOOCONF
          Se impostata è il nome del percorso completo di un file
          di configurazione  alternativo a foo.conf.  È annullata
          dall'opzione -c.

DIAGNOSTICA
     I seguenti  messaggi diagnostici  possono essere  inviati su
     stderr:

     Magic number sconosciuto.
          Il file di input non sembra essere  un file  d'archivio
          valido.
     Segmenti baz di vecchio formato.
          foo può gestire solo i  segmenti baz  di nuovo formato.
          Le librerie di  oggetti COBOL  non sono  supportate  in
          questa versione.

BUG
     Il nome del comando dovrebbe essere scelto  facendo maggiore
     attenzione al suo scopo.

AUTORE
     Jens Schweikhardt <howto at schweikhardt dot net> (mailto:howto@schweikhardt.net)

VEDERE ANCHE
     bar(1), foo(5), xyzzy(1)

Linux                        MARZO 1995                        2
```

Come promesso, ecco la spiegazione delle voci di sezione.

La sezione NOME

...è la sola sezione obbligatoria. Le pagine man senza una sezione del nome sono utili quanto un congelatore al polo nord. Questa sezione ha anche un formato standard che consiste di un elenco di nomi di programmi o funzioni separati da virgole, seguita da un trattino, seguito a sua volta da una breve descrizione (su una sola riga, di solito) della funzionalità che il programma (o la funzione, o il file) è in grado di fornire. Per mezzo di `makewhatis`(8), le sezioni del nome vengono inserite nel database dei file di `whatis`. L'uso di `makewhatis` è il motivo della necessità della sezione NOME; tale sezione, dunque deve esistere ed aderire al formato descritto. Il sorgente in formato `groff` deve apparire così:

.SH NOME foo \- sentarbinisce la libreria zac

La sequenza \- è importante: la barra rovesciata (o backslash) è necessaria a distinguere il trattino da quello della suddivisione in sillabe che può comparire sia nel nome del comando che nella descrizione disposta su una riga.

La sezione SINTASSI

...ha lo scopo di fornire una breve panoramica sulle opzioni messe a disposizione dal programma. Per quel che riguarda le funzioni, questa sezione elenca i corrispondenti file di include e il prototipo, così da informare il programmatore in merito al tipo degli argomenti di input, al loro numero e al tipo dei dati restituiti.

La sezione DESCRIZIONE

...descrive in maniera eloquente il motivo per cui la vostra sequenza di 0 e 1 valga la pena di essere usata. Questo è il posto in cui scrivere la vostra conoscenza. Questa è la Sala delle Celebrità: potrete guadagnarvi l'ammirazione degli altri programmatori ed utenti facendo di questa sezione la fonte di informazioni dettagliate e affidabili. Qui, tra le altre cose, potete descrivere l'utilizzo degli argomenti, il formato dei file e quali sono gli algoritmi delle operazioni critiche.

La sezione OPZIONI

...dà una descrizione di come ogni opzione influenza il comportamento del programma. Ma questo già si sapeva, no?

La sezione FILE

...elenca i file usati dal programma o dalla funzione; per esempio, può elencare i file di configurazione, quelli di avvio e quelli su cui il programma interviene direttamente in scrittura. È buona cosa fornire il percorso completo di questi file e permettere che il processo di installazione possa modificare parte della directory per soddisfare le preferenze dell'utente: i manuali di `groff` hanno un prefisso predefinito pari a /usr/local, cosicché faranno riferimento per default a /usr/local/lib/groff/*. Invece, se si installa usando 'make prefix=/opt/gnu', il riferimento nella pagina man diventa /opt/gnu/lib/groff/*

La sezione AMBIENTE

...elenca tutte le variabili d'ambiente che influenzano il programma o la funzione (descrivendo anche le modalità di quest'interazione, ovviamente). La maggior parte delle variabili contiene nomi di percorso, nomi di file o opzioni di default.

La sezione DIAGNOSTICA

...dovrebbe dare una panoramica dei messaggi d'errore emessi più comunemente dal vostro programma e come venirne a capo. Non c'è alcuna necessità di descrivere i messaggi d'errore di sistema (descritti in `perror`(3)) o i segnali d'errore fatale (descritti in `psignal`(3)) che dovessero comparire durante l'esecuzione, in quanto sono descritti a parte e riguardano l'esecuzione di qualsiasi programma.

La sezione BUG

...non dovrebbe esistere, in un mondo ideale. Qui, se siete intrepidi, potrete descrivere le limitazioni, i problemi noti e quelle che voi vedete come funzionalità ma che altri potrebbero interpretare come complicazioni inutili. Se invece non avete questo coraggio, cambiate il nome di questa sezione in TO DO ;-)

La sezione AUTORE

...è molto utile nel caso in cui ci siano errori grossolani nella documentazione o nel comportamento del programma (Bzzt!) e volete inviare una segnalazione del problema incontrato.

La sezione VEDERE ANCHE

...è un elenco di pagine man correlate alla presente, elencate in ordine alfabetico. Per convenzione, questa è l'ultima sezione. Se le sezioni disponibili fossero insufficienti a descrivere il contenuto che volete fornire siete liberi di inventarne altre. Per concludere: come si può generare questa pagina man? Mi aspettavo questa domanda: ecco il sorgente:

