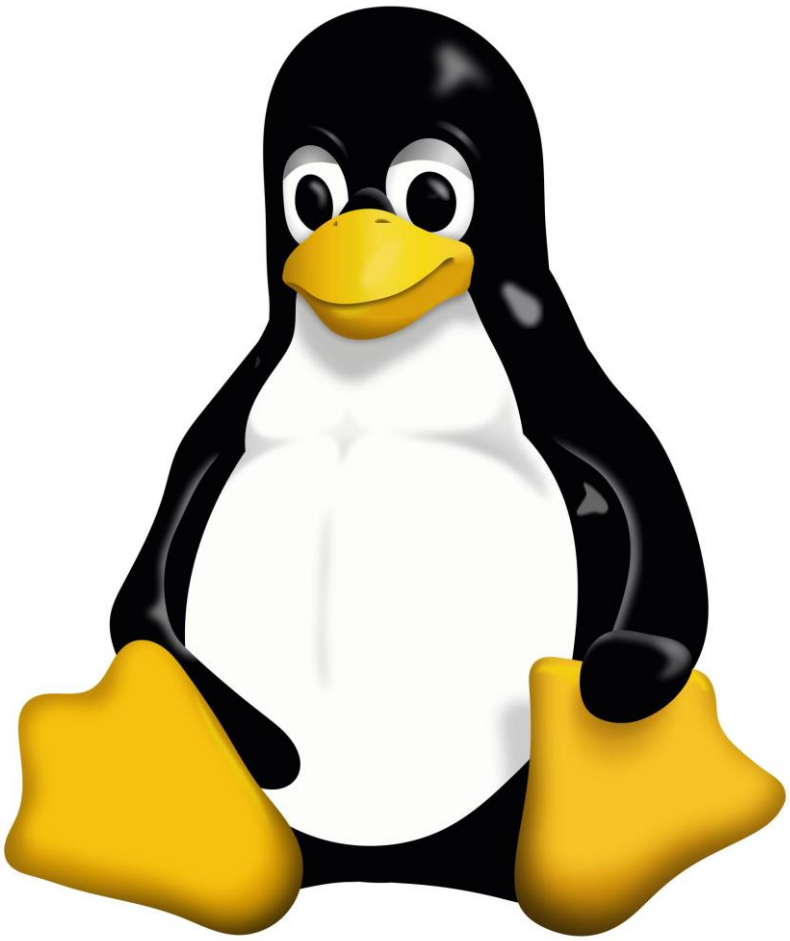
Quaderni dei Viandanti

Marco Moraschi

Da Unix a Linux

Una storia del Software Libero



Viandanti delle Nebbie

Marco Moraschi
DA UNIX A LINUX
Edito in Lerma (AL) nel dicembre 2019
Per i tipi dei **Viandanti delle Nebbie**
collana *Quaderni dei Viandanti*
https://www.viandantidellenebbie.org
https://www.facebook.com/viandantidellenebbie
https://www.instagram.com/viandantidellenebbie/

Quaderni dei Viandanti

Marco Moraschi

Da Unix a Linux

Una storia del Software Libero

Viandanti delle Nebbie

A chi pensa fuori dalla scatola

INDICE

Introduzione

Se tornassimo indietro nel tempo di 40 o 50 anni, i computer cui ci troveremmo di fronte non sarebbero gli attuali PC desktop o i notebook leggerissimi che ci portiamo appresso, ma calcolatori dalle dimensioni quasi inimmaginabili per la nostra epoca, grandi come stanze o addirittura come intere case. Se già le dimensioni costituirebbero un problema, un aggravio ulteriore sarebbe rappresentato da una pratica piuttosto comune all'epoca: ogni computer aveva un sistema operativo diverso. Tutti i sistemi di computer commercialmente disponibili erano infatti scritti in un codice specificamente sviluppato per ognuno, al fine di svolgere compiti molto specializzati, con il risultato che il software di un dato sistema non poteva girare su un altro. Essere quindi in grado di utilizzare un determinato computer, software o sistema, non significava affatto essere in grado di padroneggiare anche gli altri dispositivi; ciò rappresentava un'enorme difficoltà non solo per gli utenti privati, ma anche, e soprattutto, per gli amministratori di sistema. Altro punto critico era inoltre il fatto che i computer erano estremamente costosi e, oltre all'investimento iniziale, bisognava spendere ulteriore denaro per spiegare agli utenti come funzionavano. Le cose procedettero in questo modo per un po' di tempo, sino a che la tecnologia non fu abbastanza avanzata da iniziare a cambiare poco alla volta lo stato dell'arte. Questo testo racconta la storia di quel cambiamento epocale, una storia (anche) del Software Libero, da Unix a Linux. Buona lettura.

Storia

In principio era UNIX

I *Bell Labs*, i laboratori di ricerca della Bell Telephone Company, sentivano l'esigenza di un nuovo sistema operativo sin dalla fine degli anni '50. Il primo risultato di questa necessità fu, nel 1957, il sistema operativo BESYS che, sebbene molto utilizzato internamente, aveva però una distribuzione limitata al di fuori dei Bell Labs. A metà degli anni '60 si trovarono quindi nella condizione di dover decidere se creare un nuovo sistema operativo o se adottarne uno sviluppato da terze parti; insieme alla General Electric e al MIT (*Massachusetts Institute of Technology*) decisero, nel 1965, di sviluppare un nuovo sistema operativo che chiamarono *MULTIplexed Information and Computing Service* (MULTICS). Questo progetto condiviso voleva dimostrare che era possibile creare un sistema operativo multiutente per utilizzo generico, sicuramente un'idea nuova per l'epoca. Il sistema MULTICS si basava su un sistema operativo sperimentale sviluppato dal MIT chiamato *Compatible Time Sharing System* (CTSS). Nonostante fosse un progetto molto innovativo, che aveva portato nuovi approcci nello sviluppo dei sistemi informatici, a causa degli alti costi di sviluppo e della grande quantità di tempo richiesta, i Bell Labs decisero, nel 1969, di abbandonare lo sviluppo di MULTICS. Tra coloro che lavoravano a questo progetto c'erano però due programmatori che sarebbero in seguito diventati molto famosi: Ken Thompson e Dennis Ritchie.

Nel 1969 Ken Thompson si dedicò alla scrittura del codice di esecuzione di un gioco chiamato "*Space Travel*" che simulava i movimenti del Sole e dei pianeti, così come il movimento di una navicella spaziale che poteva atterrare in diversi luoghi nel Sistema Solare (d'altronde il 1969 è stato l'anno in cui l'uomo ha messo piede sulla Luna!). L'esecuzione di Space Travel sul GE-645, un computer della General Electric su cui era installato il sistema operativo GECOS, avveniva però a scatti ed era molto costosa; di qui nacque l'esigenza di sviluppare un supporto diverso. Thompson e alcuni suoi colleghi (Douglas McIlroy, Joe Ossanna e Rudd Canaday) si misero quindi al lavoro inizialmente su un GE-635 e svilupparono così un nuovo sistema operativo multi-tasking, che sfruttava una gestione del file system innovativa e comprendeva un interprete di comandi e alcune utility per un altro tipo di computer che avevano disponibile: un vecchio Digital Equipment Corp.

(DEC) PDP-7. Brian Kernighan lo battezzò UNICS (*UNiplexed Information and Computing Service*), poco dopo sintetizzato in UNIX dallo stesso Kernighan. Questo sistema operativo aveva un altro punto di forza, era cioè capace di riutilizzare il codice, rendendolo quindi compatibile con più sistemi. UNIX necessitava infatti solamente di una piccola porzione di codice che fosse appositamente adattata ad ogni specifico sistema: il kernel. Un po' in sordina e con scarsi finanziamenti, l'avventura continuò. Nel 1970 i pionieri di UNIX acquistarono un DEC PDP-11 con ben 24 KB di memoria! Di questi 24 ne utilizzarono 12 per il sistema operativo, avendone quindi a disposizione altri 12 per i programmi utente e la RAM. Nel frattempo, Thompson si era messo a lavorare su un compilatore Fortran, rendendolo compatibile con il linguaggio B, derivato da BCPL. Nel 1971 venne rilasciata la prima versione di UNIX, che includeva il supporto a un compilatore B.

Il numero di versione per i primi sistemi UNIX corrisponde al numero dell'edizione del "*Programmer's Reference Manual*" corrente quando veniva rilasciata la distribuzione; il codice e il manuale venivano modificati indipendentemente. Nel manuale di programmazione di UNIX del 1972 si trova scritto un dettaglio curioso: in quell'anno le installazioni di UNIX erano cresciute fino a toccare la sbalorditiva quota di...10!

Tra il 1971 e il 1973 Dennis Ritchie, a partire dal lavoro svolto da Thompson, sviluppò il primo compilatore C. Nel 1973 venne quindi rilasciata la terza edizione di UNIX: era la prima volta che il kernel di un sistema operativo era scritto con un linguaggio di programmazione di alto livello, ovvero proprio il linguaggio C su cui avevano lavorato Thompson e Ritchie (sarebbero tuttavia passati ancora 5 anni prima che venisse pubblicato il famoso libro di Thompson e Ritchie intitolato "*The C Programming Language*"). L'utilizzo del C rappresentò un'altra idea innovativa per l'epoca, in quanto fino ad allora tutti i sistemi operativi erano scritti in linguaggio assembly; ciò rese UNIX facilmente mantenibile e, soprattutto, ampiamente portabile. Da quel momento in avanti i computer di diversi costruttori sarebbero stati in grado di comunicare fra loro sulla stessa rete e gli utenti avrebbero potuto lavorare su sistemi diversi senza necessità di nuovi studi. Naturalmente non fu tutto così immediato e semplice come potrebbe sembrare: durante lo sviluppo UNIX veniva infatti ampiamente usato solo all'interno dei laboratori della AT&T. La prima versione ampiamente utilizzata al di fuori dei Bell Laboratories fu la versione 6, rilasciata nel 1975. Dicendo ampiamente utilizzata dobbiamo comunque pensare che i sistemi Unix erano presenti solo in ambienti molto

vasti con mainframe e minicomputer (i PC, come li intendiamo oggi, sono "micro" computer), ovvero le università, il governo o società molto facoltose. Bisognerà aspettare fino al 1986 prima che UNIX sia installato su oltre 100 mila differenti piattaforme e architetture a livello globale.

Tornando indietro di qualche anno, nel settembre 1973 Berkley Tague, a capo del dipartimento di computer planning della Bell, formò lo *UNIX Support Group* (USG), che iniziò a lavorare a stretto contatto con il team dei Bell Labs. L'anno successivo, nel 1974, Ken Thompson e Dennis Ritchie pubblicarono quello che è diventato il loro paper più famoso: "*The UNIX Time-Sharing System*".

Quando venne distribuita la versione 7, nel 1979, l'USG assunse il controllo amministrativo e la responsabilità del gruppo di ricerca per le distribuzioni di UNIX all'interno di AT&T. L'USG divenne in seguito USDL, ovvero *UNIX System Development Laboratory*. La versione 7 di UNIX, che girava su PDP-11/70 e su Interdata 8/32, è la progenitrice della maggior parte dei moderni sistemi UNIX-like. C'è però un dettaglio di fondamentale importanza che è bene riportare a questo punto: alla fine degli anni '70 la Bell si rese conto dell'importanza commerciale del suo progetto. A partire dalla versione 7, la licenza di UNIX proibì alle università di usare il codice sorgente di questo sistema operativo come oggetto di studio nei loro corsi. Nel 1983, con il rilascio di UNIX System V Release 1, UNIX, fino ad allora diffuso solo in ambiente universitario a fini di ricerca, divenne un prodotto commerciale a tutti gli effetti, regolarmente distribuito dalla AT&T. Da quel momento in avanti l'azienda si impegnò a mantenerne la compatibilità: tutto ciò che girava in System V Release 1 avrebbe dovuto rimanere eseguibile per sempre nelle versioni successive di UNIX. Nel 1983 gli sviluppatori originali, Ritchie e Thompson, furono premiati per il loro lavoro su UNIX dall'*Association for Computing Machinery* con il premio più prestigioso: il premio *A. M. Turing*. Tra le motivazioni del premio si legge: "*Il modello del sistema UNIX ha portato una generazione di programmatori di software a nuovi modi di pensare alla programmazione*".

Con lo smembramento di AT&T nel 1984 nelle famose 7 *baby-bells*, società più piccole che operavano a livello locale, la condivisione gratuita di UNIX ebbe fine. Solamente alcune università, come quella di Berkeley in California, continuarono a sviluppare la loro versione derivata da UNIX. Tuttavia, il sasso era ormai stato lanciato e alla fine degli anni '80 molte

persone possedevano *home computers*. A quel tempo esistevano quasi un centinaio di diverse versioni di UNIX disponibili per l'architettura PC, ma nessuna di loro era realmente libera e, ancora più importante, erano tutte terribilmente lente, cosicché la maggioranza della gente faceva girare MS DOS o Windows 3.11 nei propri computer domestici.

Probabilmente in seguito alla collaborazione tra AT&T e la Sun Microsystems per lo sviluppo di UNIX System V Release 4, nel 1988 sette importanti compagnie del settore dei computer fondarono la Open Software Foundation (OSF) con l'obiettivo di produrre insieme uno standard aperto per il sistema operativo UNIX. La risposta a questa nuova organizzazione non si fece attendere: nello stesso anno un consorzio guidato proprio da AT&T e Sun Microsystems diede vita alla *UNIX International* (UI). La mossa fu in realtà più di marketing che non di reali intenti: non avendo alle spalle solidi progetti di sviluppo, la UI si disgregò in pochi anni. La guerra per un sistema operativo aperto, però, era solo all'inizio.

BSD, il primo sistema Unix-like

La piccola dimensione, la modularità e il disegno pulito dei primi sistemi UNIX portarono a lavori basati su questo sistema operativo numerose altre organizzazioni informatiche, come DEC (*Digital Equipment Corporation*) e BBN (*Bolt Beranek and Newman*), e le Università dell'Illinois, di Harvard e di Purdue. Il maggior fermento si concentrò però presso l'*Università di Berkeley in California*, dove a partire dal 1977, anche grazie al lavoro di William Joy e Ozalp Babaoglu, si cominciò a distribuire la prima variante di UNIX: BSD (Berkeley Software Distribution). Per questo software, nacque una licenza d'uso che rimane ancora oggi alla base della filosofia del software libero: la licenza BSD. Tuttavia, questa edizione libera dello UNIX BSD non ebbe vita facile, dato che da quel momento iniziarono delle contese giudiziarie sulla proprietà di alcune porzioni di codice ritenute libere, a torto o a ragione che fosse. Da questi problemi si svilupparono in seguito altri progetti indipendenti per ottenere, finalmente, un sistema BSD libero da codice proprietario. Il primo di questi fu chiamato NetBSD, al quale si aggiunsero FreeBSD e, più tardi, anche OpenBSD.