```
.\" Processa questo file con il comando
.\" groff -man -Tascii foo.1
.\"
.TH FOO 1 "MARZO 1995" Linux "Manuale Utente"
.SH NOME
foo \- sentarbinisce la libreria zac
```

```
.SH SINTASSI
.B foo [-bar] [-c
.I file-config
.B ]
.I file
.B ...
.SH DESCRIZIONE
.B foo
sentarbinisce la libreria zac sminestrando nelle
tabelle interne dei simboli; per default parserizza
tutti i segmenti baz e li risistema in ordine
inverso di tempo per mezzo del linker
.BR xyzzy (1),
per sottoporli a ricerca. Successivamente viene compresso
l'elemento symdef utilizzando l'algoritmo
WBG (Whiz-Bang-Gizmo). Tutti i file vengono elaborati
nell'ordine indicato.
.SH OPZIONI
.IP -b
In fase di elaborazione non scrive 'busy' sullo stdout.
.IP "-c file-config"
Usa il file di configurazione alternativo
.I file-config
invece di
.IR /etc/foo.conf .
Ciò comporta l'annullamento della variabile d'ambiente
.BR FOOCONF .
.IP -a
Oltre ai segmenti baz, vengono parserizzati anche gli
header blurfl.
.IP -r
Modalità ricorsiva. Opera alla velocità della luce, al
costo di un megabyte di memoria virtuale.
.SH FILE
.I /etc/foo.conf
.RS
Il file di configurazione generale. Per altri dettagli
si veda
.BR foo (5).
.RE
.I ~/.foorc
.RS
File di configurazione specifico dell'utente. Per altri
dettagli si veda
.BR foo (5).
.SH AMBIENTE
.IP FOOCONF
Se impostata è il nome del percorso completo di un file
di configurazione alternativo a
.IR foo.conf .
È  annullata
dall'opzione
.BR -c .
```

```
.SH DIAGNOSTICA
I seguenti messaggi diagnostici possono essere inviati su
stderr:

Magic number sconosciuto.
.RS
Il file di input non sembra essere un file d’archivio valido.
.RE
Segmenti baz di vecchio formato.
.RS
.B foo
può gestire solo i segmenti baz di nuovo formato.
Le librerie di oggetti COBOL non sono supportate in
questa versione.
.SH BUG
Il nome del comando dovrebbe essere scelto facendo maggiore
attenzione al suo scopo.
.SH AUTORE
Jens Schweikhardt <howto at schweikhardt dot net>
.SH "VEDERE ANCHE"
.BR bar (1),
.BR foo (5),
.BR xyzzy (1)
```

4. Come è possibile documentare più programmi/funzioni all’interno di un’unica pagina man?

Un gruppo di programmi (`grep`, `egrep`) o di funzioni (`printf`, `fprintf`, ...) possono essere documentati in un’unica pagina di manuale. Tuttavia, queste pagine man sarebbero del tutto inservibili se fossero accessibili soltanto attraverso uno dei nomi. Non potremo pretendere che un utente si ricordi che la pagina man di `egrep` è in realtà la stessa di `grep`. Perciò si farà in modo di rendere disponibile la stessa pagina man sotto diversi nomi; ci sono svariate possibilità per ottenere questo risultato:

1. fare copie identiche per ogni nome.
2. collegare tutte le pagine man utilizzando i link fisici (o hard link).
3. utilizzare link simbolici che puntano tutti alla pagina man effettiva.
4. utilizzare il meccanismo del ‘sorgente’ di `groff` messo a disposizione dalla macro `.so`.

La prima strada, ovviamente, comporta lo spreco di spazio su disco. La seconda non è raccomandabile perché alcune versioni intelligenti del programma `catman` consentono di risparmiare molto lavoro inutile, scandagliando il tipo di file o i suoi contenuti. I link fisici impediscono a `catman` di fare queste preziose operazioni (si noti che `catman` preformatta tutte le pagine man al fine di mostrarle più rapidamente). La terza alternativa presenta un piccolo inconveniente: poiché alcuni file system non gestiscono i link simbolici, questa soluzione sarebbe poco flessibile. Se ne conclude che la miglior soluzione consiste nell’utilizzare il meccanismo del sorgente di `groff`. Ecco come si fa: se volete che la vostra pagina man sia disponibile nella sezione 1 attraverso i nomi ‘foo’ e ‘zac’, collocate la pagina man in foo.1 e realizzate la pagina zac.1 in modo che appaia così:

```
.so man1/foo.1
```

È importante indicare esplicitamente sia il nome del file 'foo.1' sia la parte `man1/` della directory; infatti, quando `groff` verrà eseguito dal visualizzatore delle pagine man, avrà come directory di lavoro quella dei manuali e `groff` interpreterà gli argomenti di `.so` come nomi di percorso relativi a tale directory.

5. Quale pacchetto di macro dovrei utilizzare?

Esistono numerosi pacchetti di macro progettati appositamente per scrivere le pagine di manuale; di solito si trovano nella directory delle macro di groff /usr/lib/groff/tmac. I nomi dei file sono `tmac.`<qualcosa>, dove <qualcosa> è l'argomento per l'opzione -m di groff. Groff utilizzerà `tmac.`<qualcosa> quando verrà data l'opzione '`-m` <qualcosa>'. Spesso viene omesso lo spazio tra '`-m`' e '<qualcosa>', cosicché potremo scrivere '`groff -man`' per formattare le pagine man con il pacchetto di macro `tmac.an`. È questa la ragione dello strano nome 'tmac.an'. Oltre a tmac.an esiste un altro pacchetto di macro che ha avuto una certa diffusione, `tmac.doc`, creato presso l'Università della California di Berkeley (UCB). Molte pagine man di BSD usano tale macro e sembra che l'UCB l'abbia resa un vero standard per la documentazione. Le macro `tmac.doc` sono molto più flessibili, ma purtroppo ci sono alcuni visualizzatori di pagine man che non le usano e che effettuano sempre la chiamata `groff -man`. Per esempio, tutte le versioni di `xman` che conosco fanno cilecca con le pagine man che richiedono `tmac.doc`. in definitiva, se volete fare un favore a voi stessi, usate `tmac.an`; l'uso di qualsiasi altro pacchetto di macro è considerato dannoso. `tmac.andoc` è un pacchetto di pseudo macro che fa un primo esame del sorgente e poi, sulla base di quest'esame, carica `tmac.an` o `tmac.doc`. In effetti sarebbe meglio che i visualizzatori di pagine man utilizzassero questo pacchetto di pseudo macro; poiché però ciò non avviene, è meglio prendere confidenza con il buon vecchio `tmac.an`. Tutto quello che d'ora in poi verrà detto sulle macro vale per `tmac.an`. Ad ogni modo, se volete usare le macro di `tmac.doc`, date un'occhiata a `mdoc.samples` (http://www.freebsd.org/cgi/man.cgi?query=mdoc.samples), un esempio utile anche come tutorial. In alcune distribuzioni (come ho già detto) sono compresi mdoc(7), mdoc.samples(7) e groff_man(7).

Il massimo della documentazione ottenibile su `troff`, con descrizione minuziosa di tutte le macro, è il *Troff User's Manual*, disponibile nei formati html (http://cm.bell-labs.com/sys/doc/troff.html), PostScript (ps, 760K) (http://cm.bell-labs.com/sys/doc/troff.ps) o Portable Document Format (pdf, 240K) (http://cm.bell-labs.com/sys/doc/troff.pdf). Gli autori sono Jospeh F. Ossanna e Brian W. Kernighan; è stato revisionato in novembre 1992 ed è stato messo a disposizione dai Bell Labs di AT&T. Non dimenticate di visitare l'eccellente homepage di W. Richard Steven (http://www.kohala.com/start/) (famoso per *Unix Network Programming* e per la trilogia *TCP/IP Illustrated*), che ha anche un elenco di risorse su Troff (http://www.kohala.com/start/troff/troff.html) tra cui `tbl`, `eqn`, `pic` e altri filtri.

6. Quali preprocessori posso utilizzare?

Groff è distribuito insieme ad almeno tre preprocessori, `tbl`, `eqn` e `pic` (che su alcuni sistemi sono chiamati `gtbl`, `geqn` e `gpic`). Essi servono a tradurre le macro del preprocessore e i loro dati in normale input per troff. `Tbl` è un preprocessore di tabelle, `eqn` di equazioni/formule matematiche e `pic` di immagini. Si faccia riferimento alle pagine man relative per altre informazioni sulle funzionalità da esse fornite. Per farla breve: non scrivete pagine man che richiedano *uno qualsiasi* di questi preprocessori. Eqn produce solitamente un output che risulta tremendo per i dispositivi di videoscrittura e similari; sfortunatamente questo è il 99% di tutti i dispositivi su cui vengono visualizzate le pagine di manuale (be', almeno questo è quello che capita a me). Per esempio, XAllocColor.3x utilizza alcune formule con l'esponenziale. A causa della natura dei dispositivi di videoscrittura, l'esponente si troverà sulla stessa riga della base. N elevato a due diventa 'N2'. `Tbl` andrebbe evitato perché non funziona con xman, in tutte le versioni da me incontrate. Xman 3.1.6 usa il seguente comando per formattare le pagine man (nell'esempio si utilizza la pagina man signal(7)):