Tornando a BSD, il lavoro sulla gestione della memoria convinse la DARPA (*Defense Advanced Research Projects Agency*) a finanziare Berkeley per lo sviluppo di un sistema standard UNIX per uso governativo, che

permettesse la creazione di una rete al fine di collegare i maggiori centri di ricerca; ne risultò 4BSD UNIX. Il progetto 4BSD è stato guidato da un comitato di coordinamento che ha incluso molte persone note delle comunità della rete e di UNIX.

Il progetto BSD si è infine arrestato nel 1995 con il rilascio da parte della Berkeley University della versione 4.4.

Tra i progetti derivati da UNIX e BSD ricordiamo Sun OS (poi diventato Solaris), ovvero la versione realizzata dalla Sun Microsystems (fondata nel 1982 da William Joy, uno degli autori di BSD), e XENIX, la versione di UNIX sviluppata dalla Microsoft, che influenzò pesantemente lo sviluppo dei suoi sistemi operativi successivi. Darwin OS, infine, un altro sistema operativo derivato da BSD, venne invece usato come base per il sistema operativo MAC OS X della Apple (ora chiamato macOS).

Un paragrafo a parte merita invece lo spin-off principe: Linux.

L'etica hacker: il progetto GNU

Il programmatore Richard Stallman cominciò a lavorare nel laboratorio di Intelligenza Artificiale del MIT nel 1971; entrò così a far parte di una comunità di hacker molto attivi, a cui piaceva scrivere programmi ed esplorare tutte le possibilità offerte dai computer. A quel tempo la parola "hacker" non aveva il significato e l'accezione negativa che le attribuiamo oggi. L'hacker era colui che si divertiva a programmare, risolvere problemi con i software, colui che, dotato di capacità informatiche eccezionali, era in grado di "spingere" i software oltre le loro funzioni originarie. Gli hacker del MIT avevano sviluppato un sistema operativo completo (chiamato Incompatible Time-Sharing), interamente scritto all'interno del laboratorio di AI. Stallman iniziò fin da subito a portare il suo contributo a questo nuovo sistema operativo, migliorandolo e aggiungendo nuove funziona-

Figura 1: Il simbolo del progetto GNU

lità. Lo sviluppo procedette con entusiasmo sino a quando dall'esterno del laboratorio furono caldamente invitati a utilizzare le password nei loro computer. Tuttavia, i primi hacker che avevano sviluppato quei computer non avevano previsto l'utilizzo di password, perché si erano resi conto che, conoscendo le password, gli amministratori avrebbero potuto avere il controllo completo di tutti gli utenti; secondo la filosofia hacker chiunque si fosse seduto al computer doveva essere lasciato libero di fare tutto ciò che desiderava. A Richard Stallman e un gruppetto di altri programmatori non piacque dunque quando su una delle macchine del laboratorio furono inserite delle password di accesso. Stallman tentò quindi un'azione eversiva: scoprì come decifrare le password e guardando il database delle credenziali cifrate fu in grado di individuare ciò che ogni persona digitava per entrare nel sistema. Inviò allora dei messaggi agli utenti dicendo loro che era a conoscenza della loro password e invitandoli a emulare il suo comportamento: usare semplicemente '*enter*' come password di accesso, "*molto più breve e facile da scrivere*". A detta di Stallman, con quel messaggio, egli stava comunicando agli utenti che "l*a sicurezza era solo una buffonata*"; alla fine, un quinto degli utenti di quel computer fece come Stallman aveva suggerito. Questa e altre esperienze negative, come l'impossibilità di poter sviluppare dei codici poiché protetti da licenza, ebbero un profondo impatto su Stallman, che a metà degli anni '80 era forse l'ultimo custode dell'etica hacker sviluppata al MIT. Egli utilizzò le esperienze e i principi assimilati nei suoi anni di permanenza all'AI Lab come linee guida per la sua opera più conosciuta, un programma di text editing chiamato Emacs che permetteva agli utenti di personalizzarlo senza limite: la sua architettura aperta incoraggiava infatti le persone ad aggiungervi nuove funzioni e a migliorarlo senza sosta. Stallman distribuiva gratis il programma a chiunque accettasse la sua unica condizione: rendere disponibili tutte le modifiche apportate in modo da collaborare al miglioramento di Emacs, che divenne quasi subito l'editor di testi standard nei dipartimenti universitari di informatica. Dopo aver lasciato il suo lavoro al MIT nel 1984, l'anno successivo fondò la FSF (*Free Software Foundation*) con lo scopo preciso di creare e diffondere la filosofia del software libero, dove con libertà s'intende la possibilità data agli utenti di distribuire e modificare il software a seconda delle proprie esigenze e di poter distribuire anche le modifiche effettuate. Queste idee filosofiche si tradussero in pratica nella redazione di un contratto di licenza d'uso, la GPL (*General Public License*), studiata appositamente per proteg-

gere il software libero in modo che non potesse essere "accaparrato" da chi poi avrebbe potuto impedirne la diffusione libera. Oggi il copyright del software protetto in questo modo viene definito copyleft (maggiori dettagli nel capitolo "*La licenza GNU GPL e il Copyleft*").

Il software libero richiedeva però delle basi, prima di tutto un sistema operativo. In questo senso, l'obiettivo pratico che si prefiggeva Richard Stallman era quello di realizzare, con l'aiuto di volontari, un sistema operativo completamente libero. Nacque così, a coronamento di un percorso di riflessione iniziato negli anni '70, il progetto GNU (*Gnu's Not Unix*), con il quale, dopo la realizzazione di un compilatore C, si desiderava costruire una serie di programmi di servizio necessari nel momento in cui il kernel, il cuore del sistema operativo, fosse stato portato a termine. Il progetto GNU, annunciato per la prima volta il 27 settembre 1983, diede così vita a una grande quantità di software utilizzabile sulla maggior parte delle piattaforme UNIX, indirizzando implicitamente il software libero nella direzione di sistemi di questo tipo.

Sul finire degli anni '80, c'era però ancora una cosa di cui aveva bisogno il sistema operativo GNU per poter essere completo e sostituire UNIX: il kernel! Lo sviluppo di un kernel aperto (conosciuto con il nome *GNU Hurd*) da parte dei programmatori che avevano aderito al progetto procedeva infatti molto a rilento, richiedendo più tempo rispetto a quello previsto. Ben presto, però, il problema sarebbe stato inaspettatamente risolto: all'Università di Amsterdam e a quella di Helsinki qualcuno stava premendo sull'acceleratore. Una nuova storia stava per essere scritta.

Il "casus belli": MINIX

Alla fine degli anni '80, nel 1987, il professore universitario olandese Andrew S. Tanenbaum, docente di sistemi di rete alla *Vrije Universiteit di Amsterdam*, completò un sistema operativo Unix-like realizzato specificamente per uso didattico: Minix. Molti studenti potevano finalmente comprendere i fondamenti della progettazione e il funzionamento di un sistema operativo mettendoci materialmente "le mani sopra"; Minix era infatti open-source. Era sufficiente acquistare il libro scritto dallo stesso professor Tanenbaum e si otteneva un sistema Minix completo di sorgenti. Tuttavia, Minix aveva dei seri problemi: poteva essere usato, distribuito e modificato, solo per fini didattici e sembrava essere comunque molto limitato sotto

numerosi aspetti, per esempio supportava male la nuova architettura i386 a 32 bit, all'epoca tanto economica e popolare. Un giovane studente finlandese al secondo anno di Informatica all'Università di Helsinki, appassionato di programmazione e insoddisfatto del sistema operativo Minix, decise di avviare un progetto sullo studio delle funzionalità di multiprogrammazione dei microprocessori i386. Egli voleva realizzare qualcosa di innovativo che portasse la tecnologia dei sistemi operativi UNIX sui Personal Computer. Quel ragazzo era Linus Torvalds.

Linux, arriva il pinguino!

Inizialmente Linux non era altro che un obiettivo personale di Linus Torvalds: egli voleva essere in grado, infatti, di gestire sul suo computer personale un ambiente simile a quello su cui era abituato a lavorare in università. Non riuscendo però a trovare nulla di simile, decise quindi di mettersi al lavoro per crearlo da sé. In breve tempo cominciò dunque a informarsi e porre delle domande sui newsgroup per cercare risposte e soluzioni che lo potessero aiutare ad avere UNIX sul suo PC. Qui sotto c'è uno dei suo primi messaggi su *comp.os.minix*, datato 3 luglio 1991:

> *Hello netlanders,*
> *Due to a project I'm working on (in minix), I'm interested in the posix standard definition. Could somebody please point me to a (preferably) machine-readable format of the latest posix rules? FTP-sites would be nice.*

Ovvero:

> *Ciao amici della rete*[1]*,*
> *a causa di un progetto su cui sto lavorando (in minix), sono interessato alla definizione degli standard posix. Qualcuno sarebbe così gentile da indicarmi un formato (preferibilmente) leggibile da macchina sulle ultime regole posix? I siti FTP andrebbero benissimo.*

Gli standard POSIX (*Portable Operating System Interface for UNIX) erano una famiglia di standard elaborati dall'Institute of Electrical and Electronic Engineers* (IEEE) a partire dal 1985. Il loro scopo era quello di definire alcuni concetti comuni (standard per l'appunto) per le interfacce di un sistema operativo. UNIX richiedeva proprio gli standard POSIX ed è per questo motivo che Linus Torvalds iniziò a informarsi su di essi.

[1] [N.d.T] Traduzione libera

Sebbene fosse nato come tale, Linux non rimase il progetto personale di una sola persona: Torvalds, servendosi di internet, domandò infatti aiuto a tutti coloro che fossero interessati al suo progetto e in breve tempo coinvolse un enorme numero di persone, unite dal fatto che si trattava di un progetto libero da qualsiasi restrizione legale. Quando ormai circa un migliaio di persone aveva già messo mano a Linux e su tutti i newsgroup non si parlava d'altro, finalmente anche il professor Tanenbaum in persona fece sentire la sua voce. Egli affermò perentorio che Linux nasceva obsoleto, in quanto era una riscrittura di qualcosa che esisteva già da vent'anni. Tanenbaum era dell'idea che un sistema operativo veramente nuovo avrebbe dovuto far riferimento a soluzioni basate su *microkernel*, nei quali la maggior parte del sistema operativo viene eseguita in termini di processi separati, e la comunicazione viene gestita da un kernel di dimensioni ridotte. Tra Torvalds e Tanenbaum era diverso anche l'approccio per lo sviluppo di un sistema operativo, in quanto Tanenbaum sosteneva e pretendeva che chiunque avesse realizzato del codice per il suo SO lo avrebbe dovuto preventivamente sottoporre alla sua approvazione. Lo scopo di Linus Torvalds fu invece fin dall'inizio quello di avere un sistema libero e completamente aderente all'originale UNIX (nelle prime fasi si ispirò in particolare a Sun OS, il sistema che utilizzava all'università in quel periodo).

Agli inizi degli anni '90 non era stato ancora inventato il plug-and-play, ma così tante persone erano interessate ad avere un sistema UNIX che questo non fu un grosso ostacolo. Nuovi driver furono resi disponibili per tutti i tipi di nuovo hardware a una velocità sempre maggiore. Non appena un nuovo componente hardware era a disposizione, qualcuno lo comprava e lo sottoponeva al *Linux test*, rilasciando codice libero per una gamma sempre più ampia di hardware. Questi programmatori non si limitarono ai loro PC: ogni pezzo di hardware che potevano trovare era utile per Linux. Quelle persone furono chiamate "*nerd*" o "*freak*".

La prima versione del kernel Linux, la 0.01, fu pubblicata su Internet il 17 settembre 1991 e la seconda, la prima "ufficiale", il 5 ottobre dello stesso anno. Il messaggio che accompagnava quella release 0.02 è ormai passato alla storia dell'informatica:

«*Rimpiangete i bei giorni di Minix 1.1, quando gli uomini erano uomini e scrivevano da soli i driver? Vi manca un bel progetto e morite dalla voglia di spezzarvi le ossa con un Sistema Operativo che potete provare a modificare*

secondo le vostre necessità? Trovate frustrante che su Minix funzioni tutto? Non passate più intere notti per ottenere un programma che lavora meravigliosamente? Allora questo post dovrebbe essere fatto apposta per voi.

Come ho detto un mese fa, sto lavorando su una versione libera di un clone di Minix per computer AT-386. Ha finalmente raggiunto uno stadio in cui è usabile (sebbene non possa dipendere da ciò che volete), e sono favorevole a rendere pubblici i sorgenti per la grande massa. È solo la versione 0.02 (è già pronta una patch molto piccola), ma ho avviato con successo la shell bash/GCC/GNU-make/GNU-sed/compress eccetera. [...]

ATTENZIONE! NOTA! Questi sorgenti necessitano di minix-386 per essere compilati (e gcc-1.40, 0, non è stato testato, 1.37.1), e impostati correttamente per il funzionamento, quindi non è ancora un sistema indipendente per chi non avesse Minix. Ci sto lavorando. Devi anche essere parecchio esperto per impostarlo correttamente, quindi per quelli che speravano in un'alternativa a minix-386, per favore ignoratemi. Al momento è indirizzata ad esperti interessati ai sistemi operativi e 386 con accesso a Minix.

Il sistema necessita di un disco rigido AT-compatibile (IDE va bene) ed EGA/VGA. Se siete interessati vi prego di scaricare il README e le note di rilascio, e/o spedirmi un'e-mail per ulteriori informazioni.

Riesco (più o meno) a sentirvi mentre vi chiedete "perché?". Hurd uscirà tra un anno (o 2, o il prossimo mese, chi lo sa), e ho già Minix. Questo è un programma per programmatori scritto da un programmatore. Mi è piaciuto scriverlo, e a qualcuno potrebbe piacere darci un'occhiata e anche modificarlo per le proprie esigenze. È abbastanza piccolo da capire, usare e modificare, e sono curioso dei vostri commenti.

Mi piacerebbe anche sentirli da chiunque abbia scritto qualsiasi utility o funzione di libreria per Minix. Se i vostri sforzi sono liberamente distribuibili (sotto copyright o anche di pubblico dominio), mi piacerebbe sentirvi, in modo che possa aggiungerli al sistema. Sto usando Earl Chews estdio al momento (grazie per un sistema carino e funzionante Earl), e lavori simili saranno molto ben accetti. Il vostro C sarà ovviamente lasciato intatto. Scrivetemi due righe se volete lasciarmi usare il vostro codice.»