```
gtbl /usr/man/man7/signal.7 | geqn | gtbl | groff -Tascii -man /tmp/xmana01760 2>
/dev/null
```

e tale comando fallisce su sorgenti che usano `gtbl`, perché l'output di `gtbl` viene ridato in pasto a `gtbl` stesso. L'effetto finale è una pagina man priva di tabelle. Non so se si tratta di un bug o di una funzionalità, ma le possibilità sono due: o `gtbl` fa semplicemente fiasco nel processare il suo stesso output, oppure xman dovrebbe fare più attenzione a non utilizzare due volte `gtbl`. Inoltre, alcuni sistemi usano `grog` per determinare l'opzione da passare a groff. Sfortunatamente, a volte grog sbaglia e suggerisce di eseguire `groff -t` anche quando `tbl` non andrebbe usata. Per gestire le tabelle ci rimangono due trucchi:

1. Formattare a mano la tabella e posizionarla tra le righe .nf e .fi, evitando di inserire altra formattazione. Con questo metodo non avremo a disposizione grassetti o corsivi, ma potremo inserire facilmente tutte le tabelle che vorremo.
2. Usare una macro qualsiasi di `tbl` avendo cura di distribuire l'output di `tbl` invece dell'input. Tuttavia, rimane ancora sospesa la questione di `grog`: per questa utility, ogni file contenente una riga che inizia per `.TS`, richiede l'uso di `tbl`. L'output di `Tbl`, per qualche ragione a me sconosciuta, contiene ancora `.TS` e `.TE`; potete semplicemente eliminarli e continuare ad avere un buon risultato. YMMV, per cui verificatelo sulla vostra pagina man particolare.

Devo ancora vedere una pagina man che richieda la preprocessazione `pic`, ma non credo mi piacerebbe; come potete aver letto più sopra, `xman` non l'userebbe e `groff` farà certamente pasticci quando l'avrà in input.

7. Dovrei distribuire il sorgente e/o la documentazione già formattata?

Ecco i pro (+) e i contro (-) riguardo alle diverse possibilità di distribuzione delle pagine man:

1. Solo in sorgente:
   - + il pacchetto della distribuzione è più piccolo.
   - - è inaccessibile sui sistemi senza `groff`.
2. Solo la documentazione formattata non compressa:
   - + è accessibile anche su sistemi senza `groff.`
   - - l'utente non può generare un file dvi o postscript.
   - - sui sistemi che gestiscono già le pagine compresse c'è spreco di spazio del disco.
3. Solo la documentazione formattata e compressa:
   - + accessibile anche sui sistemi senza `groff`.
   - - l'utente non può generare un file dvi o postscript.
   - - che formato di compressione andrebbe usato? .Z? .z? .gz? Tutti?

4. Sia in sorgente che formattata compressa:
   - + accessibile anche su sistemi senza `groff`.
   - - il pacchetto di distribuzione è più voluminoso.
   - - alcuni sistemi potrebbero aspettarsi pagine man formattate e compresse.
   - - le informazioni sono ridondanti sui sistemi muniti di `groff`.

IMHO è meglio distribuire il sorgente soltanto. L'argomento dell'inaccessibilità sui sistemi senza `groff` non ha un valore reale. Le 500 (e oltre) pagine man del Linux Documentation Project sono distribuite in sorgente soltanto, così come le pagine man di XFree86 e quelle della FSF. In effetti, mi è capitato raramente di vedere pacchetti software distribuiti con pagine man formattate. Se un amministratore di sistema è davvero interessato all'accessibilità delle pagine man, allora di sicuro ha già installato `groff`.

8. Quali sono le convenzioni sui font?

Prima di tutto: non usate gli operatori diretti dei font come `\fB`, `\fP` ecc.. Invece, usate le macro che accettano argomenti. In tal modo eviterete un errore comune: dimenticare il cambio del font alla fine della parola estendendo erroneamente il grassetto o il corsivo fino al successivo cambio di font. Succede più spesso di quanto si pensi, credetemi. Le macro di `tmac.an` forniscono i seguenti formati tipografici:

.B grassetto (bold)

.BI grassetto alternato al corsivo (italic)

.BR grassetto alternato al carattere Roman

.I corsivo

.IB corsivo alternato al grassetto

.IR corsivo alternato al Roman

.RB Roman alternato al grassetto

.RI Roman alternato al corsivo

.SM small: piccolo (di dimensioni pari a 9/10 di quella normale)

.SB piccolo e grassetto (*non* piccolo alternato al grassetto)

X alternato con Y significa che gli argomenti di ordine dispari vengono resi nel formato X, mentre quelli pari sono in formato Y. Per esempio:

.BI "Arg 1 è grassetto, " "Arg 2 è corsivo, " "questo è grassetto, " "e questo corsivo."

Le doppie virgolette sono necessarie per includere gli spazi in un argomento; senza di esse non avremmo modo di far apparire gli spazi tra i diversi formati tipografici. A tutti gli effetti, quando al passaggio tra un formato tipografico e l'altro *vorrete* evitare la comparsa di spazi bianchi, saranno necessarie solo le macro con formato tipografico alternato. Ciò vale per la maggior parte dei casi. Ecco come dovreste far uso dei diversi formati tipografici (userò un frammento spudoratamente rubato da man(7)):

Benché nel mondo UNIX esistano molte convenzioni arbitrarie per le pagine man, l'esistenza di svariate centinaia di pagine man specifiche per Linux definiscono i nostri standard: nel caso delle funzioni, gli argomenti sono sempre indicati in corsivo, anche nella sezione SINTASSI, mentre il resto della funzione è indicata in grassetto:

.BI "miafunzione(int " argc ", char **" argv);

I nomi dei file sono sempre in corsivo, tranne che nella sezione SINTASSI, in cui i file inclusi sono in grassetto. Quindi andrebbero utilizzati i formati

.I /usr/include/stdio.h

e

.B #include <stdio.h>

Le macro speciali (quelle della programmazione C, da non confondere con le macro delle pagine man, NdT), solitamente indicate in maiuscolo, sono in grassetto:

.B MAXINT

Quando si elenca una serie di codici d'errore, questi vanno resi in grassetto. Di solito, per questo elenco si utilizza la macro .TP (il paragrafo con i tag appesi) come segue:

```
.TP
.B EBADF
.I fd non è un descrittore di file valido.
.TP
.B EINVAL
.I fd è inaffidabile in lettura
```

Ogni riferimento ad un'altra pagina man (o al soggetto della pagina man corrente) è in grassetto. Se presente, il numero di sezione del manuale viene reso in carattere roman, senza spazi:

.BR man (7)

Gli acronimi vengono resi meglio usando i caratteri tipografici piccoli, per cui raccomando di fare come negli esempi che seguono

.SM UNIX

.SM ASCII

.SM TAB

.SM NFS

.SM LALR(1)

9. Come affinare una pagina man

Di seguito vengono mostrati alcuni suggerimenti per accrescere affidabilità, leggibilità e 'formattabilità' della vostra documentazione.

- Provate voi stessi gli esempi da inserire nella pagina man, così da verificarne il reale funzionamento (usate il copia-e-incolla dalla pagina man per assicurarvi di immettere l'esatta sequenza sulla riga di comando usata per i test). Non riportate l'output del comando come voi *pensate* che venga prodotto, ma copiate l'output reale.
- Rileggete, usate ispell e fate leggere la documentazione a qualcun altro, specialmente se l'inglese non è la vostra lingua madre. L'HOWTO che state leggendo, ad esempio, ha appena passato il suo ultimo test (un ringraziamento speciale va a Michael Miller per il suo contributo particolarmente eroico! Tutte le imperfezioni rimanenti sono interamente attribuibili a me). Altri contributi sono sempre bene accetti.

- Verificate la vostra pagina man: quando la passate alla formattazione con `groff` vengono visualizzati errori? Il comando `man`(1) si lamenta quando eseguite `man mioprogramma`? Fornisce il risultato atteso? `xman`(1x) e `tkman`(1tk) fanno cilecca con il vostro manuale? Con XFree86 3.1 c'è xman 3.1.6 - X11R6, che prova ad effettuare una decompressione usando `gzip -c -d < %s > %s zcat < %s > %s`
- Il comando `makewhatis`(8) è in grado di estrarre la descrizione dalla sezione NOME?
- Traducete la vostra pagina man in formato HTML usando `rman` prelevabile da http://polyglotman.sourceforge.net/ e visualizzate il risultato con molti browser Web (netscape, mozilla, opera, lynx, ...). Verificate che i riferimenti incrociati tra le vostre pagine man funzionino, nel codice HTML generato, come link ipertestuali. Se il vostro programma ha un sito dedicato, pubblicateci le sue pagine man e tenetele aggiornate.
- La utility `rman` può tradurre le pagina man anche in LaTeX, RTF, SGML e altri formati; verificate anche questi, se volete accludere le pagine man in un libro o in un documento più ampio.
- Provate a tradurre le pagine man in HTML usando `man2html`, che è parte del pacchetto man di Linux fin dalla sua versione man-1.4. La utility `man2html` è un traduttore meno ambizioso di `rman`, ma quasi tutti gli utenti Linux lo possiedono, cosicché è importante assicurarsi che `man2html` non manchi di elaborare correttamente la vostra pagina man.

10. Come è possibile ottenere una pagina man in formato testo senza tutti quei ^H^_ ?

Date un'occhiata al funzionamento del comando `col`(1), che può filtrare le sequenze di backspace. Ma se proprio non volete perder tempo a leggere, ecco la soluzione nuda e cruda:

```
funnyprompt$ groff -t -e -mandoc -Tascii paginaman.1 | col -bx > paginaman.txt
```

Le opzioni `-t` e `-e` impongono a `groff` di effettuare l'elaborazione con `tbl` e `eqn`. Questa è una soluzione esagerata per le pagine di manuale che non necessitano di essere preprocessate, ma non fa alcun danno, tranne per qualche ciclo di CPU sprecato. D'altro canto, se non si usasse l'opzione `-t` nei casi in cui fosse necessaria, le tabelle verrebbero formattate in modo pessimo. Potreste anche trovare (o meglio: "indovinare") il comando necessario a formattare un dato documento `groff` (non è necessariamente una semplice pagina man) immettendo

```
funnyprompt$ grog /usr/man/man7/signal.7
groff -t -man /usr/man/man7/signal.7
```

"Grog" sta per "GROff Guess" e fa proprio quello che dice: guess (indovina). Se fosse perfetto, non dovremmo usare altre opzioni. Però ho notato che sbaglia a indovinare sia i pacchetti di macro sia i preprocessori. Ecco un mio piccolo script in Perl per eliminare intestazioni e piè di pagina; quando vorrete stampare le pagine man, questo script farà risparmiare a voi alcune pagine, e un albero a madre natura. Salvate lo script in un file chiamato `strip-headers` e dategli i permessi con chmod 755.

```
#!/usr/bin/perl -wn
#  elabora l'intero file tutto in una volta:
undef $/;
#  cancella l'intestazione:
s/^\n*.*\n+//;
#  cancella il pié di pagina:
s/\n+.*\n+$/\n/g;
```

```
#  cancella l'interruzione di pagina:
s/\n\n+[^ \t].*\n\n+(\S+).*\1\n\n+/\n/g;
#  riduce le righe vuote multiple ad una sola:
s/\n{3,}/\n\n/g;
#  ecco quel che rimane...
print;
```

Questo script va usato come primo filtro dopo il comando `man`, in quanto fa affidamento sul numero di caratteri di fine riga (newline) emessi in output da `groff`. Per esempio:

```
funnyprompt$ man bash | strip-headers | col -bx > bash.txt
```

11. Come si può ottenere una pagina man in formato PostScript di elevata qualità?