Torvalds preferiva chiamare *Freax* il kernel a cui stava lavorando, ovvero Freaks con la x d'ordinanza, ma Ari Lemmke, assistente alla Helsinki University of Technology, che gli aveva offerto lo spazio FTP per il progetto

(*ftp.funet.fi*), preferì assegnare alla subdirectory dedicata il nome di lavorazione *Linux*, ovvero *Linus* + *UNIX*.

Due anni dopo il messaggio di Linus c'erano già 12000 utenti Linux.

Fino alla versione 0.10 era richiesto un computer con Minix per configurare, compilare e installare Linux, perché quest'ultimo usava il filesystem del sistema sul quale si appoggiava. Ben presto, però, i sistemi Linux avrebbero superato Minix in termini di funzionalità.

La versione 0.12 del kernel Linux, uscita nel 1992, portò con sé una grande novità: Linux, fino ad allora "*copyrighted by Linus Torvalds*", adottò la licenza GPL. Questa decisione, che ha decretato il successo e la rapida espansione di Linux, portò quindi all'integrazione dei componenti GNU di Richard Stallman, dando vita a un sistema operativo completo, pienamente funzionante e, soprattutto, libero. Era nato *GNU/Linux*.

Nella primavera del 1992 l'hacker Orest Zborowski riuscì a rendere eseguibile il gestore grafico noto come "X Window System" sulla versione 0.12 di Linux. Per far ciò, Zborowski dovette implementare tutta la struttura degli Unix Domain Socket indispensabili a X Window e quindi un primo livello socket sul quale venne poi costruita tutta l'infrastruttura di rete di Linux. In realtà, il tutto era imbastito in maniera un po' caotica e non era ben integrato all'interno del kernel, ma Linus accettò comunque la patch, perché con essa era possibile sia utilizzare X, sia utilizzare tale infrastruttura per dotare Linux di uno stack di rete. Entusiasta della novità, Linus rilasciò, dopo la versione 0.12, la versione 0.95, senza pensare a tutti i problemi di sicurezza che la rete avrebbe comportato. Per rimediare alla leggerezza, nei due anni che trascorsero dalla 0.95 alla 1.0, Linus dovette utilizzare sia un ulteriore numero per indicare il livello di patch sia le lettere dell'alfabeto (*sino alla versione 0.99.15Z, 0.99 150 livello di patch, revisione Z*).

Nei primi anni '90 nacquero i primi LUGs (*Linux User Groups*) e le prime distribuzioni Linux, tra cui RedHat, Debian e SUSE, tuttora fra le distribuzioni più diffuse. Nel 1996, probabilmente in seguito a un piccolo incidente che lo stesso Linus Torvalds ebbe in uno zoo, fu scelto come logo ufficiale di Linux un pinguino disegnato da Larry Ewing, chiamato "*Tux*" da James Hughesin come abbreviazione di *Torvalds Unix*. Nel 1997 la storia di Linux divenne sempre più indipendente da Torvalds, che lasciò la Finlandia per trasferirsi a Santa Clara, nella Silicon Valley, dove assunse un ruolo non molto chiaro alla Transmeta, una misteriosa azienda tra i cui soci figurava uno dei

cofondatori di Microsoft, Paul Allen. La Transmeta progettava di produrre microprocessori a basso consumo e assunse programmatori e tecnici molto esperti, pur pagando Torvalds perché continuasse a sviluppare il suo kernel. Nel 2003 Linus Torvalds pose fine alla sua avventura alla Transmeta per dedicarsi interamente allo sviluppo di Linux all'*Open Source Development Lab*. L'ODSL, fondata nel 2000, era un'organizzazione non-profit dedicata alla diffusione di Linux come sistema operativo. Nel 2007 l'associazione si è fusa con la Free Standards Group per formare la *Linux Foundation*.

Nel 2004, sotto la spinta del milionario sudafricano Michael Shuttleworth, nacque Ubuntu, una distribuzione basata su Debian tra le più utilizzate al mondo per la sua facilità d'uso e la sua interfaccia grafica user-friendly. Il termine "*Ubuntu*", mutuato dalla lingua zulu, significa letteralmente "*umanità verso gli altri*", e si riferisce a una filosofia africana il cui motto centrale è "*Io sono ciò che sono, per merito di ciò che siamo tutti*".

Nel luglio del 2011, per festeggiare il XX anniversario della nascita di Linux, Torvalds decise di rilasciare la versione 3.0 del kernel Linux, passando a un sistema di numerazione a 2 cifre.

Linus Torvalds dirige ancora oggi lo sviluppo del kernel Linux. Il compito di fornire un sistema integrato, che combini tutte le componenti di base con le varie interfacce grafiche (come per esempio GNOME o KDE) e con il software applicativo, è svolto dalle distribuzioni. Per quanto riguarda il kernel vero e proprio, Torvalds già nel settembre 2009 dichiarò che esso è diventato "*gonfio e grosso*", non così veloce e scattante come avrebbe voluto quando l'aveva progettato. Questo "ingrassamento", riconosce però Torvalds, non va visto solo come una cosa negativa, perché significa che Linux ha molta più compatibilità hardware rispetto al passato.

Ci sono voluti quasi 30 anni per arrivare dove siamo ora. È stata fatta tanta strada, costruendo qualcosa di unico e prima d'oggi neppure pensabile. Sviluppatori, tester, sistemisti, utenti, hanno dato ciascuno il loro contributo e si sono tutti guadagnati un pezzetto di merito nella realizzazione del più grande progetto informatico libero esistente. Oggi il kernel Linux viene utilizzato in tutti i continenti, nei laboratori di ricerca scientifica, nelle basi spaziali della NASA e gestisce la maggior parte dei server che diffondono internet nelle nostre case. Linux è inoltre alla base di Android, il sistema operativo per dispositivi mobili più diffuso.

Free Software e Open Source

Eric Raymond, un hacker controverso, autore del testo "*La cattedrale e il bazaar*", descrive nel suo libro due modalità contrapposte per lo sviluppo del software: il modello "*a cattedrale*" e il modello "*a bazaar*". Secondo il primo, il programma viene realizzato da un limitato numero di "esperti" che scrivono il codice in condizioni di quasi totale isolamento. L'organizzazione gerarchica per lo sviluppo del programma è in questo caso molto stretta e ogni sviluppatore si preoccupa esclusivamente della piccola porzione di codice da lui sviluppata. Le revisioni si susseguono quindi con relativa lentezza; è compito degli sviluppatori ("*pochi*") cercare di rilasciare programmi che siano il più possibile completi e senza bug. Il modello a cattedrale è quello predominante per i software commerciali.

La seconda modalità, invece, quella *a bazaar*, prevede che il codice sorgente del programma sia disponibile gratuitamente e liberamente per tutti, in modo tale che gli utenti possano interagire con gli sviluppatori e, se ne sono in grado, possano anche contribuire allo sviluppo del software. Non esiste quindi in questo caso una scala gerarchica stretta, in quanto lo sviluppo è "decentralizzato" e i programmatori possono modificare e integrare qualsiasi porzione del codice. Il modello *a bazaar* è il modello alla base del software libero e dell'open source.

Il Software Libero

In accordo con quanto esplicitamente dichiarato dalla Free Software Foundation, con l'espressione "*Software Libero*" s'intende la "*libertà dell'utente di eseguire, copiare, distribuire, studiare, cambiare e migliorare il software*". Più nello specifico quando si parla di software libero si fa riferimento a quattro libertà essenziali:

- *Libertà 0*: Libertà di eseguire il programma come si desidera, per qualsiasi scopo.
- *Libertà 1:* Libertà di studiare come funziona il programma e di modificarlo in modo da adattarlo alle proprie necessità. L'accesso al codice sorgente ne è un prerequisito.
- *Libertà 2*: Libertà di ridistribuire copie in modo da aiutare il prossimo.
- *Libertà 3*: Libertà di migliorare il programma e distribuirne pubblica-

mente i miglioramenti, in modo tale che tutta la comunità ne tragga beneficio. L'accesso al codice sorgente ne è un prerequisito.

Il progetto GNU e le sue quattro libertà sono i fondamenti di GNU/Linux, un sistema operativo libero e condiviso, non necessariamente gratuito, rispettoso dei diritti dell'utente e aperto al contributo di tutti.

Secondo Richard Stallman, fondatore del progetto GNU, e secondo la Free Software Foundation, la dicitura "Linux" (senza prefisso "GNU/") per i sistemi operativi che utilizzano software GNU sarebbe erronea, in quanto il nome Linux è attribuibile al solo kernel. Il sistema operativo completo, strutturato a partire dai componenti dell'originale progetto GNU, in unione al kernel sviluppato da Linus Torvalds, dovrebbe quindi essere chiamato più correttamente *GNU/Linux*. Secondo l'uso della maggior parte degli utenti, degli sviluppatori e delle società coinvolti nello sviluppo del sistema operativo e del software a esso collegato, il nome Linux è ormai invece divenuto sinonimo di sistema "*Linux based*", cioè di sistema basato sul kernel Linux.

Differenze con l'Open Source

Una volta compresa l'importanza del software libero, nel momento in cui hanno cominciato a delinearsi interessi economici, o di altro genere, si è posto il problema di definire in modo preciso e inequivocabile cosa sia effettivamente il «software libero».

Il 3 febbraio 1998 Christine Peterson, futurista e docente nel campo delle nanotecnologie, coniò il termine "*Open Source*" ("*sorgente aperto*"), allo scopo di identificare i principi secondo cui il software può essere ritenuto libero. Qualche mese più tardi, Eric Raymond e Bruce Perens diedero vita alla *Open Source Initiative* (OSI), un'organizzazione educativa e di difesa dedicata a promuovere l'utilizzo di software open source. Nonostante le buone intenzioni, però, il termine open source resta ancora oggi piuttosto ambiguo, dal momento che non sintetizza il significato che vorrebbe avere, soprattutto se relazionato a quello di *software libero*. Semplificando, si può infatti dire che l'Open Source si basa su principi pragmatici ed economici, mentre il Free Software (così come descritto nella GPL) ha radici più etiche e filosofiche. In inglese, dove la terminologia utilizzata potrebbe creare più confusione, si usa definire l'open source come "*free as beer*", ovvero *free* inteso come gratuito, e il free software come "*free as speech*", ovvero *free* inteso come *libertà* (di parola). Secondo l'OSI si dovrebbe rilasciare il codice

del proprio software non solo perché è una scelta commercialmente sostenibile e valida, ma anche perché quando il codice sorgente di un programma è disponibile per tutti i programmatori che vogliano leggerlo, distribuirlo e/o modificarlo, allora questo arriverà a essere maturo. Tale programma sarà cioè più flessibile e di qualità superiore, perché molte più persone lo avranno testato nelle condizioni più varie, rispetto a quanto avviene per i software tradizionali. Il software open source sembra rispondere più a una necessità pratica, piuttosto che a una filosofia. In particolare, vale per il software open source quella che è comunemente conosciuta come "Legge di Linus" (così battezzata dallo stesso Eric Raymond): "*Dato un numero sufficiente di occhi, tutti i bug vengono a galla*".

Per la FSF (*Free Software Foundation*) il codice dovrebbe invece essere aperto per rispettare la libertà degli autori e degli utenti.

Per spiegare meglio le differenze tra software libero e open source è forse bene, così come è stato fatto per il free software, elencare le caratteristiche generali che un software deve possedere per poter essere definito open source; leggendole non potrà che emergere subito la differenza tra le due visioni, differenza che, occorre dirlo, è quasi puramente filosofica. Ecco quindi i 10 diritti illustrati dalla "Open-Source Definition":

1. Ridistribuzione libera, ovvero libertà di ridistribuire il proprio software a qualcun altro, chiedendo o meno un corrispettivo in denaro.
2. Disponibilità del codice sorgente.
3. I prodotti derivati devono essere consentiti: se qualcuno migliora un programma deve anche avere la possibilità di distribuirne il risultato.
4. Integrità del codice sorgente dell'autore: se si apporta un cambiamento si dovrebbe modificare il nome del programma o evidenziare chiaramente il cambiamento fatto, in modo che questo non venga attribuito all'autore
5. Nessuna discriminazione verso singoli o gruppi di persone.
6. Nessuna discriminazione verso i settori di applicazione: il software dev'essere utilizzabile in un'azienda così come in una scuola.
7. La licenza dev'essere distribuibile.
8. La licenza non può essere specifica per un prodotto.
9. La licenza non può contaminare altri software.

10. La licenza dev'essere tecnologicamente neutrale.

Se è sempre vero che un software libero è anche open source, può non essere vero il contrario, ovvero un software open source potrebbe anche non essere libero. Un software che sia invece sia libero che open source può essere talvolta definito come FLOSS (*Free/Libre and Open Source Software*).

Infine, occorre citare un vantaggio non trascurabile nell'utilizzo di software libero e open source. L'utilizzo che si fa di questo tipo di software è infatti spesso più etico rispetto a quanto avviene per i software proprietari. Con etico non ci si riferisce allo scopo finale per il quale si sta utilizzando il software, quanto all'utilizzo in sé. La maggior parte delle volte che si utilizzano software commerciali, infatti, si è tentati di crackarne la licenza piuttosto che acquistarla regolarmente (*chi non l'ha fatto?*), compromettendo quindi il lavoro di chi si è dedicato allo sviluppo del codice proprietario. Con l'utilizzo del free software e dell'open source, quasi sempre gratuito, si ha invece sempre pieno diritto di utilizzo del software.

La licenza GNU GPL e il Copyleft

In breve, la licenza GNU GPL è una licenza per il software libero originariamente ideata da Richard Stallman nel 1989, ampiamente utilizzata negli applicativi GNU e per il kernel Linux. La licenza GNU GPL racchiude in termini giuridici le 4 libertà fondamentali del software libero ma, a differenza di altre licenze per il software libero, richiede che le opere derivate da quella protetta da questa licenza restino anch'esse libere. L'ultima versione pubblicamente disponibile della licenza GNU GPL è la versione 3 del 29 giugno 2007.

La licenza GNU GPL si basa su ciò che è comunemente conosciuto come *copyleft*, un termine evidentemente ispirato a quello di *copyright*, ma con un "ribaltamento": la parola "right", "diritto" in inglese, viene sostituita da "left", ovvero "ceduto". Nel copyleft, cioè, il detentore dei diritti di autore di un'opera (l'autore) ne concede l'utilizzo e la libera distribuzione, purché le eventuali versioni modificate dell'opera originaria siano distribuite con lo stesso regime giuridico, ovvero la stessa licenza.