```
funnyprompt$ groff -t -e -mandoc -Tps paginaman.1 > paginaman.ps
```

Stampate o visualizzate questa pagina usando il vostro programma preferito per leggere/stampare i file PostScript. Per una spiegazione delle opzioni usate, si veda la domanda 10.

12. Come far funzionare correttamente 'apropos' e 'whatis'?

Supponiamo di voler sapere quali compilatori siano installati sul nostro sistema e come si possano utilizzare. Per rispondere a queste domande (poste frequentemente) si può digitare

```
funnyprompt$ apropos compiler
f77 (1) - Fortran 77 compiler
gcc (1) - GNU C and C++ compiler
pc (1) - Pascal compiler
```

`Apropos` e `whatis` sono usati per fornire un breve resoconto su quali pagine di manuale contengono informazioni riguardo a un certo argomento. Entrambi i comandi effettuano una ricerca in numerosi file chiamati 'whatis' che possono essere trovati in ognuna delle directory di base dei manuali. Come già precedentemente affermato, i file del database di whatis contengono una sola riga per ogni pagina di manuale presente nel rispettivo albero di directory. In effetti, questa riga è esattamente la sezione NOME (per esser precisi: il contenuto della sezione è riunito su una sola riga e privato della sillabazione; si noti che il numero di sezione è citato all'interno di parentesi). I file del database di whatis sono creati con il programma `makewhatis`(8). Poiché ne esistono svariate versioni, per verificare le opzioni disponibili è importante far riferimento alla pagina di manuale. Per far sì che `makewhatis` estragga correttamente le sezioni NOME, è importate che chi scrive il manuale si adegui al suo formato, descritto alla domanda 3. La differenza tra `apropos` e `whatis` sta semplicemente nel tipo di ricerca che fanno sulla riga. `Apropos` (che è equivalente a `man -k`) cerca la stringa ovunque posizionata sulla riga, mentre `whatis` (equivalente a `man -f`) prova a verificare la coincidenza della stringa con il nome completo del comando, posizionato nella parte che precede il trattino. Di conseguenza, '`whatis cc`' ci informerà se esiste un manuale relativo a `cc` e sarà reticente riguardo a `gcc`.

Correzioni e suggerimenti sono benvenuti!

13. Diritto d’autore

14. Ringraziamenti

- Ringrazio Michael Miller per aver verificato la leggibilità dell'intero HOWTO (in febbraio 2001); Gordon Torrie per le tante utili annotazioni grammaticali (in agosto 2001). Le imperfezioni grammaticali e stilistiche rimanenti sono dovute interamente a me.
- S.u.S.E. (.de) (http://www.SuSE.de/) (o .com (http://www.SuSE.com/)) che è stato l'unico distributore a inviarmi una copia gratuita del loro ultimo prodotto, riconoscendo il mio lavoro come autore di howto.
- George B. Moody per i suggerimenti aggiuntivi sul perfezionamento di una pagina man.

Se ho dimenticato qualcuno, fatemelo sapere.

15. Storia dell'HOWTO

- Marzo 6 2001: Ora il sorgente HTML supera l'esame di `weblint -pedantic`. Paragrafo 6: Aggiunti i metodi per aggirare i malfunzionamenti di `tbl`. Aggiunti Ringraziamenti e Storia dell'HOWTO. Aggiunto l'Id RCS.
- Agosto 9 2001: L'Howto è posto sotto le due clausole della licenza BSD.
- Agosto 20 2001: Migliorato sotto il profilo grammaticale. Uso di una lista numerata nel sommario (TOC).
- Ottobre 28 2001: Aggiunti i riferimenti a mdoc(7), mdoc.samples(7) e groff_man(7).
- Aprile 28 2002: Corretto un errore grammaticale con s/particular/particularly/.
- Aprile 30 2002: Aggiornato il link al tutorial BSD di groff_mdoc.
- Novembre 29 2002: Aggiunti altri suggerimenti per raffinare le pagine man.
- Dicembre 15 2002: Pubblicato in versione SGML derivata da HTML. Eliminato il link defunto a LSM.