Perché usare Linux

- Linux è *gratuito*; non ci sono tasse di registrazione o costi di licenza. Gli aggiornamenti sono gratuiti.
- Linux è *libero*: Linux è distribuito sotto licenza GNU GPL. Chiunque voglia farlo ha diritto di modificare Linux e, eventualmente, di ridistribuirne una versione modificata (con la stessa licenza).
- Linux è *portabile* su qualsiasi piattaforma hardware. Il kernel linux può essere reso funzionante e adattato a qualsiasi hardware in quanto la documentazione relativa a tale attività è liberamente accessibile.
- Linux è progettato per essere *sempre funzionante*. Come nel caso di UNIX, un sistema Linux può restare in esecuzione tutto il tempo senza alcuna necessità di essere riavviato. Ciò è valido addirittura anche durante l'aggiornamento del kernel.
- Linux è *scalabile*, può infatti essere adattato a qualunque sistema, indipendentemente dalla memoria disponibile, aggiungendo o rimuovendo i pacchetti appropriati.
- Linux *non rende* i PC obsoleti. Con l'avanzamento tecnologico e le sempre maggiori risorse richieste dai software, i PC tendono a invecchiare precocemente, risultando inadatti ai nuovi aggiornamenti o subendo rallentamenti significativi nelle prestazioni. Linux, grazie alla sua modularità e alla sua struttura, può essere installato su PC vecchi e ritenuti obsoleti, permettendo di sfruttare nuovamente questi sistemi, evitando che finiscano precocemente in discarica.

Perché NON usare Linux

- Esistono *migliaia di distribuzioni diverse*. Se da un lato la varietà può essere un vantaggio, dall'altro può essere un ostacolo iniziale a chi si avvicina per la prima volta al mondo Linux e non abbia molta dimestichezza con l'informatica.
- Linux non è *user-friendly*. Sebbene Linux per PC sia un sistema operativo "di nicchia", utilizzato dagli smanettoni e dagli utenti più consapevoli, in realtà sono stati compiuti enormi passi avanti rispetto al passato. L'avanzamento tecnologico delle GUI (*Graphical User Interface*) ha notevolmente semplificato l'utilizzo di Linux, rendendolo adatto anche agli utenti meno esperti.
- *Problemi di compatibilità.* Essendo poco diffuso a livello desktop, la maggior parte dei produttori hardware rilascia driver solamente per il sistema operativo di casa Microsoft, ovvero Windows. Rispetto al passato, occorre però evidenziare che attualmente i problemi di compatibilità hardware sono enormemente diminuiti.
- *Carenza di software.* Per carenza di software non si intende qui la scarsa disponibilità di programmi compatibili con Linux, quanto il fatto che molti software commerciali sono disponibili esclusivamente per Windows, talvolta per macOS, eccezionalmente anche per Linux. Dovendo quindi lavorare obbligatoriamente con un determinato software, non è detto che questo sia disponibile anche per Linux. Esistono tuttavia moltissime alternative compatibili con Linux, spesso di qualità anche superiore.

Foto

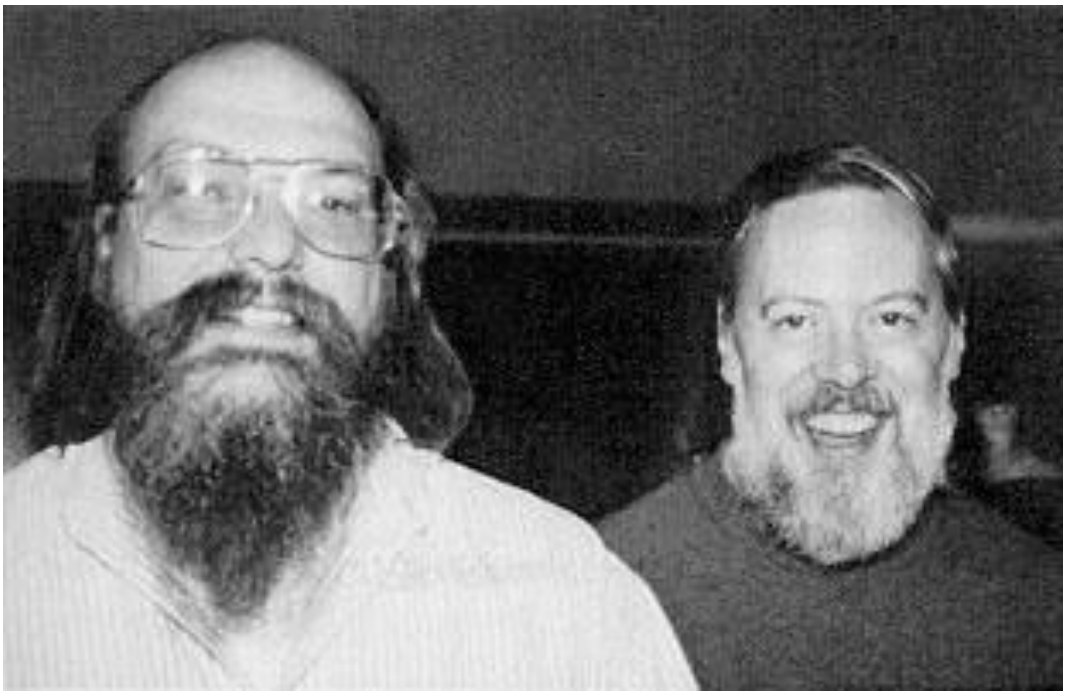


Figura 2: Ken Thompson e Dennis Ritchie

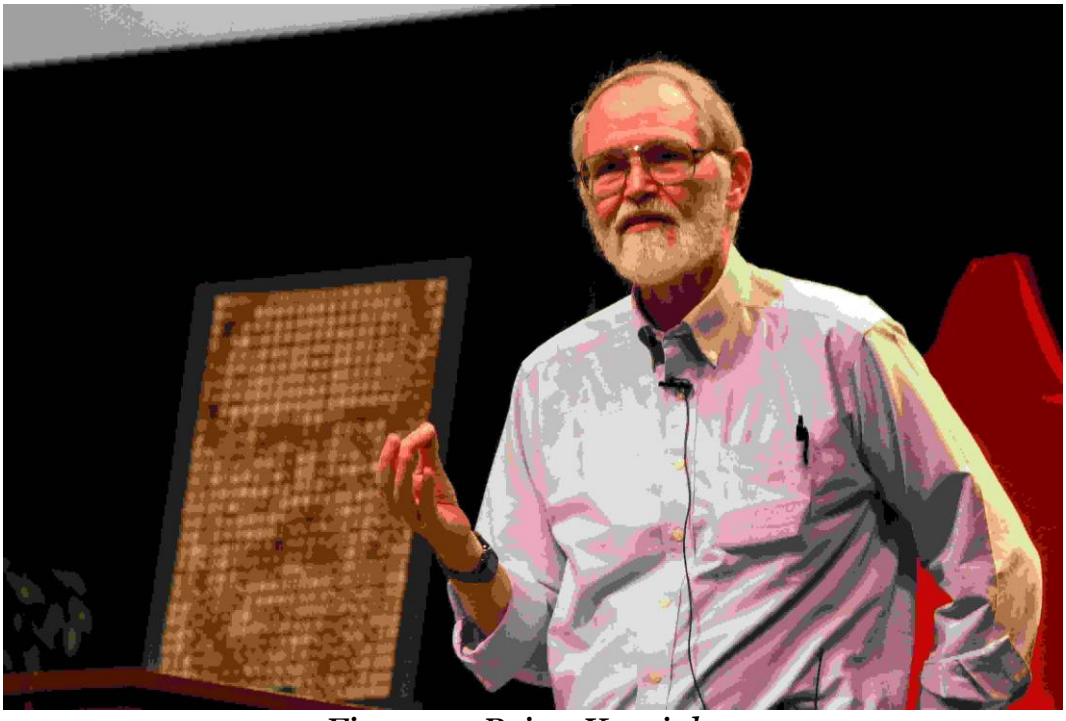


Figura 3: Brian Kernighan

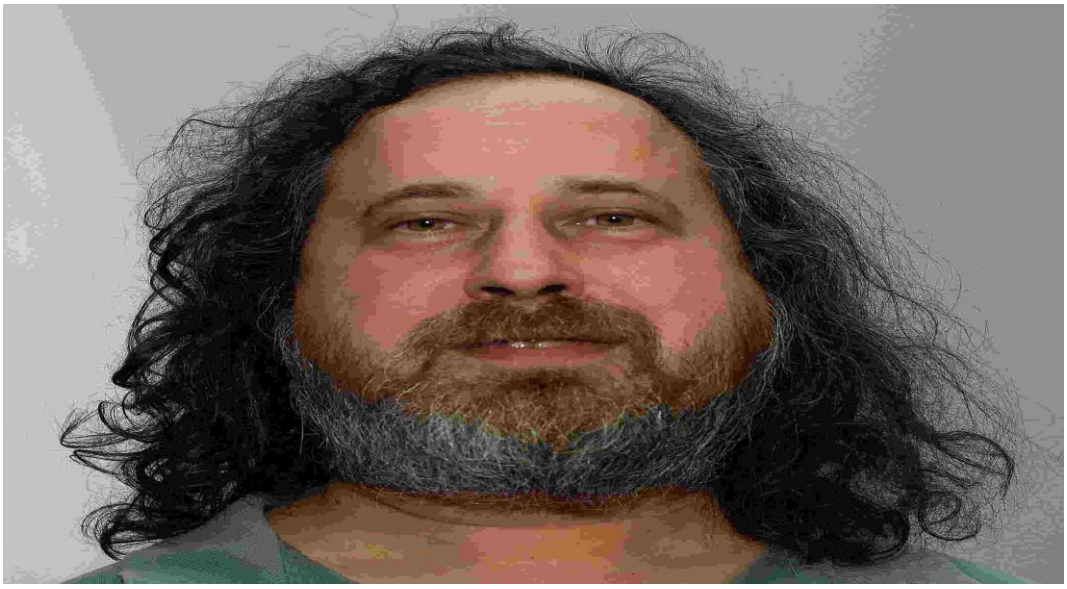


Figura 4: Richard Stallman

Figura 5: Andrew Tanenbaum

Figura 6: Linus Torvalds

Ringraziamenti

Ringrazio per la redazione di questo testo tutti coloro che hanno reso disponibile la propria conoscenza condividendola con la rete. Ringrazio il mio amico Lorenzo S. che da esperto e appassionato ha accettato di revisionare l'opera alla ricerca di errori e imprecisioni. Ringrazio inoltre tutti coloro che ogni giorno supportano e sopportano ciò che esce dalla tastiera del mio computer: scusatemi, il debito che ho con voi è inestinguibile.

Se questo testo è riuscito in qualche modo a creare in voi interesse e curiosità per il software libero, Linux e l'open source, qui di seguito potete trovare, oltre alle fonti consultate, anche un elenco non esaustivo di ulteriori siti internet di approfondimento.

Il miglior regalo che potete fare al mondo del software libero, e non solo, è quello di condividere la conoscenza, coinvolgendo amici e parenti nella speranza di creare insieme un mondo più equo, pulito e leale. Ricordatevi di Ubuntu: se siamo ciò che siamo, lo dobbiamo a tutti coloro che condividono con noi questo cammino. Grazie.

Fonti

Bibliografia

Introduzione a Unix: un po' di storia, C. Baroglio, Laboratorio di Sistemi Operativi, a. a. 2002-2003. UniTO.

Fondamenti di Linux, Davide Benvegnù.

Introduzione a Linux: una guida pratica, Machtelt Garrels, versione 1.25.0.

Linux: la nascita e la storia, Rocco Camera.

Personaggi ed aneddoti protagonisti della storia di GNU-Linux, Rosario Paone.

Software Libero, pensiero libero, Volume 1. Richard Stallman, Nuovi equilibri, 2003.

Software Libero, pensiero libero, Volume 2. Richard Stallman, Nuovi equilibri, 2004.

Just for Fun – The Story of an Accidental Revolutionary. Linus Torvalds and David Diamond. Harper Collins, 2001.

Codice libero – Richard Stallman e la crociata per il software libero. APOGEO, 2003.

Revolution OS II. Arturo Di Corinto. APOGEO, 2003.

Sitografia

http://catalogimages.wiley.com/images/db/pdf/0471164836.01.pdf

https://www.oakton.edu/user/2/rjtaylor/CIS118/Handouts/A%20History%20of%20UNIX.pdf

http://www.windoweb.it/edpstory_new/ex.htm

https://www.computerhope.com/history/unix.htm

https://opensource.org/osd.html

https://www.html.it/articoli/breve-storia-di-linux/

https://www.edscuola.it/archivio/antologia/corso_linux/2-1.htm

https://groups.google.com/forum/#!msg/comp.os.minix/4995SivOl90/

GwqLJlPSlCEJ

https://www.fastweb.it/web-e-digital/che-cos-e-linux/

https://it.wikipedia.org/wiki/OSDL

https://it.wikipedia.org/wiki/Unix

Revolution OS (J. T. S. Moore, 2001): https://www.youtube.com/watch?v=109RaSdGlOI&t=409s

Immagini

TUX:

lewing@isc.tamu.edu Larry Ewing and The GIMP [Attribution or CC0]

GNU:

Aurelio A. Heckert <aurium@gmail.com> - gnu.org, CC BY-SA 2.0, https://en.wikipedia.org/w/index.php?curid=33285325

Ken Thompson e Dennis Ritchie:

https://upload.wikimedia.org/wikipedia/commons/4/46/Ken_Thompson_and_Dennis_Ritchie.jpg

Brian Kernighan:

Ben Lowe [CC BY 2.0 (https://creativecommons.org/licenses/by/2.0)], via Wikimedia Commons

Richard Stallman:

NicoBZH from Saint Etienne - Loire, France [CC BY-SA 2.0 (https://creativecommons.org/licenses/by-sa/2.0)], via Wikimedia Commons

Andrew Tanenbaum:

Daniel [CC BY-SA 2.0 (https://creativecommons.org/licenses/by-sa/2.0)], via Wikimedia Commons

Linus Torvalds:

Krd [CC BY-SA 3.0 (https://creativecommons.org/licenses/by-sa/3.0) or CC BY-SA 4.0 (https://creativecommons.org/licenses/by-sa/4.0)], from Wikimedia Commons

Link per l'approfondimento

Free Software Foundation: https://www.fsf.org

Il Progetto GNU: http://www.gnu.org

Linux Foundation: https://www.linuxfoundation.org

Open Source Initiative: https://opensource.org

The Linux Kernel Archives: https://www.kernel.org

Note finali

 Questo libro è reso disponibile gratuitamente sul mio blog in PDF: *www.marcomoraschi.it*

Su Amazon sono disponibili la versione kindle e cartacea, che hanno un costo minimo compatibile con le politiche di Amazon e le spese di stampa.

Qualora lo vogliate, in caso di errori, suggerimenti o riflessioni potete scrivermi all'indirizzo mm@marcomoraschi.it

Grazie!

Marco Moraschi

Viandanti delle Nebbie